Optimierung von Problemstellungen aus der diskreten und der Prozess- Industrie unter Verwendung physikalischer Verfahren

Dissertation

zur Erlangung des Doktorgrades
der Naturwissenschaften (Dr. rer. Nat.)
der Naturwissenschaftlichen Fakultät II - Physik
der Universität Regensburg

vorgelegt von

Markus Puchta

aus Regensburg

2004

Das Promotionsgesuch wurde eingereicht am: 22. Juni 2004

Diese Arbeit wurde angeleitet von Prof. Dr. Ingo Morgenstern.

Prüfungsausschuss:
Vorsitzender: Prof. Dr. Dietmar Göritz
1. Gutachter: Prof. Dr. Ingo Morgenstern
2. Gutachter: Dr. Heinrich Braun
3. Gutachter: Prof. Dr. Uwe Krey
weiterer Prüfer: Prof. Dr. Uwe Krey

Termin des Promotionskolloquiums: 23. September 2004

Inhaltsverzeichnis

Einleitung **1**

Kapitel 1 Problemstellung **3**
1.1 Einleitung 3
1.2 Maschine 3
1.3 Alternative Maschinen 4
1.4 Umrüsten der Maschinen 4
1.5 Produktionszeit des Auftrags 4
1.6 Zeitliche Bedingungen 4
1.6.1 Zeitliche Bedingungen der Maschinen 5
1.6.2 Zeitliche Bedingungen der Aufträge 5
1.6.3 Zeitliche Bedingungen zwischen Aufträgen 5
1.7 Multiressourcen 6
1.8 Lager 6
1.9 Pausen 6
1.10 Maschinenkosten 6
1.11 Losgröße 7
1.12 Zielkriterien 7

Kapitel 2 Optimierungsverfahren **9**
2.1 Einleitung 9
2.2 Exakte Verfahren 9
2.2.1 Vollständige Enumeration 9
2.2.2 Branch&Bound (B&B) 10
2.3 Heuristiken 10
2.3.1 Zufallssuche 10
2.3.2 Konstruktionsheuristiken 10
2.3.3 Local Search 11
2.4 Details zu den verwendeten Verfahren 16
2.4.1 Simulated Annealing (SA) 16
2.4.2 Threshold Accepting (TA) 18
2.4.3 Erwartungswerte für Observablen 20
2.4.4 Abkühlstrategie (Cooling Schedule) 23
2.4.5 Temperature Bouncing (TB) 25
2.4.6 SA basierend auf der Akzeptanzrate (Acceptance SA) 26

Kapitel 3 Modelle der Produktionsplanung 27
3.1 Abbildung auf ein Spinsystem ... 27
3.1.1 Einleitung 27
3.1.2 1 Maschine, N Produktionsvorgänge 27
3.1.3 M Maschinen, N Produktionsvorgänge 28
3.1.4 Zeitliche Bedingungen zwischen Produktionsvorgängen 29
3.1.5 Modellierung JobShop, OpenShop, FlowShop 29
3.1.6 Zeitliche Bedingungen für die Maschinen 31
3.1.7 Zeitliche Bedingungen für die Produktionsvorgänge 31
3.1.8 Umrüsten der Maschinen 32
3.1.9 Maschinenkosten 33
3.1.10 Multiressourcen 34
3.1.11 Lager 34
3.1.12 Pausen 35
3.1.13 Konfiguration 35
3.1.14 Nachbarschaft 35
3.1.15 Nachteile des Spinmodells 36
3.2 Modell mit direkter Repräsentation ... 38
3.2.1 Einleitung 38
3.2.2 Ressourcen 38
3.2.3 Produktionsauftrag 43
3.2.4 Konfiguration 49
3.2.5 Nachbarschaft 50
3.2.6 Aufbau des Produktionsplans (*Schedule*) 54
3.2.7 Optimierungsziele 60

Kapitel 4 Vergleich der Produktionsplanungsmodelle mit Spinglasmodellen 63
4.1 Spinglasmodelle ... 63
4.1.1 Das dreidimensionale $\pm J$ Edward-Anderson-Modell 63
4.1.2 Das $\pm J$ Sherrington-Kirkpatrick-Modell 68
4.2 Modelle zur Simulation von Produktionsplanungsproblemen ... 71
4.2.1 Spinmodell 71
4.2.2 Modell mit direkter Repräsentation 77
4.3 Zusammenfassung ... 83

Kapitel 5 Das MT10x10 Problem 85
5.1 Problemstellung ... 85
5.2 Modell ... 85
5.3 Ergebnisse ... 85
5.4 Erweiterung ... 88
5.5 Weitere JobShop-Probleme ... 90
5.6 Zusammenfassung ... 92

Kapitel 6 Scheduling under Labour Resource Constraints 93
6.1 Problemstellung 93
6.2 Modell 94
6.3 Ergebnisse 96
6.3.1 Ergebnisse verschiedener Verfahren aus der Literatur 96
6.3.2 Ergebnisse der Verfahren SA und GR 97
6.4 PSPLib 101
6.5 Zusammenfassung 104

Kapitel 7 Production Flow Planning with Machine Assignment 105
7.1 Problemstellung 105
7.2 Modell 106
7.3 Ergebnisse 107
7.3.1 Ein-Tages-Pakete 107
7.3.2 Mehr-Tages-Pakete 109
7.4 Zusammenfassung 114

Kapitel 8 Das Benchmark Bench01 115
8.1 Problemstellung 115
8.2 Modell 117
8.3 Theoretische Überlegungen zu unteren Schranken 118
8.4 Ergebnisse 121
8.4.1 Vergleich verschiedener Nachbarschaften 121
8.4.2 Ergebnisse zu verschiedenen Prioritäten der Kriterien 125
8.5 Zusätzliche Aktivitätslinks im Modell 127
8.5.1 Fixiertes Pegging 127
8.5.2 Reihenfolgebeziehung zwischen Aufträgen 129
8.6 Veränderung der Produktionskapazitäten 131
8.6.1 Ergebnisse zur Kosteneinsparung 133
8.6.2 Ergebnisse zur erweiterten Produktionskapazität 134
8.7 Hinzufügen erforderlicher Wartungspausen 135
8.8 Vergleich mit anderen Verfahren 136
8.9 Reduzierung der benötigten Parameter 141
8.10 Zusammenfassung 144

Kapitel 9 Der Westenberger-Kallrath-Benchmark 145
9.1 Problemstellung ... 145
9.2 Modell ... 148
9.2.1 Veränderungen an der Modellierungsfunktionalität 148
9.2.2 Modellierung 150
9.3 Ergebnisse ... 151
9.3.1 Ergebnisse verschiedener Verfahren aus der Literatur 151
9.3.2 Ergebnisse des Verfahrens SA 153
9.3.3 Gegenüberstellung der Ergebnisse der verschiedenen Verfahren 155
9.3.4 Vergleich der Ergebnisse bei fixierter Losgröße 158
9.4 Bearbeitung einer neuen Aufgabenstellung ... 160
9.5 Zusammenfassung ... 162

Kapitel 10 Das Fließbandproblem 163
10.1 Problemstellung ... 163
10.2 Modell ... 164
10.2.1 Konfiguration 164
10.2.2 Kostenfunktion 164
10.2.3 Nachbarschaft 168
10.3 Spinglasverhalten ... 169
10.4 Benchmarkinstanzen CSPLib ... 171
10.4.1 Heuristiken 172
10.4.2 Lokale Suche 173
10.4.3 Ant Colony Optimization (ACO) 174
10.4.4 Ergebnisse 174
10.5 Realistische Problemstellung ... 178
10.6 Zusammenfassung ... 180

Zusammenfassung 183

Anhang 187
Anhang A - Flussdiagramme ... 187
Anhang B - Suchraum ... 191
B.1 Suchraum und Nachbarschaft der Konfiguration 191
B.2 Eine Maschine 192
B.3 Mehrere unabhängige Maschinen 193
B.4 Mehrere alternative Maschinen 193
B5 FlowShop 1 (Sonderfall eine Maschine) 194
B6 FlowShop 2 (Sonderfall mehrere unabhängige Maschinen) 194
B7 OpenShop 194
B8 JobShop 194
Anhang C – Production Flow Planning ... 195

Anhang D – Bench01 197
D.1 Aktivitäten 197
D.2 Simulationsdaten 198
D.2 Probleminstanzen 200
D.3 Lösungen 201
Anhang E – Westenberger-Kallrath 202
E.1 Aktivitäten 202
E.2 Lösungen 207

Danksagung **215**

Abbildungsverzeichnis **217**

Tabellenverzeichnis **225**

Literaturverzeichnis **233**

Einleitung

Die optimale Ausnutzung zur Verfügung stehender Produktionskapazitäten ist für Unternehmen unerlässlich, um die Produktionskosten zu senken und damit die Wettbewerbsfähigkeit zu steigern. Eine zielorientierte, durchdachte Planung bildet hierfür die Grundlage. Ihre Aufgabe besteht zunächst einmal darin, den zu fertigenden Produkten bestimmte Produktionsvorgänge zuzuordnen. Diese wiederum müssen in einem weiteren Schritt auf geeignete Maschinen verteilt werden, wobei bestimmte Zeitrahmen festgelegt werden. Um erfolgreich zu sein, ist es für eine Planung außerdem zwingend erforderlich, eine Reihe von Nebenbedingungen zu beachten. So müssen beispielsweise einzuhaltende Produktionspausen, alternativ einsetzbare Maschinen oder kapazitätsbeschränkte Lager berücksichtigt werden. Ein wesentliches Ziel dabei stellt die Minimierung der Durchlaufzeit, also der benötigten Produktionsdauer dar, neben der jedoch auch die Verringerung der Rüstzeit und die Einhaltung der Lieferfrist von großer Bedeutung sind. Weitere Ziele sind denkbar, beispielsweise die Minimierung anfallender Rüstkosten, die bei der Reinigung von Maschinen entstehen können. Diese Ziele werden meist in einer zu minimierenden Kostenfunktion zusammengefasst.

Zur Lösung der Problemstellungen soll ein auf statistischen Verfahren aus der Physik basierender Optimierungsalgorithmus entwickelt werden. Ziel dieser Arbeit ist es, ein möglichst einfaches Verfahren zu finden, mit Hilfe dessen man in möglichst kurzen Rechenzeiten möglichst gute Ergebnisse für realistische Problemstellungen erhalten kann. Oder anders ausgedrückt: Es wird *nicht* nach einem komplexen Verfahren gesucht, das auf einem stark vereinfachten, eigens konstruierten akademischen Benchmark in beliebig langer Rechenzeit ein Optimum finden und die Optimalität der gefundenen Lösung beweisen kann. Zusätzlich wird versucht, die für den Algorithmus nötigen Parameter automatisch zu steuern, so dass die Simulationen, nachdem das Modell von einem Experten erstellt wurde, auch von Produktionsplanern, die mit Optimierungsproblemen nicht vertraut sind, durchgeführt werden können.

Ein gutes Verfahren allein garantiert jedoch nicht den Erfolg. Ebenso wichtig ist die Darstellung des Optimierungsproblems in Form eines Modells. Im Modell werden die für lokale Suchverfahren benötigte Nachbarschaft und die Abbildung der Nebenbedingungen festgelegt. Diese müssen auf das Verfahren zugeschnitten sein. So ist eine günstige Kombination aus Modell und darauf arbeitendem Optimierungsverfahren erforderlich, um erfolgreich Optimierungsprobleme bestmöglich zu lösen. Das Modell soll dabei die benötige Funktionalität für eine Vielzahl an Problemstellungen zur Verfügung stellen, so dass die erstellte Software in einem breiten Problemfeld ohne spezielle Anpassungen eingesetzt werden kann. Zu den Standardproblemstellungen aus dem Bereich Scheduling, den JobShop- und FlowShop-Problemen, soll vor allem noch die Funktionalität einer variablen Losgröße zur Verfügung gestellt werden. Dies soll jedoch nicht zu einer ‚eierlegenden Wollmilch-

sau' führen, mit deren Hilfe sämtliche denkbare Problemstellungen besser gelöst werden können als auf einzelne Optimierungsprobleme spezialisierte Verfahren. Eine Erweiterung der Modellierung auf bestimmte Problemstellungen kann in Einzelfällen durchaus erforderlich sein.

Getestet wird die so erstellte Kombination aus Modell und darauf arbeitendem Verfahren an verschiedenen realistischen Benchmarks aus dem Bereich der Prozessindustrie und der diskreten Industrie. Dabei werden Problemstellungen verschiedener Firmen verwendet, die die Funktionalität mehrerer zum Teil real existierender Produktionsanlagen miteinander kombinieren, um den Einsatz verschiedenster Verfahren hinsichtlich ihrer Bedürfnisse zu vergleichen.

Die Arbeit ist in zehn Kapitel aufgeteilt. In Kapitel 1 wird die zugrundeliegende Problemstellung der Produktionsplanung mit der benötigten Funktionalität erläutert. Kapitel 2 gibt einen kurzen Überblick über Verfahren, die zur Lösung der gestellten Aufgabe verwendet werden können und erläutert die in dieser Arbeit verwendeten Verfahren aus dem Bereich *Local Search*. In Kapitel 3 werden zwei grundsätzlich unterschiedliche Ansätze gezeigt, mit deren Hilfe sich Produktionsplanungsprobleme abbilden lassen. Es handelt sich dabei um ein Modell basierend auf einer Spindarstellung, wobei die meisten der Nebenbedingungen als Strafterm in der Kostenfunktion modelliert sind, und ein Modell mit einer direkten Repräsentation der benötigten Funktionalität, bei dem die meisten Nebenbedingungen von jeder Lösung beachtet werden. Ein Vergleich der beim S*imulated Annealing* temperaturabhängigen Observablen wird in Kapitel 4 zwischen verschiedenen Modellen eines Spinglases und der zwei zuvor beschriebenen Modelle für Produktionsplanungsprobleme durchgeführt. In den Kapiteln 5 bis 8 wird das erarbeitete Modellierungs- und Optimierungskonzept auf verschiedenen Problemstellungen getestet. Neben einfachen *JobShop*-Problemen, wobei sich der Begriff ‚einfach' hier auf die benötigte Funktionalität bezieht, werden vor allem Benchmarks der BASF AG und SAP AG ausführlicher diskutiert. Im folgenden Kapitel 9 wird die bereits erwähnte Funktionalität der variablen Losgrößenplanung dem Modell als Funktionalität hinzugefügt und anhand einer komplexen Problemstellung von H. Westenberger und J. Kallrath (**[9.1]**), das in einer Kooperation der BASF AG und der Bayer AG entstand, getestet. Im zehnten Kapitel werden schließlich verschiedene Modellierungen für eine Problemstellung aus der diskreten Industrie, dem Fließbandproblem, vorgestellt.

Kapitel 1

Problemstellung

1.1 Einleitung

Das hier betrachtete Produktionsplanungsproblem besteht aus *n* Aufträgen, die auf *m* Maschinen bearbeitet werden. Ein Produktionsplan legt die Produktionsintervalle der einzelnen Aufträge auf den Maschinen fest. Der Produktionsplan ist gültig, falls sich die Produktionsintervalle eines Auftrages und die Produktionsintervalle der Aufträge auf einer Maschine nicht überschneiden. Ein Auftrag kann somit nicht auf mehreren Maschinen gleichzeitig bearbeitet werden und eine Maschine kann nur einen Auftrag zu einer bestimmten Zeit bearbeiten. Zusätzlich muss ein gültiger Produktionsplan noch weitere problemspezifische Restriktionen einhalten. Zu diesen Restriktionen gehören zeitliche Bedingungen, sekundäre Ressourcen, Lager usw.. Das Produktionsplanungsproblem wird durch die Maschinen, Aufträge und Restriktionen bestimmt (siehe dazu auch **[1.1]**). Die einzelnen benötigten Funktionalitäten werden im Folgenden der Reihe nach eingeführt.

1.2 Maschine

Bei mehreren Maschinen unterscheidet man die Problemstellung nach der Reihenfolge, in der die Maschinen von den Aufträgen benötigt werden:

- Existiert eine feste, für alle Aufträge gültige Reihenfolge, so bezeichnet man das Problem im Allgemeinen als ein **FlowShop-Problem**.
- Unterscheiden sich die Reihenfolgen der einzelnen Aufträge voneinander, so handelt es sich um ein **JobShop-Problem**.
- Existiert keine feste Reihenfolge, so hat man ein **OpenShop-Problem** zu bearbeiten, bei dem neben den Produktionsintervallen noch die Reihenfolge, in der die Maschinen benötigt werden, für jeden Auftrag festzulegen sind.
- Neben diesen drei Standard-Problemstellungen existieren bei real vorkommenden Problemstellungen natürlich noch Probleme, die aus einer Kombination der beschriebenen Bedingungen bestehen. Zusätzlich kann es vorkommen, dass manche Aufträge nicht alle Maschinen benötigen, wie z.B. im Westenberger-Kallrath-Problem (vgl. **Kapitel 9**).

1.3 Alternative Maschinen

Statt einer einzelnen Maschine kann auch eine Gruppe von Maschinen existieren, die alternativ zueinander die Produktion durchführen können. Solche alternativen Maschinen werden in Produktionsstufen zusammengefasst. Bei den Produktionsproblemen spricht man bei alternativen Maschinen dann vom **Flexible-FlowShop**, **Flexible-JobShop** oder **Flexible-OpenShop-Problem**.

1.4 Umrüsten der Maschinen

Zwischen der Ausführung der einzelnen Aufträge auf einer Maschine kann eine Reinigung oder Umrüstung nötig sein. Dadurch wird die Maschine auf die Ausführung eines anderen Auftrags vorbereitet (z.B. Austausch eines Werkzeugs an der Maschine). Die Dauer des Rüstvorgangs kann dabei von der Maschine und der Reihenfolge der zu fertigenden Aufträge abhängen. Teilweise sind bei der Umrüstung noch zeitliche Bedingungen zu beachten, so ist z.B. bei Westenberger-Kallrath eine Reinigung auf jeden Fall durchzuführen, falls nach dem Ende eines Auftrags nicht sofort ein anderer Auftrag gestartet wird. Zusätzlich zur Rüstdauer können Kosten bei der Umrüstung auftreten, z.B. Material- und Energieverbrauch oder die Entsorgung anfallender Reststoffe. Neben diesen unmittelbaren Ausgaben bestehen die Rüstkosten noch aus sog. Opportunitätskosten. Darunter versteht man die Deckungsbeiträge, die dem Unternehmen deshalb entgehen, weil die Anlage während der Rüstzeit nicht produzieren kann. Da die Fertigungskapazitäten regelmäßig knapp sind, besteht der überwiegende Teil aus Opportunitätskosten. Die Bestimmung dieser Kosten ist meist problematisch. Bei der Optimierung werden deshalb bei den Rüstkosten nur die unmittelbaren Kosten betrachtet.

1.5 Produktionszeit eines Auftrags

Die Produktionszeit eines Auftrags ist abhängig von den Eigenschaften des Auftrags und denen der zur Produktion gewählten Maschinen. Manche Maschinen sind neuerer Bauart und deshalb schneller beim Bearbeiten des Auftrags.

1.6 Zeitliche Bedingungen

Es existieren zeitliche Bedingungen für die Maschinen und Aufträge, die in der Regel strikt einzuhalten sind. Eine Verletzung des Liefertermins (vgl. **Kapitel 1.6.2**) kann unter bestimmten Umständen jedoch zulässig sein. Die Minimierung solcher Verletzungen stellt dann ein weiteres Optimierungsziel dar.

1.6.1 Zeitliche Bedingungen der Maschinen

Bei der Planung sind frühest mögliche Zeitpunkte, ab denen die Maschinen für die Produktion einsetzbar werden, zu beachten. Folgende Umstände können diesen Zeitpunkt beeinflussen:

- Aufgrund vorhergehender Planung kann es sein, dass eine Maschine bis zu einem bestimmten Zeitpunkt bereits ausgelastet, also nicht verfügbar ist (rollierende Planung).
- Bei der Neuanschaffung einer Maschine steht diese erst nach ihrer Aktivierung zur Verfügung.
- Durch die Wartung einer Maschine kann es zu längeren Ausfallzeiten kommen.

1.6.2 Zeitliche Bedingung der Aufträge

Bei den Aufträgen kann neben dem frühesten Startzeitpunkt auch das letztmögliche Ende der Produktion vorgegeben sein. Im Unterschied zum frühesten Startzeitpunkt (*release date*), der immer einzuhalten ist, unterscheidet man bei dem letzten Produktionszeitpunkt, dem Lieferzeitpunkt, zwischen einer *deadline*, die bei der Produktion ebenfalls eine harte Schranke darstellt, und einem *due date*, einer weichen Schranke, die nicht unbedingt eingehalten werden muss, deren Einhaltung aber wünschenswert ist. Je nachdem, wie wichtig die Einhaltung der weichen Lieferfrist (*due date*) ist, lassen sich verschiedene Prioritäten definieren.

Die zeitlichen Bedingungen können von folgenden Gegebenheiten beeinflusst werden:

- Benötigte Materialien/Rohstoffe stehen noch nicht zur Verfügung und werden erst zu einem bestimmten Zeitpunkt geliefert. Dies verzögert womöglich die Ausführung der Aufträge und erhöht evtl. deren frühest möglichen Startzeitpunkt (*release date*).
- Eine Nichteinhaltung der festen Lieferfrist (*deadline*) führt zu Umsatzverlusten, da der Kunde seinen Auftrag zurückzieht (*lost sales*).
- Eine Überschreitung der weichen Lieferfrist (*due date*) bedingt eine Konventionalstrafe bzw. Verzugskosten. Zusätzlich ist ein Imageverlust die Folge, der die Hinwendung des Kunden zu einer konkurrierenden Firma wahrscheinlich macht (*goodwill* Verlust).

1.6.3 Zeitliche Bedingung zwischen Aufträgen

Falls Material zwischen verschiedenen Aufträgen ausgetauscht wird, kommt es zu zeitlichen Bedingungen zwischen den Aufträgen. Ein Auftrag kann z.B. erst dann starten, wenn ein anderer, der Rohstoffe liefert, bereits teilweise oder vollständig abgearbeitet ist. Neben dieser Minimalabstandsbedingung, bei der ein minimaler Abstand zwischen Zeitpunkten der Aufträge (z.B. den Startzeitpunkten) festgelegt wird, existieren noch Maximalabstände. Nötig können solche Maximalabstände möglicherweise aufgrund von Verfallszeiten (*shelf live time*) von Materialien werden. Ein solcher Maximalabstand kann dazu führen, dass zwei Aufträge direkt nacheinander auszuführen sind, falls z.B. Materialien nicht gelagert werden können.

1.7 Multiressourcen

Neben den bereits eingeführten Maschinen, die nur jeweils einen Auftrag gleichzeitig ausführen können, bedarf es möglicherweise zusätzlich noch weiterer Ressourcen. Dabei kann es sich z.B. um Arbeitskräfte oder erforderliches Werkzeug handeln, die eventuell gleichzeitig von mehreren Maschinen benötigt werden. Jede dieser Multiressourcen besitzt eine zeitabhängige Kapazität. Die für einem Auftrag benötigte Kapazität kann ebenfalls zeitabhängig sein. Stehen alternative Multiressourcen zur Verfügung, so kann es sein, dass die Wahl der Multiressource die Produktionszeit beeinflusst, z.B. durch Experten-Arbeiter, die einen Arbeitsvorgang schneller durchführen können als die ‚normalen' Arbeitskräfte.

1.8 Lager

Lager dienen zur Aufbewahrung anfallender Zwischen- und Endprodukte und können einen maximalen Füllstand und/oder einen Mindestfüllstand aufweisen. Es gibt Lager, in denen nur ein Material gelagert werden kann (z.B. Silos) oder solche, in denen verschiedene Materialien gleichzeitig gelagert werden können (z.B. Kühlhäuser; hier werden die Materialien verpackt auf Paletten gelagert).

Die bisherigen Funktionalitäten **1.2** bis **1.8** werden im Allgemeinen auch als **MRCPSP** (*Multi-Mode Resource Constrained Project Scheduling Problem*) bezeichnet.

1.9 Pausen

Pausen treten auf, wenn es zu Produktionsunterbrechungen kommt. Dies kann in Mittagspausen, nicht vorhandenen Nachtschichten oder am Wochenende der Fall sein. Manche der Pausen sorgen für eine harte Produktionsunterbrechung, bei der keiner der Aufträge unterbrochen werden darf, d.h. wird ein Auftrag vor einer solchen Pause begonnen, so muss er auch vor der Pause beendet werden. Andere Pausen erlauben eine Pausierung bei der Produktion, d.h. Aufträge dürfen vor der Pause begonnen werden und können nach der Pause weiterbearbeitet werden. Die Produktionszeit des Auftrags wird dabei um die Pausendauer erhöht. Allerdings gibt es Aufträge, die niemals durch Pausen unterbrochen werden dürfen, z.B. weil Reaktionen mit fest vorgegebener Reaktionszeit stattfinden.

1.10 Maschinenkosten

Bei der Benutzung einzelner Maschinen, Multiressourcen oder Lager kann es zu Kosten kommen. Eine Maschine älterer Bauart benötigt möglicherweise mehr Energie bei der Produktion oder erzeugt höhere Wartungskosten.

1.11 Losgröße

Unter einer Losgröße eines Auftrags versteht man die Menge der durch den Auftrag hergestellten Produkte. Die Losgröße eines Auftrags kann bei der Planung festgelegt sein (*fixierte Losgröße*) oder erst durch diese bestimmt werden (*variable Losgröße*), wobei die Losgröße innerhalb bestimmter Grenzen frei gewählt werden kann, ohne dabei die Produktionsdauer des Auftrags zu verändern. Diese Grenzen sind in der chemischen Industrie z.B. vorgegeben durch die maximale Kapazität oder die Mindestfüllmenge eines Reaktionsgefäßes.

1.12 Zielkriterien

Bei der Optimierung in der Produktionsplanung sollen die Kosten der Produktion gesenkt werden. Da diese Kosten in der Regel nicht direkt ermittelbar sind, strebt man die Optimierung zeitbezogener Ziele an. Neben der Durchlaufzeit der Standard-Probleme existieren bei den meisten realen Problemen verschiedene andere Ziele mit unterschiedlichen Prioritäten, die bei der Optimierung zu beachten sind.

a) Durchlaufzeit (*makespan*):

Unter der Durchlaufzeit soll im weiteren Verlauf der Arbeit die Zeitspanne zwischen dem Beginn der Produktion und der Fertigstellung aller Aufträge, also die gesamte Produktionsdauer, verstanden werden.

Die Minimierung der Durchlaufzeit stellt ein wesentliches Ziel der Planung dar. Eine Verkürzung der Durchlaufzeit führt zu einer besseren Auslastung der Maschinen und damit zu einer erhöhten Effizienz. Die relativen Produktionskosten der Aufträge sinken.

b) Verspätung (*lateness*):

Eine Verspätung stellt eine Verletzung der weichen Lieferfrist (*due date*) eines Auftrags dar.

Hier unterscheidet man zwischen:

- der Summe der gewichteten Verspätungen,
- der maximalen Verspätung und
- der Anzahl der Verspätungen.

Man ist bestrebt, die vorgegebenen Fertigstellungstermine einzuhalten, da bei einer Überschreitung derselben mit Kosten zu rechnen ist (vgl. **Kapitel 1.6.2**). Eine Optimierung erfolgt mit Hilfe eines oder mehrerer der angesprochenen Ziele.

c) Rüstzeiten (*setup time*)

Die Reduzierung der Rüstzeiten (vgl. **Kapitel 1.4**) soll eine Reduzierung der Opportunitätskosten zur Folge haben, da bei knappen Fertigungskapazitäten mit einer Steigerung der Produktivität der Anlage bei reduzierten Rüstzeiten gerechnet werden kann.

d) Rüstkosten (*setup cost*)

Die Minimierung der Summe der anfallenden Rüstkosten (vgl. **Kapitel 1.4**) stellt ein weiteres Optimierungsziel dar.

e) Maschinenkosten

Werden bestimmten Maschinen ‚virtuelle' Kosten zugeordnet, so kann nach einer Simulation mit dem Optimierungsziel der Minimierung der Maschinenkosten (vgl. **Kapitel 1.10**) festgestellt werden, ob diese Maschinen für die Produktion benötigt werden.

Kapitel 2

Optimierungsverfahren

2.1 Einleitung

Bei den im Folgenden dargestellten Scheduling Problemen wird nach einer Lösung gesucht, deren Kosten minimal sind. Die Darstellung des Optimierungsproblems erfolgt mittels einer Konfiguration. Die Menge aller Konfigurationen wird als Konfigurationsraum *K* bezeichnet. Ist *K* endlich oder abzählbar unendlich, so spricht man von einem *kombinatorischen* Optimierungsproblem. Durch eine Zielfunktion $f_z : K \to R$ werden jeder Konfiguration Kosten zugeordnet. Eine Konfiguration $k \in K$ ist ein globales Optimum, falls ihre Kosten minimal sind $\left(f(k) \leq f(i) \quad \forall i \in K\right)$.

2.2 Exakte Verfahren

2.2.1 Vollständige Enumeration

Die konzeptuell einfachste Methode, alle global optimalen Lösungen zu finden, besteht darin, den gesamten Konfigurationsraum abzusuchen. Die exakte Enumeration ist bei realistischen Problemen meist nicht möglich, da bereits bei kleinen Problemstellungen *K* sehr groß ist. Angenommen, man hat ein Optimierungsproblem mit *n* binären Entscheidungsvariablen und die benötigte Rechenzeit für die Berechnung der Zielfunktion beträgt 10^{-10} Sekunden, so würde man die in **Tabelle 2.1** angegebene Gesamtrechenzeit für eine vollständige Enumeration benötigen.

n	**25**	**50**	**75**	**100**
t	$3.4 \cdot 10^{-3}$ Sekunden	31 Stunden	$1.2 \cdot 10^{5}$ Jahre	$4.0 \cdot 10^{12}$ Jahre

Tabelle 2.1: Benötigte Rechenzeit (***t***) für die vollständige Enumeration bei einer Problemstellung mit ***n*** binären Entscheidungsvariablen (z.B. Ising-Spins), falls die Berechnung einer Lösung 10^{-10} Sekunden benötigt.

Insgesamt gesehen ist es bereits bei mittleren Problemstellungen zwischen 50 und 75 binären Entscheidungsvariablen unmöglich, dieses Problem zu lösen. Auch bei kleineren Problemstellungen kann dieses Verfahren bei begrenzter Rechenzeit nicht durchgeführt werden.

2.2.2 Branch&Bound (B&B)

Ein globales Optimum lässt sich möglicherweise auch finden, ohne alle Konfigurationen zu bewerten. Bei diesem Verfahren werden die beiden folgenden Komponenten benötigt:

- **Branching**
  Die Konfigurationen bilden die Spitzen der Äste einer Baumstruktur. Jede Verzweigung/Knoten stellt bestimmte Systemeigenschaften dar, die alle Konfigurationen besitzen, die von diesem Knoten aus erreicht werden können.
- **Bounding**
  Es existiert ein Verfahren, das für jeden Knoten eine untere Schranke für die Lösungsqualität angeben kann, die für alle von diesem Knoten erreichbaren Konfigurationen gilt. Eine obere Schranke für die Lösungsqualität ergibt sich aus der besten bisher gefundenen Konfiguration. Ist nun die untere Schranke eines Knotens größer als die obere Schranke, so kann der Teilbaum unter diesem Ast ‚abgeschnitten' werden. Die erreichbaren Konfigurationen müssen nicht mehr beachtet werden, da ihre Lösungsqualität schlechter ist als die der besten bisher gefundenen.

Durch geeignete Wahl des **branching** und des **bounding** kann so die Anzahl der zu bewertenden Konfigurationen eingeschränkt werden und damit die Problemstellung in vertretbarer Rechenzeit gelöst werden. Allerdings ist hier nicht sichergestellt, dass jede auf einer Instanz der Problemstellung erfolgreiche Wahl auf anderen Instanzen der gleichen Problemstellung ebenso erfolgreich ist. Unter ‚erfolgreich' wird hier eine möglichst kurze Rechenzeit verstanden.

2.3 Heuristiken

2.3.1 Zufallssuche

Die einfachste, wenn auch nicht unbedingt erfolgreichste Heuristik ist die Zufallssuche. Bei dieser werden aus K zufällig Lösungen gewählt und die beste gespeichert. Allerdings kann man dabei nicht sicher sein, ein globales Optimum gefunden zu haben. Meist ist die auf diese Weise erreichte Lösungsqualität nicht besonders überzeugend.

2.3.2 Konstruktionsheuristiken

Diese Heuristiken versuchen, schrittweise eine Lösung aufzubauen. Ähnlich der Vorgehensweise eines Menschen wird dabei jede neue Komponente möglichst optimal in die bereits bestehende Teillösung integriert.

So wird beim *BestInsertion* (**[2.1]**) ein Ort in einem **TSP** (*travelling salesperson problem*) an der Stelle in eine noch unvollständige Tour einsetzt, an der er die gesamte Wegstrecke am wenigsten vergrößert.

Bei Sequenzproblemen in der Automobilindustrie versucht man, die nächste zu planende Position mit einem Fahrzeug zu belegen, das die wenigsten (gewichteten) Restriktionen verletzt (siehe dazu z.B. **[2.2]**).

Bei *Scheduling*-Problemen, wie sie in dieser Arbeit diskutiert werden, werden bestimmte zu planende Operationen nach ihren Eigenschaften priorisiert und dann nacheinander aktiv oder semiaktiv (vgl. **Kapitel 3.2.6.4**) in den bereits bestehenden Plan eingefügt. Dabei gibt es die verschiedensten Vorgehensweisen (vgl. z.B. **[2.3]**, **[2.4]**), die davon abhängig sind, welche Funktionalitäten vorhanden sind.
Die einzelnen Vorgehensweisen unterscheiden sich beispielsweise dadurch voneinander, dass zunächst Operationen

- mit kürzester Produktionsdauer,
- mit längster Produktionsdauer,
- mit kleinstem Lieferzeitpunkt,
- mit der geringsten Zeitspanne zwischen frühestem und spätestem Startzeitpunkt (z.B. gegeben durch Lieferzeitpunkt),
- mit längster Restlaufzeit, gegeben durch die Produktionsdauern der Nachfolger,
- mit größtem Ressourcenverbrauch oder
- mit den meisten direkten Nachfolgern

geplant werden.
Existieren verschiedene Möglichkeiten eine Operation auszuführen, so kann man diejenige

- mit kürzester Produktionsdauer,
- mit minimalem Verhältnis von Ressourcenbedarf zu Ressourcenkapazität oder
- mit minimalem Verhältnis von Ressourcenauslastung zu Ressourcenkapazität

wählen.

Im Folgenden werden Verbesserungsheuristiken vorgestellt, die ausgehend von einer vollständigen Lösung, durch Veränderungen eine weitere erzeugen und so versuchen, schrittweise die Lösungsqualität zu verbessern.

2.3.3 Local Search (LS)

LS startet mit einer beliebigen gültigen Lösung und versucht, diese schrittweise zu verbessern. Dazu werden Veränderungen an der aktuellen Lösung vorgenommen und mittels eines Auswahlkriteriums entschieden, welche davon beibehalten werden sollen. Die erlaubten Veränderungen definieren die Nachbarschaft *N* im Konfigurationsraum. Jede Konfiguration aus der Menge $N(k) \subseteq K$ ist dabei ein Nachbar der Konfiguration *k*. Verknüpft man den Konfigurationsraum mit der Nachbarschaftsstruktur, so erhält man den Suchraum. Die Bewertung aller Konfigurationen im Suchraum mit ihrer Zielfunktion ergibt die Energielandschaft. Bei **LS** bewegt man sich nun schrittweise durch diesen Suchraum, mit Hilfe der *Moves*, die aus einer Konfiguration *k k'* erzeugen, wobei $k' \in N(k)$. Eine Konfiguration *k* ist ein lokales Optimum, falls $(f(k) \leq f(i) \quad \forall i \in N(k))$. Die Menge der lokalen Optima hängt somit im Unterschied zu den globalen Optima sehr stark von der gewählten Nachbarschaft ab. Unter einer Ebene oder einem Plateau soll im Folgenden eine Konfiguration *k* verstanden werden, die bzgl. ihrer Nachbarn die gleiche Energie besitzt $(f(k) = f(i) \quad \forall i \in N(k))$. Nach der obigen Definition stellt eine solche Konfiguration (Ebene/Plateau) ein lokales Optimum

dar. Es soll nun noch ein striktes lokales Optimum *k'* definiert werden durch $\left(f(k') < f(i) \quad \forall i \in N(k')\right)$. Dieses existiert jedoch nicht unbedingt. Die folgenden Verfahren können dem Bereich **LS** zugeordnet werden. Sie unterscheiden sich meist in der Akzeptanzwahrscheinlichkeit der vorgenommenen Veränderungen. Zur vertiefenden Lektüre der folgenden Verfahren bieten sich u.a. folgende Werke an: **[2.5], [2.6]**.

2.3.3.1 Random walk

Bei der Suche wird jede Lösung akzeptiert. Die beste wird dabei gespeichert. Diese lokale Suche ist meist den folgenden Versionen unterlegen. Nur bei schwierigen Problemstellungen, bei denen die Nachbarschaft die Suche nicht unterstützt, z.B. bei *golf-hole*-Problemen, schneiden die folgenden Verfahren ähnlich schlecht ab. *Golf-hole*-Probleme sind solche, bei denen die ‚guten' Konfigurationen vereinzelt im Suchraum liegen und zu all ihren Nachbarn hohe Kostendifferenzen aufweisen. Der *random walk* entspricht einer Zufallssuche (vgl. **Kapitel 2.3.1**).

2.3.3.2 Greedy (GR)

Bei **GR**-Verfahren werden bei der Suche grundsätzlich nur Veränderungen akzeptiert, die Verbesserungen herbeiführen. Man unterscheidet dabei zwischen den *first improvement* Strategien, bei denen die erste Lösung aus der Nachbarschaft *N(k)* akzeptiert wird, die besser ist als *k*, und den *best improvement* Strategien, bei denen die beste Lösung aus *N(k)* akzeptiert wird, wenn sie besser als *k* ist. Bei beiden Strategien wird die Suche spätestens abgebrochen, sobald ein lokales Optimum gefunden ist.

Eine einfache Möglichkeit, lokalen Optima zu entkommen, die Plateaus darstellen, besteht darin, neben den verbessernden Veränderungen auch solche zu akzeptieren, die die Lösungsqualität nicht verschlechtern. Hier wird die Suche dann nicht mehr notwendigerweise in einem lokalen Optimum beendet, sondern nur noch in strikten lokalen Optima. Es kann nun vorkommen, dass zum Ende der Suche eine Menge an Lösungen *K'* endlos durchsucht wird, für die gilt: $\left(f(k) = f(i) \quad \forall i,k \in K'\right)$. Somit sollte noch ein weiteres Kriterium zum Beenden der Suche festgelegt werden, z.B. die maximale Anzahl an erzeugten Lösungen. Diese Variante des *first improvement* konnte beim Einsatz bei Scheduling Problemen Verbesserungen gegenüber der anfangs definierten **GR**-Strategie erzielen (**[2.7]**). Im Vergleich zu anderen Verfahren (z.B. *Simulated Annealing* (**SA**)) konnte sich **GR** jedoch nicht als eigenständiges Optimierungsverfahren behaupten. Nur bei solchen Problemen, die wenige Täler im Suchraum besitzen, arbeitet dieses Verfahren wirklich erfolgreich. Meist wird **GR** in Kombination mit anderen Verfahren eingesetzt, um ein lokales Optimum in der Nachbarschaft einer gegebenen Lösung zu finden, z.B. am Ende der Simulation mit **SA**.

2.3.3.3 Simulated Annealing (SA) und Threshold Accepting (TA)

Mit den **GR** Strategien ist es nicht möglich, aus (strikten) lokalen Optima zu entkommen. Die beiden Strategien **SA** und **TA** versuchen diesen Nachteil auszugleichen, indem auch schlechtere Lösungen akzeptiert werden. Ein Kontrollparameter steuert, wie stark die Verschlechterung sein darf. Grundsätzlich werden am Anfang größere Verschlechterungen akzeptiert als gegen Ende der Simulation. Diese Heuristiken werden in **Kapitel 2.4** ausführlicher diskutiert.

2.3.3.4 Great Deluge Algorithm (GDA)

Hier wird das biblische Szenario einer Sintflut simuliert, wobei Noah, diesmal ohne seine Arche, versucht, den höchsten Gipfel trockenen Fußes zu erreichen. Bei der Optimierung sucht man den tiefsten Punkt in der Energielandschaft, ähnlich einem Fisch in einem austrocknenden Tümpel, d.h. der Wasserstand beginnt bei einem hohen Wert und sinkt schließlich, bis das System in einem lokalen Optimum gefangen ist. Bei diesem *Sintflutalgorithmus* (**[2.8], [2.9], [2.10]**) werden bei der lokalen Suche alle Lösungen akzeptiert, deren Zielfunktionswert unterhalb der gegebenen Schwelle liegt, unabhängig von der Qualität der vorhergehenden Lösung. Dieses Verfahren hat den Nachteil, bei tiefem Schwellwert in lokalen Optima einzufrieren. Der Absenkung der Schwelle muss deshalb besondere Beachtung geschenkt werden. Eine Inselbildung bei hohem Schwellwert kann durch die Wahl einer geeigneten Nachbarschaft verhindert werden. Im Vergleich zu **SA** konnte **GDA** leider nicht erfolgreich auf Produktionsplanungsprobleme angewendet werden (**[2.7]**). **GDA** konvergiert im allgemeinen langsamer als **SA**.

2.3.3.5 Tabu Search (TS)

Falls im Unterschied zum **GR** noch Lösungen akzeptiert werden, die keine Verbesserung ergeben, um aus lokalen Optima zu entkommen, tritt nun das Problem auf, dass bereits bewertete Lösungen evtl. nochmals bewertet werden. Im schlimmsten Fall entstehen dadurch Zyklen bei der Suche. Um dies zu verhindern, soll daher ein Gedächtnis definiert werden, das, zusätzlich zur besten Lösung, der aktuellen Lösung und deren Nachbarschaft, weitere Informationen über die bisherige Lösungssuche speichert. Bei der weiteren Suche werden diese Informationen jeweils genutzt, um die Nachbarschaft der aktuellen Lösung zu verkleinern. Die Nachbarschaft ist deshalb abhängig von der bisherigen Suche, weshalb man dieses Verfahren auch der Gruppe der *dynamic neighborhood search techniques* zuordnen kann. Eine detailliertere Erklärung zu **TS** ist z.B. in **[2.11]** zu finden.
Die folgenden Komponenten werden benötigt:

- **Tabu Liste**
  In dieser Liste werden Lösungen gespeichert, die im nächsten Schritt nicht akzeptiert werden. Diese Liste wird bei jedem Schritt aktualisiert, ist dabei aber nicht notwendigerweise immer von der gleichen Länge. Ist das Speichern ganzer Lösungen zu aufwendig, kann die Liste auch mit Nachbarschaftsoperatoren gebildet werden, die im nächsten Zug nicht ausgeführt werden. Dadurch werden jedoch möglicherweise auch Lösungen verboten, die noch nicht bewertet wurden. Es existiert deshalb ein Kriterium (**aspiration level condition**), solche Lösungen dennoch zu akzeptieren.
- **Aspiration level condition**
  Dieses Kriterium sorgt dafür, dass Lösungen akzeptiert werden, die durch die Tabu Liste eigentlich verboten sind. Die einfachste Bedingung erlaubt Lösungen, die besser sind als die beste bisher gefundene.
- **Akzeptanzkriterium**
  Dieses Kriterium steuert, welche der Lösungen aus der verringerten Nachbarschaft akzeptiert werden. Ähnlich zu **SA** werden auch Lösungen akzeptiert, die eine Verschlechterung zur aktuellen Lösung darstellen.

- **Intensification - diversification**
  Bisher wurde das Gedächtnis verwendet, um eine Rückkehr zu bereits bewerteten Lösungen zu verhindern. Nun wird das Gedächtnis darüber hinaus noch eingesetzt, um bestimmte Regionen abzusuchen. Durch einen zusätzlichen Term in der Zielfunktion werden Lösungen bevorzugt, die nahe einer bestimmten Lösung liegen (*intensification*), oder solche, die wenig gemeinsam haben mit einer bestimmten Lösung (*diversification*). So kann man abwechselnd zuerst in der Nähe einer bestimmten Lösung eine intensive Suche durchführen, um dann wieder in einen anderen Teil des Suchraums zu springen.

2.3.3.6 Genetischer Algorithmus (GA) und Evolutionsstrategien (ES)

Diese Verfahren sind angelehnt an die genetischen Algorithmen von Holland **[2.12]** und die Evolutionsstrategien von Rechenberg **[2.13]**. Sie nutzen Konzepte aus der Genetik und der Evolutionstheorie, um die Fitness einer Population aus Individuen mittels Kombination und Mutation ihrer Gene zu optimieren. Es existieren viele verschiedene Ansätze, diese Ideen auf kombinatorische Optimierungsprobleme anzuwenden, z.B. **[2.14]**, **[2.15]**. Grundsätzlich wird bei diesen Verfahren mit einer Menge an Lösungen, den **Individuen**, gearbeitet, die als **Population** bezeichnet wird. Ein Optimierungsschritt, auch **Generation** genannt, besteht dann darin, aus der gegebenen Population Individuen auszuwählen (**Selektion**) und Nachkommen mit Hilfe von Nachbarschaftsoperatoren zu erzeugen. Es wird dabei entweder ein Individuum mutiert (***mutation***) oder die Information von mehreren kombiniert an die Nachkommen übergeben (***crossover***). Welche der Individuen, Eltern und/oder Nachkommen, in die Population der nächsten Generation übergeben werden bzw. überleben, steuert ein **Selektionsmechanismus** (***survival of the fittest***). Die Suche wird **terminiert**, falls ein bestimmtes Kriterium erfüllt ist, z.B. die maximale Anzahl der Generationen.

2.3.3.7 Iterated Local Search (ILS)

Unter **ILS** versteht man die iterative Anwendung von **LS**. Anstatt sich wie bei **LS** im Konfigurationsraum zu bewegen, besteht die Nachbarschaft nun aus lokal optimalen Lösungen, die durch ein Verfahren, basierend auf **LS**, im Folgenden **routineLS** genannt, erstellt werden. Man kann diese Suche nun ähnlich dem **LS** steuern und erhofft sich dabei Verbesserungen gegenüber einer Zufallsuche (*random restart*). Insgesamt gesehen handelt es sich also um ein **LS** Verfahren im Raum der lokal optimalen Lösungen. **ILS** wurde bereits in verschiedenen Optimierungsbereichen eingesetzt. Eine Anwendung des Verfahrens auf Produktionsplanungsprobleme kann man z.B. in **[2.16]** nachlesen.

Es werden die folgenden vier Komponenten benötigt (**[2.17]**, **[2.18]**):

- **Initiallösung**
  Dies ist die erste Lösung im Konfigurationsraum, auf die **routineLS** angewendet wird. Im einfachsten Fall handelt es sich um eine zufällige Lösung. Sie kann aber auch von anderen Verfahren erzeugt werden. Meist ist es von Vorteil, mit einer guten Lösung zu starten, jedoch sollte man die dafür aufgewendete Rechenzeit nicht außer acht lassen.

- **routineLS**
  Hier können die verschiedensten Verfahren eingesetzt werden, die im Konfigurationsraum, ausgehend von einer Initiallösung, ein lokales Optimum finden. Neben einfachen **GR**-Heuristiken, können der LinKernighan (z.B. für das **TSP [2.19]**) oder auch ausgefeiltere Heuristiken wie z.B. **SA** oder **TS** eingesetzt werden. Hier gilt im Grunde das Gleiche wie bei der Initiallösung; es ist besser, ein gutes Verfahren einzusetzen, jedoch sollte die dazu verwendete Rechenzeit beachtet werden. Es kann sein, dass es sich als Vorteil erweist, eine schnelle Heuristik öfters anwenden zu können als eine gute, aber langsame.
- **Veränderung (Perturbation)**
  Sie dient dazu, aus den lokalen Optima zu entkommen, indem sie eine Lösung im Konfigurationsraum erzeugt. Die Größe der Veränderung ist dabei eine wichtige Eigenschaft. Ist die Veränderung zu klein, so landet die anschließende **routineLS** im gleichen lokalen Optimum. Ist sie zu groß, wird die nächste **routineLS** von einer fast zufälligen Lösung aus starten. Es handelt sich dann um eine nicht gewünschte Zufallssuche.
- **Akzeptanzkriterium**
  Ähnlich zur **LS** wird nun noch ein Auswahlkriterium benötigt, das entscheidet, ob eine neue lokal optimale Lösung akzeptiert wird, oder ob die Suche bei der zuletzt gefundenen erneut gestartet wird. Hier kann ein Greedy-Kriterium ausreichend sein. Im Gegensatz dazu kann auch jede Lösung akzeptiert werden. Dies führt dann zu einer Zufallssuche im Raum lokal optimaler Lösungen. Die Auswahl des Akzeptanzkriteriums entscheidet, ob die Suche eher *intensification* oder *diversification* bewirkt. Meist ist es von Vorteil, ein Kriterium zu wählen, das zwischen den beiden Extrema liegt.

2.3.3.8 Ruin&Recreate (R&R)

Dem **ILS** ähnlich ist der Ansatz des **R&R**. Beginnend mit einer **Initiallösung**, wird in jedem Schritt ein Teil der aktuellen Lösung zerstört (**Veränderung**) und dann neu aufgebaut (**routineLS**). Ob die so erzeugte neue Lösung akzeptiert wird, entscheidet ein **Akzeptanzkriterium**.
Dabei werden die folgenden Komponenten benötigt:

- **Initiallösung**
  Es sollte hier nicht mit einer zufälligen Lösung begonnen werden. Zu empfehlen ist meist eine Lösung, die mit dem im **Recreate** verwendeten Ansatz erzeugt wird.
- **Ruin**
  Es werden bestimmte Teile aus einer vollständigen Lösung entfernt. Anhand des **TSP** werden in **[2.20]** verschiedene Vorgehensweisen vorgestellt. Neben dem zufälligen Entfernen von Orten aus der Rundtour können auch solche der Tour entnommen werden, die benachbart sind. Die Anzahl der entnommenen Teile aus der Lösung entspricht hier einem Parameter. Es empfiehlt sich, nicht mehr als 50% der Lösung zu verändern.
- **Recreate**
  Die aus der Lösung entfernten Teile werden in diesem Teil wieder in die Lösung integriert. Beim **TSP** geschieht dies in **[2.20]**, indem man einen Ort zufällig aus der Menge der entnommenen Orte auswählt und ihn an der Stelle in die Tour einfügt, an der er sie am wenigsten verlängert.

Die richtige Kombination aus **Ruin** und **Recreate** ist bei diesem Verfahren keine triviale Aufgabe. Nicht jede Kombination aus einem günstigen **Ruin** mit einem guten **Recreate** ist erfolgreich. Bei einer Zerstörung der Lösung um 50% beim **Ruin** kann eigentlich nicht davon gesprochen werden, dass es sich hier um ein lokales Verfahren (**ILS**) handelt, sondern vielmehr um die iterative Anwendung konstruktiver Verfahren auf eine bereits teilweise erstellte Ausgangssituation.
Neben dem **TSP** wurde das Verfahren bereits auf andere Problemstellungen übertragen, z.B. auf **VRP** (vehicle resource planning) und Netzwerkprobleme (**[2.21]**).

2.4 Details zu den verwendeten Verfahren

2.4.1 Simulated Annealing (SA)

SA ist ein in der Physik oft verwendeter Ansatz, um Zustände niedriger Energie für komplexe Probleme zu finden, die analytisch nicht lösbar sind. Wie der Name erwarten lässt, handelt es sich um ein Verfahren zur Simulation eines Abkühlvorganges **[2.22]**. Man versucht dabei, das Verhalten eines stark erhitzten geschmolzenen Festkörpers bei langsamem Abkühlen zu simulieren, der schließlich in einem Grundzustand mit minimaler Energie erstarrt. Das Verfahren basiert dabei auf einem *Monte-Carlo*-Prozess **[2.23]**.
Metropolis führte den Gedanken ein, bei der Suche einen *Markov*-Prozess mit einer geeigneten Übergangswahrscheinlichkeit zu verwenden. Bei einem *Markov*-Prozess werden schrittweise aufeinanderfolgende Zustände $\{k_i\}$ mit Hilfe eines *Moves* erzeugt, wobei die Wahrscheinlichkeit, einen Zustand k_{i+1} zu erhalten, nur von k_i und nicht k_j, wobei $j<i$, abhängt. Die Übergangswahrscheinlichkeit $p(k_i \to k_{i+1})$ sollte so gewählt werden, dass die Verteilungsfunktion der Zustände, die durch diesen *Markov*-Prozess erzeugt werden, im Limes $i \to \infty$ gegen die gewünschte kanonische Gleichgewichtsverteilung (*Boltzmann*-Verteilung)

$$P_{eq}(k_i) = \frac{1}{Z} \cdot \exp\left(-\frac{E(k_i)}{k_B \cdot T} \right) \qquad \textbf{(2.1)}$$

konvergiert. Dabei ist E die zugrundeliegende Energiefunktion, T die Temperatur, k_B die *Boltzmann*-Konstante, und

$$Z = \sum_{k_i \in L} \exp\left(-\frac{E(k_i)}{k_B \cdot T} \right) \qquad \textbf{(2.2)}$$

die Zustandssumme.
Ein stationäres Gleichgewicht stellt sich ein, falls die *Master-Gleichung* erfüllt ist:

$$P_{eq}(k_i) \cdot \sum_{k_j} p(k_i \to k_j) = \sum_{k_j} P_{eq}(k_j) \cdot p(k_j \to k_i) \quad \forall k_i \,. \qquad \textbf{(2.3)}$$

Eine hinreichende Bedingung dafür erhält man durch das Prinzip der *detailed balance*:

$$P_{eq}(k_i)\cdot p(k_i \to k_j) = P_{eq}(k_j)\cdot p(k_j \to k_i). \qquad \textbf{(2.4)}$$

Das Verhältnis der Übergangswahrscheinlichkeiten für einen *Move* $k_i \to k_j$ und den inversen *Move* $k_j \to k_i$ hängt also nur von der Energieänderung $\Delta E = E(k_j) - E(k_i)$ ab:

$$\frac{p(k_i \to k_j)}{p(k_j \to k_i)} = \exp\left(-\frac{\Delta E}{k_B \cdot T}\right). \qquad \textbf{(2.5)}$$

Durch **(2.5)** ist $p(k_i \to k_j)$ nicht eindeutig bestimmt (**[2.23]**). Eine häufig verwendete Wahl ist das sog. *Metropolis*-Kriterium **[2.24]**:

$$p(k_i \to k_j) = \min\left\{1, \exp\left(-\frac{\Delta E}{k_B \cdot T}\right)\right\} = \begin{cases} \exp\left(-\frac{\Delta E}{k_B \cdot T}\right) & falls\ \Delta E > 0 \\ 1 & sonst \end{cases}. \qquad \textbf{(2.6)}$$

Dies führt dazu, dass bessere Zustände immer angenommen werden, schlechtere nur mit einer bestimmten Wahrscheinlichkeit. Die Temperatur *T* ist ein Kontrollparameter, der während der Simulation von einem Startwert bis auf 0 gesenkt wird. Er wird bei der üblichen Wahl von k_B=1 bei der Optimierung, in Einheiten von *E* gesetzt. Im Laufe der Simulation wird also die Wahrscheinlichkeit, einen schlechteren Zustand anzunehmen, immer geringer.
Das *Metropolis*-Kriterium erfüllt die Bedingung der *detailed balance*.
SA erfüllt ferner die Bedingung der *Ergodizität*. Nach *Ehrenfest* (1911) wird ein System als ergodisch bezeichnet, wenn die im Phasenraum an eine Hyperfläche *E=const*. gebundene Phasenraumtrajektorie im Laufe der Zeit jedem Punkt beliebig nahe kommt. Für den Fall, dass der Phasenraum aus diskreten Punkten besteht, muss die Phasenraumtrajektorie jeden Punkt erreichen können.
Seit seiner Einführung durch *Kirkpatrick* (**[2.22], [2.25])** ist **SA** ein anerkanntes Verfahren in der kombinatorischen Optimierung.
Senkt man die Temperatur bei **SA** ausreichend langsam ab, so konvergiert die Verteilung der Zustände gegen eine Verteilung, bei der nicht-optimale Zustände die Wahrscheinlichkeit 0 besitzen (**[2.26]**). Ein geeignetes Temperaturschema ist durch

$$T = \frac{a}{b + \log(t)} \qquad \textbf{(2.7)}$$

gegeben, wobei a und b positive Konstanten sind, die von dem gestellten Optimierungsproblem abhängen und t die vergangene Rechenzeit darstellt.

2.4.2 Threshold Accepting (TA)

Dieses Verfahren (**[2.27]**) ist vom Prinzip her verwandt mit **SA**. Die Übergangswahrscheinlichkeit ist beim **TA** jedoch gegeben durch die *Heaviside*-Stufenfunktion:

$$p = \Theta(th - \Delta E) = \begin{cases} 1 & falls\ \Delta E \le th \\ 0 & sonst \end{cases}. \tag{2.8}$$

Die Schwelle (engl. *threshold*) *th* wirkt als Pseudotemperatur, die von einem anfänglich hohen Wert gegen 0 gesenkt wird. Dies führt ähnlich wie bei **SA** dazu, dass während der Simulation immer weniger Verschlechterungen angenommen werden.
Allerdings verletzt **TA** im Gegensatz zu **SA** die Bedingung der *detailed balance*. Dies wird deutlich bei einem System aus drei Zuständen k_i, k_j, k_m bei einem *threshold th* mit folgenden Eigenschaften:

$$E(k_i) < E(k_j) < E(k_m)$$
$$E(k_j) - E(k_i) < th$$
$$E(k_m) - E(k_j) < th$$
$$E(k_m) - E(k_i) > th$$

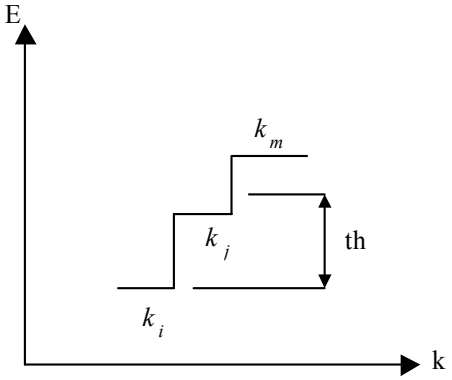


Abbildung 2.1: Die Bedingung der *detailed balance* ist nicht erfüllt!

Dies führt zu folgenden Übergangswahrscheinlichkeiten:

$$\begin{aligned} &p(k_i \to k_j) = 1,\ p(k_j \to k_i) = 1, \\ &p(k_j \to k_m) = 1,\ p(k_m \to k_j) = 1, \\ &p(k_i \to k_m) = 0,\ p(k_m \to k_i) = 1. \end{aligned} \tag{2.9}$$

Unter der Annahme, dass das Prinzip der *detailed balance* erfüllt ist, ergibt sich:

$$\begin{aligned} &p(k_i \to k_m) \cdot \pi(k_i) = p(k_m \to k_i) \cdot P(k_m) \Rightarrow P(k_m) = 0, \\ &p(k_j \to k_m) \cdot \pi(k_j) = p(k_m \to k_j) \cdot P(k_m) \Rightarrow P(k_j) = P(k_m), \\ &p(k_i \to k_j) \cdot \pi(k_i) = p(k_j \to k_i) \cdot P(k_j) \Rightarrow P(k_i) = P(k_j). \end{aligned} \tag{2.10}$$

Die Besetzungswahrscheinlichkeit $P(k_m)$ für Zustand k_m verschwindet, und damit auch die Besetzungswahrscheinlichkeiten der anderen Zustände. Die Bedingung der *detailed balance* ist somit nicht erfüllt, da die Summe aus den Besetzungswahrscheinlichkeiten 1 ergeben muss.

Auch verletzt **TA** die Bedingung der *Ergodizität*. Ein Zustand k_i mit der Nachbarschaft $N(k_i)$, für den gilt: $E(k_i) - E(k_j) > th \quad \forall k_j \in N(k_i)$, kann von einem beliebigen anderen Zustand aus nicht mehr erreicht werden (vgl. **Abbildung 2.2**). Es kann sich kein thermisches Gleichgewicht einstellen.

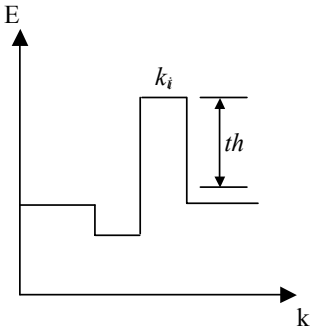


Abbildung 2.2: Die Bedingung der *Ergodizität* ist nicht erfüllt!
k_i bezeichnet man auch als ‚Zuckerhut'.

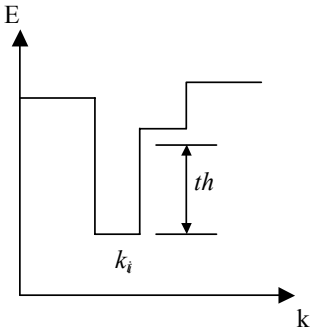


Abbildung 2.3: *Golf hole* k_i.

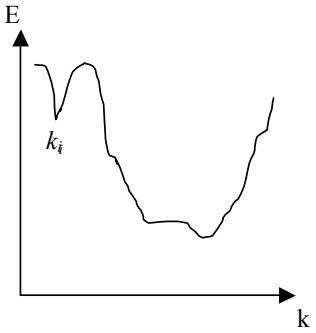


Abbildung 2.4: Ist das System im *golf hole* bei k_i gefangen, so werden andere, evtl. bessere, Zustände nicht mehr erreicht.

Die Verletzung der Bedingung der *Ergodizität* wirkt sich sehr stark bei Problemstellungen aus, bei denen viele Zustände k_i existieren, die zu all ihren Nachbarn Energiedifferenzen ΔE aufweisen, die größer als der gegebene *threshold th* sind: $\Delta E = E(k_j) - E(k_i) > Th \quad \forall k_j \in N(k_i)$ (vgl. **Abbildung 2.3**). Befindet sich das System in solch einen Zustand (*golf hole*), so ist es gefangen, d.h. es werden keine anderen, evtl. bessere, Zustände mehr erreicht (vgl. **Abbildung 2.4**). Neben diesen *golf holes* existieren womöglich auch ‚Inseln' *I*, die die Eigenschaft besitzen, dass alle Zustände außerhalb der Insel, die Nachbarn der Zustände der Insel sind, von diesen aus nicht mehr erreicht werden können: $E(k_j) - E(k_i) > th \quad (\forall k_i, k_j | k_i \in I \wedge k_j \notin I \wedge k_j \in N(k_i))$. Ein *golf hole* ist nach dieser Definition ebenfalls eine Insel. Eine Inselbildung findet beim Absenken des *thresholds* immer statt. Existieren

am Anfang der Suche bereits viele solcher Inseln, so ist die Lösungssuche stark abhängig von der anfänglich gewählten Lösung. Eine frühe Inselbildung lässt sich durch Erweiterung der Nachbarschaft jeder Lösung verhindern. Jedoch wird das System bei kleinem *threshold* immer auf solch einer Insel (in einem Tal der Energielandschaft) gefangen sein, wodurch es evtl. von Vorteil ist, die vorhandene Rechenzeit aufzuteilen und viele schnelle Simulationen durchzuführen.
TA kann man als einfache Näherung von **SA** betrachten, wobei das exponentielle Verhalten der Übergangswahrscheinlichkeit durch eine Stufenfunktion ersetzt wurde.

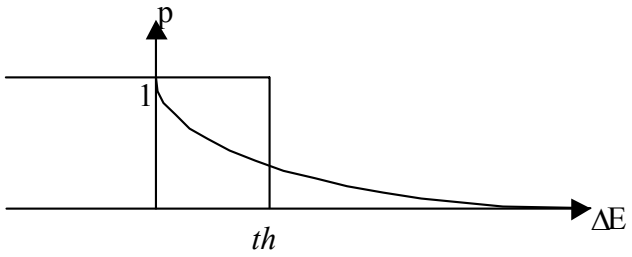


Abbildung 2.5: Näherung des exponentiellen Verhaltens der Übergangswahrscheinlichkeit von **SA** durch eine Stufenfunktion bei **TA**.

Der Übergangsthreshold und die Übergangstemperatur von einem ungeordneten energiereichen zu einem geordneten energiearmen System sollten in etwa dieselbe Größenordnung haben. Zur Vereinfachung wird im Folgenden auch bei **TA** von einer Temperatur und nicht mehr vom *threshold* gesprochen.
Wird der *threshold* auf 0 gesetzt, so handelt es sich um ein *first improvement* **GR**-Verfahren.

2.4.3 Erwartungswerte für Observablen

Für die folgenden Berechnungen sei vorausgesetzt, dass sich das System ergodisch ist und sich im thermischen Gleichgewicht befindet. Streng genommen ist dies allerdings nur für **SA** erfüllt. Trotzdem werden aber die erhaltenen Ergebnisse auch bei **TA** angewendet.
Der Erwartungswert einer Observablen A bezüglich der *Boltzmann*-Verteilung ist definiert als

$$\langle A \rangle = \sum_{k_i \in L} A(k_i) \cdot P_{eq}(k_i) = \frac{\sum_{k_i \in L} A(k_i) \cdot \exp(-\beta \cdot E(k_i))}{\sum_{k_i \in L} \exp(-\beta \cdot E(k_i))} = \frac{1}{Z} \cdot \sum_{k_i \in L} A(k_i) \cdot \exp(-\beta \cdot E(k_i)), \qquad \textbf{(2.11)}$$

wobei E die dem System zugrundeliegende *Hamiltonfunktion* ist, $\beta = 1/k_B T$ und

$Z = \sum_{k_i \in L} \exp(-\beta \cdot E(k_i))$ die Zustandssumme.

Der Erwartungswert der Energie ist gegeben durch

$$\langle E \rangle = \frac{1}{Z} \cdot \sum_{k_i \in L} E(k_i) \cdot \exp(-\beta \cdot E(k_i)). \qquad \textbf{(2.12)}$$

Er lässt sich formal aber auch als Ableitung der Zustandssumme ausdrücken:

$$-\frac{\partial}{\partial\beta}\ln Z = -\frac{1}{Z}\cdot\sum_{k_i\in L}\frac{\partial}{\partial\beta}\exp\left(-\beta\cdot E\left(k_i\right)\right) = \frac{1}{Z}\cdot\sum_{k_i\in L}E\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right) = \left\langle E\right\rangle . \qquad \textbf{(2.13)}$$

Der Erwartungswert der spezifischen Wärme ist definiert als:

$$C = \frac{\partial\left\langle E\right\rangle}{\partial T} . \qquad \textbf{(2.14)}$$

Mit

$$\begin{aligned}\frac{\partial\left\langle E\right\rangle}{\partial T} &= \frac{1}{Z^2}\cdot Z\cdot\frac{1}{k_B T^2}\cdot\sum_{k_i\in L}E^2\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right)\\ &\quad -\frac{1}{Z^2}\cdot\sum_{k_i\in L}E\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right)\cdot\frac{1}{k_B T^2}\cdot\sum_{k_i\in L}E\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right)\\ \rightarrow\frac{\partial\left\langle E\right\rangle}{\partial T} &= \frac{1}{k_B T^2}\cdot\left\{\frac{1}{Z}\cdot\sum_{k_i\in L}E^2\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right)-\frac{1}{Z^2}\cdot\left[\sum_{k_i\in L}E\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right)\right]^2\right\}\end{aligned}$$

ergibt sich dadurch

$$C = \frac{\partial\left\langle E\right\rangle}{\partial T} = \frac{1}{k_B T^2}\cdot\left\{\left\langle E^2\right\rangle-\left\langle E\right\rangle^2\right\} = \frac{1}{k_B T^2}\cdot Var\left(E\right). \qquad \textbf{(2.15)}$$

Oft sind neben der Energie und Wärmekapazität des Systems auch noch andere Observablen wichtig, z.B. Ordnungsparameter. Die *Hamiltonfunktion* ist bei den untersuchten Problemen gegeben durch:

$$E = E_1 + E_2 + \ldots . \qquad \textbf{(2.16)}$$

Der Einfachheit halber wird im Folgenden nur $E = E_1 - \lambda\cdot M$ betrachtet, wobei λ ein veränderbarer Kontrollparameter und M eine Observable ist. Bei einem magnetischen System ist M die Magnetisierung und λ ein Magnetfeld.
Der Erwartungswert von M kann geschrieben werden als

$$\left\langle M\right\rangle = \frac{\sum_{k_i\in L}M\left(k_i\right)\cdot\exp\left(-\beta\cdot\left(E_1\left(k_i\right)-\lambda\cdot M\left(k_i\right)\right)\right)}{\sum_{k_i\in L}\exp\left(-\beta\cdot\left(E_1\left(k_i\right)-\lambda\cdot M\left(k_i\right)\right)\right)} = \frac{1}{Z}\cdot\sum_{k_i\in L}M\left(k_i\right)\cdot\exp\left(-\beta\cdot E\left(k_i\right)\right). \qquad \textbf{(2.17)}$$

Der Erwartungswert der Suszeptibilität ist definiert als

$$\chi = \frac{\partial \langle M \rangle}{\partial \lambda}. \tag{2.18}$$

Dies lässt sich umformen

$$\frac{\partial \langle M \rangle}{\partial \lambda} = \frac{\partial}{\partial \lambda}\left\{\frac{1}{Z} \cdot \sum_{k_i \in L} M(k_i) \cdot \exp(-\beta \cdot (E_1(k_i) - \lambda \cdot M(k_i)))\right\}$$

$$\rightarrow \frac{\partial \langle M \rangle}{\partial \lambda} = \frac{1}{Z^2} \cdot \left\{Z \cdot \beta \cdot \sum_{k_i \in L} M^2(k_i) \cdot \exp(-\beta \cdot E(k_i)) - \beta \cdot \left[\sum_{k_i \in L} M(k_i) \cdot \exp(-\beta \cdot E(k_i))\right]^2\right\}$$

zu

$$\chi = \frac{\partial \langle M \rangle}{\partial \lambda} = \beta \cdot \left\{\langle M^2 \rangle - \langle M \rangle^2\right\} = \beta \cdot Var(M). \tag{2.19}$$

Bei den meisten Problemen ist es jedoch nicht möglich, über alle Konfigurationen k_i zu summieren. Es werden nur m Messungen durchgeführt.
Bei **SA** wird, wie bereits angesprochen, kein *randomwalk* durchgeführt und dann der Mittelwert gebildet, was als *Simple Sampling* bezeichnet wird. Statt dessen wird mittels eines Auswahlkriteriums (*Metropolis*-Kriterium) entschieden, welcher *Move* akzeptiert wird, wodurch sich eine Wahrscheinlichkeitsverteilung P_{eq} für die Zustände k_i einstellt. Dadurch erhält man hauptsächlich ‚wichtige Zustände', also solche, die bei der Temperatur T eine hohe Wahrscheinlichkeit besitzen. Man nennt dieses Verfahren *Importance Sampling*.
Formel **(2.12)**

$$\langle A \rangle \approx \frac{\sum_{i=1}^{m} A(k_i) \cdot \exp(-\beta \cdot E(k_i)) \Big/ P_{eq}(k_i)}{\sum_{i=1}^{m} \exp(-\beta \cdot E(k_i)) \Big/ P_{eq}(k_i)} \tag{2.20}$$

vereinfacht sich dadurch mit $P_{eq}(k_i) = \frac{1}{Z} \cdot \exp(-\beta \cdot E(k_i))$ zu

$$\langle A \rangle \approx \frac{1}{m} \cdot \sum_{i=1}^{m} A(k_i). \tag{2.21}$$

Durch die Wahl des *Metropolis*-Kriteriums wird die exponentielle Gewichtung aufgehoben.

2.4.4 Abkühlstrategie (Cooling Schedule)

Für das Abkühlschema aus **Kapitel 2.4.1 (2.7)** benötigt man leider unendlich viel Rechenzeit. Deshalb gibt man sich auch mit Lösungen zufrieden, die nahe am globalen Optimum liegen. Verwendet werden dazu rein empirische Abkühlschemata, die schneller gegen 0 konvergieren.

a) Lineares Abkühlschema

Es folgt der einfachen Regel

$$T_i = T_0 - i \cdot \Delta T \,, \qquad \textbf{(2.22)}$$

d.h. die Abkühlung erfolgt auf einer linearen Temperaturskala in äquidistanten Schritten.

b) Logarithmisches oder exponentielles Abkühlschema

Hier wird die neue Temperatur durch Multiplikation der alten mit einem Faktor α berechnet:

$$T_i = \alpha^i \cdot T_0 \,. \qquad \textbf{(2.23)}$$

Bewährt hat sich die Wahl von α im Bereich [0.9,0.99].

Die Temperaturschritte sind jetzt auf einer logarithmischen Skala äquidistant.

Welches der beiden Schemata man verwendet, hängt vom zu optimierenden System ab. Die Wärmekapazität sollte auf der entsprechenden Skala in etwa symmetrisch zur Temperatur des Maximums liegen. Bei den folgenden Produktionsplanungsproblemen hat sich das logarithmische Schema mit α=0.98 bewährt.

Entscheidend für die Güte der Lösung eines Optimierungsproblems sind noch die Wahl der Starttemperatur T_0 und der Endtemperatur T_E, bei der der Programmablauf abgebrochen wird (vgl. dazu auch (**[2.20]**).

c) Starttemperatur:

Die Starttemperatur sollte so gewählt werden, dass die Akzeptanzrate jedes einzelnen *Moves* bei mindestens 75% liegt. Dazu führt man zuerst mehrere *random*-Schritte im Lösungsraum durch und protokolliert den maximalen Energieunterschied ΔE_{max}. Für **SA** erhält man dann die Anfangstemperatur aus der Faustformel

$$T_0 = -\frac{\Delta E_{\max}}{\log(0.75)} \,. \qquad \textbf{(2.24)}$$

Die *Moves* sollten jedoch, wenn möglich, so gewählt werden, dass die zugehörigen ΔE_{max} nicht zu stark voneinander abweichen.

Eine andere Möglichkeit, die Starttemperatur zu ermitteln, besteht darin, die Wärmekapazität zu betrachten, das Maximum τ und die Breite des Peaks $\Delta\tau$ zu bestimmen. Für die Starttemperatur ergibt sich dann $T_0 = \tau+\Delta\tau$.

d) Endtemperatur:

Beim Erreichen der Endtemperatur T_E sollte die Akzeptanzrate der einzelnen *Moves* bis auf 0% zurückgehen. Triviale *Moves* ($\Delta E = 0$) werden dabei nicht beachtet.

Meist bezieht man die Einfriertemperatur auf den kleinsten Energieunterschied $\Delta E_{min}>0$ zwischen benachbarten Lösungen.

Die Wärmekapazität könnte hier ebenfalls wieder einen Hinweis geben. Verschwindet sie über mehrere Temperaturschritte, so ist die Endtemperatur erreicht.

Eine extreme Variante für ein Abkühlschema besteht darin, auf das Abkühlen ganz zu verzichten und die Simulation bei konstanter Temperatur durchzuführen (siehe z.B. **[2.28]**). Jedoch ist die Wahl dieser konstanten Temperatur entscheidend. Eine zu hohe Temperatur führt zu einem *random walk* im Suchraum. Ist die Temperatur zu niedrig, verhält sich die Suche wie ein *first improvement* **GR**.

Bei den zuvor genannten *statischen* Abkühlschemata besteht das Problem, dass kritische Temperaturbereiche, in denen große Systemumordnungen geschehen, nicht angemessen viel Rechenzeit erhalten. *Adaptive* oder *dynamische* Abkühlschema versuchen diesen Nachteil auszugleichen.
Bei *EnsembleBased-SA/TA* **EBSA**/**EBTA** (**[2.20]**, **[2.29]**) wird die Temperatur, abhängig vom Energiemittelwert einer Menge an Lösungen, gesenkt. Die Lösungssuche der N Lösungen ist, abgesehen von der gleichen Temperatur, voneinander unabhängig und kann parallel ausgeführt werden. Die Temperatur wird gesenkt, falls der Mittelwert der Energie bei einer neuen Iteration um mehr als einen bestimmten Betrag ε steigt: $\overline{H}_{i+1} \geq \overline{H}_i + \varepsilon$. E ist dabei abhängig von der Varianz, z.B. $\varepsilon \sim \sqrt{Var\left(\overline{H}_{i+1}\right)}$.
Es existieren auch Verfahren, die die Temperatur entsprechend der vorangegangenen Suche einstellen. So wird z.B. in **[2.30]** die Temperatur abhängig von der letzten Akzeptanzwahrscheinlichkeit und Annahme des *Moves* verändert. Die Temperatur kann dabei erhitzt oder abgekühlt werden.
Eine Version einer lokalen Suche, bei der die Temperatur zu bestimmten Zeitpunkten gezielt erhitzt wird, das *Temperature Bouncing*, wird in **Kapitel 2.4.5** erläutert.

Bei der Optimierung führt man am Ende des Abkühlvorgangs meist noch ein paar *first improvement* **GR** Schritte durch, um dann mit Sicherheit ein lokales Optimum zu erreichen.

2.4.5 Temperature Bouncing (TB)

Es wurde bereits in **Kapitel 2.4.2** erwähnt, dass **TA**, im Unterschied zu **SA**, keine Möglichkeit hat, bei kleiner Temperatur aus einem lokalen Optimum zu entkommen. Eine einfache Möglichkeit, ein lokales Optimum wieder zu verlassen, besteht darin, die Temperatur zu erhöhen, damit schlechtere Lösungen kurzfristig akzeptiert werden können. Anschließend wird die Temperatur wieder gesenkt. Wird diese Vorgehensweise wiederholt angewendet (*temperature bouncing* **[2.31]**), bewegt man sich von einem lokalen Optimum zum nächsten durch den Suchraum. Die Wahl des zum **TA** zusätzlichen Parameters, der *bouncing* Temperatur T_B, hat einen starken Einfluss auf die Suche. Wird T_B zu groß gewählt, so wird zuviel der vorhandenen Lösung zerstört und die neue Suche beginnt bei einer nahezu zufälligen Lösung (*random restart*). Bei einem zu kleinen T_B bleibt die Suche in dem zuletzt gefundenen lokalen Optimum gefangen. Bei der richtigen Wahl von T_B wird die erneute lokale Suche im Unterschied zum *random restart* von einer relativ guten Startlösung begonnen. Neben T_B ist als neuer Parameter noch die Anzahl der *bouncing* Schritte festzulegen, wobei jeder bei einer anderen Temperatur $T_B(i)$ begonnen werden kann. In **Abbildung 2.6** sind Beispiele für verschiedene Temperaturverläufe zu sehen.

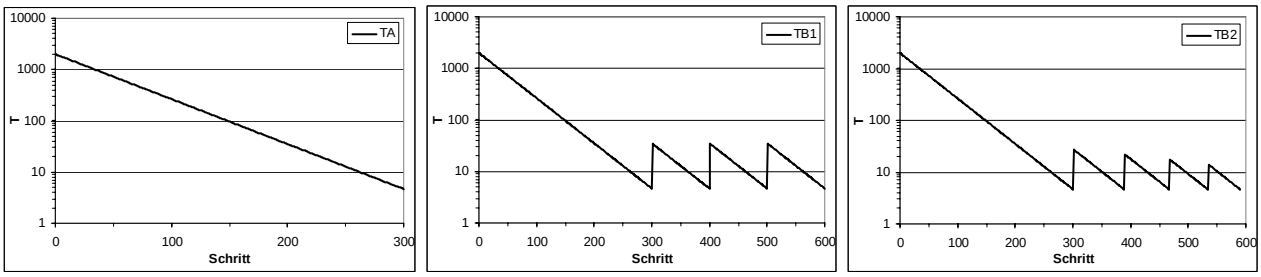


Abbildung 2.6: Temperaturverlauf des einfachen **TA** und zweier *bouncing* Varianten, bei konstantem $T_B(i)$ und bei logarithmisch gesenktem: $T_B(i) = \alpha * T_B(i-1)$ mit α=0,8.

Auf der Menge der, durch die *bouncing* Schritte erzeugten, lokalen Optima kann nun jedes der bisher beschriebenen Verfahren arbeiten. In **[2.7]** wurde **TB** bereits erfolgreich auf Scheduling Probleme angewendet, wobei bei der Suche jedes lokale Optimum akzeptiert wurde. Arbeitet man hingegen immer mit der besten bekannten Lösung als Initiallösung für den nächsten *bouncing* Schritt, so kann die Suche in deren näheren Umgebung intensiviert werden.

2.4.6 SA basierend auf der Akzeptanzrate (Acceptance SA)

Die Übergangswahrscheinlichkeit ist bei **SA-Acc** gegeben durch das *Metropolis*-Kriterium:

$$p = \begin{cases} \exp\left(-\frac{\Delta E}{T}\right) & falls\ \Delta E > 0 \\ 1 & sonst \end{cases}. \qquad \textbf{(2.25)}$$

Im Unterschied zu **SA** wird hier jedoch während der Simulation nicht mehr direkt die Temperatur T gesenkt. Die Bestimmung des Abkühlschemas der Temperatur stellt bei **SA** ein Problem dar, da dieser Parameter problemspezifisch anzupassen ist. Bei diesem Verfahren soll deshalb die Temperatur aus der Simulation selbst bestimmt werden. Dazu wird ein problemunabhängiger Parameter, die Akzeptanzrate, definiert, der während der Simulation von einem anfangs hohen Wert gesenkt wird. Die für die Übergangswahrscheinlichkeit benötigte Temperatur T bestimmt sich nun aus der Akzeptanzwahrscheinlichkeit W und der näheren Vergangenheit der Suche. Die Temperatur wird so eingestellt, dass eine Verschlechterung, deren Betrag dem Durchschnitt ΔE der letzten n Verbesserungen entspricht, mit der Wahrscheinlichkeit W akzeptiert wird. Die Temperatur kann dabei zwischenzeitlich erhöht werden:

$$T = \frac{-\Delta E}{\ln(W)}. \qquad \textbf{(2.26)}$$

Die Akzeptanzwahrscheinlichkeit wird im Folgenden immer logarithmisch von einem Anfangswert W_S bis auf einen Wert W_E gesenkt:

$$W_i = \alpha^i \cdot W_S, \qquad \textbf{(2.27)}$$

wobei $0 < W_S < 1$, $0 < W_E < 1$, $0 < \alpha < 1$. Empfohlen werden dabei die Werte $0.5 \le W_S \le 0.8$, $0.01 \le W_E \le 0.1$, α=0.98.

Kapitel 3

Modelle der Produktionsplanung

3.1 Abbildung auf ein Spinsystem

3.1.1 Einleitung

Im Folgenden sollen die einzelnen Komponenten der Problemstellung auf ein Spinsystem abgebildet werden. Die einzelnen Nebenbedingungen werden durch Strafterme im Hamiltonian abgebildet, die Optimierungsziele (z.B. die Durchlaufzeit) durch Zielfunktionen. Der gesamte Hamiltonian ergibt sich aus der gewichteten Summe der Ziel- und Straffunktionen:

$$E = \sum_i \lambda_i^Z \cdot Z_i + \sum_j \lambda_j^S \cdot S_j \,, \qquad \textbf{(3.1.1)}$$

mit den Zielfunktionen Z_i, den Straffunktionen S_j und den Gewichten $\lambda_i^Z \geq 0$ und $\lambda_j^S \geq 0$. Die gegebene Energiefunktion E ist bei der Optimierung zu minimieren.

3.1.2 Eine Maschine, *N* Produktionsvorgänge

Es soll hier eine Maschine betrachtet werden, die N Produktionsvorgänge ausführt. Die Durchlaufzeit stellt in diesem Fall das einzige Optimierungsziel dar.
Unter $P^D{}_i$ wird dabei die Produktionsdauer des Produktionsvorgangs i verstanden, während $P^S{}_{i,t}$ als Produktionsstartzeitpunkt definiert wird, und zwar derart, dass

$$P^S{}_{i,t} = \begin{cases} 1 & \textit{falls Produktionsvorgang i zum Zeitpunkt t startet} \\ 0 & \textit{sonst} \end{cases},$$

wobei für die Zeitpunkte gilt: $t \geq 0$.
Es versteht sich von selbst, dass ein Produktionsvorgang nur zu einem bestimmten Zeitpunkt starten kann:

$$\sum_t P^S{}_{i,t} = 1 \quad \forall i \,. \qquad \textbf{(3.1.2)}$$

Weiterhin ist gefordert, dass die Maschine nur einen Produktionsvorgang zu einem Zeitpunkt t bearbeiten kann:

$$\sum_{i,t} P^S{}_{i,t} \cdot \left(\sum_{j \neq i} \sum_{t \leq t' < t + P^D{}_i} P^S{}_{j,t'} \right) = \sum_{\substack{i,j \\ i \neq j}} \sum_t \sum_{t \leq t' < t + P^D{}_i} P^S{}_{i,t} \cdot P^S{}_{j,t'} = 0 \,. \qquad \textbf{(3.1.3)}$$

Diese Bedingungen lassen sich durch die zusätzlichen Straffunktionsterme erreichen. Die Zielfunktion ist gegeben durch

$$Z_1 = MAX_{i,t}\left(P^S{}_{i,t} \cdot \left(t + P^D{}_i\right)\right), \tag{3.1.4}$$

und die beiden Straffunktionen durch

$$S_1 = \sum_i \left| \sum_t P^S{}_{i,t} - 1 \right|, \tag{3.1.5}$$

$$S_2 = \sum_{\substack{i,j \\ i \neq j}} \sum_t \sum_{t \le t' < t + P^D{}_i} P^S{}_{i,t} \cdot P^S{}_{j,t'} \cdot \left(t + P^D{}_i - t'\right). \tag{3.1.6}$$

Diese Problemstellung ist damit auf ein Spinsystem abgebildet, wobei die Spins $P^S{}_{i,t}$ auf einem zweidimensionalen Gitter sitzen, die die Werte 0 oder 1 annehmen können und in einer Wechselwirkung zueinander stehen.

3.1.3 *M* Maschinen, *N* Produktionsvorgänge

Das erstellte Modell soll nun auf mehrere Maschinen erweitert werden. Einige Produktionsvorgänge können möglicherweise von mehreren alternativen Maschinen ausgeführt werden, wobei die dabei entstehende Produktionsdauer variieren kann. Als Optimierungsziel wird wieder die Durchlaufzeit angesehen.

Es sind $P^D{}_{i,m}$ die Produktionsdauern, $P^M{}_{i,m}$ die erlaubten Maschinen und $P^S{}_{i,t,m}$ die Produktionsstartzeitpunkte. Es ist

$$P^M{}_{i,m} = \begin{cases} 1 & \textit{falls Produktionsvorgang i Maschine m nutzen kann} \\ 0 & \textit{sonst} \end{cases},$$

$$P^S{}_{i,t,m} = \begin{cases} 1 & \textit{falls Produktionsvorgang i zum Zeitpunkt t auf Maschine m startet} \\ 0 & \textit{sonst} \end{cases}.$$

Jeder Produktionsvorgang hat genau einen Startzeitpunkt auf genau einer Maschine:

$$\sum_{t,m} P^S{}_{i,t,m} = 1 \quad \forall i\,. \tag{3.1.7}$$

Jede Maschine bearbeitet nur einen Produktionsvorgang zu einem Zeitpunkt t:

$$\sum_{i,t} P^S{}_{i,t,m} \cdot \left(\sum_{j \neq i} \sum_{t \le t' < t + P^D{}_{i,m}} P^S{}_{j,t',m} \right) = \sum_{\substack{i,j \\ i \neq j}} \sum_t \sum_{t \le t' < t + P^D{}_{i,m}} P^S{}_{i,t,m} \cdot P^S{}_{j,t',m} = 0 \quad \forall m\,. \tag{3.1.8}$$

Ein Produktionsvorgang ist auf einer seiner erlaubten Maschinen geplant:

$$\sum_{t,m} \left(1 - P^M{}_{i,m}\right) \cdot P^S{}_{i,t,m} = 0 \quad \forall i\,. \tag{3.1.9}$$

Die Zielfunktion ist hier gegeben durch:

$$Z_1 = MAX_{i,t,m}\left(P^S{}_{i,t,m} \cdot \left(t + P^D{}_{i,m}\right)\right), \tag{3.1.10}$$

und die Straffunktionen durch

$$S_1 = \sum_i \left| \sum_{t,m} P^S{}_{i,t,m} - 1 \right|, \tag{3.1.11}$$

$$S_2 = \sum_m \sum_{\substack{i,j \\ i \neq j}} \sum_t \sum_{t \le t' < t + P^D{}_{i,m}} P^S{}_{i,t,m} \cdot P^S{}_{j,t',m} \cdot \left(t + P^D{}_{i,m} - t'\right), \tag{3.1.12}$$

$$S_3 = \sum_{i,t,m} \left(1 - P^M{}_{i,m}\right) \cdot P^S{}_{i,t,m} \,. \tag{3.1.13}$$

Die Spins $P^S{}_{i,t,m}$ liegen jetzt auf einem dreidimensionalen Gitter mit den Werten 0 oder 1.

3.1.4 Zeitliche Bedingungen zwischen Produktionsvorgängen

Nun sollen zeitliche Abstände zwischen Produktionsvorgängen definiert werden. Diese werden immer auf die Startzeitpunkte bezogen.

Existiert ein zeitlicher Abstand zwischen zwei Produktionsvorgängen, so wird er durch die Beziehung $Z_{i,j}$ und den zeitlichen Mindestabstand zwischen den Startzeitpunkten, $Z^T{}_{i,j}$, definiert, wobei

$$Z_{i,j} = \begin{cases} 1 & \textit{falls Produktionsvorgang i Vorgänger von j ist} \\ 0 & \textit{sonst} \end{cases}.$$

Dabei muss gelten:

$$Z_{i,i} = 0, \tag{3.1.14}$$

$$Z_{i,j} \cdot \sum_{m,m'} \sum_t \sum_{t' > t - Z^T_{i,j}} P^S{}_{i,t',m'} \cdot P^S{}_{j,t,m} = 0 \quad \forall i, j \,. \tag{3.1.15}$$

Damit lässt sich ein zeitlicher Maximalabstand zwischen Produktionsvorgang *i* und *j* definieren mit

$$Z_{j,i} = 1 \wedge Z^T_{j,i} \le 0. \tag{3.1.16}$$

Die Hamiltonfunktion wird um den Straffunktion S_4 erweitert, wobei

$$S_4 = \sum_{i,j} Z_{i,j} \cdot \sum_{m,m'} \sum_t \sum_{t' > t - Z^T_{i,j}} P^S{}_{i,t',m'} \cdot P^S{}_{j,t,m} \cdot \left(t' - t + Z^T{}_{i,j}\right). \tag{3.1.17}$$

3.1.5 Modellierung JobShop, OpenShop, FlowShop

Mit den gegebenen Funktionalitäten lassen sich bereits die Problemstellungen **JobShop**, **OpenShop** und **FlowShop** definieren. Ziel der Optimierung ist die Minimierung der Durchlaufzeit. Es kann also der bisher entwickelte Hamiltonian mit der Zielfunktion und den vier Straftermen verwendet werden.

a) JobShop

Es existieren *M* Maschinen, die *N* Aufträge ausführen sollen. Jeder Auftrag besteht dabei aus *M* Produktionsvorgängen mit den Produktionsdauern d_i, von denen jeder genau einer Maschine zugeordnet wird. Die Reihenfolge, in der diese Produktionsvorgänge ausgeführt werden, ist fest vorgegeben, kann sich jedoch von Auftrag zu Auftrag unterscheiden.

Für jeden Produktionsvorgang eines Auftrags gilt:

$$P^{D}{}_{i,m} = \begin{cases} d_i & falls\ m = i \wedge 1 \le i \le M \\ 0 & sonst \end{cases}, \tag{3.1.18}$$

$$P^{M}{}_{i,m} = \begin{cases} 1 & falls\ m = i \wedge 1 \le i \le M \\ 0 & sonst \end{cases}. \tag{3.1.19}$$

Die *M-1* Zeitbeziehungen $Z_{i,j}$ sollten keinen Zyklus bilden. Die Reihenfolge σ_i ist vorgegeben mit $\sigma_i \neq \sigma_j \ \forall i \neq j$, so dass

$$Z_{i,j} = \begin{cases} 1 & falls\ i = \sigma_k \wedge j = \sigma_{k+1} \wedge 1 \le k < M \\ 0 & sonst \end{cases}, \tag{3.1.20}$$

$$Z^{T}_{i,j} = \begin{cases} d_i & falls\ i = \sigma_k \wedge j = \sigma_{k+1} \wedge 1 \le k < M \\ 0 & sonst \end{cases}. \tag{3.1.21}$$

b) OpenShop

Im Gegensatz zum **JobShop** ist hier die Reihenfolge, in der die Produktionsvorgänge eines Auftrags ausgeführt werden, nicht vorgegeben, jedoch dürfen zwei Produktionsvorgänge nicht gleichzeitig bearbeitet werden.

Zusätzlich zu den *M* gegebenen Produktionsvorgängen jedes Auftrags werden nun noch weitere ‚virtuelle' *M* Produktionsvorgänge benötigt, die von einer pro Auftrag zusätzlichen ‚virtuellen' Maschine bearbeitet werden. Diese stellt eine Art theoretisches, also real nicht existierendes Hilfskonstrukt dar, mit dessen Hilfe gewährleistet wird, dass tatsächlich keine zeitlichen Überschneidungen zwischen den einzelnen Produktionsvorgängen auftreten.

Für jeden Auftrag *a* ($1 \le a \le N$) gilt für den Produktionsvorgang *i*:

$$P^{D}{}_{i,m} = \begin{cases} d_i & falls\ m = i \wedge 1 \le i \le M \\ d_{i-M} & falls\ m = M + a \wedge M < i \le 2 * M \\ 0 & sonst \end{cases}, \tag{3.1.22}$$

$$P^{M}{}_{i,m} = \begin{cases} 1 & falls\ m = i \wedge 1 \le i \le M \\ 1 & falls\ m = M + a \wedge M < i \le 2 * M \\ 0 & sonst \end{cases}, \tag{3.1.23}$$

$$Z_{i,j} = \begin{cases} 1 & falls\ j = i + M \wedge 1 \le i \le M \\ 1 & falls\ i = j + M \wedge 1 \le j \le M \\ 0 & sonst \end{cases}, \tag{3.1.24}$$

$$Z^{T}_{i,j} = 0 \quad \forall i, j. \tag{3.1.25}$$

c) FlowShop

Anders als beim **JobShop** ist hier die Reihenfolge der Produktionsvorgänge stets dieselbe.
Für jeden Produktionsvorgang eines Auftrags gilt:

$$P^{D}{}_{i,m} = \begin{cases} d_i & falls\ m = i \wedge 1 \le i \le M \\ 0 & sonst \end{cases}, \tag{3.1.26}$$

$$P^{M}{}_{i,m} = \begin{cases} 1 & falls\ m = i \wedge 1 \le i \le M \\ 0 & sonst \end{cases}. \tag{3.1.27}$$

Die folgenden zeitlichen Bedingungen definieren die Reihenfolge:

$$Z_{i,j} = \begin{cases} 1 & falls\ j = i+1 \wedge 1 \le i < M \\ 0 & sonst \end{cases}, \tag{3.1.28}$$

$$Z^{T}_{i,j} = \begin{cases} d_i & falls\ j = i+1 \wedge 1 \le i < M \\ 0 & sonst \end{cases}. \tag{3.1.29}$$

Bei dieser Modellierung sind Zwischenlager vorhanden, d.h. die Auftragssequenzen für die einzelnen Maschinen können sich voneinander unterscheiden.

3.1.6 Zeitliche Bedingungen für die Maschinen

Bisher war die Produktion auf jeder Maschine ab dem Zeitpunkt t=0 möglich. Dies ist jedoch nicht immer der Fall. Der früheste Zeitpunkt, zu dem die Maschine belegt werden darf, ist durch $M^{Smin}{}_{m}$ gegeben. Für die Produktionsvorgänge muss gelten:

$$\sum_{t<M^{S\min}{}_{m}} P^{S}{}_{i,t,m} = 0 \quad \forall i,m\,. \tag{3.1.30}$$

Daraus ergibt sich die zusätzliche Straffunktion S_5 im Hamiltonfunktion:

$$S_5 = \sum_{i,m} \sum_{t<M^{S\min}{}_{m}} P^{S}{}_{i,t,m} \cdot \left(M^{S\min}{}_{m} - t\right). \tag{3.1.31}$$

3.1.7 Zeitliche Bedingungen für die Produktionsvorgänge

Jeder Produktionsvorgang muss einen bestimmten Zeitrahmen erfüllen, innerhalb dessen er begonnen und beendet sein sollte.

a) Frühester Start (*release date*)

Dieser stellt eine harte Nebenbedingung bei der Planung dar und wird deshalb als Strafterm in den Hamiltonian übernommen. Er ist gegeben durch $P^{Smin}{}_{i}$. Für die Produktionsvorgänge muss gelten:

$$\sum_{t<P^{S\min}{}_{i}} P^{S}{}_{i,t,m} = 0 \quad \forall i,m\,. \tag{3.1.32}$$

Im Hamiltonian wird S_6 benötigt, um diese Nebenbedingung zu berücksichtigen, wobei

$$S_6 = \sum_{i,m} \sum_{t<P^{S\min}{}_{i}} P^{S}{}_{i,t,m} \cdot \left(P^{S\min}{}_{i} - t\right). \tag{3.1.33}$$

b) Strikte Lieferfrist (*deadline*)

Hier handelt es sich ebenfalls um eine harte Nebenbedingung. Die *deadline* eines Produktionsvorgangs ist durch $P^{Emax}{}_i$ definiert. Es muss gelten:

$$\sum_{t>P^{E\max}{}_i - P^D{}_{i,m}} P^S{}_{i,t,m} = 0 \quad \forall i,m\,. \tag{3.1.34}$$

Der Hamiltonian erhält die zusätzliche Straffunktion:

$$S_7 = \sum_{i,m}\left(\sum_{t>P^{E\max}{}_i - P^D{}_{i,m}} P^S{}_{i,t,m}\cdot\left(t + P^D{}_{i,m} - P^{E\max}{}_i\right)\right). \tag{3.1.35}$$

c) Gewünschte Lieferfrist (*due date*)

Im Unterschied zur strikten Frist, die keinerlei Verzögerungen zulässt, ist die gewünschte Frist eine weiche Nebenbedingung. Ihre Einhaltung stellt neben der bereits erwähnten Durchlaufzeit eine weitere Zielfunktion im Hamiltonian dar. Eine Verletzung der Frist (Verspätung) ist unerwünscht und zu minimieren. Dabei können verschiedene Definitionen angegeben werden. Es kann bei der Verletzung der Liefertermine $P^L{}_i$

- die Summe der Verspätungen

$$Z_2 = \sum_{i,m}\left(\sum_{t>P^L{}_i - P^D{}_{i,m}} P^S{}_{i,t,m}\cdot\left(t + P^D{}_{i,m} - P^L{}_i\right)\right) \tag{3.1.36}$$

- die maximale Verspätung

$$Z_2 = MAX_i\left(\sum_m\left(\sum_{t>P^L{}_i - P^D{}_{i,m}} P^S{}_{i,t,m}\cdot\left(t + P^D{}_{i,m} - P^L{}_i\right)\right)\right) \tag{3.1.37}$$

- die Anzahl der Verspätungen

$$Z_2 = \sum_{i,m}\left(\sum_{t>P^L{}_i - P^D{}_{i,m}} P^S{}_{i,t,m}\right) \tag{3.1.38}$$

beachtet werden.

3.1.8 Umrüsten der Maschinen

Zwischen der Durchführung der einzelnen Produktionsvorgänge auf einer Maschine *m* können Rüstzeiten und Rüstkosten entstehen. $M^{RZ}{}_{m,i,j}$ sind die Rüstzeiten und $M^{RK}{}_{m,i,j}$ die Rüstkosten zwischen dem Vorgänger *i* und dem Nachfolger *j* auf Maschine *m*.

Falls der Produktionsvorgang *i* der Vorgänger von *j* auf der Maschine *m* ist, d.h.

$$\sum_t \sum_{t'>t} P^S{}_{i,t,m}\cdot P^S{}_{j,t',m} = 1 \tag{3.1.39}$$

und es keinen anderen Produktionsvorgang *k* gibt, der auf Maschine *m* zwischen *i* und *j* geplant wurde:

$$\sum_t \sum_{t'>t} P^S{}_{i,t,m}\cdot P^S{}_{k,t',m} = 0 \quad \vee \quad \sum_t \sum_{t'>t} P^S{}_{k,t,m}\cdot P^S{}_{j,t',m} = 0 \quad \forall k\,, \tag{3.1.40}$$

dann sind die Rüstzeiten einzuhalten:

$$\sum_t \left(\sum_{t'>t-P^D{}_{i,m}-M^{RZ}{}_{m,i,j}} P^S{}_{i,t',m} \cdot P^S{}_{j,t,m} \right) = 0 \,. \tag{3.1.41}$$

Dies führt zu einem weiteren Strafterm im Hamiltonian:

$$\begin{aligned} S_8 = \sum_m \sum_{\substack{i,j \\ i \neq j}} \Bigg\{ \Bigg\{ \sum_t \sum_{t'>t} P^S{}_{i,t,m} \cdot P^S{}_{j,t',m} \Bigg\} \bullet \\ \Bigg\{ \prod_k \Bigg[1 - \Bigg(\sum_t \sum_{t'>t} P^S{}_{i,t,m} \cdot P^S{}_{k,t',m} \Bigg) \cdot \Bigg(\sum_t \sum_{t'>t} P^S{}_{k,t,m} \cdot P^S{}_{j,t',m} \Bigg) \Bigg] \Bigg\} \bullet \\ \Bigg\{ \sum_t \Bigg(\sum_{t'>t-P^D{}_{i,m}-M^{RZ}{}_{m,i,j}} P^S{}_{i,t',m} \cdot P^S{}_{j,t,m} \cdot \left(t' - t + P^D{}_{i,m} + M^{RZ}{}_{m,i,j} \right) \Bigg) \Bigg\} \Bigg\} \Bigg\} \,. \end{aligned} \tag{3.1.42}$$

Es können noch zwei Zielfunktionen definiert werden:

- Summe der Rüstzeiten:

$$\begin{aligned} Z_3 = \sum_m \sum_{\substack{i,j \\ i \neq j}} \Bigg\{ M^{RZ}{}_{m,i,j} \bullet \Bigg\{ \sum_t \sum_{t'>t} P^S{}_{i,t,m} \cdot P^S{}_{j,t',m} \Bigg\} \bullet \\ \Bigg\{ \prod_k \Bigg[1 - \Bigg(\sum_t \sum_{t'>t} P^S{}_{i,t,m} \cdot P^S{}_{k,t',m} \Bigg) \cdot \Bigg(\sum_t \sum_{t'>t} P^S{}_{k,t,m} \cdot P^S{}_{j,t',m} \Bigg) \Bigg] \Bigg\} \Bigg\} \,. \end{aligned} \tag{3.1.43}$$

- Summe der Rüstkosten:

$$\begin{aligned} Z_4 = \sum_m \sum_{\substack{i,j \\ i \neq j}} \Bigg\{ M^{RK}{}_{m,i,j} \bullet \Bigg\{ \sum_t \sum_{t'>t} P^S{}_{i,t,m} \cdot P^S{}_{j,t',m} \Bigg\} \bullet \\ \Bigg\{ \prod_k \Bigg[1 - \Bigg(\sum_t \sum_{t'>t} P^S{}_{i,t,m} \cdot P^S{}_{k,t',m} \Bigg) \cdot \Bigg(\sum_t \sum_{t'>t} P^S{}_{k,t,m} \cdot P^S{}_{j,t',m} \Bigg) \Bigg] \Bigg\} \Bigg\} \,. \end{aligned} \tag{3.1.44}$$

3.1.9 Maschinenkosten

Nutzt ein Produktionsvorgang *i* eine Maschine *m*, so kann dies spezifische Kosten $P^{MK}{}_{i,m}$ erzeugen. Ziel der Optimierung kann eine Reduzierung dieser Kosten sein:

$$Z_5 = \sum_{i,m} P^{MK}{}_{i,m} \,. \tag{3.1.45}$$

3.1.10 Multiressourcen

Die einfachste Möglichkeit, diese zu modellieren, besteht darin, die bisherige Funktionalität zu nutzen und statt einer Multiressource mehrere Maschinen einzusetzen.
Existiert also eine Multiressource mit Kapazität *M*, so sind *M* Maschinen erforderlich. Die benötigte Kapazität *N* eines Auftrags der Dauer *d* wird durch *N* Produktionsvorgänge mit den folgenden Eigenschaften modelliert:

$$P^{D}{}_{i} = d \quad \forall i\,, \tag{3.1.46}$$

$$P^{M}{}_{i,m} = \begin{cases} 1 & falls\ 1 \le i \le N \wedge 1 \le m \le M \\ 0 & sonst \end{cases}, \tag{3.1.47}$$

$$Z_{i,j} = \begin{cases} 1 & falls\ j = i+1 \wedge 1 \le i < N \\ 1 & falls\ i = j+1 \wedge 1 \le j < N \\ 0 & sonst \end{cases}, \tag{3.1.48}$$

$$Z^{T}{}_{i,j} = 0 \quad \forall i, j\,. \tag{3.1.49}$$

Für diese Funktionalität werden auf diese Weise keine zusätzlichen Strafterme im Hamiltonian benötigt.

3.1.11 Lager

Sie dienen dazu, evtl. hergestellte Produkte/Materialien aufzubewahren und verfügen über eine gewisse Kapazität. Die Produktionsvorgänge können dem Lager Produkte hinzufügen und/oder entnehmen. Jedes Lager kann immer nur identische Produkte aufnehmen.
$P^{LV}{}_{i,l}$ ist die vom Produktionsvorgang *i* dem Lager *l* entnommene Menge des Produkts, $P^{LP}{}_{i,l}$ die hinzugefügte Menge und $L^{K}{}_{l,t}$ die maximale Kapazität des Lagers *l* zur Zeit *t*.
Eine Überlastung des Lagers soll vermieden werden:

$$\sum_{i,m}\left\{\left(\sum_{t'\le t-P^{D}{}_{i,m}} P^{S}{}_{i,t',m}\right)\cdot P^{LP}{}_{i,l} - \left(\sum_{t'\le t} P^{S}{}_{i,t',m}\right)\cdot P^{LV}{}_{i,l}\right\} \le L^{K}{}_{l,t} \quad \forall l,t\,. \tag{3.1.50}$$

Die entnommene Menge darf höchstens der gelagerten Menge des Produkts entsprechen:

$$\sum_{i,m}\left\{\left(\sum_{t'\le t-P^{D}{}_{i,m}} P^{S}{}_{i,t',m}\right)\cdot P^{LP}{}_{i,l} - \left(\sum_{t'\le t} P^{S}{}_{i,t',m}\right)\cdot P^{LV}{}_{i,l}\right\} \ge 0 \quad \forall l,t\,. \tag{3.1.51}$$

Die zugehörigen Strafterme sind:

$$S_9 = \sum_{l,t} MAX\left\{0, \sum_{i,m}\left\{\left(\sum_{t'\le t-P^{D}{}_{i,m}} P^{S}{}_{i,t',m}\right)\cdot P^{LP}{}_{i,l} - \left(\sum_{t'\le t} P^{S}{}_{i,t',m}\right)\cdot P^{LV}{}_{i,l}\right\} - L^{K}{}_{l.t}\right\}, \tag{3.1.52}$$

$$S_{10} = \sum_{l,t} MAX\left\{0, -\sum_{i,m}\left\{\left(\sum_{t'\le t-P^{D}{}_{i,m}} P^{S}{}_{i,t',m}\right)\cdot P^{LP}{}_{i,l} - \left(\sum_{t'\le t} P^{S}{}_{i,t',m}\right)\cdot P^{LV}{}_{i,l}\right\}\right\}. \tag{3.1.53}$$

3.1.12 Pausen

Im Folgenden unterbrechen die Pausen die Produktionsvorgänge. Diese werden nach der Pause fortgesetzt. $M^{PS}{}_{m,p}$ sind die Startzeitpunkte und $M^{PD}{}_{m,p}$ die Dauern der Pausen auf der Maschine m.
Kein Produktionsvorgang darf während einer Pause starten:

$$\sum_{M^{PS}{}_{m,p} \le t < M^{PS}{}_{m,p} + M^{PD}{}_{m,p}} P^{S}{}_{i,t,m} = 0 \quad \forall i, p, m . \tag{3.1.54}$$

Die Pause verlängert die Produktionsdauer, d.h. falls eine Überlappung eines Produktionsvorgangs mit einer Pause vorliegt:

$$\sum_{M^{PS}{}_{m,p} - P^{D}{}_{i,m} < t < M^{PS}{}_{m,p}} P^{S}{}_{i,t,m} = 1 \quad \exists i, p, m , \tag{3.1.55}$$

kann ein folgender Produktionsvorgang erst später starten:

$$\sum_{t} \left(\sum_{t \le t' < t + P^{D}{}_{i,m} + M^{PD}{}_{m,p}} P^{S}{}_{i,t,m} \cdot P^{S}{}_{j,t',m} \right) = 0 \quad \forall j . \tag{3.1.56}$$

Daraus folgen die Strafterme:

$$S_{11} = \sum_{i,p,m} \left(\sum_{M^{PS}{}_{m,p} \le t < M^{PS}{}_{m,p} + M^{PD}{}_{m,p}} P^{S}{}_{i,t,m} \cdot \left(M^{PS}{}_{p,m} + M^{PD}{}_{p,m} - t \right) \right), \tag{3.1.57}$$

$$S_{12} = \sum_{\substack{i,j,p,m \\ i \ne j}} \left(\sum_{M^{PS}{}_{m,p} - P^{D}{}_{i,m} < t < M^{PS}{}_{m,p}} P^{S}{}_{i,t,m} \right) \bullet \left(\sum_{t \le t' < t + P^{D}{}_{i,m} + M^{PD}{}_{m,p}} P^{S}{}_{i,t,m} \cdot P^{S}{}_{j,t',m} \cdot \left(t + P^{D}{}_{i,m} + M^{PD}{}_{p,m} - t' \right) \right). \tag{3.1.58}$$

3.1.13 Konfiguration

Die Konfiguration besteht bei dieser Modellierung aus dem dreidimensionalen Spinfeld $P^{S}{}_{i,t,m}$ mit den Werten 0 oder 1.

3.1.14 Nachbarschaft

Im Folgenden werden drei Nachbarschaften unterschieden:

- Im einfachsten Fall besteht die Nachbarschaft aus einem SingleSpinFlip ($P^{S}{}_{i,t,m}{}^{neu} = 1 - P^{S}{}_{i,t,m}$), der allerdings sofort die Nebenbedingung **(3.1.7)** verletzt, falls diese zuvor erfüllt war.
- Nachbarschaft (2) besteht aus Veränderungen, die die Nebenbedingungen **(3.1.7)** und **(3.1.9)** beachten. Diese Veränderungen befolgen folgende Regeln:
  - Verändere den Startzeitpunkt und evtl. die Maschine eines Produktionsvorganges. Vertausche die Spins $P^{S}{}_{i,t,m}$ und $P^{S}{}_{i,t',m'}$, so dass $P^{S}{}_{i,t,m}{}^{neu} = P^{S}{}_{i,t',m'}$ und $P^{S}{}_{i,t',m'}{}^{neu} = P^{S}{}_{i,t,m}$, die folgende Bedingung erfüllen: $P^{S}{}_{i,t,m} + P^{S}{}_{i,t',m'} = 1 \wedge P^{M}{}_{i,m} = 1 \wedge P^{M}{}_{i,m'} = 1$.
  - Vertausche zwei Produktionsvorgänge, d.h. vertausche die Spins $P^{S}{}_{i,t,m}$ und $P^{S}{}_{i',t',m'}$, so dass $P^{S}{}_{i,t,m}{}^{neu} = 0 \wedge P^{S}{}_{i,t',m'}{}^{neu} = 1$ und $P^{S}{}_{i',t',m'}{}^{neu} = 0 \wedge P^{S}{}_{i',t,m}{}^{neu} = 1$, wobei Folgendes gilt: $P^{S}{}_{i,t,m} + P^{S}{}_{i',t',m'} = 2 \wedge P^{M}{}_{i,m'} = 1 \wedge P^{M}{}_{i',m} = 1$.

- Nachbarschaft (3) besteht aus den gleichen Veränderungen wie Nachbarschaft (2), jedoch wird nun der Spin, der verändert wird, bei manchen Aufrufen gezielt ausgesucht. Es wird dann der Spin des Produktionsvorgangs i gewählt, der am spätesten startet:
  $\sum_{t,m} P^S{}_{i,t,m} \cdot t \geq \sum_{t',m'} P^S{}_{i',t',m'} \cdot t' \quad \forall i'$.
  Zusätzlich darf der so ausgewählte Produktionsvorgang nur zu früheren Zeiten starten, d.h. $P^S{}_{i,t,m}{}^{neu} = 0 \wedge P^S{}_{i,t',m'} = 1$, wobei $t'<t$.

3.1.15 Nachteile des Spinmodells

Im Folgenden sind nun einige negative Eigenschaften des Modells, basierend auf einer Abbildung auf ein Spinsystem, aufgeführt:

- **Variable Zeit t**
  Zu beachten ist nun allerdings, dass die bisherige Variable der Zeit t zwar nach unten ($t \geq 0$), nicht jedoch nach oben beschränkt ist. Der benötigte Speicherplatz und die Berechnung des Hamiltonians sind jedoch von t abhängig, weshalb ein möglichst kleiner Bereich günstig ist. Es kann die Zeitspanne zur Planung auf folgenden Maximalwert eingeschränkt werden:
  $$t \leq \sum_i P^D{}_i + \sum_{i,j} Z_{i,j} \cdot Z^T{}_{i,j} + \sum_{i,j,m} M^{RZ}{}_{i,j,m} + \sum_{p,m} M^{PD}{}_{p,m} \qquad \textbf{(3.1.59)}$$
  Dieser pessimistische Wert ist immer noch sehr groß. Es lässt sich leicht eine engere Grenze finden. Während der Simulation kann dieser Maximalwert auf x% der Durchlaufzeit der besten bisher erzeugten gültigen Lösung gesetzt werden, wobei x>100 ist. Bei zu kleinem x reduziert man die Nachbarschaft evtl. zu stark, so dass der Suchraum in Inseln zerfällt und man die optimale Lösung nicht mehr erreichen kann.

- **Konfiguration**
  Die Konfiguration ist ebenfalls von der Zeit t abhängig. Es zeigt sich, dass der Konfigurationsraum in dieser Modellierung sehr viel größer als benötigt ist. Eine Modellierung, bei der bei jeder Konfiguration darauf geachtet wird, möglichst früh die Produktionsvorgänge einzuplanen, könnte die Anzahl der möglichen Konfigurationen deutlich senken.

- **Nebenbedingungen**
  Es existieren sehr viele Nebenbedingungen. Diese sind immer als Straffunktionsterm mit einem Lagrange Multiplikator an den Hamiltonian angehängt. Es ist bei der Optimierung mit dem SingleSpinFlip bereits bei kleinen Aufgabenstellungen (z.B. **MT6** siehe **Kapitel 4.2**) schwierig, Konfigurationen zu finden, die alle Nebenbedingungen einhalten. Es ist deshalb zu empfehlen, nur mit Nachbarschaft 2 oder 3 zu arbeiten, die einige der Nebenbedingungen jederzeit erfüllen, falls die Initiallösung diese erfüllt.
  Die Ergebnisse können noch weiter verbessert werden, falls die veränderten Spins zeitlich ‚nahe' liegen: $|t - t'| \leq T$. Dabei kann die Produktionszeit der beteiligten Produktionsvorgange verwendet werden: $T = \left(D_{im} + D_{im'} + D_{i'm} + D_{i'm'}\right)/4$.

Neben den hier angeführten Nachteilen verfügt die beschriebene Modellierung auch über einige Vorteile, die in der einfachen Definition der Konfiguration und der Nachbarschaft liegen. Die negativen Aspekte überwiegen jedoch bei weitem. Im Folgenden wird deshalb eine auf den ersten Blick kompliziertere Modellierungsbasis entwickelt, die die benötigten Funktionalitäten sehr viel direkter abbildet und so die meisten Nebenbedingungen in jeder Konfiguration erfüllt. Der Konfigurationsraum ist dann erheblich kleiner. Er wird noch weiter reduziert um alle Konfigurationen, die keine *semiaktiven* Lösungen darstellen.

3.2 Modell mit direkter Repräsentation

3.2.1 Einleitung

Bei dem vorliegenden Produktionsplanungsproblem sucht man nach einem gültigen Produktionsplan, also einer Zuordnung der Aufträge zu den Ressourcen zu bestimmten Zeitpunkten, wobei die problemspezifischen Restriktionen eingehalten werden. Mit dem Begriff Ressourcen werden die Primärressourcen, Sekundärressourcen und die Lager/Silos zusammengefasst. Dabei soll eine gegebene Zielfunktion minimiert werden.
Im Folgenden soll nun die formale Beschreibung der einzelnen Ressourcen, Auftragsbestandteile und Restriktionen folgen. Im Anschluss daran wird die Definition der Nachbarschaft mit Hilfe der Konfiguration vorgestellt. Aus der Konfiguration lässt sich der Produktionsplan erstellen. Zum Schluss werden die möglichen Optimierungsziele definiert.

3.2.2 Ressourcen

Drei grundsätzlich verschiedene Ressourcentypen sind zu unterschieden. Die Primärressource stellt eine Maschine dar. Die Sekundärressource ist im Unterschied zur Primärressource eine Multiressource, die evtl. mehrere verschiedene Arbeitsschritte gleichzeitig ausführen kann. Als Lager wird nur das Silo modelliert.

3.2.2.1 Primärressource

Eine Primärressource definiert eine Maschine. Die Produktion kann ab einem bestimmten Zeitpunkt starten, wird jedoch möglicherweise zu bestimmten Zeiten durch Pausen unterbrochen. Zwischen verschiedenen Produktionsvorgängen kann eine Umrüstung der Maschine nötig sein. Soll ein Auftrag auf einer Primärressource bearbeitet werden, so muss er als Produktionsvorgang die folgenden Bedingungen einhalten.

a) Produktionsvorgang

Sei P_i ein Produktionsvorgang der Primärressource, $P^S{}_i$ sein Startzeitpunkt, $P^E{}_i$ das Ende und $P^D{}_i$ die Produktionsdauer mit

$$P^S{}_i \le P^E{}_i \wedge P^D{}_i = P^E{}_i - P^S{}_i . \qquad \textbf{(3.2.1)}$$

Eine Primärressource kann immer nur einen Produktionsvorgang zu einem Zeitpunkt t bearbeiten:

$$P^S{}_i \ge P^E{}_j \vee P^E{}_i \le P^S{}_j \quad \left(\forall i, j \middle| i \ne j\right). \qquad \textbf{(3.2.2)}$$

Falls $P^S{}_i \ge P^E{}_j$, so bezeichnet man P_i als den Nachfolger von P_j und P_j als den Vorgänger von P_i. P_i ist der direkte Nachfolger von P_j (bzw. P_j der direkte Vorgänger von P_i), falls es keinen Produktionsvorgang P_k gibt mit $P^S{}_k \ge P^E{}_j \wedge P^E{}_k \le P^S{}_i$.

b) Funktionalität ‚frühester Start'

Die Primärressource steht erst zu einem bestimmten Zeitpunkt M^{Smin} zur Verfügung. Für die Produktionsvorgänge gilt:

$$P^S{}_i \geq M^{S\min} \quad \forall i . \tag{3.2.3}$$

c) Funktionalität ‚Umrüstung'

Eine Umrüstung definiert einen ‚speziellen' Produktionsvorgang $P^R{}_i$, dessen Produktionsdauer, im Unterschied zu einem ‚einfachen' Produktionsvorgang, abhängig von seinem direkten Vorgänger ist (reihenfolgeabhängig). Die Produktionsdauer ist in der, für die Primärressource gültigen, Rüstzeitmatrix $M^{RZ}{}_{i,j}$ definiert.

Eine Umrüstung kann zusätzlich noch reihenfolgeabhängige Rüstkosten verursachen. Diese sind in einer Rüstkostenmatrix $M^{RK}{}_{i,j}$ definiert.

Die Produktionsdauer $P^D{}_i$ der Umrüstung entspricht somit der Rüstzeit $P^D{}_i = M^{RZ}{}_{j,i}$, falls P_j der direkte Vorgänger von $P^R{}_i$ ist.

d) Funktionalität ‚Pausen'

In einer Pause $M^P{}_p$ ruht die Produktion. $M^{PS}{}_p$ ist ihr Start, $M^{PE}{}_p$ das Ende und $M^{PD}{}_p$ die Pausendauer mit $M^{PS}{}_p \leq M^{PE}{}_p \wedge M^{PD}{}_p = M^{PE}{}_p - M^{PS}{}_p$. Unterschieden wird bei Pausen und Produktionsvorgängen nun noch nach deren Unterbrechbarkeit:

$$P^{PU}{}_i = \begin{cases} 1 & \textit{falls } P_i \textit{ unterbrochen werden kann} \\ 0 & \textit{sonst} \end{cases},$$

$$M^{PU}{}_p = \begin{cases} 1 & \textit{falls } P_i \textit{ durch Pause } M^P{}_p \textit{ unterbrochen werden können} \\ 0 & \textit{sonst} \end{cases}.$$

Bei der Planung des Produktionsvorgangs müssen bzgl. der Pausen folgende Regeln beachtet werden:

(1) Ein Produktionsvorgang darf nicht in einer Pause starten:

$$P^S{}_i < M^{PS}{}_p \vee P^S{}_i \geq M^{PE}{}_p \quad \forall i, p . \tag{3.2.4}$$

(2) Ein Produktionsvorgang darf nicht in einer Pause enden:

$$P^E{}_i \leq M^{PS}{}_p \vee P^E{}_i > M^{PE}{}_p \quad \forall i, p . \tag{3.2.5}$$

(3) Eine Überschneidung eines Produktionsvorgangs mit einer Pause ist nicht erlaubt, falls der Produktionsvorgang oder die Pause nicht unterbrechbar sind:

$$P^E{}_i \leq M^{PS}{}_p \vee P^S{}_i \geq M^{PE}{}_p \quad \left(\forall i, p \middle| M^{PU}{}_p = 0 \vee P^{PU}{}_i = 0\right). \tag{3.2.6}$$

(4) Die Dauer des Produktionsvorgangs wird bei einer Überschneidung um die Pausendauer erhöht. In diesem Zusammenhang wird auch von der Netto- $P^{D\_N}{}_i$ und der Brutto-Dauer $P^{D\_B}{}_i$ gesprochen. Die Netto-Dauer bezeichnet dabei die ursprüngliche Produktionsdauer ohne Pausen und die Brutto-Dauer die tatsächliche Produktionsdauer:

$$P^{D\_B}{}_i = P^{D\_N}{}_i + \sum_p M^{PD}{}_p \quad \left(\forall i, p \middle| P^S{}_i < M^{PS}{}_p \wedge P^S{}_i + P^{D\_B}{}_i > M^{PS}{}_p\right). \tag{3.2.7}$$

Folgernder iterativer Ansatz erzeugt für einen Produktionsvorgang P_i mit gegebener Netto-Dauer $P^{D\_N}{}_i$ einen pausengültigen möglichen Start $P^{S}{}_i$ und die Brutto-Dauer $P^{D\_B}{}_i$ von einem gegeben Zeitpunkt $P^{Smin}{}_i$ aus. Die N Pausen sind dabei so geordnet, dass
$M^{PE}{}_p < M^{PS}{}_{p+1} \quad (\forall p | 1 \le p < N)$.

$P^{S}{}_i = P^{S\min}{}_i$
$P^{D\_B}{}_i = P^{D\_N}{}_i$
p=1
while($p \le N \wedge M^{PE}{}_p \le P^{S}{}_i$) ++p
while($p \le N \wedge P^{S}{}_i + P^{D\_B}{}_i > M^{PS}{}_p$) {
  if($P^{S}{}_i \ge M^{PS}{}_p \vee M^{PU}{}_p = 0 \vee P^{PU}{}_i = 0$) {
    $P^{S}{}_i = M^{PE}{}_p$
    $P^{D\_B}{}_i = P^{D\_N}{}_i$
  } else {
    $P^{D\_B}{}_i = P^{D\_B}{}_i + M^{PD}{}_p$
  }
  ++p
  }

Der so gefundene Startzeitpunkt $P^{S}{}_i$ für den Produktionsvorgang erfüllt die Bedingungen (1) bis (4).

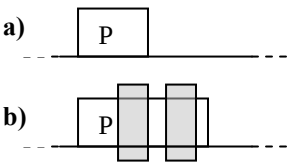


Abbildung 3.2.1: Beispiel für die Bestimmung der Brutto-Dauer **b)** eines Produktionsvorgangs P mit gegebener Netto-Dauer **a)**.

e) Planungsfunktionalität:

Beim seriellen Beplanen der Primärressource werden die Produktionsvorgänge *semiaktiv* auf der Ressource eingelastet. Sei P_i noch nicht geplant und P_j die M bereits geplanten Produktionsvorgänge, dann gilt:

$$P^{S}{}_i \ge \begin{cases} P^{E}{}_j & falls\ M > 0 \\ M^{S\min}{}_m & sonst \end{cases}. \qquad \textbf{(3.2.8)}$$

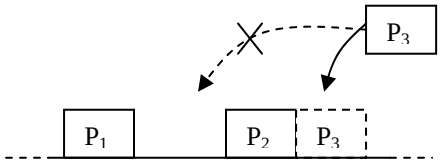


Abbildung 3.2.2: Die Produktionsvorgänge werden auf der Primärressource semiaktiv geplant. Produktionsvorgang P_3 wird erst nach P_2 geplant, obwohl zu einem früheren Zeitpunkt (zwischen P_1 und P_2) noch ausreichend Kapazität zur Verfügung steht.

3.2.2.2 Sekundärressource

Eine Sekundärressource (Multiressource) kann im Unterschied zur Primärressource von mehr als nur einem Produktionsvorgang gleichzeitig benutzt werden. Es müssen bei der Nutzung der Sekundärressource durch einen Auftrag als Produktionsvorgang die folgenden Bedingungen eingehalten werden.

a) Kapazität

Die maximale Kapazität M^K der Ressource ist zu beachten. Diese kann eine zeitliche Abhängigkeit besitzen: $M^K(t)$. Die benötigte Kapazität eines Produktionsvorgang P_i kann ebenfalls eine zeitliche Abhängigkeit besitzen $P^{MK}{}_i(t)$, wobei $P^{MK}{}_i(t) = 0 \quad \left(\forall t \middle| t < 0 \vee t \geq P^D{}_i\right)$. Im Normalfall ist diese jedoch zeitlich konstant: $P^{MK}{}_i(t) = P^{MK}{}_i \quad \left(\forall t \middle| 0 \leq t < P^D{}_i\right)$.

Es muss gelten:

$$\sum_i P^{MK}{}_i\left(t - P^S{}_i\right) \leq M^K(t) \quad \forall t\,. \tag{3.2.9}$$

b) Planungsfunktionalität:

Im Unterschied zur Primärressource werden die Produktionsvorgänge bei der Sekundärressource *aktiv* geplant, d.h. es wird von einem frühest möglichen Startzeitpunkt $P^{Smin}{}_i$ des Produktionsvorgangs aus der nächste mögliche Zeitpunkt $P^S{}_i \geq P^{S\min}{}_i$ mit ausreichend Kapazität gesucht und dort der Produktionsvorgang geplant. Sei $M^{Kfrei}(t)$ die freie Kapazität der Sekundärressource, dann muss der zu planende Produktionsvorgang P_i zum Zeitpunkt $P^S{}_i$ kapazitätsgerecht eingeplant werden können, d.h.

$$P^{MK}{}_i\left(t - P^S{}_i\right) \leq M^{Kfrei}(t) \quad \left(\forall t \middle| P^S{}_i \leq t < P^E{}_i\right). \tag{3.2.10}$$

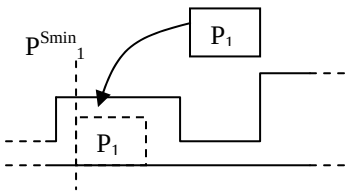


Abbildung 3.2.3: Die Produktionsvorgänge werden auf der Sekundärressource aktiv geplant. Produktionsvorgang P_1 wird zu seinem frühesten Zeitpunkt geplant.

3.2.2.3 Silo

Das Silo definiert ein mögliches Lager. Es sind folgende Bedingungen zu beachten:

a) Kapazität

Die maximale Kapazität S^K des Silos ist zu beachten. Diese kann eine zeitliche Abhängigkeit besitzen: $S^K(t)$. Stellt ein Produktionsvorgang P_i Material her, so ist die benötigte Kapazität des Materials $P^{SK}{}_i$ positiv, wird Material verbraucht, ist sie negativ. Ist $P^{ST}{}_i$ der Zeitpunkt, zu dem das Material dem Silo entnommen (bzw. im Silo eingelagert) wird, so gilt:

$$0 \le \sum_{\left(i \,\middle|\, P^{ST}{}_i \le t\right)} P^{SK}{}_i \le S^K(t) \quad \forall t\,. \tag{3.2.11}$$

Bei der Lagerung oder der Entnahme von Materialien unterscheidet man zwischen *batch* und *continuous*. Bei *batch* Betrieb erfolgt die Lagerung bzw. Entnahme des Materials zu einem Zeitpunkt, während bei *continuous*-Betrieb die Lagerung bzw. Entnahme einer bestimmten Zeit bedarf. Die zeitlichen Materialverläufe eines Produktionsvorgangs P_1 sind in **Abbildung 3.2.4** zu sehen.

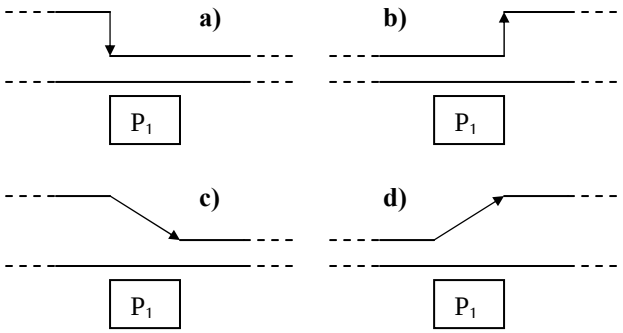


Abbildung 3.2.4: Zu sehen ist die Entnahme (**a**,**c**) und Lagerung (**b**,**d**) eines Materials durch einen Produktionsvorgang P_1 im *batch*- (**a**,**b**) bzw. im *continuous*- (**c**,**d**) Betrieb.

b) Weiche Grenze für die maximale Kapazität:

Neben der bereits definierten harten Grenze S^K für die maximale Kapazität, die während der Optimierung in jeder Konfiguration einzuhalten ist, soll nun eine weiche Nebenbedingung S^{Ksoft} für die maximale Kapazität definiert werden, deren Nichteinhaltung möglich ist, jedoch als zusätzliche Kosten in die Zielfunktion eingeht. Durch geeignete Wahl der Priorität dieser Nebenbedingung sollte es dann möglich sein, am Ende der Optimierung eine Lösung zu erhalten, bei der diese Nebenbedingung erfüllt ist.

Ist $S^K{}_{belegt}(t)$ der zeitliche Verlauf der bereits genutzten Lagerkapazität, so ist die Straffunktion definiert durch:

$$S_{Silo} = \sum_t MAX\left(0, S^K{}_{belegt}(t) - S^{Ksoft}\right). \tag{3.2.12}$$

c) Planungsfunktionalität:

Grundsätzlich wird das Silo *aktiv* beplant. Wird das zu lagernde (bzw. gelagerte) Material vom Produktionsvorgang P_i produziert (bzw. verbraucht), so sind folgende Bedingungen einzuhalten, falls $S^K_{belegt}(t)$ den aktuellen zeitlichen Füllstand des Silos darstellt:

- *batch*-Entnahme:

$$-P^{SK}{}_i \le S^K{}_{belegt}(t) \quad \left(\forall t \middle| t \ge P^S{}_i\right) \tag{3.2.13}$$

- *batch*-Lagerung:

$$P^{SK}{}_i + S^K{}_{belegt}(t) \le S^K \quad \left(\forall t \middle| t \ge P^E{}_i\right) \tag{3.2.14}$$

- *continuous*-Entnahme:

$$\left(-P^{SK}{}_i * \frac{t - P^S{}_i}{P^E{}_i - P^S{}_i} \le S^K{}_{belegt}(t) \quad \left(\forall t \middle| P^S{}_i \le t < P^E{}_i\right)\right) \wedge \left(-P^{SK}{}_i \le S^K{}_{belegt}(t) \quad \left(\forall t \middle| t \ge P^E{}_i\right)\right) \tag{3.2.15}$$

- *continuous*-Lagerung:

$$\left(P^{SK}{}_i * \frac{t - P^S{}_i}{P^E{}_i - P^S{}_i} + S^K{}_{belegt}(t) \le S^K \quad \left(\forall t \middle| P^S{}_i \le t < P^E{}_i\right)\right) \wedge \left(P^{SK}{}_i + S^K{}_{belegt}(t) \le S^K \quad \left(\forall t \middle| t \ge P^E{}_i\right)\right) \tag{3.2.16}$$

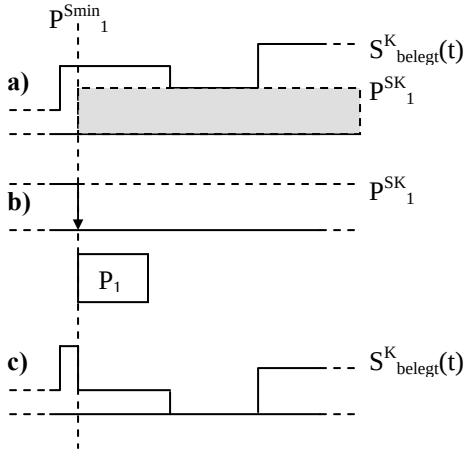


Abbildung 3.2.5: Hier ist ein Beispiel einer Befüllung eines Silos im *batch* zu sehen. **a)** zeigt den Füllstand des Silos vor der Planung des Produktionsvorgangs, **c)** nach der Planung. Ab dem Startzeitpunkt des Produktionsvorgangs wird eine bestimmte Menge des Materials verbraucht (**a)**, **b)**).

3.2.3 Produktionsauftrag

Ein Produktionsauftrag besteht aus einer Menge an Aktivitäten, die ausgeführt werden müssen. Jede Aktivität kann möglicherweise auf verschiedene Arten ausgeführt werden. Diese Möglichkeiten werden durch alternative Multimodi dargestellt. Jeder Multimodus besteht aus einer Menge Modi, die nacheinander ausgeführt werden müssen. Jeder Modus stellt dabei einen Arbeitsschritt/Produktionsvorgang mit einer bestimmten Ressourcenbelegung dar. Die Aktivitäten des Auftrags können noch durch zeitliche Nebenbedingungen, die sog. Aktivitätslinks, verbunden sein.

3.2.3.1 Modus

Ein Modus stellt einen Produktionsvorgang dar. Er kann eine Primärressource und mehrere Sekundärressourcen (Multiressourcen) benötigen. Zusätzlich kann er noch andere Primärressourcen sperren. Falls der Modus Material produziert oder verbraucht, sind ihm Silos zugeordnet.
Sei Mo^D_i die Dauer des Modus Mo_i, Mo^S_i der Startzeitpunkt und Mo^E_i das Ende, wobei $Mo^S_i \leq Mo^E_i$ und $Mo^D_i = Mo^E_i - Mo^S_i$. Die folgenden Bedingungen müssen eingehalten werden.

a) Primärressource
Ein Modus kann höchstens eine Primärressource belegen.

b) Sekundärressource
Ein Modus kann mehrere Sekundärressourcen mit verschiedenen Kapazitätsprofilen belegen.

c) Silo
Es können mehrere Silos vom Modus verwendet werden, wobei der Modus dabei Material erzeugen und/oder verbrauchen kann. Der Modus kann Material bei einem Silo jedoch nur verbrauchen oder erzeugen.

d) Sperren anderer Primärressourcen
Primärressourcen werden semiaktiv beplant. Durch einen einzelnen Eintrag (kurzer Produktionsvorgang) wird die Ressource somit für die Planung zu früheren Zeitpunkten gesperrt. Der Sperrzeitpunkt Mo^{TS}_i kann relativ zum Start des Modus vorgegeben werden. $Mo^{TS}_i = Mo^S + t$ mit beliebigem t. Diese Funktionalität wird verwendet, um **FlowShop-Probleme** zu modellieren, bei denen keine Silos für die Lagerung existieren und die Primärressourcen zur Lagerung des Materials benutzt werden (vgl. **Kapitel 7**).

e) Funktionalität ‚Umrüstung'
Falls der Modus eine Umrüstung darstellt, so muss er eine Primärressource belegen. Die Dauer des Modus hängt dann von der Reihenfolge der geplanten Modi auf dieser Ressource und der zur Primärressource gehörigen Rüstmatrix ab. Zusätzlich kann eine Umrüstung noch Kosten verursachen. Diese hängen ebenfalls von der geplanten Reihenfolge auf der Primärressource ab.
Die Rüstzeit Mo^{RZ}_i des Modus entspricht seiner Dauer: $Mo^D_i = Mo^{RZ}_i$. Mo^{RK}_i entsprechen den Rüstkosten des Modus.
Es existieren in solch einer Rüstmatrix meist viele gleiche Zeilen und Spalten, d.h. es gibt viele Modi, die bzgl. der Umrüstung gleich sind. Um die Rüstmatrix möglichst klein zu halten, wird deshalb jedem Modus ein Rüstschlüssel zugeordnet, mit dessen Hilfe die Rüstdauer bzw. die Rüstkosten aus der Rüstmatrix ermittelt werden. Die Rüstzeiten der Primärressource m, $M^{RZ}_{m,i,j}$, sind gegeben durch den Rüstschlüssel i des Vorgängermodus und den Rüstschlüssel j des Rüstmodus, die m belegen. Somit ist dann nicht mehr direkt die Reihenfolge der Modi, sondern vielmehr die Reihenfolge der Rüstschlüssel entscheidend.

Im Folgenden wird ein Modus durch $\langle (B,S,b)|D|PR|(SR,P^{MK}(t))|GR(t)|RS|(P,S,b)\rangle$ definiert, wobei

- **D** der Dauer des Modus entspricht oder ‚*', falls es sich um einen Rüstmodus handelt
- **PR** seiner Primärressource
- **(SR, $P^{MK}(t)$)** den Sekundärressourcen **SR** mit den evtl. zeitabhängigen Kapazitätsprofilen $P^{MK}(t)$
- **GR(t)** den gesperrten Primärressourcen zum Zeitpunkt t
- **RS** dem Rüstschlüssel des Modus
- **(B,S,b)** dem benötigten Material **B** aus dem Silo **S**, wobei die Entnahme *batch* (**b**=1) oder *kontinuierlich* (**b**=0) sein kann. Wird **B** keine Mengenangabe hinzugefügt, so wird 1 Einheit des entsprechenden Materials verbraucht.
- **(P,S,b)** dem produzierten Material **P** in das Silo **S**, wobei die Lagerung *batch* (**b**=1) oder *kontinuierlich* (**b**=0) sein kann. Wird **P** keine Mengenangabe hinzugefügt, so wird 1 Einheit des entsprechenden Materials hergestellt.

Bei den Ressourcen **(SR, $P^{MK}(t)$)**, **GR(t)**, **(B,S)** und **(P,S)** sind Mehrfachnennungen möglich; diese werden durch ‚;' voneinander getrennt. Falls jedoch keine Ressource benötigt wird, so ist dies durch ‚*' gekennzeichnet.

Zwei Modi sind identisch ($Mo_i = Mo_j, i \neq j$), falls sie identische Eigenschaften aufweisen, nämlich:

- gleiche Dauer,
- gleiche Primärressource,
- gleiche Liste aus Sekundärressourcen,
- gleiche Liste aus gesperrten Primärressourcen mit gleichen Sperrzeitpunkten,
- gleicher Rüstschlüssel,
- gleiche Liste aus benötigten Materialien sowie
- gleiche Liste aus produzierten Materialien.

$$Mo_i = \langle (B,S,b)|D|PR|(SR,P^{MK}(t))|GR(t)|RS|(P,S,b)\rangle_i = \langle (B,S,b)|D|PR|(SR,P^{MK}(t))|GR(t)|RS|(P,S,b)\rangle_j = Mo_j$$

3.2.3.2 Multimode

Ein Multimodus Mu_k sammelt mehrere Produktionsvorgänge, die nacheinander ausgeführt werden. Mo_i $(1 \leq i \leq N)$ sind die N zugehörigen Modi. Es können zusätzlich noch Zeitbeziehungen zwischen den Modi festgelegt werden (Moduslink).

a) Funktionalität ‚Moduslink'

Zwei Modi Mo_i und Mo_j ($i<j$) eines Multimodus können über einen Moduslink miteinander verbunden werden, wobei Mo_i als der Vorgänger von Mo_j bezeichnet wird. Ein Link legt eine zeitliche Bedingung (Mindestabstand T_{min}, Maximalabstand T_{max}) zwischen dem Vorgänger und Nachfolger fest, wobei man zwischen vier Typen: Start-Start (SS), Start-Ende (SE), Ende-Start

(ES), Ende-Ende (EE) unterscheidet. Es gibt höchstens *2*N*(N-1)* solcher Moduslinks innerhalb eines Multimodus.

Für die einzelnen Typen gilt:

$$\begin{aligned}&\left(Typ \equiv SS \wedge Mo^S{}_j \geq Mo^S{}_i + T_{\min}\right) \vee\\ &\left(Typ \equiv SE \wedge Mo^E{}_j \geq Mo^S{}_i + T_{\min}\right) \vee\\ &\left(Typ \equiv ES \wedge Mo^S{}_j \geq Mo^E{}_i + T_{\min}\right) \vee\\ &\left(Typ \equiv EE \wedge Mo^E{}_j \geq Mo^E{}_i + T_{\min}\right)\end{aligned} \qquad \textbf{(3.2.17)}$$

Falls ein Maximalabstand definiert ist, muss zusätzlich noch gelten:

$$\begin{aligned}&\left(Typ \equiv SS \wedge Mo^S{}_j \leq Mo^S{}_i + T_{\max}\right) \vee\\ &\left(Typ \equiv SE \wedge Mo^E{}_j \leq Mo^S{}_i + T_{\max}\right) \vee\\ &\left(Typ \equiv ES \wedge Mo^S{}_j \leq Mo^E{}_i + T_{\max}\right) \vee\\ &\left(Typ \equiv EE \wedge Mo^E{}_j \leq Mo^E{}_i + T_{\max}\right)\end{aligned} \qquad \textbf{(3.2.18)}$$

Im Folgenden wird durch $Mo_i \xrightarrow{XX,T_{\min}} Mo_j$ ein Moduslink zwischen den Modi Mo_i und Mo_j ($i<j$) eines Multimodus vom Typ XX (XX=SS, SE, ES oder EE) mit zeitlichem Mindestabstand T_{min} festgelegt. Mo_i ist dabei der Vorgänger von Mo_j. $Mo_i \xrightarrow{XX,T_{\min},T_{\max}} Mo_j$ definiert zusätzlich einen Maximalabstand T_{max}.

Zwei Moduslinks $Link_1 = Mo_1 \xrightarrow{XX_1,T_1,T_2} Mo_2$ und $Link_2 = Mo_3 \xrightarrow{XX_2,T_3,T_4} Mo_4$ sind identisch $Link_1=Link_2$, falls sie identische Modi verbinden ($Mo_1 = Mo_3 \wedge Mo_2 = Mo_4$), der Typ und die zeitlichen Bedingungen übereinstimmen ($XX_1 = XX_2 \wedge T_1 = T_3 \wedge T_2 = T_4$).

b) Funktionalität ‚Umrüstung'

Falls ein Rüstmodus Mo_i existiert, muss zusätzlich noch mindestens ein Modus Mo_j mit $i<j$ existieren, der die gleiche Primärressource belegt.

c) Funktionalität ‚Kosten'

Dem Multimodus können Kosten zugeordnet werden. Diese können möglicherweise durch die Benutzung der Ressourcen entstehen.

Im Folgenden wird ein Multimodus durch

K	$\langle (B,S,b) \mid D \mid PR \mid (SR, P^{MK}(t)) \mid GR(t) \mid RS \mid (P,S,b) \rangle$	$Mo_i \xrightarrow{XX,T_{\min},T_{\max}} Mo_j$
	$\langle (B,S,b) \mid D \mid PR \mid (SR, P^{MK}(t)) \mid GR(t) \mid RS \mid (P,S,b) \rangle$	...
	...	...

dargestellt, wobei dieser aus drei Teilen besteht:

- den Kosten **K**,
- einer Liste aus Modi und
- einer Liste aus Moduslinks.

Zwei Multimodi Mu_1 und Mu_2, die aus jeweils N Modi Mo_{1i} und Mo_{2i} und M Moduslinks $Link_{1j}$ und $Link_{2j}$ bestehen, sind identisch ($Mu_1=Mu_{12}$), falls ihre Eigenschaften im Hinblick auf:

- gleiche Kosten,
- identische Modi ($Mo_{1i} = Mo_{2i}\ \left(\forall i \middle| 1 \le i \le N\right)$) und
- identische Moduslinks ($Link_{1i} = Link_{2i}\ \left(\forall i \middle| 1 \le i \le M\right)$)

übereinstimmen.

3.2.3.3 Aktivität

Die Arbeitsschritte/Produktionsvorgänge, die in einem Multimodus zusammengefasst sind, können möglicherweise auch auf alternativen Ressourcen durchgeführt werden. Durch die Aktivität wird dies modelliert. Eine Aktivität kann aus mehreren zueinander alternativen Multimodi bestehen. Mu_k ($1 \le k \le M$) sind die M zueinander alternativen Multimodi der Aktivität. Nur einer der Multimodi wird selektiert, d.h. dessen Modi werden ausgeführt.
Der Aktivität wurde neben einem frühesten Startzeitpunkt, der immer einzuhalten ist, noch ein Lieferzeitpunkt zugeordnet, der eine weiche Grenze darstellt. Die Rüstzeiten, bzw. Rüstkosten sind durch die Modi des selektierten Multimodus gegeben. Zwischen den Modi verschiedener Aktivitäten lässt sich noch eine Zeitbeziehung definieren (Aktivitätslink).
Der Startzeitpunkt A^S_i der Aktivität A_i ergibt sich aus dem Minimum der Startzeitpunkte der Modi Mo_j ($1 \le j \le N$) des selektierten Multimodus. $A^S_i = MIN_j(Mo^S_j)$ ($1 \le j \le N$). Für den Endzeitpunkt A^E_i ergibt sich: $A^E_i = MAX_j(Mo^E_j)$.
Der Index k gibt in der Schreibweise A_i^k den selektierten Multimodus der Aktivität an. Die Funktion $HasResource\left(R, A_i^k\right) = (TRUE, FALSE)$ gibt an, ob der Multimodus k der Aktivität A_i mindestens einen Modus enthält, der die Ressource R belegt. Bei der Ressource R kann es sich dabei um eine Primärressource oder eine Sekundärressource handeln.

a) Funktionalität ‚frühester Startzeitpunkt'

Der früheste Startzeitpunkt A^{Smin}_i der Aktivität ist eine immer einzuhaltende Bedingung. Für alle Modi Mo_j des selektierten Multimodus gilt somit:

$$Mo^S_j \ge A^{S\min}_i \qquad \textbf{(3.2.19)}$$

b) Funktionalität ‚Lieferzeitpunkt'

Im Gegensatz zur Bedingung für den frühesten Startzeitpunkt ist der Lieferzeitpunkt A^L_i, bis zu dem die Aktivität beendet sein muss, also $A^E_i \le A^L_i$, keine harte Bedingung. Das Einhalten dieser Bedingung kann eines der Optimierungsziele sein, wobei jede Aktivität zusätzlich zu dem Lieferzeitpunkt noch ein Gewicht G_i mitgegeben wird. Für die gewichtete Verspätung ergibt sich:

$$V_i = \begin{cases} 0 & falls\ A^E_i \le A^L_i \\ G_i \cdot \left(A^L_i - A^E_i\right) & sonst \end{cases}. \qquad \textbf{(3.2.20)}$$

c) Funktionalität ‚Umrüstung'

Die Rüstzeit $A^{RZ}{}_i$, bzw. die Rüstkosten $A^{RK}{}_i$ der Aktivität entspricht der Rüstzeit, bzw. Rüstkosten der Rüstmodi des selektierten Multimodus, falls dieser eine Umrüstung definiert:

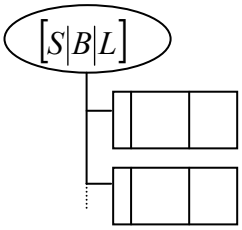
$$A^{RZ}{}_i = \sum_j Mo^{RZ}{}_j \,, \qquad \textbf{(3.2.21)}$$

wobei $Mo^{RZ}{}_j = 0$ falls Mo_j kein Rüstmodus ist, bzw.

$$A^{RK}{}_i = \sum_j Mo^{RK}{}_j \,, \qquad \textbf{(3.2.22)}$$

wobei $Mo^{RK}{}_j = 0$ falls Mo_j kein Rüstmodus ist.

Eine Aktivität wird im Folgenden auch durch



definiert mit

- dem frühesten Start **S**,
- der Bezeichnung der Aktivität **B**,
- dem Lieferzeitpunkt **L** und
- den zugehörigen Multimodi.

Existiert kein frühester Start oder Liefertermin, so ist **S** oder **L** mit ‚*' gekennzeichnet. Ist beim Liefertermin kein Gewicht (Priorität) angegeben, so ist es 1.

Zwei Aktivitäten A_1 und A_2, die jeweils N Multimodi besitzen Mu_{1k} und Mu_{2k}, sind identisch (A_1=A_2), falls ihre Eigenschaften in folgenden Bereichen übereinstimmen:

- gleiche früheste Startzeitpunkte,
- gleiche Lieferzeitpunkte und
- identische Multimodi ($Mu_{1k} = Mu_{2k}\ \left(\forall k \middle| 1 \le k \le N\right)$).

3.2.3.4 Aktivitätslink

Zwischen den Modi zweier Aktivitäten wird ein Moduslink definiert. Dazu werden die Nummern der Modi in den Multimodi angegeben. Es sei Mu_k der selektierte Multimodus der ersten Aktivität und Mu_l der selektierte Multimodus der zweiten Aktivität, wobei N_k und N_l die Anzahl der zugehörigen Modi darstellen. Zwischen dem i-ten Modus Mo_i $(1 \le i \le N_k)$ der ersten Aktivität und dem j-ten Modus Mo_j $(1 \le j \le N_l)$ der zweiten Aktivität sei nun ein Moduslink mit zeitlichem Mindestabstand T_{min} definiert. Im Unterschied zum Moduslink innerhalb eines Multimodus existiert kein Aktivitätslink mit zeitlicher Maximalabstand T_{max}. Für die vier verschiedenen Typen des Moduslinks gilt:

$$\begin{aligned}
&\left(Typ \equiv SS \wedge Mo^S{}_j \geq Mo^S{}_i + T_{\min}\right) \quad \vee \\
&\left(Typ \equiv SE \wedge Mo^E{}_j \geq Mo^S{}_i + T_{\min}\right) \quad \vee \\
&\left(Typ \equiv ES \wedge Mo^S{}_j \geq Mo^E{}_i + T_{\min}\right) \quad \vee \\
&\left(Typ \equiv EE \wedge Mo^E{}_j \geq Mo^E{}_i + T_{\min}\right)
\end{aligned} \qquad \textbf{(3.2.23)}$$

Im Folgenden wird durch $A_k(i) \xrightarrow{XX,T_{\min}} A_l(j)$ ein Moduslink zwischen dem *i*-ten Modus der Aktivitäten A_k und dem *j*-ten Modus von A_l vom Typ XX (XX=SS, SE, ES oder EE) mit zeitlichem Mindestabstand T_{min} festgelegt. A_k (bzw. Mo_i) ist dabei der Vorgänger von A_l (bzw. Mo_j); A_l (bzw. Mo_j) ist der Nachfolger von A_k (bzw. Mo_i).
Falls ein Link $A_l(e) \xrightarrow{XX,T_{\min}} A_m(f)$ existiert, also A_m Nachfolger von A_l ist, so ist A_m auch Nachfolger von A_k.
Beim Erzeugen der Links zwischen den Aktivitäten darf kein Zyklus entstehen, d.h. es darf keine Aktivität A_m geben, die gleichzeitig Vorgänger und Nachfolger von A_k ist (vgl. **Abbildung 3.2.6**).

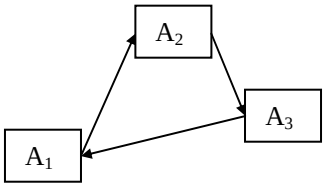


Abbildung 3.2.6: Zyklus bei den 3 Aktivitäten A_1, A_2 und A_3, die durch die Aktivitätslinks $A_1(1) \xrightarrow{ES,0} A_2(1)$, $A_2(1) \xrightarrow{ES,0} A_3(1)$ und $A_3(1) \xrightarrow{ES,0} A_1(1)$ miteinander verbunden sind. A_2 ist z.B. Vorgänger und Nachfolger bzgl. A_3.

3.2.4 Konfiguration

Unter einer Konfiguration $\langle K \rangle$ soll im Folgenden eine Sequenz aus Aktivitäten A_i verstanden werden. Für diese Konfiguration werden nun einige Definitionen gegeben:

- $|\langle K \rangle|$ gibt dabei die Anzahl der Aktivitäten in der Konfiguration $\langle K \rangle$ an.
- Jede Konfiguration, die A_i enthält, lässt sich darstellen als $\langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle$ mit zwei möglicherweise leeren Konfigurationen $\langle K_1 \rangle$ und $\langle K_2 \rangle$ ($|\langle K_1 \rangle| \geq 0, |\langle K_2 \rangle| \geq 0$). Man spricht von einer leeren Konfiguration $\langle K \rangle$, falls $|\langle K \rangle| = 0$.
- Unter $\overline{\langle K \rangle}$ versteht man eine gespiegelte Konfiguration: $\overline{\langle K_1 \rangle \bullet A_i^k} = A_i^k \bullet \overline{\langle K_1 \rangle}$.
- $\langle K_2 \rangle \subseteq \langle K \rangle$ stellt eine Teilkonfiguration von $\langle K \rangle$ dar. Es gilt: $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$. Insbesondere ist $A_i^k \subseteq \langle K \rangle$, falls $\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_3 \rangle$
- Mit $GetKonfRes(R, \langle K \rangle) = \langle K_R \rangle$ erhält man eine Konfiguration, für die gilt: $HasResource(R, A_i) = TRUE \quad \forall A_i \subseteq \langle K_R \rangle$.

- Zwei Aktivitäten A_i^k und A_j^l sind im strengen Sinne identisch ($A_i^k \equiv A_j^l$), falls sie identisch ($A_i = A_j$) sind, die selektierten Multimodi identisch (*k=l*) sind und sie keine Vorgänger oder Nachfolger besitzen.
- Zwei Konfigurationen $\langle K_1 \rangle = A_1^k \bullet \langle K_3 \rangle$ und $\langle K_2 \rangle = A_2^l \bullet \langle K_4 \rangle$ sind identisch ($\langle K_1 \rangle = \langle K_2 \rangle$), falls sie aus identischen Aktivitäten bestehen, also falls $|\langle K_1 \rangle| = |\langle K_2 \rangle| \wedge A_1^k \equiv A_2^l \wedge \langle K_3 \rangle = \langle K_4 \rangle$

3.2.5 Nachbarschaft

Die Nachbarschaft wird mit Hilfe von Operatoren (*Moves*) definiert, die die Konfiguration verändern. Die Darstellung der Lösung durch eine Konfiguration und die darauf definierte Nachbarschaft wurde bereits bei anderen Problemstellungen eingesetzt, so z.B. bei Sequenzbildungsproblemen in der Automobilindustrie (diskrete Industrie) **[2.2]**.
Im Folgenden sind $\langle K \rangle$ und $\langle K' \rangle$ benachbarte Konfigurationen.

3.2.5.1 Nachbarschaftsoperatoren

- **Push1:**
  $\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet A_i^k \bullet \langle K_3 \rangle$
  oder
  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet A_i^k \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$,
  wobei
  $|\langle K_2 \rangle| > 0$

- **Swap**
  $\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet A_j^l \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet A_j^l \bullet \langle K_2 \rangle \bullet A_i^k \bullet \langle K_3 \rangle$,
  wobei
  $|\langle K_2 \rangle| \geq 0$

- **Lin2Opt**
  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \overline{\langle K_2 \rangle} \bullet \langle K_3 \rangle$,
  wobei
  $|\langle K_2 \rangle| > 1$

- **Change**

  $\langle K \rangle = \langle K_1 \rangle \bullet A_i^{\,k} \bullet \langle K_2 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet A_i^{\,l} \bullet \langle K_2 \rangle$,

  wobei

  $k \neq l$

- **PushChange**

  $\langle K \rangle = \langle K_1 \rangle \bullet A_i^{\,k} \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet A_i^{\,l} \bullet \langle K_3 \rangle$

  oder

  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet A_i^{\,k} \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet A_i^{\,l} \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$,

  wobei

  $\left|\langle K_2 \rangle\right| > 0 \wedge k \neq l$

- **Swap&Change**

  $\langle K \rangle = \langle K_1 \rangle \bullet A_i^{\,k} \bullet \langle K_2 \rangle \bullet A_j^{\,l} \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet A_j^{\,n} \bullet \langle K_2 \rangle \bullet A_i^{\,m} \bullet \langle K_3 \rangle$,

  wobei zwei Ressourcen R_1 und R_2 existieren mit

  $HasResource(R_1, A_i^{\,k}) = TRUE \wedge HasResource(R_2, A_i^{\,m}) = TRUE \wedge k \cong m \wedge$

  $HasResource(R_2, A_j^{\,l}) = TRUE \wedge HasResource(R_1, A_j^{\,n}) = TRUE \wedge l \cong n$

- **SwapSameResource**

  $\langle K \rangle = \langle K_1 \rangle \bullet A_i^{\,k} \bullet \langle K_2 \rangle \bullet A_j^{\,l} \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet A_j^{\,l} \bullet \langle K_2 \rangle \bullet A_i^{\,k} \bullet \langle K_3 \rangle$,

  wobei eine Ressource R existiert mit

  $HasResource(R, A_i^{\,k}) = TRUE \wedge HasResource(R, A_j^{\,l}) = TRUE$

- **Lin2OptSameResource**

  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \langle K_2' \rangle \bullet \langle K_3 \rangle$,

  wobei

  $\left|GetKonfRes(R, \langle K_2 \rangle)\right| > 1$

  Hat $\langle K_2 \rangle$ N Positionen ($\left|\langle K_2 \rangle\right| = N$) und besteht aus den Aktivitäten A_i ($1 \leq i \leq N$): $\langle K_2 \rangle = A_1 \bullet A_2 \bullet ... \bullet A_{N-1} \bullet A_N$, dann wird $\langle K_2' \rangle = A_1' \bullet A_2' \bullet ... \bullet A_{N-1}' \bullet A_N'$ folgendermaßen aus $\langle K_2 \rangle$ erzeugt:

```
i=1, j=N
while( i<j ) {
    while( HasResource(R, A_i) = FALSE ) {
        A_i' = A_i
        ++i
    }
    while( HasResource(R, A_j) = FALSE ) {
        A_j' = A_j
        --j
    }
    A_i' = A_j , A_j' = A_i
    ++i, --j
}
```

3.2.5.2 Probleme

Bei der Definition der Nachbarschaft durch Operatoren, die die Konfiguration verändern, kann es zu folgenden Problemen kommen:

(1) Zwei benachbarte Konfigurationen können zum gleichen Produktionsplan führen. Dies ist z.B. der Fall, falls zwei benachbarte Aktivitäten, die verschiedene Ressourcen belegen, getauscht werden.

$\langle K_1 \rangle \bullet A_i^k \bullet A_j^l \bullet \langle K_2 \rangle$ und $\langle K_1 \rangle \bullet A_j^l \bullet A_i^k \bullet \langle K_2 \rangle$

führen zum gleichen Produktionsplan, falls keine Ressource R existiert mit $HasResource(R, A_i^k) = TRUE \wedge HasResource(R, A_j^l) = TRUE$.

Es ist deshalb darauf zu achten, dass bei jeder Veränderung Aktivitäten betroffen sind, die die gleiche Ressource belegen (siehe **Kapitel 3.2.5.3**).

(2) Ein Operator verändert den Produktionsplan nicht, falls nur identische Aktivitäten ausgetauscht werden, z.B. ein Swap der beiden Aktivitäten A_i^k und A_j^l, wobei $A_i^k \equiv A_j^l$. Dies sollte bei der Ausführung der Operatoren beachtet werden (siehe **Kapitel 3.2.5.3**).

(3) Nicht jede mögliche Lösung (Produktionsplan) lässt sich durch diese Planungsmethode (vgl. **Kapitel 3.2.6**) aus den möglichen Konfigurationen erzeugen. Bei manchen Problemstellungen kann dies dazu führen, dass die optimale Lösung nicht im Suchraum enthalten ist (siehe z.B. **Kapitel 7.3.2**).

(4) Die Operatoren beachten einige der Nebenbedingungen nicht, so z.B., dass bestimmte Aktivitäten Vorgänger anderer Aktivitäten sind. Es wird deshalb beim Aufbau des Produktionsplans noch ein Reparaturmechanismus (**Kapitel 3.2.6.3**) definiert.

3.2.5.3 Verbesserung der Operatoren

Damit die in **Kapitel 3.2.5.2** beschriebenen Probleme (1) und (2) nicht mehr auftreten und durch die Anwendung eines Operators ein neuer Produktionsplan erzeugt wird, muss sich für mindestens eine Ressource eine neue Ausführungsreihenfolge ergeben. Jeder Ressource werden zwei Konfigurationen $\langle K_R \rangle$ und $\langle K_R' \rangle$ zugeordnet, die aus der Ausgangskonfiguration $\langle K \rangle$ und der durch den Operator erzeugten Konfiguration $\langle K' \rangle$ erzeugt werden: $\langle K_R \rangle = GetKonfRes(R, \langle K \rangle)$, $\langle K_R' \rangle = GetKonfRes(R, \langle K' \rangle)$. Mindestens eine Ressource sollte durch die Anwendung des Operators eine neue Konfiguration erhalten: $(\exists R | \langle K_R \rangle \neq \langle K_R' \rangle)$. Wird diese Bedingungen verletzt, so wird der Operator nicht ausgeführt und ein neuer ausgewählt.

3.2.5.4 Auswahl der Operatoren

Es muss nun noch festgelegt werden, in welcher Weise die zuvor definierten Operatoren angewendet werden. Folgende Punkte sind dabei zu beachten:

- Es erscheint nicht bei jeder Aufgabenstellung sinnvoll, alle Operatoren zu verwenden. So ist z.B. bei einer Problemstellung ohne alternative Ressourcen ein Operator überflüssig, der den ausgewählten Multimodus wechselt (**Change**).
- Es stellt sich die Frage, ob eine festgelegte Reihenfolge bei der Anwendung der Operatoren (Nachbarschaften) zu verwenden ist, wie z.B. bei **ILS**, bei dem bestimmte Simulationsschritte in zwei verschiedenen Nachbarschaften stattfinden **[2.16]**, oder ob eine gewisse Auswahlwahrscheinlichkeit vorliegt.
- Bei einer zufälligen Wahl der Operatoren ist die zugehörige Auswahlwahrscheinlichkeit zu bestimmen.

Im Folgenden wird die Nachbarschaft immer durch eine zufällige Wahl der Operatoren erstellt. Dazu erhält jeder Operator eine Auswahlwahrscheinlichkeit. Sollen Operatoren bei einer bestimmten Problemstellung nicht benutzt werden, so ist ihre Auswahlwahrscheinlichkeit 0. Oft wird eine solche Auswahl auch mit einem *roulette wheel* verglichen, bei dem den einzelnen Operatoren Kreissegmente zugeordnet sind, deren Kreisbogen proportional zu ihren Auswahlwahrscheinlichkeiten ist. Es wird dann bei jeder Auswahl derjenige Operator angewendet, auf dessen Kreissegment die Roulettekugel liegen bleibt. Bei genetischen Algorithmen wird das *roulette-wheel*-Verfahren z.B. benutzt, um aus einer Menge von Individuen diejenigen auszuwählen, auf die dann anschließend ein Crossover-Operator angewendet wird (vgl. z.B. **[2.15]**). Bei Minimierungsproblemen, also bei solchen Problemen, bei denen ein Individuum mit möglichst geringer Fitness gesucht wird, erhält das Individuum einen umso größeren Kreisabschnitt (Auswahlwahrscheinlichkeit), je kleiner seine Fitness ist. Die Gesamtheit der Kreisabschnitte ergibt immer einen vollständigen Kreis.

Zu unterscheiden ist hier noch, ob die Auswahlwahrscheinlichkeiten während der Optimierung festgelegt sind, oder ob sie adaptiv angepasst werden.

a) fixierte Auswahlwahrscheinlichkeit
Die Auswahlwahrscheinlichkeit W_i jedes Operators O_i wird am Anfang der Optimierung festgelegt. Dabei ist zu beachten, dass $\sum W_i = 1$.

b) adaptive Auswahlwahrscheinlichkeit
Die Auswahl der Operatoren bei dem adaptiven Verfahren ähnelt der Individuenauswahl beim **GA**. Zuerst erhält jeder Operator eine bestimmte Auswahlwahrscheinlichkeit. Zu bestimmten festgelegten Zeitpunkten (z.B. alle N Sweeps) wird nun der Kreisbogen jedes Operators neu bewertet. Dabei wird die Größe festgelegt durch die relative Energieverbesserung, die eine Durchführung des Operators im letzten Zeitintervall (also den letzten N Sweeps) erbrachte. Eine Nebenbedingung sollte allerdings beachtet werden: Jeder Operator sollte immer eine gewisse Mindestwahrscheinlichkeit besitzen, falls er anfangs eine Auswahlwahrscheinlichkeit >0 hatte.
Wurde der Operator O_i D_i-mal verwendet und ist E_i die gesamte resultierende Energieverbesserung $E_i = \sum_j MAX(0, -\Delta E_j)$ mit der Energieänderung ΔE_j beim Anwenden des Operators, dann ergibt sich für die Auswahlwahrscheinlichkeit $W_i = K * (E_i / D_i)$ mit dem Normierungsfaktor $K = \left(\sum_i D_i\right) / \left(\sum_i E_i\right)$. Jede Auswahlwahrscheinlichkeit kleiner 1% wird nun auf Kosten, der größten Auswahlwahrscheinlichkeit auf 1% erhöht, wobei $\sum W_i = 1$ zu beachten ist.
Neben der relativen Energieverbesserung könnte noch die bisherige Akzeptanzwahrscheinlichkeit die Größe des Kreisabschnitts definieren.

In den folgenden Kapiteln wird meist mit einer fixierten Auswahlwahrscheinlichkeiten gearbeitet.

3.2.6 Aufbau des Produktionsplans (*Schedule*)

Die Konfiguration definiert die Reihenfolge der Aktivitäten auf jeder Ressource. Die Wahl der Multimodi der Aktivitäten ist dort bereits festgelegt. Der Produktionsplan legt nun zusätzlich noch die Startzeitpunkte der Modi fest.

3.2.6.1 Darstellung des Produktionsplans

Die Darstellung des Produktionsplans erfolgt in Form sog. Gantt Diagramme. Dabei werden die geplanten Modi in einem Diagramm eingetragen, dessen Achsen die Ressourcen und die Zeit darstellen. Im Folgenden wird dafür ein Graphikprogramm der folgenden Erscheinungsform (**Abbildung 3.2.7**) gewählt.

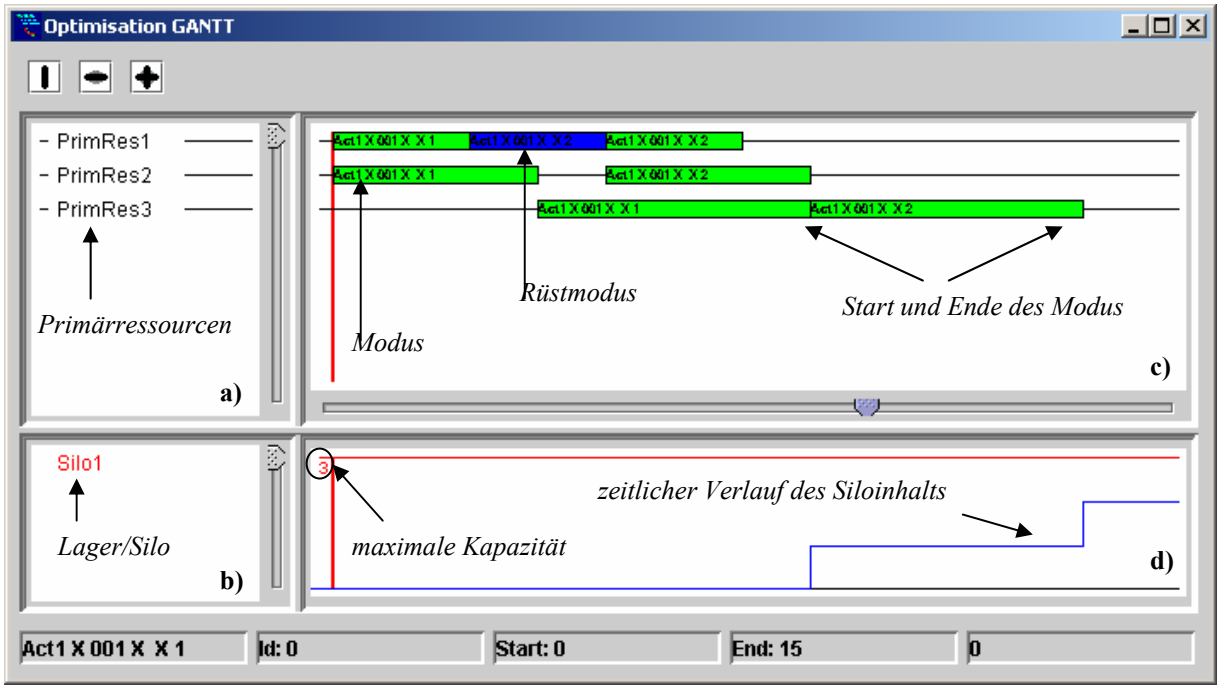


Abbildung 3.2.7: Oberfläche eines Programms zur Anzeige eines Gantt Diagramms einer Lösung. Vier Bereiche sind vorhanden: Liste der Primärressourcen (**a)**), Liste der Lager/Silo und Multiressourcen (**b)**), zeitliche Lage der einzelnen Modi (**c)**) und der zeitliche Verlauf des Lagerfüllstandes (**d)**).

3.2.6.2 Konfiguration → *Schedule*

Ein Plan wird nun aus der Konfiguration erzeugt, indem jede Aktivität A_i^k in der Reihenfolge der Sequenz der Konfiguration nacheinander geplant wird. Bei dem Einplanen der Aktivität wird nun folgendermaßen vorgegangen (5 Einplanungsschritte):

1. Zuerst muss geprüft werden, ob A_i^k geplant werden kann. Folgende Gründe können gegen eine sofortige Planung sprechen:
   - Es existiert ein Vorgänger A_j^l $(i \neq j)$, der noch nicht geplant wurde:
     $\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet A_j^l \bullet \langle K_3 \rangle$
   - A_i^k benötigt Material aus einem Silo, das nicht mehr genügend davon zur Verfügung stellt. Bedingung **(3.2.11)** ist verletzt. Es muss also zuerst die Aktivität A_j^l $(i \neq j)$ geplant werden, die das benötigte Material produziert und damit das Silo füllt.
   - A_i^k produziert Material und füllt damit ein Silo, das nicht mehr über genügend freie Kapazität verfügt. Bedingung **(3.2.11)** ist verletzt. Es muss also zuerst die Aktivität A_j^l $(i \neq j)$ geplant werden, die Material aus dem Silo verbraucht.

2. Falls A_i^k nicht geplant werden kann, wird die Konfiguration repariert (**Kapitel 3.2.6.3**) und dann bei **Schritt 1** weitergearbeitet.

3. Sei nun A_i^k planbar. Es sind Mo_m ($1 \le m \le N$) die N Modi des selektierten Multimodus der Aktivität. Diese Mo_m werden nun nacheinander frühestmöglich geplant, wobei der Startzeitpunkt $Mo^S{}_m$ jedes Modus folgende Bedingungen erfüllen muss:
   - Stellt der Modus einen Rüstmodus dar, so wird seine Dauer festgelegt durch die der Primärressource zugeordneten Rüstmatrix und dem letzten bereits geplanten Modus auf dieser Primärressource (siehe **Kapitel 3.2.2.1.c**).
   - Jeder Modus Mo_m muss den frühest möglichen Startzeitpunkt der zugehörigen Aktivität A_k einhalten **(3.2.19).**
   - Falls ein Modus durch einen Moduslink mit einem anderen Modus oder durch einen Aktivitätslink mit einem Modus einer anderen Aktivität verbunden ist und der Nachfolger bezüglich des Links ist, so muss der durch den Link festgelegte zeitliche Mindestabstand eingehalten werden (siehe Moduslink **(3.2.17)** bzw. Aktivitätslink **(3.2.23)**).
   - Falls der Modus eine Primärressource benötigt, so wird er als Produktionsvorgang auf der Primärressource immer *semiaktiv* geplant. (siehe **Kapitel 3.2.2.1.e (3.2.8)**)
   - Falls der Modus eine Primärressource belegt und diese Pausen besitzt, so muss $Mo^S{}_m$ die Bedingungen der Funktionalität Pausen der Primärressourcen einhalten. (siehe **Kapitel 3.2.2.1.d (3.2.4)**, **(3.2.5)**, **(3.2.6)**, **(3.2.7))**
   - Ein Modus kann mehrere Sekundärressourcen belegen. Auf jeder dieser Sekundärressourcen wird der Modus als Produktionsvorgang *aktiv* geplant. Zum Zeitpunkt $Mo^S{}_m$ muss der Modus auf jeder Sekundärressource kapazitätsgerecht geplant werden können. (siehe **Kapitel 3.2.2.2.b (3.2.10)**)

     Falls der Modus eine Primärressource belegt und diese Pausen besitzt, so wird nicht mit der Netto-Dauer und dem zugehörigen von der Sekundärressource benötigten Kapazitätsprofil gearbeitet, sondern mit einem der Brutto-Dauer angeglichenen Kapazitätsprofil. Dieses Brutto-Kapazitätsprofil muss zum Zeitpunkt $Mo^S{}_m$ kapazitätsgerecht geplant werden können. Aus dem Netto-Kapazitätsprofil $P^{MK\_N}{}_n(t)$, der Brutto-Dauer $Mo^{D\_B}{}_m$ des Modus und den Pausen $M^P{}_i$, die die Brutto-Dauer des Modus beeinflussen (vgl. **Kapitel 3.2.2.1.d (3.2.7)**),

     $Mo^S{}_m < M^{PS}{}_i \wedge Mo^S{}_m + Mo^{D\_B}{}_m > M^{PE}{}_i$, $M^{PS}{}_i < M^{PS}{}_{i+1}$,

     lässt sich das Brutto-Kapazitätsprofil $P^{MK\_B}{}_n(t)$ bestimmen.

```
i=0, j=0, k=0
while( j < Mo^{D_B}_m ) {
    if( Mo^{Smin}_m + j < M^{PS}_i ) {
        P^{MK_B}_n(j) = P^{MK_N}_n(k)
        ++j, ++k
    } else {
        P^{MK_B}_n(j) = 0
        ++j, ++i
    }
}
```

- Falls der Modus Material aus einem Silo entnimmt oder ein Silo damit füllt, muss er als Produktionsvorgang alle Bedingungen erfüllen. (siehe **Kapitel 3.2.2.3.c (3.2.13)**, **(3.2.14), (3.2.15), (3.2.16)**)

4. Nun müssen noch alle Links innerhalb der Aktivität mit maximalen Zeitabstand überprüft werden. Ist einer der maximalen Zeitabstände verletzt **(3.2.18)**, so werden die frühesten Startzeitpunkte des zugehörigen Vorgängers (Modus) entsprechend angepasst und **Schritt 3** nochmals durchgeführt. Existiert z.B. $Mo_1 \xrightarrow{SS,T_{\min},T_{\max}} Mo_2$ und haben die Modi die zugehörigen Startzeitpunkte $Mo^S{}_1$ und $Mo^S{}_2$, dann ist der Link verletzt, falls $Mo^S{}_2 > Mo^S{}_1 + T_{\max}$. Der früheste Startzeitpunkt $Mo^{Smin}{}_1$ wird erhöht: $Mo^{S\min}{}_1 = Mo^S{}_2 - T_{\max}$.

5. A_i^k wurde nun geplant. Falls A_i^k nicht die letzte Aktivität in der Konfiguration ist ($\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle$, wobei $|\langle K_2 \rangle| > 0$), starte erneut mit **Schritt 1** und dem nächsten Planungsobjekt A_j^l in der Konfiguration ($\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet A_j^l \bullet \langle K_2 \rangle$).

3.2.6.3 Reparaturmechanismus einer ungültigen Konfiguration

Das Planungsobjekt A_i^k an Position *P* der Konfiguration konnte nicht in einen Schedule übersetzt werden. Dies kann mehrere Gründe haben (**Kapitel 3.2.6.2 Schritt 1**). Bei jedem Aufruf wird genau *ein* Problem ‚repariert' bzw. behoben, wobei folgende Reihenfolge beachtet wird:

a) Problem: Es sind nicht alle Vorgänger geplant.

Vorgehen zur Behebung des Problems:

Von allen Vorgängern von A_i^k, die in der Konfiguration nach A_i^k stehen, wird derjenige mit minimaler Position gewählt. Der so gewählte Vorgänger A_j^l mit

$\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet A_j^l \bullet \langle K_3 \rangle$

wird nun vor A_i^k verschoben

$\langle K' \rangle = \langle K_1 \rangle \bullet A_j^l \bullet A_i^k \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$

und mit der neuen Konfiguration an Position P und der Aktivität A_j^l weitergearbeitet.

b) Problem: Die Lagerkapazität ist nicht ausreichend.

Vorgehen zur Behebung des Problems:

A_i^k produziert Material *X*, für das nicht mehr ausreichen Lagerkapazität in einem Silo vorhanden ist. Zwei Konfigurationen $\langle K^P \rangle$ und $\langle K^V \rangle$ werden erstellt, deren Aktivitäten aus der Teilkonfiguration $A_i^k \bullet \langle K_2 \rangle$ gebildet werden:

$A_p{}^m \subseteq A_i^k \bullet \langle K_2 \rangle \quad \forall A_p{}^m \subseteq \langle K^P \rangle$, $A_v{}^n \subseteq A_i^k \bullet \langle K_2 \rangle \quad \forall A_v{}^n \subseteq \langle K^V \rangle$.

Alle $A_p{}^m \subseteq \langle K^P \rangle$ produzieren dabei Material *X*, und alle $A_v{}^n \subseteq \langle K^V \rangle$ verbrauchen Material *X*.

Aus $\langle K^V \rangle$ wird nun eine Aktivität A_j^l mit minimaler Position gesucht, deren Vorgänger nicht in $\langle K^P \rangle$ vorhanden sind. Die so bestimmte A_j^l

$\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet A_j^l \bullet \langle K_3 \rangle$

wird nun vor A_i^k verschoben

$\langle K' \rangle = \langle K_1 \rangle \bullet A_j^l \bullet A_i^k \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$

und mit der neuen Konfiguration an Position P und der Aktivität A_j^l weitergearbeitet.

c) Problem: Es ist nicht genügend benötigtes Material vorhanden.

Vorgehen zur Behebung des Problems:

A_i^k benötigt Material *X*, das nicht mehr ausreichend zur Verfügung steht. Zwei Konfigurationen $\langle K^P \rangle$ und $\langle K^V \rangle$ werden erstellt, deren Aktivitäten aus der Teilkonfiguration $A_i^k \bullet \langle K_2 \rangle$ gebildet werden:

$A_p^m \subseteq A_i^k \bullet \langle K_2 \rangle \quad \forall A_p^m \subseteq \langle K^P \rangle,\ A_v^n \subseteq A_i^k \bullet \langle K_2 \rangle \quad \forall A_v^n \subseteq \langle K^V \rangle.$

Alle $A_p^m \subseteq \langle K^P \rangle$ produzieren dabei Material *X*, und alle $A_v^n \subseteq \langle K^V \rangle$ verbrauchen Material *X*.

Aus $\langle K^P \rangle$ wird nun eine Aktivität A_j^l mit minimaler Position gesucht, deren Vorgänger nicht in $\langle K^V \rangle$ vorhanden sind. Die so bestimmte A_j^l

$\langle K \rangle = \langle K_1 \rangle \bullet A_i^k \bullet \langle K_2 \rangle \bullet A_j^l \bullet \langle K_3 \rangle$

wird nun vor A_i^k verschoben

$\langle K' \rangle = \langle K_1 \rangle \bullet A_j^l \bullet A_i^k \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$

und mit der neuen Konfiguration an Position P und der Aktivität A_j^l weitergearbeitet.

Repariert wird immer direkt die Konfiguration und nicht die Lösung. Somit ist während der lokalen Suche jede Ausgangssituation eines *Moves* eine gültige Konfiguration. Dies hat den Vorteil, dass immer nur der Bereich repariert werden muss, der von dem letzten *Move* verändert wurde. Die beschriebene Vorgehensweise ist damit deutlich schneller als andere Vorgehensweisen, bei denen nur die Lösung repariert wird und ungültige Konfigurationen als Startpunkt der *Moves* zugelassen werden. Meist muss dabei nämlich die gesamte Lösung repariert werden. Im ungünstigsten Fall führt dieser Reparaturmechanismus dazu, dass der letzte *Move* zurückgenommen und mit der alten Konfiguration weitergearbeitet wird.

Falls man durch einen *Move* eine ungültige Lösung erzeugt, also den Reparaturmechanismus startet, kann man nicht durch einen inversen *Move* zur letzten gültigen Konfiguration zurückkehren. Bei der lokalen Suche (z.B. bei **SA**) wird allerdings ein Mechanismus benötigt, um zu der letzten Konfiguration zurückzukehren. Dazu wird immer eine Kopie der letzten gültigen Konfiguration angelegt und bei Bedarf der veränderte Teil wiederhergestellt.

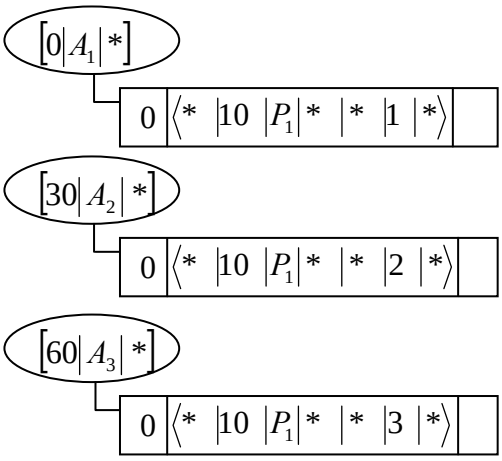
3.2.6.4 Aktive und semiaktive Schedules

Der so erzeugte Produktionsplan ist *semiaktiv* bzgl. der Primärressourcen. Es existiert noch die Möglichkeit, diesen Produktionsplan *aktiv* zu erzeugen, oder eine zusätzliche Freiheit bei der Planung vorzugeben und die Modi bei beliebigen Zeitpunkten zu planen, die die in Schritt 3 und 4 (**Kapitel 3.2.6.2**) gestellten Bedingungen erfüllen.

Semiaktiv bedeutet dabei, dass keine Aktivität(Modus) früher starten kann, ohne dabei die Reihenfolge auf den Ressourcen zu verändern. Bei einem *aktiven* Plan kann keine Aktivität(Modus) früher starten, ohne dabei eine andere zu verzögern.

Die Menge aller *aktiven* Pläne ist in der Menge aller *semiaktiven* enthalten. Bei **JobShop-**Problemen ist die optimale Lösung in der Menge der *aktiven* Pläne enthalten. Es wird in dieser Arbeit jedoch absichtlich die meist viel größere Menge der *semiaktiven* Pläne genutzt, da eine optimale Lösung im Falle einer Optimierung der Rüstzeiten nicht immer in der Menge der *aktiven* Pläne enthalten ist. Dies ist in folgendem Beispiel zu sehen:

Gegeben ist eine Primärressource P_1 mit den Rüstzeiten $M^{RZ}_{i,j}$ zwischen Vorgängerrüstschlüssel i und Nachfolgerrüstschlüssel j und drei Aktivitäten A_1, A_2, A_3:



$$M^{RZ}_{i,j} = \begin{cases} 10 & falls\ i < j \\ 0 & sonst \end{cases}$$

Es existiert genau ein *aktiver* Plan und sechs *semiaktive* Pläne. **Abbildung 3.2.8** zeigt zwei *semiaktive* Pläne, wobei einer eine minimale Durchlaufzeit (70) und der andere minimale Rüstzeiten (0) aufweist. Der einzige *aktive* Plan entspricht hier dem *semiaktiven* Plan mit minimaler Durchlaufzeit. Bei diesem Beispiel kann ein Plan mit minimaler Rüstzeit nicht durch *aktive* Planung erzeugt werden.

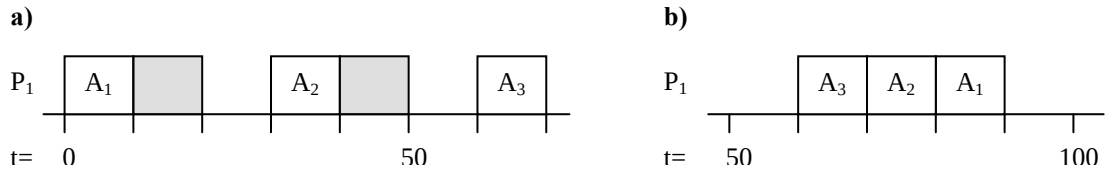


Abbildung 3.2.8: 2 der 6 *semiaktiven* Pläne sind zu sehen. Während Plan **a)** eine minimale Durchlaufzeit hat, ist bei Plan **b)** die Rüstzeit (schattiert) minimiert. Plan **a)** ist bei diesem Beispiel der einzige *aktive* Plan.

3.2.6.5 Nebenbedingungen

Im Unterschied zu dem in **Kapitel 3.1** definierten Modell werden die meisten Nebenbedingungen hier hart definiert, d.h. sie werden bei jeder Konfiguration eingehalten. Eingeführt wurde diese Definition, um den Suchraum zu verkleinern. Es existiert nur noch bei der maximalen Kapazität des Silos die Möglichkeit, diese als weiche Grenze zu definieren und die Einhaltung durch einen Straffunktionsterm in der Zielfunktion zu erzwingen. Bei einigen Problemstellungen hat dies den Vorteil, dass der Suchraum nicht in Inseln zerfällt (vgl. **Abbildung 3.2.9** und **Kapitel 9**).

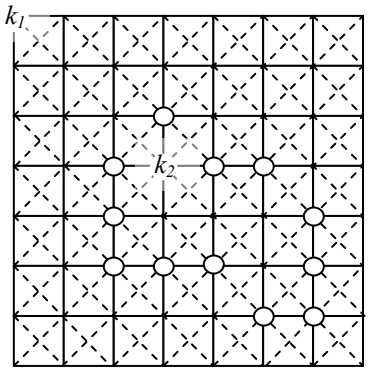


Abbildung 3.2.9: Suchraum: Die Konfigurationen sind hier als Eckpunkte in einem quadratischen Gitter dargestellt. Die Kreise zeigen Konfigurationen, die die Nebenbedingung verletzen. Die Linien entsprechen den Nachbarschaften (durchgezogene: N1, gestrichelt: N2). Es ist nicht möglich, mit N1 von k_1 zu k_2 zu gelangen, falls die Konfigurationen, die die Nebenbedingung verletzen, verboten sind. Diese Inselbildung durch die Nebenbedingung lässt sich durch N2 oder eine weiche Definition der Nebenbedingung mithilfe von Straftermen verhindern.

3.2.7 Optimierungsziele

3.2.7.1 Einzelne Zielfunktionswerte

Es wurden die Aktivitäten A_i ($1 \le i \le N_A$) geplant. Für jede Aktivität A_i ist nun Folgendes gegeben:

- der Liefertermin L_i
- der Startzeitpunkt $A^S{}_i$
- der Endzeitpunkt $A^E{}_i$
- die Rüstzeit $A^{RZ}{}_i$
- die Rüstkosten $A^{RK}{}_i$
- die Kosten des selektierten Multimodus $Mu^K{}_i$

Dadurch lassen sich nun die einzelnen Optimierungskriterien f_k definieren:

a) Maximale Verspätung

$$f_1 = MAX_i\left(MAX\left(0, A^E{}_i - L_i\right)\right) \qquad \textbf{(3.2.24)}$$

b) Summe der Verspätungen

Die Summe der Verspätungen ist definiert als

$$f_2 = \sum_i MAX\left(0, A^E{}_i - L_i\right) \quad \textbf{(3.2.25)}$$

oder als Summe der gewichteten Lieferzeitpunkte

$$f^g{}_2 = \sum_i \left(G^L{}_i * MAX\left(0, A^E{}_i - L_i\right)\right) \quad \textbf{(3.2.26)}$$

mit den Gewichten $G^L{}_i$.

c) Durchlaufzeit

$$f_3 = MAX_i(A^E{}_i) - MIN_i(A^S{}_i) \quad \textbf{(3.2.27)}$$

d) Summe der Rüstzeiten

$$f_4 = \sum_i A^{RZ}{}_i \quad \textbf{(3.2.28)}$$

e) Summe der Rüstkosten

$$f_5 = \sum_i A^{RK}{}_i \quad \textbf{(3.2. 29)}$$

f) Summe der Multimoduskosten

$$f_6 = \sum_i Mu^K{}_i \quad \textbf{(3.2.30)}$$

g) Überlastungskosten der Silos

Falls bei mindestens einem Silo eine weiche Grenze für die maximale Kapazität gegeben ist, so wird die Summe der Überlastungen aller Silos **(3.2.12)** als Kosten definiert:

$$f_7 = \sum_j S_j \quad \textbf{(3.2.31)}$$

3.2.7.2 Globales Optimierungsziel

Wird bei der Optimierung mehr als nur ein Kriterium verwendet (multikriterielle Optimierung), so lassen diese sich auf verschiedene Arten kombinieren:

a) Gewichtete Summe

Hier ist bei der Optimierung eine Lösung gesucht, die eine, nach den Prioritäten der Optimierungszielen, gewichtete Summe **(3.2.32)** minimiert. Diese gewichtete Summe wird in physikalischen Systemen als Energie E bezeichnet. Bei der Optimierung von Produktionsplanungsproblemen spricht man auch von den zu minimierenden Gesamtkosten. Bei der Anwendung genetischer Algorithmen stellt sie die Fitness der Individuen dar.

$$E = \sum_{k=1}^{7} \lambda_k * f_k \,, \qquad \textbf{(3.2.32)}$$

mit den Gewichten λ_k und den Zielfunktionswerten f_k.
Eine Lösung L_i ist hier, bei Minimierungsproblemen, somit besser als eine andere L_j, falls $E(L_i)<E(L_i)$. Es existieren möglicherweise viele verschiedene optimale Lösungen zu einem optimalen Energiewert E_{opt}.

b) Prioritätsbasierter Ansatz

Es sollen die einzelnen Optimierungskriterien unabhängig voneinander aufgrund ihrer Prioritäten optimiert werden. Optimierungskriterien mit gleicher Priorität werden aufsummiert. Optimierungskriterien verschiedener Prioritäten werden nacheinander betrachtet. Sind P_k die Prioritäten der Kriterien f_k , dann definiert $P_{k1} > P_{k2}$ die Priorität P_{k1} als ‚wichtiger' als P_{k2}. Es sind $f^S{}_k$ die aufsummierten Kriterien gleicher Priorität.

$$f^S{}_k = \sum_{l=1}^{6} K_{kl} \text{ mit } K_{kl} = \begin{cases} 0 & falls\ P_k \neq P_l \\ f_l & sonst \end{cases} \qquad \textbf{(3.2.33)}$$

Eine Lösung L_i wird bei Minimierungsproblemen als ‚besser' als L_j eingestuft, falls Folgendes gilt: $\left(\exists k \middle| f^S{}_k(L_i) < f^S{}_k(L_j) \wedge f^S{}_l(L_i) = f^S{}_l(L_j)\left(\forall l \middle| P_k < P_l\right)\right)$

Es wird nun eine Lösung gesucht, für die keine andere bessere Lösung existiert. Möglicherweise existieren mehrere solcher Lösungen. Diese stimmen dann in allen ihren $f^S{}_k$ überein. In dieser Arbeit wird kein prioritätsbasierter Ansatz verwendet. Jedoch wurde er bereits in **[2.2]** angewendet, um bei einem Sequenzierungsproblem Restriktionen verschiedener Prioritäten möglichst gut zu erfüllen.

c) Pareto-optimale Lösungen

Bei der Pareto-Optimalität, benannt nach Vilfredo Pareto, werden Lösungen gesucht, bei denen sich keines der Optimierungskriterien verbessern lässt, ohne mindestens ein anderes zu verschlechtern. Eine Lösung L_i ist Pareto-optimal, falls es kein L_j gibt mit $\left(\exists k \middle| f_k(L_j) < f_k(L_i)\right) \wedge \left(f_l(L_j) \leq f_l(L_i)\ \forall l\right)$.

d) Optimierungskriterien als Nebenbedingung

Es ist nun eine Optimierung der f_l mit der Bedingung $f_k \leq MAX_k$ gewünscht, wobei $l \neq k$. Dabei kann bei der Optimierung der Kriterien die gewichtete Summe, der prioritätsbasierte Ansatz oder eine Pareto-Optimierung erforderlich sein.
Eingesetzt wird dies z.B., um bei eingehaltenen Lieferzeitpunkten $f_2 \leq 0$ die anderen Optimierungskriterien, z.B. Durchlaufzeit und Rüstzeit ($E = \lambda_3 * f_3 + \lambda_4 * f_4$) zu minimieren.

Kapitel 4

Vergleich der Produktionsplanungsmodelle mit Spinglasmodellen

In **Kapitel 3.1** wurde gezeigt, dass sich Produktionsplanungsprobleme mit Hilfe von Spins definieren lassen, die die Werte 0 und 1 annehmen können und in einem dreidimensionalen Gitter verteilt sind. Die Wechselwirkung findet dabei nicht nur zwischen den nächsten Nachbarn statt, jedoch auch nicht zwischen allen vorhandenen Spins. Im Folgenden werden die beiden Modellierungen von Produktionsplanungsproblemen zwei Modellen gegenübergestellt, die zur Simulation eines Spinglases verwendet werden.

4.1 Spinglas-Modelle

Das Verhalten eines Spinglases bei Temperatursenkung soll anhand der folgenden vereinfachten Modelle, dem Edward-Anderson- (EA) und dem Sherrington-Kirkpatrick-Modell (SK), simuliert werden. Bei beiden Modellen wird mit Ising-Spins gearbeitet und eine Wechselwirkung verwendet, die die Werte –J,0,J annehmen kann. Die beiden Modelle wurden ausgewählt, da die Wechselwirkungen zum einen nur nächste Nachbarn und zum anderen alle Spins miteinander verbindet.

4.1.1 Das dreidimensionale ± J Edward-Anderson-Modell

Die Ising-Spins $S_i = \pm 1$ (*up*, *down*) sind bei diesem Modell auf einem dreidimensionalen kubischen Gitter an den Eckplätzen angeordnet. Die Wechselwirkung bei periodischen Randbedingungen (**Abbildung 4.1**) ist gegeben durch J_{ij}, wobei diese nur für die nächsten Nachbarn ungleich 0 ist. Der Hamiltonian für solch ein System in einem äußeren magnetischen Feld λ lautet

$$H = -\sum_{i<j} J_{ij} \cdot S_i \cdot S_j - \lambda \cdot M \qquad \textbf{(4.1)}$$

mit den Wechselwirkungen

$$J_{ij} = \begin{cases} \pm J & i \text{ ist nächster Nachbar von } j \\ 0 & \text{sonst} \end{cases}$$

und der Magnetisierung

$$M = \sum_i S_i \,. \qquad \textbf{(4.2)}$$

Existieren nur positive Werte (+J) für die Austauschwechselwirkung, so handelt es sich um ein ferromagnetisches System. Bei rein negativen Wechselwirkungen (-J) spricht man von einem antiferromagnetischen System.

Dieses Modell erscheint auf den ersten Blick sehr einfach, enthält jedoch zwei wesentliche Merkmale eines Spinglases: Unordnung und Konkurrenz.
Als Unordnung bezeichnet man die scheinbar zufällige Anordnung der Spins im Grundzustand. Unter Konkurrenz versteht man die gleichzeitige Existenz ferromagnetischer Wechselwirkungen, die versuchen, die Spins parallel anzuordnen, und antiferromagnetischer, die die Spins antiparallel ausrichten möchten. Dadurch kann es zu einer optimalen Anordnung von Spins kommen, die nicht gleichzeitig allen Nachbarn bezüglich ihrer Werte und Austauschwechselwirkungen J_{ij} genügen kann. Diesen Effekt nennt man *Frustration*. Frustration führt zu einer *Entartung* der Energiezustände, da verschiedene Einstellungen zu gleichen Energien führen. In **Abbildung 4.2** ist eine Grundzustandskonfiguration eines Spinglases zu sehen, bei der ein Spin frustriert ist.

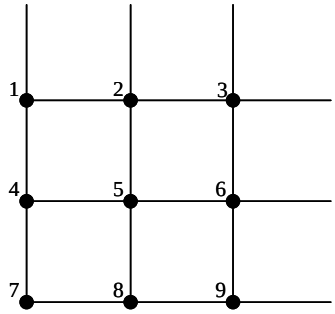


Abbildung 4.1: Zu sehen ist ein zweidimensionales kubisches Gitter mit 3*3 Positionen. Aufgrund der periodischen Randbedingung ist Position 1 nicht nur der nächste Nachbar der Positionen 2 und 4, sondern auch der Positionen 3 und 7.

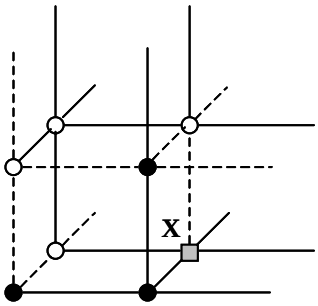


Abbildung 4.2: Einer der möglichen Grundzustände eines dreidimensionalen ± J EA-Modells mit periodischen Randbedingungen, ohne äußeres Magnetfeld, mit 2*2*2 Ising-Spins (der gefüllte Kreis entspricht +1, der leere -1) und Wechselwirkungen zwischen nächsten Nachbarn (die durchgezogene Linie entspricht +J, die gestrichelte -J). Der Spin an Position **X** ist frustriert, er kann nicht alle Wechselwirkungen zu seinen nächsten Nachbarn gleichzeitig erfüllen. Beide Einstellungen des Spins **X** führen zur gleichen Energie.

Bei den folgenden Simulationen mit **SA** werden die Wechselwirkungsbeträge auf 1 gesetzt (J=1). Das äußere Magnetfeld wird nicht beachtet (λ=0). Die Energie und Magnetisierung wird mit der Anzahl der Spins normiert. Im dreidimensionalen Fall existieren pro Spin drei Wechselwirkungsterme (jeweils einer in x-, y- und z-Richtung). Werden alle Wechselwirkungen erfüllt, ergibt das eine normierte Energie von –3. Die normierte Magnetisierung kann Werte zwischen –1 und 1 annehmen.

Im Folgenden wird zuerst ein System mit rein positiven Wechselwirkungen (0% negative Wechselwirkung) und 32*32*32 Spins simuliert. Die einzelnen Messgrößen dieses Ferromagneten sind in **Abbildung 4.3** zu sehen.

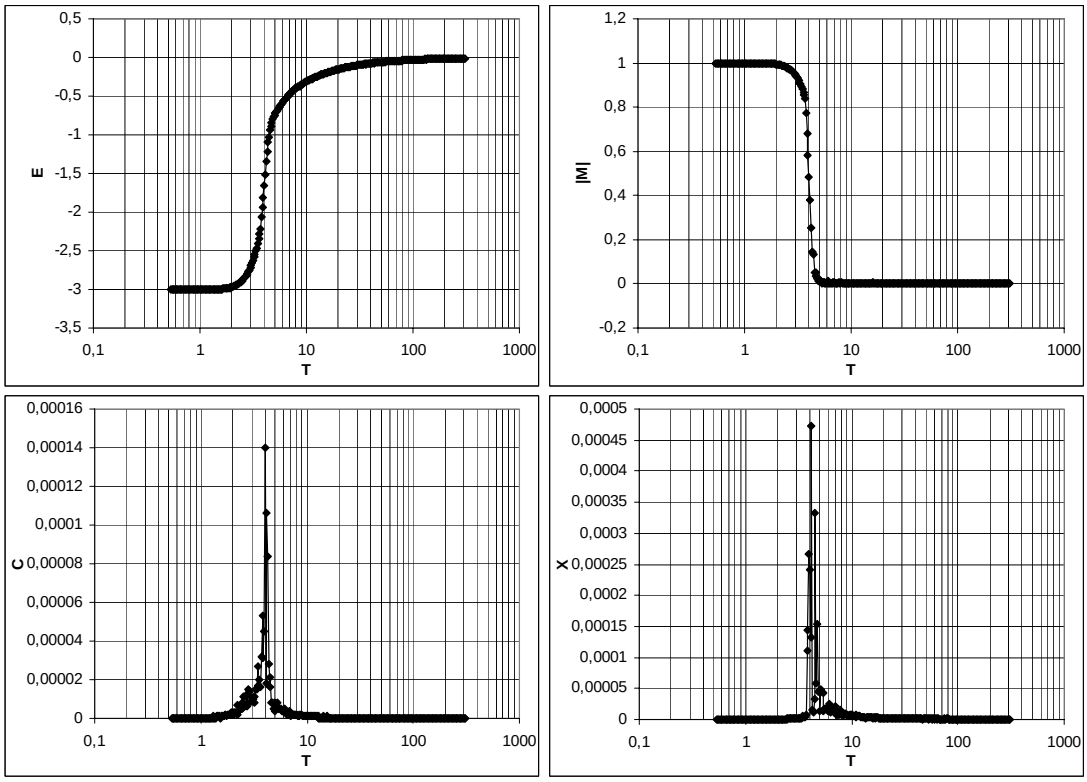


Abbildung 4.3: Es sind die Messgrößen Energie (**E**), Magnetisierungsbetrag (|**M**|), spezifische Wärme (**C**) und Suszeptibilität (**X**) in Abhängigkeit der Temperatur (**T**) für ein ferromagnetisches System (0% negative Wechselwirkungen) mit 32*32*32 Spins abgebildet.

Das ferromagnetische System hat bei einer bestimmten Temperatur einen Peak in der spezifischen Wärme. Dort vollzieht sich der Übergang vom ungeordneten zum geordneten Zustand. Als Ordnungsparameter dient beim Ferromagneten der Betrag der Magnetisierung. Dieser ist im Grundzustand 1. Alle Spins sind somit gleich ausgerichtet. Eine Entartung, alle Spins –1 oder +1, liegt nur aufgrund der Symmetrie des Hamiltonians vor. Die Grundzustandsenergie beträgt –3, d.h. alle Wechselwirkungen sind erfüllt. Die Entartung des Grundzustands lässt sich brechen, falls ein schwaches äußeres Magnetfeld (λ >0) angelegt wird.
Die Simulation eines antiferromagnetischen Systems erzielt ähnliche Resultate. Auch hier liegt die Grundzustandsenergie (für $N*N*N$ Systeme mit geradem N) bei –3. Die Magnetisierung ist im Grundzustand jedoch 0, da zwei benachbarte Spins aufgrund der Wechselwirkung unterschiedliche Richtung (1 oder -1) aufweisen. Der Magnetisierungsbetrag kann hier nicht mehr als Ordnungsparameter verwendet werden. Als Ordnungsparameter O ließe sich eine Projektion auf den Grundzustand verwenden:

$$O = \left| \sum_i S_i * S'_i \right|, \qquad \textbf{(4.3)}$$

wobei S'_i den Spin in einem Grundzustand darstellt. Der Grundzustand beim Ferromagneten besteht aus Spins gleicher Richtung (+1 oder -1), wodurch der Betrag der Magnetisierung dem Ordnungsparameter entspricht. Beim Antiferromagneten haben zwei benachbarte Spins immer unterschiedliche Richtung. Ein äußeres Magnetfeld bestimmter Stärke (λ=6*J) führt bei einem antiferromagnetischen System zu einer Frustration der Spins und somit höheren Entartung. Ist das Magnetfeld zu klein, so hat es keine Auswirkungen auf die Konfiguration im Grundzustand, ist es zu groß, so dominiert es den Hamiltonian und der Grundzustand besteht aus Spins gleicher Richtung.
Systeme ohne äußeres Magnetfeld zeigen bei x% negativen Wechselwirkungen, wobei $0<x<100$, Frustration. Es ist dann nicht mehr möglich, alle Wechselwirkungen zu erfüllen, die Energie pro Spin steigt auf Werte >-3 und die Magnetisierung liegt zwischen 0 und 1 (vgl. **Abbildung 4.4**).

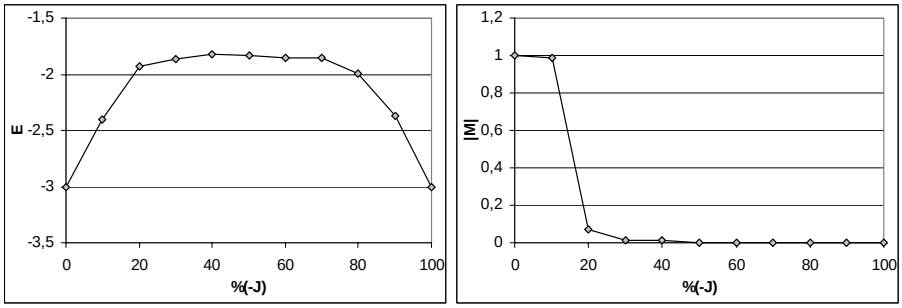


Abbildung 4.4: Es ist die Abhängigkeit der beiden Messgrößen Energie (**E**) und Magnetisierungsbetrag (|**M**|) der jeweils besten gefundenen Lösung für verschiedene Anteile an negativen Wechselwirkungen (**%-J**) für ein System aus 32*32*32 Spins zu sehen.

Bisher wurde zur Simulation nur der SingleSpinFlip (Nachbarschaft **N1**) verwendet. Dieser ist ausreichend für die relativ einfach zu lösenden Modelle mit 0% und 100% negativen Wechselwirkungen. Möglicherweise können jedoch effektivere Nachbarschaften definiert werden, um bei Spingläsern den Grundzustand in begrenzter Rechenzeit zu finden. Es sollen deshalb noch Nachbarschaften beachtet werden, die gezielt solche Spins verändern, die ihre Wechselwirkungen verletzen. Nachbarschaft **N2** verwendet den SingleSpinFlip, wobei Spins, die alle Wechselwirkungen erfüllen, nicht gedreht werden. Bei Nachbarschaft **N3** werden zusätzlich zu Nachbarschaft **N2** solche Spins nicht gedreht, die nur eine ihrer Wechselwirkungen nicht erfüllen. Die letzte getestete Nachbarschaft zielt nun nicht mehr auf einen Spin ab, sondern auf eine Wechselwirkung. Bei Nachbarschaft **N4** wird eine Wechselwirkung per Zufall ausgewählt, die nicht erfüllt ist. Einer der beteiligten Spins wird gedreht, so dass diese Wechselwirkung erfüllt wird. In **Abbildung 4.5** sind die Ergebnisse für drei verschiedene Instanzen mit jeweils 16*16*16 Spins mit unterschiedlicher Prozentsatz an negativen Wechselwirkungen (0%, 50%, 100%) für die verschiedenen Nachbarschaften und maximale Rechenzeiten (Anzahl Sweeps pro Temperaturschritt) zu sehen. Bei jeder Simulation wurde mit einem *Simulated Annealing* Ansatz gearbeitet und das gleiche Temperaturschema verwendet.

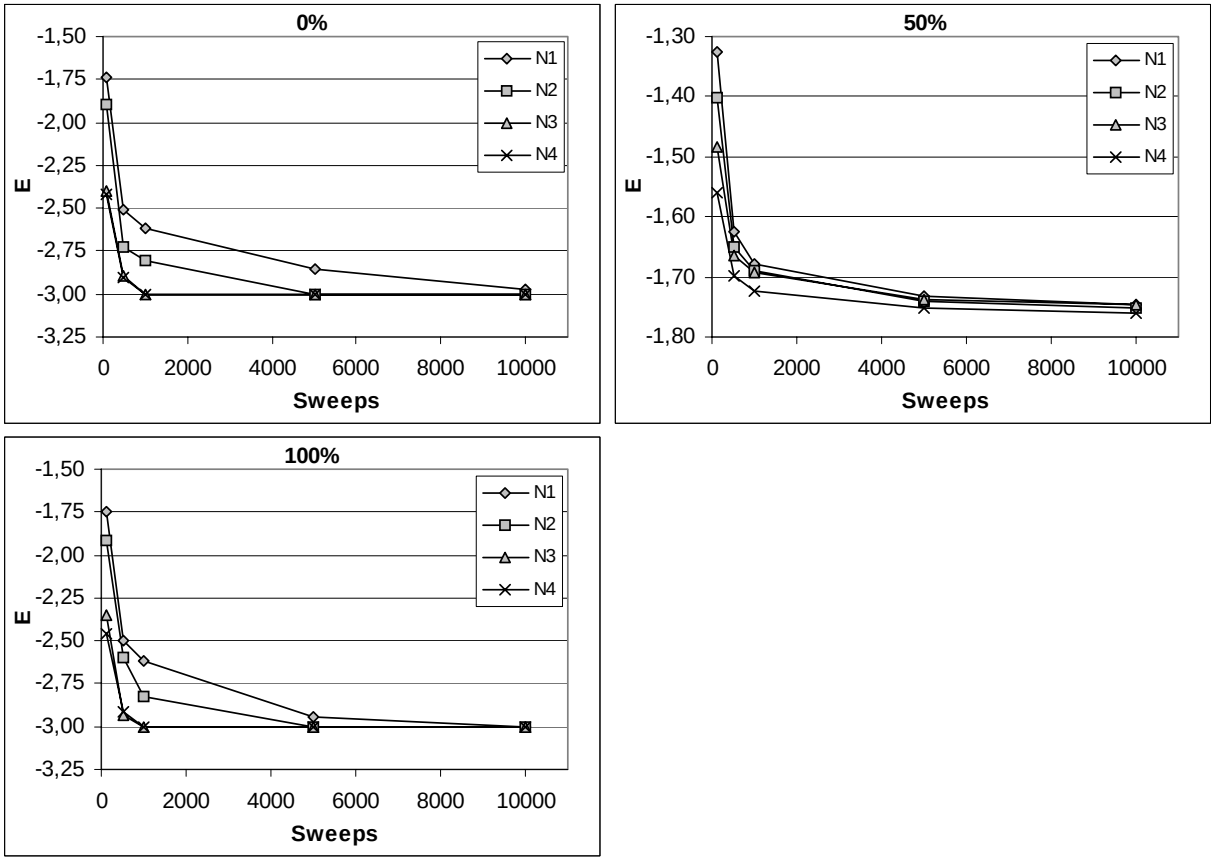


Abbildung 4.5: Zu sehen ist ein Vergleich verschiedener Nachbarschaften zur Lösung des dreidimensionalen $\pm$ J EA-Modells mit verschiedener Anzahl an negativen Wechselwirkungen (0%, 50%, 100%) und 16*16*16 Spins.

Bei den Ergebnissen kann man einen Unterschied der einzelnen Nachbarschaften bei kleinen Rechenzeiten erkennen. Nachbarschaft **N4** schneidet bei den verschiedenen Instanzen immer am besten ab, **N1** am schlechtesten. Bei großen Rechenzeiten verschwindet der Unterschied. Bei den Instanzen 0% und 100% sind keine Unterschiede zwischen den Ergebnissen von **N3** und **N4** zu sehen. Bei der Instanz 50% ist kein Unterschied zwischen **N2** und **N3** zu sehen. Der Grund für die unterschiedlichen Ergebnisse der Nachbarschaften **N2** und **N4**, die eigentlich beide Spins drehen, die mindestens eine Wechselwirkung nicht erfüllen, liegt in der unterschiedlichen Wahrscheinlichkeit, mit der diese Spins ausgewählt werden. Während bei **N2** alle Spins mit gleicher Wahrscheinlichkeit gewählt werden, werden bei **N4** die Wechselwirkungen mit gleicher Wahrscheinlichkeit versehen, was zu einer erhöhten Wahrscheinlichkeit für Spins führt, die viele Wechselwirkungen nicht erfüllen. **N4** ist deshalb der Nachbarschaft **N3** ähnlicher als der Nachbarschaft **N2**. Das ähnliche Abschneiden von **N1**, **N2** und **N3** auf die Instanz 50% liegt wahrscheinlich daran, dass die meisten der Spins mehr als eine Wechselwirkung nicht erfüllen.

4.1.2 Das ± J Sherrington-Kirkpatrick-Modell

Bei diesem Modell steht jeder Ising-Spin S_i in einer Wechselwirkung mit jedem anderen Spin. Der Hamiltonian ist hier wie zuvor gegeben durch

$$H = -\sum_{i<j} J_{ij} \cdot S_i \cdot S_j - \lambda \cdot \sum_i S_i \,. \qquad \textbf{(4.4)}$$

Eigentlich wird bei diesem Modell mit gaußverteilten Wechselwirkungen gearbeitet:

$$P(J_{ij}) \propto \exp(-J_{ij}^{\,2}/2) \,. \qquad \textbf{(4.5)}$$

Es soll hier jedoch nur eine Vereinfachung betrachtet werden, bei der die Wechselwirkung gegeben ist durch $J_{ij}=\pm J$.

Bei den folgenden Simulationen mit **SA** werden die Wechselwirkungsbeträge auf 1 gesetzt (J=1). Das äußere Magnetfeld wird nicht beachtet (λ=0). Die Energie und Magnetisierung wird mit der Anzahl der Spins normiert. Bei N Spins existieren $N*(N-1)/2$ Wechselwirkungen. Es kann also im besten Fall, falls alle Wechselwirkungen erfüllt sind, eine Energie $-(N-1)/2$ pro Spin erreicht werden. Die normierte Magnetisierung kann Werte zwischen –1 und 1 annehmen.
Im Folgenden wird zuerst ein System mit rein positiven Wechselwirkungen (0% negative Wechselwirkung) und 50 Spins simuliert. Die einzelnen Messgrößen dieses Ferromagneten sind in **Abbildung 4.6** zu sehen.

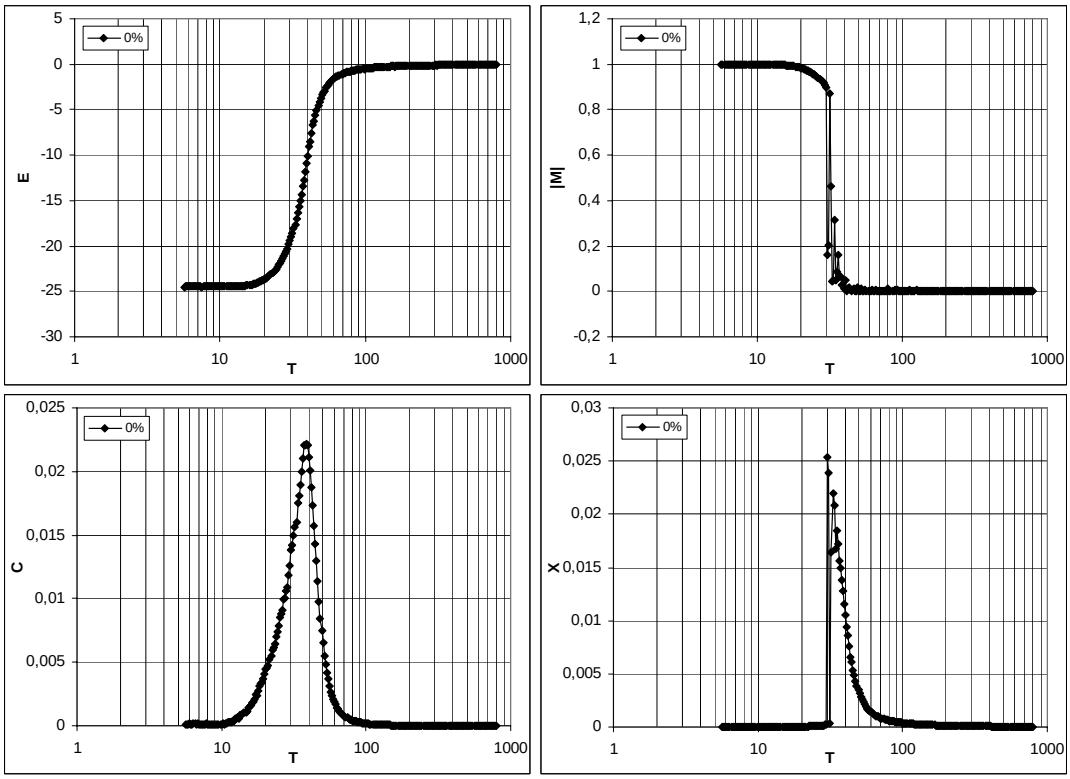


Abbildung 4.6: Es sind die Messgrößen Energie (**E**), Magnetisierungsbetrag (|**M**|), spezifische Wärme (**C**) und Suszeptibilität (**X**) in Abhängigkeit der Temperatur (**T**) für ein ferromagnetisches System (0% negative Wechselwirkungen) mit 50 Spins eines ± J SK-Modells abgebildet.

Bei der Energie ist gut zu sehen, dass sie im Fall rein ferromagnetischer Wechselwirkung im Grundzustand den Wert -24,5 pro Spin annimmt. Es werden somit alle Wechselwirkungen erfüllt. Zwei Konfigurationen mit dieser Energie existieren, alle Spins auf 1 oder alle Spins auf –1. Der Betrag der Magnetisierung ist im Grundzustand 1. Der Peak in der spezifische Wärme zeigt, dass auch hier ein Übergang von einem ungeordneten in ein geordnetes System stattfindet. Dieser findet jedoch bei höheren Temperaturen als beim EA-Modell statt. Der Betrag der Magnetisierung dient beim Ferromagneten wieder als Ordnungsparameter. Dies ist beim Antiferromagneten nicht möglich.

Im Unterschied zum EA-Modell ist hier der reine Antiferromagnet (**Abbildung 4.7**) nicht mehr gekennzeichnet durch einen zweifach entarteten Grundzustand. Falls N>1 und gerade, existieren $\binom{N}{N/2}$ Konfigurationen mit der Grundzustandsenergie –0,5 pro Spin und einer Magnetisierung 0. Bei ungeradem N>0 existieren $2*\binom{N}{(N-1)/2}$ Konfigurationen im Grundzustand mit der Energie –0.5*(N-1)/N und dem Magnetisierungsbetrag 1/N.

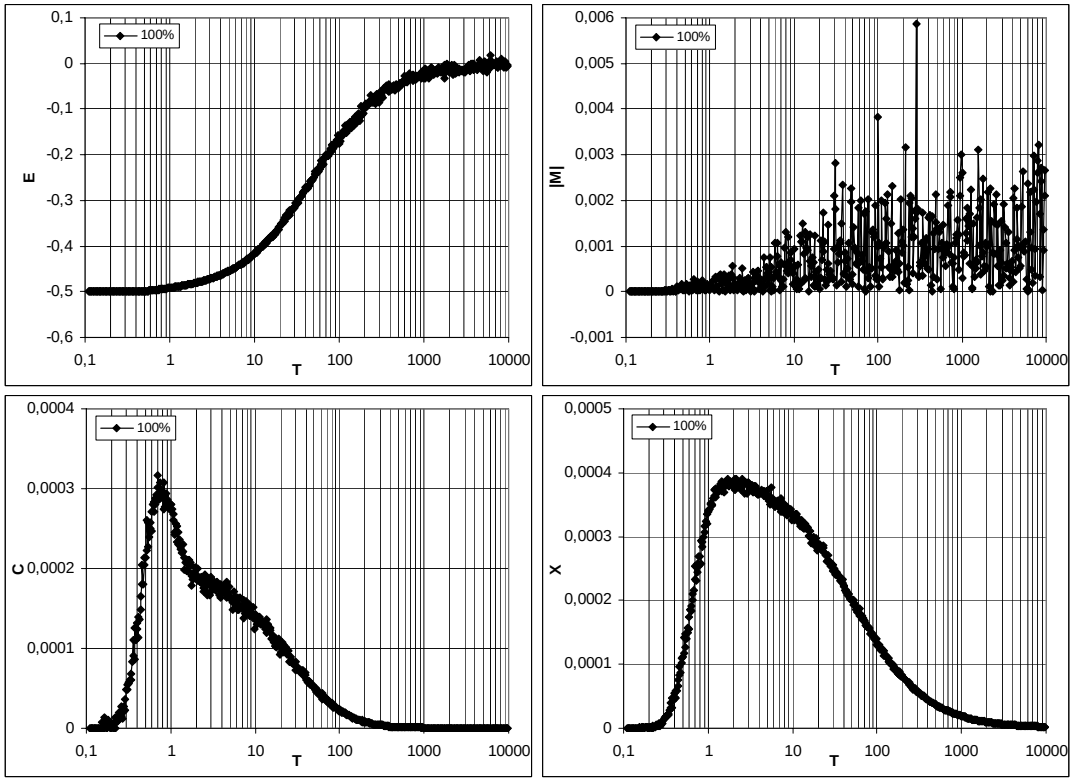


Abbildung 4.7: Es sind die Messgrößen Energie (**E**), Magnetisierungsbetrag (|**M**|), spezifische Wärme (**C**) und Suszeptibilität (**X**) in Abhängigkeit der Temperatur (**T**) für ein antiferromagnetisches System (100% negative Wechselwirkungen) mit 50 Spins eines $\pm$J SK-Modells abgebildet.

Für dieses Modell sollen nun auch verschiedene Nachbarschaften getestet werden, um die Effektivität eines einfachen *Simulated Annealings* zu steigern. Neben der Nachbarschaft **N1**, bei der ein Spin per Zufall gedreht wird, soll noch mit den Nachbarschaften **N2** und **N3** gearbeitet werden. Bei **N2** wird eine verletzte Wechselwirkung per Zufall gewählt und dann einer der zugehörigen Spins ge-

dreht. Dies ist vergleichbar mit der Nachbarschaft **N4** aus **Kapitel 4.1.1**. Bei **N3** wird Nachbarschaft **N1** zu 90% verwendet. Zusätzlich wird mit 10%iger Wahrscheinlichkeit derjenige Spin gedreht, der die wenigsten seiner Wechselwirkungen erfüllt, falls dadurch die Gesamtenergie verbessert wird.
Die auf der Auswahl einer nicht erfüllten Wechselwirkung basierende Nachbarschaft **N2** schneidet hier besser ab als der einfache zufällige SingleSpinFlip (**N1**) (vgl. **Abbildung 4.8**), wie dies bereits annähernd bei dem Vergleich der Nachbarschaften des EA-Modells der Fall war. Jedoch kann der einfache SingleSpinFlip verbessert werden durch die Auswahl eines speziellen Spins für den SingleSpinFlip in 10% der Fälle (**N3**). Eine Nachbarschaft, basierend alleine auf dieser Erweiterung, ist unabhängig von der investierten Rechenzeit nicht in der Lage, die Instanz zu lösen.

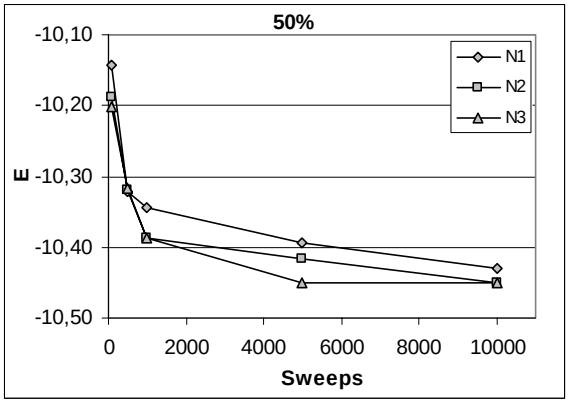


Abbildung 4.8: Zu sehen ist ein Vergleich verschiedener Nachbarschaften zur Lösung des $\pm J$ SK-Modells mit 50% negativen Wechselwirkungen und 200 Spins.

Zu Testzwecken wurden verschiedene Instanzen von Prof. Dr. S. Kobe (TU Dresden) bearbeitet. Diese hatten 50-60 Spins und positive wie negative Wechselwirkungen. Diese Instanzen und deren Grundzustände wurden mir freundlicherweise von Dr. J. Schneider zur Verfügung gestellt. Zur schnellen Lösung wurde eine Kombination der Nachbarschaften **N2** und **N3** zu jeweils 50% verwendet. Es war hier durchaus möglich, bei allen Instanzen das Optimum zu finden**.**

4.2 Modelle zur Simulation von Produktionsplanungsproblemen

Im Folgenden werden die beiden Modelle aus **Kapitel 3** den Modellen zur Simulation eines Spinglases gegenübergestellt. Dabei werden die einzelnen Observablen und deren temperaturabhängiger Verlauf miteinander verglichen.

4.2.1 Spinmodell

In **Kapitel 3.1** wurde bereits gezeigt, dass sich Produktionsplanungsprobleme auf ein dreidimensionales Spinsystem abbilden lassen, wobei die einzelnen Spins die Werte 0 oder 1 annehmen.
Der Hamiltonian ist bei **JobShop**-Problemstellungen gegeben durch

$$E = \lambda_1^Z \cdot Z_1 + \lambda_1^S \cdot S_1 + \lambda_2^S \cdot S_2 + \lambda_3^S \cdot S_3 + \lambda_4^S \cdot S_4 \qquad \textbf{(4.6)}$$

mit der Zielfunktion

$$Z_1 = MAX_{i,t,m}\left(P^S{}_{i,t,m} \cdot \left(t + P^D{}_{i,m}\right)\right), \qquad \textbf{(4.7)}$$

den Gewichten λ_1^Z, λ_1^S, λ_2^S, λ_3^S, λ_4^S und den Straffunktionstermen

$$S_1 = \sum_i \left| \sum_{t,m} P^S{}_{i,t,m} - 1 \right|, \qquad \textbf{(4.8)}$$

$$S_2 = \sum_m \sum_{\substack{i,j \\ i \neq j}} \sum_t \sum_{t \le t' < t + P^D{}_{i,m}} P^S{}_{i,t,m} \cdot P^S{}_{j,t',m} \cdot \left(t + P^D{}_{i,m} - t'\right), \qquad \textbf{(4.9)}$$

$$S_3 = \sum_{i,t,m} \left(1 - P^M{}_{i,m}\right) \cdot P^S{}_{i,t,m}\,, \qquad \textbf{(4.10)}$$

$$S_4 = \sum_{i,j} Z_{i,j} \cdot \sum_{m,m'} \sum_t \sum_{t' \ge t - Z^T_{i,j}} P^S{}_{i,t',m'} \cdot P^S{}_{j,t,m} \cdot \left(t' - t + Z^T{}_{i,j}\right), \qquad \textbf{(4.11)}$$

wobei gültige Lösungen bei dieser Problemstellung die einzelnen Straffunktionen minimieren, so dass $S_1=S_2=S_3=S_4=0$. Dies kann durch eine geeignete Wahl der Lagrange-Multiplikatoren λ_i^S erreicht werden. Eine gültige Lösung hat dann eine Energie E, die der Durchlaufzeit entspricht. Allgemein gesprochen entspricht die Energie des physikalischen Systems den Kosten bei der Optimierung von Produktionsplanungsproblemen, wobei diese Kosten meist nicht direkt ermittelbar sind und durch zeitbezogene Ziele in der Kostenfunktion ersetzt werden (vgl. **Kapitel 1.12**). Die Nebenbedingungen zwingen die Konfiguration der Spins in eine bestimmte Anordnung, ähnlich dem äußeren Feld λ in den Spinglasmodellen. Aufgrund der Wechselwirkungen ist dieses Modell zwischen dem EA-Modell, bei dem die Wechselwirkung nur zwischen nächsten Nachbarn stattfindet und einem SK-Modell, bei dem jeder Spin in einer Wechselwirkung mit jedem anderen Spins steht, einzuordnen. Es enthält ebenfalls die beiden Merkmale eines Spinglases: Unordnung und Konkurrenz.
Die Straffunktionen der Nebenbedingungen können hier als Beispiel konkurrierender Wechselwirkungen herangezogen werden. Zum einen versucht die Zielfunktion, alle Spins zu späten Zeitpunkten t auf 0 zu setzen, andererseits führen die Nebenbedingungen dazu, nicht alle Spins gleich 0 (S_1) zu setzen und nicht zu gleichen Zeiten auf 1 (S_2), wodurch einige Spins auch zu späteren Zeitpunkten auf 1 stehen müssen. Dies führt zu Frustrationseffekten und zu einer Entartung der Zustände.

Eine Entartung ist jedoch aufgrund der komplexen Wechselwirkungen meist nicht mehr nur bei einzelnen Spins gegeben, sondern durch Kombination vieler.
Eine Optimierung mit der einfachsten Nachbarschaft (1), gebildet aus dem SingleSpinFlip Operator, erbrachte selbst bei kleinen Problemstellungen keine annehmbaren Ergebnisse. Im Folgenden werden deshalb Simulationen mit der erweiterten Nachbarschaft (2) (vgl. **Kapitel 3.1.14**) durchgeführt. Dies führt dazu, dass $S_1=S_3=0$. Als Ordnungsparameter O kann hier die Auslastung der Maschinen betrachtet werden. Diese wird als Prozentsatz der tatsächlich belegten Zeit zur gesamtem Produktionszeit verstanden:

$$O = \frac{1}{M}\sum_{m=1}^{M}\frac{B_m}{E_m - S_m} \qquad \textbf{(4.12)}$$

mit dem Produktionsende E_m, dem Produktionsstart S_m und der belegten Produktionszeit B_m der Maschine m:

$$E_m = MAX_{i,t}\left(P^S{}_{i,t,m}\cdot\left(t + P^D{}_{i,m}\right)\right), \qquad \textbf{(4.13)}$$

$$S_m = E_m - MAX_{i,t}\left(E_m - P^S{}_{i,t,m}\cdot t\right), \qquad \textbf{(4.14)}$$

$$B_m = \sum_{i,t} P^S{}_{i,t,m}\cdot P^D{}_{i,m}\ . \qquad \textbf{(4.15)}$$

Während der Simulation sollte dieser Ordnungsparameter von einem kleinen Wert bei hohen Temperaturen mit sinkender Temperatur ansteigen. Der Maximalwert 1 ist bei den meisten Problemstellungen jedoch nicht immer erreichbar.

Im Folgenden wird ein **JobShop**-Problem bearbeitet, bei dem 6 Maschinen und 6 Aufträge vorhanden sind (**MT6**). Bei diesem Benchmark ist es mit der gegebenen Modellierung möglich, die optimale Lösung von 55 zu finden. Auf Simulationen von größeren Instanzen wird verzichtet, da diese nur unter erheblichem Zeitaufwand bearbeitet werden können.
Zuerst wird eine Simulation mit den Gewichten $\lambda_1^Z = \lambda_4^S = 1,\ \lambda_2^S = 10$ durchgeführt. Auf die Gewichte λ_1^S und λ_3^S wird verzichtet, da die erweiterte Nachbarschaft (2) die zugehörigen Nebenbedingungen immer erfüllt. Der Planungszeitraum wird auf 0 bis 2000 festgelegt.
Der Grund für die Einstellung der Gewichte ist in **Abbildung 4.9** zu sehen. Die Maxima der Suszeptibilitäten sollten bei den harten Nebenbedingungen bei höheren Temperaturen liegen als die der Zielfunktionen, um sicherzustellen, dass diese in der letzten Lösung eingehalten werden. Dies kann durch eine geeignete Wahl der Gewichte erreicht werden, wobei ein höheres Gewicht dazu führt, dass der zugehörige Term in der Kostenfunktion bei höheren Temperaturen optimiert wird. Die Maxima der einzelnen Straffunktionen sollten jedoch in einem gemeinsamen Temperaturbereich liegen. Durch die Trennung der Temperaturbereiche, in denen die Zielfunktion und die Straffunktionen optimiert werden, entstehen in der spezifischen Wärme zwei benachbarte Maxima. Diese sollten jedoch nicht zu weit voneinander entfernt sein, um dem System noch einige Freiheit zu geben, während der Optimierung der Durchlaufzeit noch Lösungen zu akzeptieren, die Nebenbedingungen verletzen. So erkennt man, dass, während die Durchlaufzeit sinkt, noch Verletzungen der Nebenbedingungen auftreten, die Verbesserung der Straffunktionen aber größtenteils bereits stattfindet, während die Durchlaufzeit noch um einen konstanten Wert schwankt. An den Suszeptibilitäten ist eine wichtige Eigenschaft der Gewichte abzulesen. Obwohl die Gewichte der Nebenbedingungen sich

um einen Faktor 10 unterscheiden, werden die gewichteten Straffunktionen im gleichen Temperaturbereich optimiert, d.h., sie sind mit dieser unterschiedlichen Gewichtung für die Optimierung von gleicher Priorität. Bei der Einstellung der Gewichte sollte darauf geachtet werden, dass zwei Optimierungsziele mit gleichem Gewicht nicht unbedingt die gleiche Priorität bei der Optimierung besitzen.
Am Ende der Temperaturabsenkung friert das System zwar in einem Zustand bestimmter Energie ein, zu sehen in der verschwindenden spezifischen Wärme, hat aber noch die Freiheit, im Konfigurationsraum entartete Zustände zu besuchen. Dies ist der Grund für das Ansteigen der Suszeptibilität des Ordnungsparameters bei kleinen Temperaturen. Es existieren somit mehrere bzgl. des definierten Ordnungsparameters unterschiedliche Lösungen, die zueinander benachbart sind.
Insgesamt kann eine gute Übereinstimmung des temperaturabhängigen Verhaltens der Observablen mit denen der Simulationen der Spingläser festgestellt werden. Dies begründet die Verwendung der, in der Festkörperphysik erfolgreich eingesetzten, statistischen Verfahren wie z.B. *Simulated Annealing* im Bereich der Produktionsplanung. Das bisher verwendete Modell scheint zu viele Freiheiten zu gewähren und benötigt daher mehr Rechenzeit, als dies zur Lösung solcher Problemstellungen erforderlich ist. Ein weiterer Nachteil dieser Freiheit ist, dass erst gegen Ende der Suche gültige Lösungen gefunden werden. Es wurde ja bereits eine Reduzierung der Freiheit durch Veränderung der Nachbarschaft eingeführt und auf den einfachen SingleSpinFlip verzichtet, der bei Simulationen eines Spinglases ausreichend erschien.

Im Folgenden sollen noch Ergebnisse der erweiterten Nachbarschaft (3) aus **Kapitel 3.1.14** vorgestellt werden. Die Veränderung ist an und für sich nicht sehr groß und wird nur mit einer Wahrscheinlichkeit <1 benutzt, trotzdem unterscheiden sich die Observablen (vgl. **Abbildung 4.10**) sehr deutlich von den Simulationen der Nachbarschaft (2). Die Gewichte müssen für diese Simulation angepasst werden: $\lambda_1^Z = 1,\ \lambda_2^S = \lambda_4^S = 20$. Die Veränderung der Nachbarschaft wirkt sich direkt auf die Durchlaufzeit aus, da jede Anwendung des veränderten Nachbarschaftsoperators zu einer Verringerung der Durchlaufzeit führt. Bei der Simulation beginnt die Durchlaufzeit deshalb bei kleinen Werten, im Unterschied zur Simulation mit der Nachbarschaft (2), bei der die Durchlaufzeit bei hohen Temperaturen im Bereich der maximal vorgegebenen Zeit 2000 liegt. Eine hohe Verletzung der beiden Nebenbedingungen, vor allem von S_2, ist die Folge, deshalb auch die höheren Gewichte bei den Nebenbedingungen. Es versteht sich von selbst, dass die Nebenbedingung S_2 nur bei ausreichend großer Durchlaufzeit eingehalten werden kann. Die Durchlaufzeit steigt deshalb beim Senken der Temperatur an, während die Verletzung der Nebenbedingung sinkt. Bei kleinen Temperaturen erreicht die Durchlaufzeit ein Maximum und sinkt dann ab. Dieser Peak ist die Folge der Gewichte. Bevor die Optimierung der Durchlaufzeit abgeschlossen ist, existieren bereits nur noch gültige Lösungen, da die Optimierung der Nebenbedingungen bei höheren Temperaturen stattfindet. Die besten gefundenen Ergebnisse der beiden Nachbarschaften (2) und (3) unterscheiden sich nicht bzgl. ihrer Energie. Hier ist das System jedoch nicht im Zustand niedrigster Energie eingefroren, sondern in einem etwas höheren lokalen Optimum.
Der Ordnungsparameter weist in dieser Nachbarschaft, wie die Durchlaufzeit, erhebliche Unterschiede zur Nachbarschaft (2) auf. Dieser beginnt bei hohen Werten und sinkt dann ab. Werte über 1 deuten auf eine Überlastung der Maschinen und somit Verletzung der Nebenbedingung S_2 hin.

Der letzte Zustand ist entartet. In der Nachbarschaft existieren mehrere Lösungen mit der gleichen Energie, was an der, bei kleinen Temperaturen, ansteigenden Suszeptibilität zu sehen ist.

Im Folgenden wird von der bisherigen Modellierung Abstand genommen, da sich diese in weiteren Tests als wenig erfolgreich erwies. Es war bei dieser Modellierung zwar verhältnismäßig einfach, neue Nebenbedingungen hinzuzufügen oder die Nachbarschaft anzupassen, jedoch erwiesen sich die vielen Freiheiten bei der Modellierung der harten Nebenbedingungen durch Strafterme als störend. Die nächste Idee, ein Modell für ein Produktionsplanungsproblem zu definieren, behandelt deshalb viele der Nebenbedingungen als hart, d.h. während der Suche werden diese von jeder Lösung erfüllt. Zusätzlich kommt der Gedanke, so früh wie möglich zu planen, der ja bereits in der Nachbarschaft (3) des Spinmodells eingeführt wurde, beim nächsten Modell viel stärker zum Tragen.

Vergleicht man die Suchräume der verschiedenen Nachbarschaften miteinander, so kann man den hauptsächlichen Grund für die schlechten Ergebnisse der Nachbarschaft (1) erkennen. Der Suchraum besteht bei Nachbarschaft (1), verwendet man als Zeitintervall 0 bis 2000, aus $N(1)=2^{sp1}$ verschiedenen Lösungen, wobei sp1=36*6*2000 (36 Produktionsvorgänge auf 6 Maschinen mit 2000 möglichen Zeitpunkten). Nachbarschaft (2), in der die Nebenbedingungen S_1 und S_3 eingehalten sind, besteht nur noch aus $N(2)=2000^{sp2}$ Lösungen, wobei sp2=36 (36 Produktionsvorgänge, 2000 mögliche Zeitpunkte). Die Lösungen, die in der Nachbarschaft (1) und nicht in Nachbarschaft (2) vorkommen, sind bzgl. der Nebenbedingungen nicht gültig. Auf sie kann deshalb verzichtet werden, ohne optimale Lösungen zu verlieren. Die definierte Nachbarschaft (2) ist ebenso wie Nachbarschaft (1) *optimum connected*, d.h., sie ist ergodisch und enthält das Optimum. Nachbarschaft (1) hat somit keine Vorteile im Vergleich zur Nachbarschaft (2), besteht jedoch aus sehr viel mehr abzusuchenden Lösungen. Der Suchraum der Nachbarschaft (3) entspricht dem der Nachbarschaft (2), setzt man das gleiche Zeitintervall voraus. Allerdings ist gut zu erkennen, dass bei Anwendung der Nachbarschaft (3) das Zeitintervall nur aus 0 bis 70 bestehen müsste und somit einen Suchraum mit $N(2)=70^{sp3}$ Lösungen, wobei sp3=36 (36 Produktionsvorgänge, 70 mögliche Zeitpunkte), besitzt.

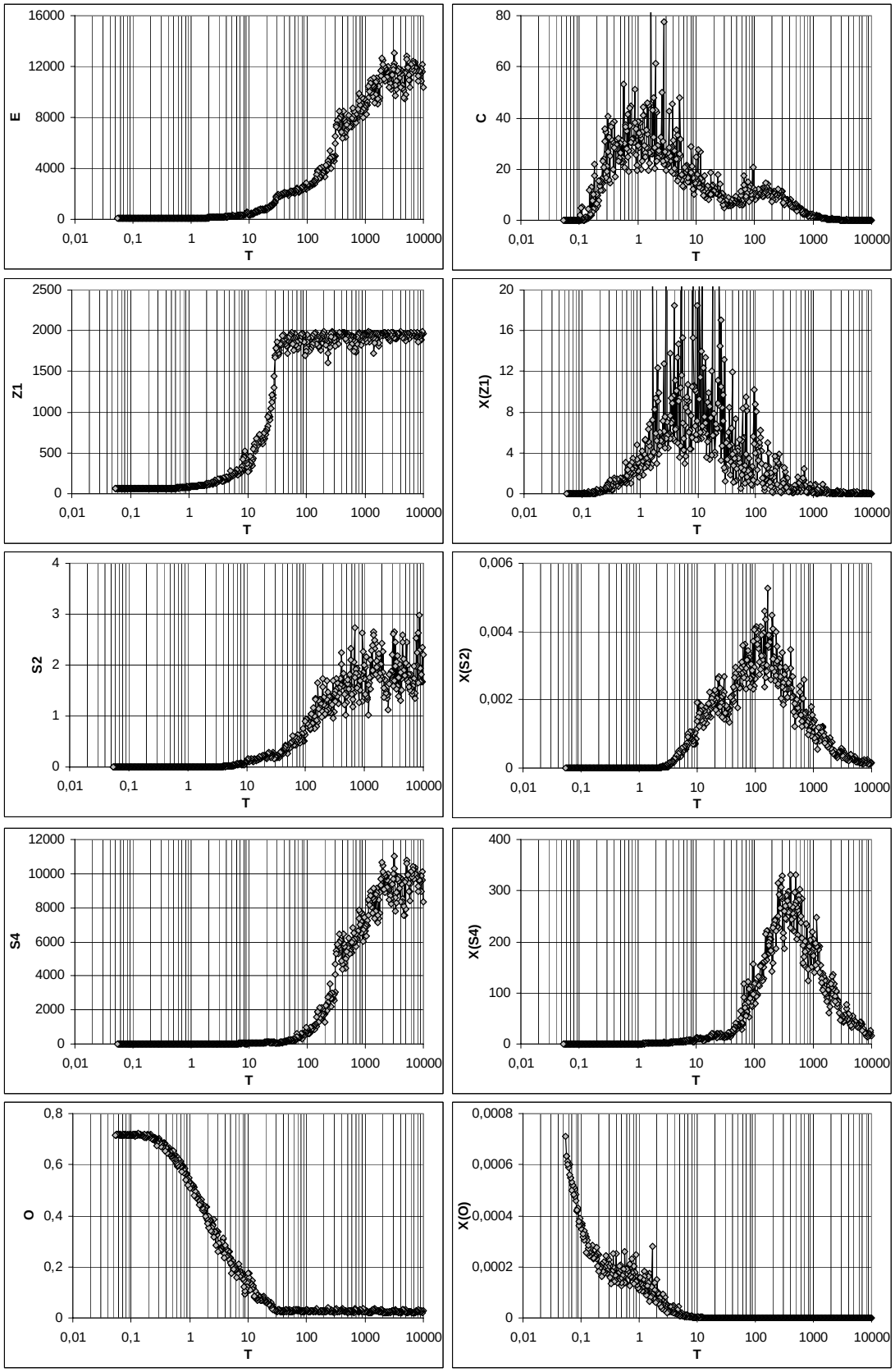


Abbildung 4.9: In dieser Abbildung sind die Energie (**E**), Ziel- (**Z1**) und Straffunktionswerte (**S2**, **S4**) und der Ordnungsparameter (**O**) in Abhängigkeit von der Temperatur (**T**) in der linken Spalte zu sehen und die spezifische Wärme (**C**) und die einzelnen Suszeptibilitäten (**X**) rechts. Die Simulationen zu **MT6** wurden dabei mit Nachbarschaft (2) durchgeführt.

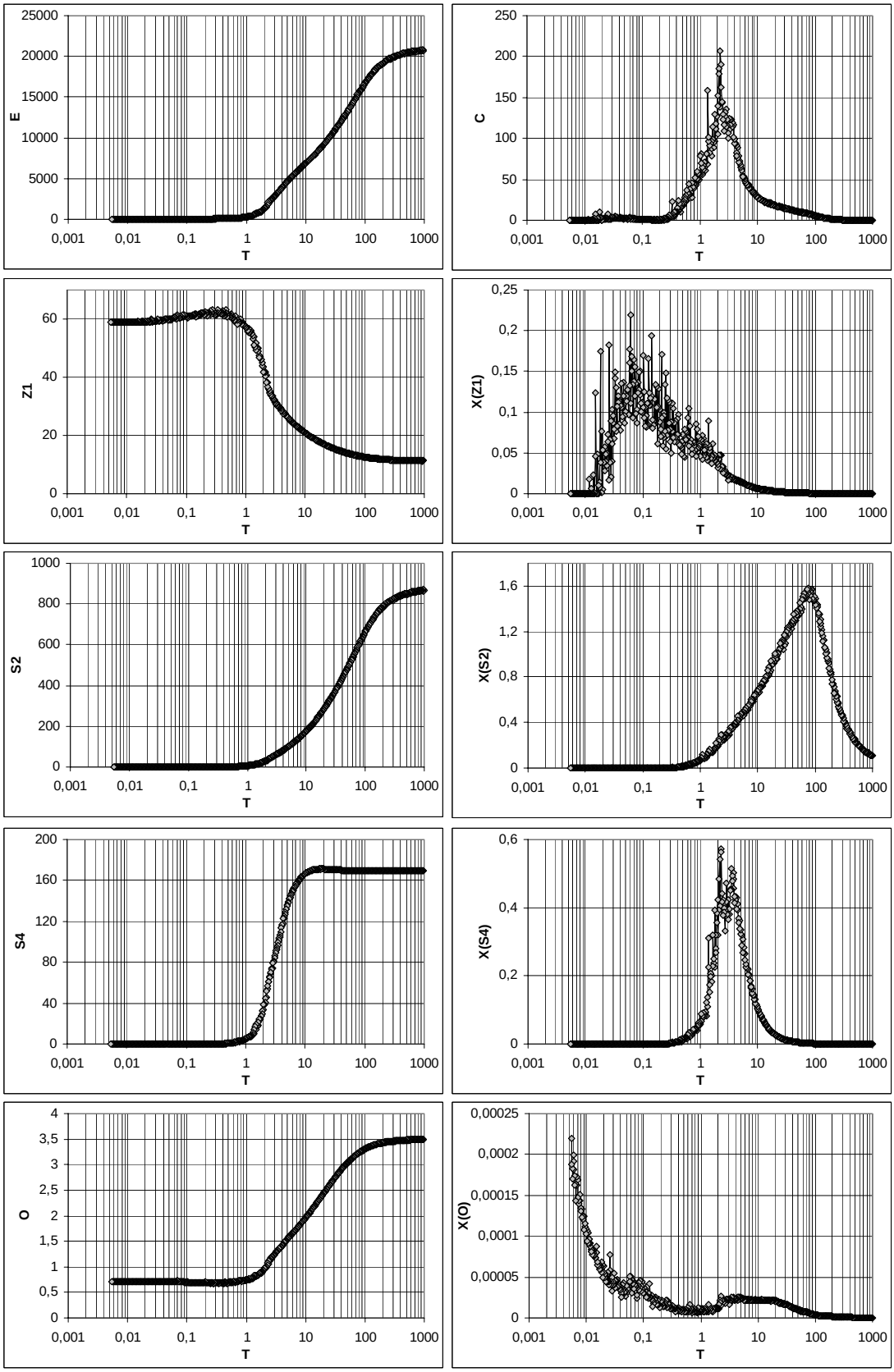


Abbildung 4.10: In dieser Abbildung sind die Energie (**E**), Ziel- (**Z1**) und Straffunktionswerte (**S2**, **S4**) und der Ordnungsparameter (**O**) in Abhängigkeit von der Temperatur (**T**) in der linken Spalte zu sehen und die spezifische Wärme (**C**) und die einzelnen Suszeptibilitäten (**X**) rechts. Die Simulationen zu **MT6** wurden dabei mit Nachbarschaft (3) durchgeführt.

4.2.2 Modell mit direkter Repräsentation

Bei dieser Modellierung sind die zuvor benötigten Nebenbedingungen alle automatisch eingehalten. Jede Konfiguration erfüllt diese. Es ist deshalb nur noch auf die Minimierung der Durchlaufzeit zu achten. In **Abbildung 4.12** ist die Energie, die der Durchlaufzeit entspricht, zusammen mit der spezifischen Wärme und der Suszeptibilität für eine Simulation des **MT6** Problems gegeben. Der in **Kapitel 4.2.1** eingeführte Ordnungsparameter für Produktionsplanungsprobleme kann auch bei dieser Modellierung verwendet werden. Der temperaturabhängige Verlauf ist ebenfalls in **Abbildung 4.12** zu sehen.
Die konkurrierende Wechselwirkung ergibt sich nun z.B. aus der Eigenschaft, dass nicht gleichzeitig zwei Produktionsvorgänge auf einer Maschine zum gleichen Zeitpunkt geplant werden können. Die einzelnen Produktionsvorgänge können somit nicht alle gleichzeitig zum frühestmöglichen Zeitpunkt starten um damit die Durchlaufzeit zu reduzieren. Eine Entartung ist auch hier die Folge (vgl. **Abbildung 4.11**).

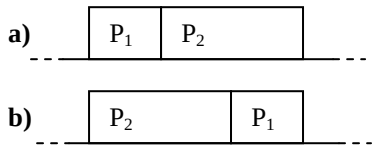


Abbildung 4.11: Die beiden Lösungen **a)** und **b)** weisen die gleiche Durchlaufzeit auf. Es ist nicht möglich, beide Produktionsvorgänge zur gleichen Zeit auf der Maschine zu starten (Frustration); entartete Zustände sind die Folge.

Die spezifische Wärme (vgl. **Abbildung 4.12**) weist Gemeinsamkeiten mit derer des Spinmodells auf. So liegt das Hauptmaximum in gleichen Temperaturbereichen. Die Energie ist jedoch aufgrund der immer eingehaltenen Nebenbedingungen sehr viel geringer als beim Spinmodell. Auch die Durchlaufzeit ist hier immer kürzer als beim Spinmodell mit Nachbarschaft (2). Dies liegt an der Planung. Während beim Spinmodell der Startzeitpunkt eines Produktionsvorgangs frei gewählt werden kann, wird hier jeder Produktionsvorgang so früh wie möglich eingeplant. Die Durchlaufzeit ist damit bereits bei hohen Temperaturen am Anfang der Simulation sehr viel kleiner als beim Spinmodell und weist eine hohe Auslastung (Ordnungsparameter) bei hohen Temperaturen auf. Der Ordnungsparameter erreicht hier nicht die gleichen Werte wie beim Spinmodell bei kleinen Temperaturen. Am Ende der Simulation werden manche Produktionsvorgänge beim Spinmodell später eingeplant, ohne die Durchlaufzeit zu erhöhen und können damit im Vergleich eine höhere Auslastung erzielen. Die existierende Entartung ist an der, bei kleinen Temperaturen, ansteigenden Suszeptibilität zu sehen. Diese ist jedoch geringer als beim Spinmodell, da nur die Reihenfolge der Produktionsvorgänge auf einer Maschine entscheidend ist, nicht jedoch deren Startzeitpunkte.
Insgesamt gesehen weist auch diese Modellierung ein Spinglasverhalten auf, obwohl sie nicht im geringsten an Spinglas-Modelle erinnert.

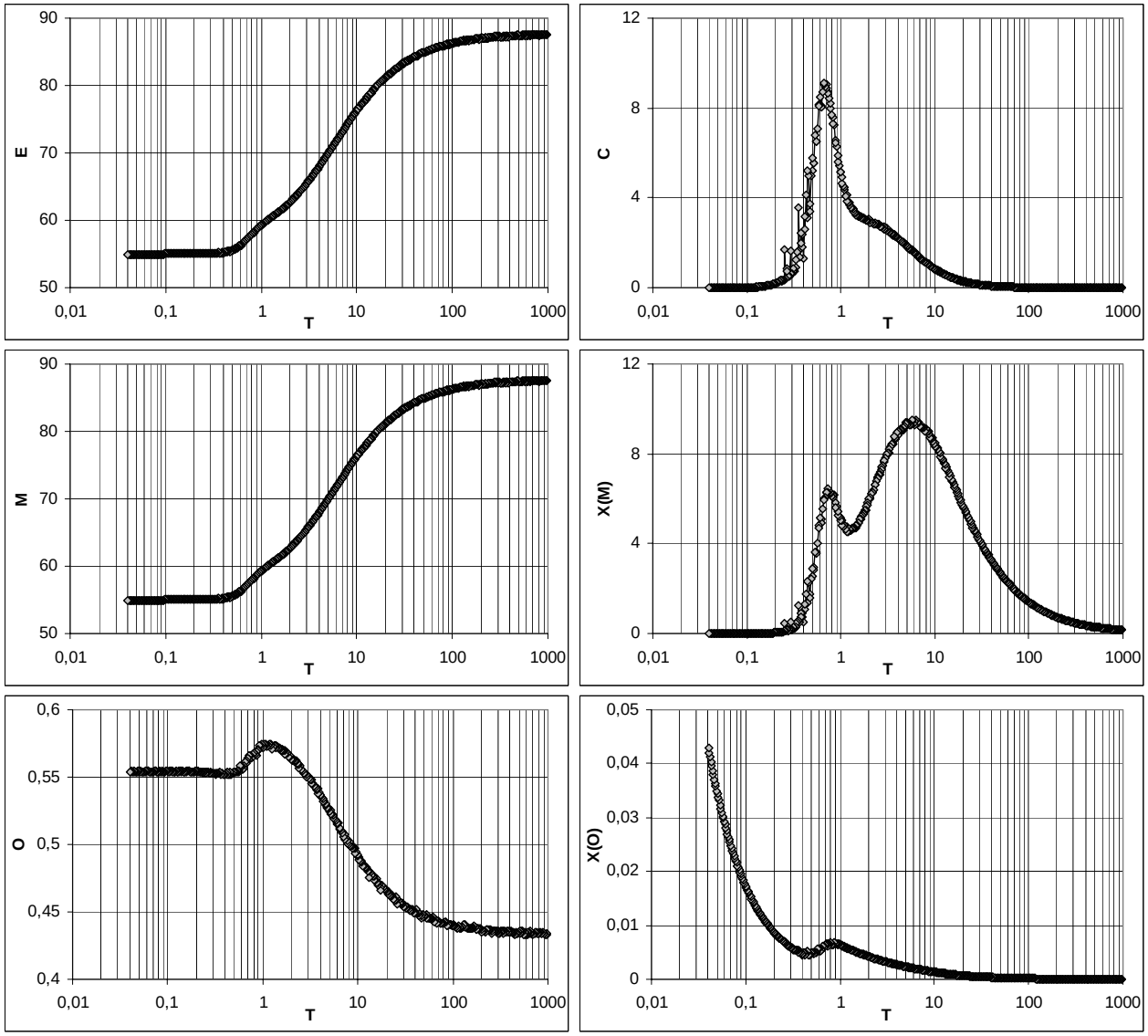


Abbildung 4.12: Neben der Energie (**E**) und der spezifischen Wärme (**C**) sind hier noch die Durchlaufzeit (**M**) und der Ordnungsparameter (**O**) zusammen mit deren Suszeptibilitäten (**X**) gegeben.

Der Suchraum besteht bei der Problemstellung **MT6** nun aus maximal $N=(6!)^6$ Lösungen (6 Maschinen mit jeweils 6 Produktionsvorgängen; vgl. **Anhang B**). Diese Anzahl wird durch die vorhandenen Reihenfolgebeziehungen innerhalb eines Auftrags eingeschränkt, da diese bei der vorliegenden Modellierung bei jeder Konfiguration bzw. Lösung eingehalten sind. Der gewählte Suchraum ist bei **JobShop**-Problemen *optimum connected* (vgl. **Kapitel 3.2.6.4**). Im Vergleich zum Suchraum der Spinglasmodellierung mit Nachbarschaft (2) sind nun neben den Konfigurationen, die eine der Nebenbedingungen S_2 oder S_4 verletzen auch gültige Lösungen entfernt worden, die nicht semiaktiv geplant sind.

In den folgenden Kapiteln wird ausschließlich mit diesem Modell gearbeitet, um größere Optimierungsprobleme mit mehr Funktionalität zu bearbeiten. Hier soll bereits auf **Kapitel 8** vorgegriffen werden und eine Problemstellung (**Bench01**) optimiert werden, bei der neben der Durchlaufzeit noch Verspätungen gegenüber Lieferzeitpunkten und Rüstzeiten minimiert werden sollen. Die zu-

sätzlichen Optimierungsziele führen zu verstärkter Frustration im System, da nicht alle Ziele gleichzeitig optimiert werden können (vgl. **Kapitel 6.4.2** und **Abbildung 4.13**).

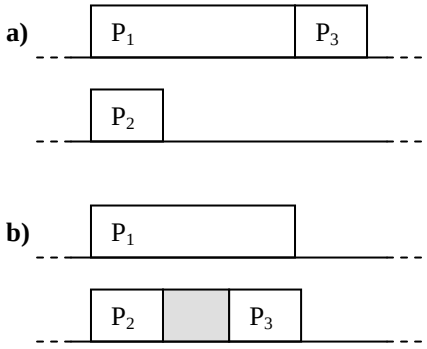


Abbildung 4.13: 3 Produktionsvorgänge sind auf 2 Maschinen zu planen. Zwischen P_2 und P_3 ist eine Rüstzeit nötig. Die beiden Lösungen **a)** und **b)** unterscheiden sich zwar in der Durchlaufzeit und der Rüstzeit, haben jedoch die gleichen Kosten, falls die Summe aus Durchlaufzeit und Rüstzeit minimiert werden soll.

Bei dieser Problemstellung (**Bench01**) sollten die Lieferzeitpunkte eingehalten werden und die Summe aus Durchlaufzeit und Rüstzeit minimiert werden. Als Gewichte werden bei der Zielfunktion 10,5,5 für **D**elay, **M**akespan, **S**etup gewählt. In **Abbildung 4.14** sind die Ergebnisse einer Simulation zu sehen. An den Maxima der Suszeptibilitäten erkennt man, dass die Gewichte richtig eingestellt sind. Während die Verspätung hinsichtlich der Lieferzeitpunkte (**D**) bereits bei höheren Temperaturen minimiert wird, werden die Durchlaufzeit (**M**) und die Rüstzeit (**R**) bei niedrigen Temperaturen reduziert. Die Maxima der Suszeptibilitäten **X(M)** und **X(S)** liegen dabei in etwa im gleichen Temperaturbereich und deuten somit auf eine gleichzeitige Optimierung der Kriterien Durchlaufzeit (**M**) und Rüstzeit (**S**) hin. Der Ordnungsparameter beginnt auch bei dieser Problemstellung bereits bei Werten über 50%. Allerdings wird am Ende der Simulation eine Lösung gefunden, in deren Nachbarschaft nur schlechtere Lösungen existieren. Es kommt hier zu einem Einfrieren in einem Tal, bestehend aus einer einzigen Lösung. Die Suszeptibilität des Ordnungsparameters nimmt an diesem Punkt schlagartig ab.

Das zu produzierende Auftragspaket besteht aus 10 verschiedenen Endprodukten, wobei insgesamt 25 Aufträge mit Lieferzeitpunkten existieren. Im Folgenden wird nun untersucht, inwieweit sich die einzelnen Observablen verändern, falls nur ein Auftragspaket bestehend aus 10 Aufträgen, wobei für jedes Endprodukt jeweils ein Auftrag existiert, simuliert wird (vgl. **Abbildung 4.15**). Zusätzlich wird die Anzahl der Sweeps pro Temperaturschritt gesenkt, so dass die Ergebnisse nach sehr kurzer Rechenzeit vorliegen. Insgesamt gesehen ergibt sich eine gute Übereinstimmung des Verhaltens der einzelnen Observablen. Eine Übereinstimmung in den Werten existiert nicht, da verschiedene Auftragspakete existieren. Auch kommt es aufgrund der reduzierten Rechenzeit zu einer größeren Streuung der einzelnen Messpunkte. Jedoch lässt sich ein wichtiger Punkt festhalten. Die Maxima der spezifischen Wärme und der Suszeptibilitäten verschieben sich nicht, obwohl ein anderes Auftragspaket simuliert wurde. Bei weiteren Simulationen zu anderen Auftragspaketen dieser Problemstellung ergibt sich das gleiche Resultat. Es kann somit davon ausgegangen werden, dass einmal definierte Parameter, die für die Simulation mit **SA** nötig sind, z.B. die Starttemperatur und das Abkühlschema oder die Gewichte der einzelnen Optimierungskriterien, beibehalten werden können,

solange nur der Mix an zu optimierenden Aufträgen verändert wird. Dies erleichtert den täglichen Einsatz dieses Optimierungsverfahren im Bereich der Produktionsplanung erheblich und ist meist eine wichtige Eigenschaft, die von einem Optimierungsverfahren von Produktionsplanern gefordert wird. Eine Anpassung der Parameter wird somit erst dann nötig, falls an der Produktionsstruktur Veränderungen vorgenommen werden, was in der Realität nicht allzu häufig der Fall sein sollte. Ein zusätzlicher Vorteil ergibt sich für die erste Auswahl der Parameter. So kann auf einem reduzierten Satz an Aufträgen gearbeitet werden und bei kurzen Rechenzeiten die Parameter bestimmt werden. Eine Überprüfung der Parameter kann bei der täglichen Planung regelmäßig erfolgen und evtl. nötige Anpassungen können vorgenommen werden.
Anhand der spezifischen Wärme und der Suszeptibilitäten lässt sich auch ein guter Wert für die Starttemperatur bestimmen. Dieser liegt bei dieser Problemstellung bei etwa 1000. Abgekühlt werden sollte bis auf Werte unter 5. Die Wahl der Starttemperatur ist dabei nicht besonders kritisch. Sie könnte in diesem Fall (**Bench01**) Werte zwischen 500 und 2000 aufweisen ohne die Simulationsergebnisse stark zu verändern. In **Kapitel 8** wird die Starttemperatur auf 2000 gesetzt, da dies eine günstige Einstellung zu sein scheint, betrachtet man sich die Simulationsergebnisse bei kurzen Rechenzeiten (**Abbildung 4.15**). Ein zu hoher Wert hat dabei nur den kleinen Nachteil, etwas zuviel Rechenzeit zu investieren. Eine Wahl einer zu kleinen Starttemperatur würde sich negativ auf die Ergebnisse auswirken, da das System dann evtl. zu schnell in einem ‚schlechten' lokalen Optimum einfriert. Während der täglichen Planung könnten jedoch die Ergebnisse aus Simulationen mit längeren Rechenzeiten verwendet werden, um eine optimalere Einstellung für die Starttemperatur der nächsten Planung zu finden, um so etwas Rechenzeit (in diesem Fall etwa 10-20%) einzusparen. Aus der Faustformel **(2.24)** aus **Kapitel 2.4.4** würde sich bei dieser Problemstellung eine Starttemperatur von 12000-15000 ergeben. Diese ist jedoch zu groß und man würde zuviel Rechenzeit bei zu hohen Temperaturen verlieren, bei denen es sich beinahe zu eine Zufallssuche handelt. Die Bestimmung der Starttemperatur aus der spezifischen Wärme ist sehr viel genauer. Dies liegt in diesem Fall wohl an den hohen Strafkosten für die Verspätungen. Bei einer Starttemperatur >10000 würde noch das Maximum der Suszeptibilität der Verspätungen **X(D)** zu sehen sein. Dies ist jedoch nicht unbedingt nötig. Wichtig ist nur, dass **X(D)** im gewählten Temperaturbereich nicht verschwindet, da dies bedeuten würde, dass sich das Optimierungsverfahren bzgl. der Verspätungen wie ein Greedy-Verfahren verhalten würde. Reduziert man das Gewicht der Verspätungen, so verschiebt sich das Maximum der Suszeptibilität **X(D)** zu niedrigeren Temperaturen und die Faustformel **(2.24)** ergibt eine ähnliche Starttemperatur.

In den folgenden Kapiteln werden einzelne Problemstellungen betrachtet und verschiedene Verfahren miteinander bzgl. benötigter Rechenzeit und Lösungsqualität verglichen. Die hier festgestellten Ergebnisse stellen die Begründung für die Wahl eines bestimmten Temperaturschemas für alle Instanzen einer Problemstellung dar.

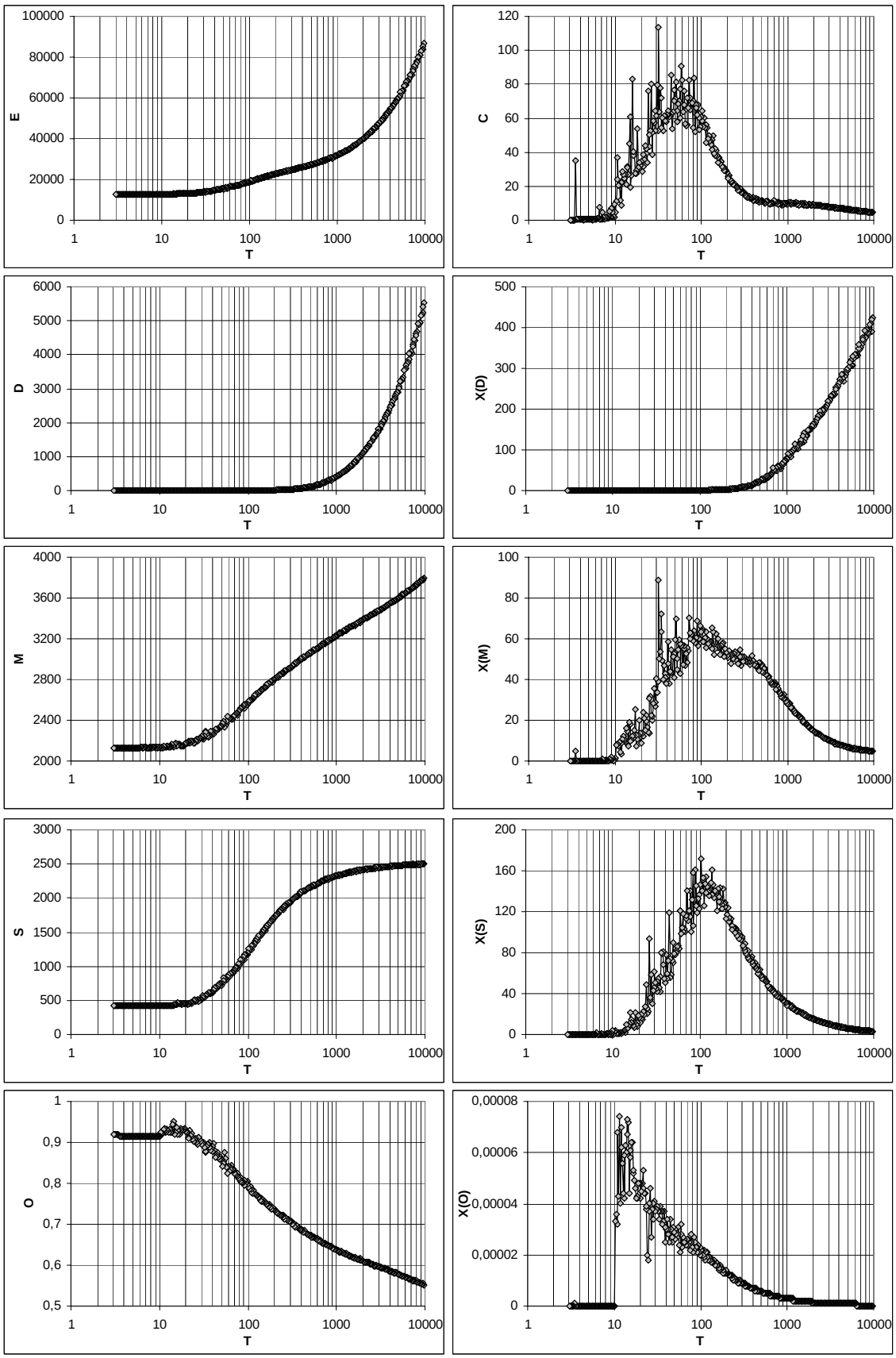


Abbildung 4.14: Die Energie (**E**) der Simulation der Problemstellung Bench01 setzt sich zusammen aus den Verspätungen (**D**), der Durchlaufzeit (**M**) und der Rüstzeit (**S**). Neben diesen Größen sind noch der Ordnungsparameter (**O**), die spezifische Wärme (**C**) und die Suszeptibilitäten (**X**) gegen die Temperatur abgebildet.

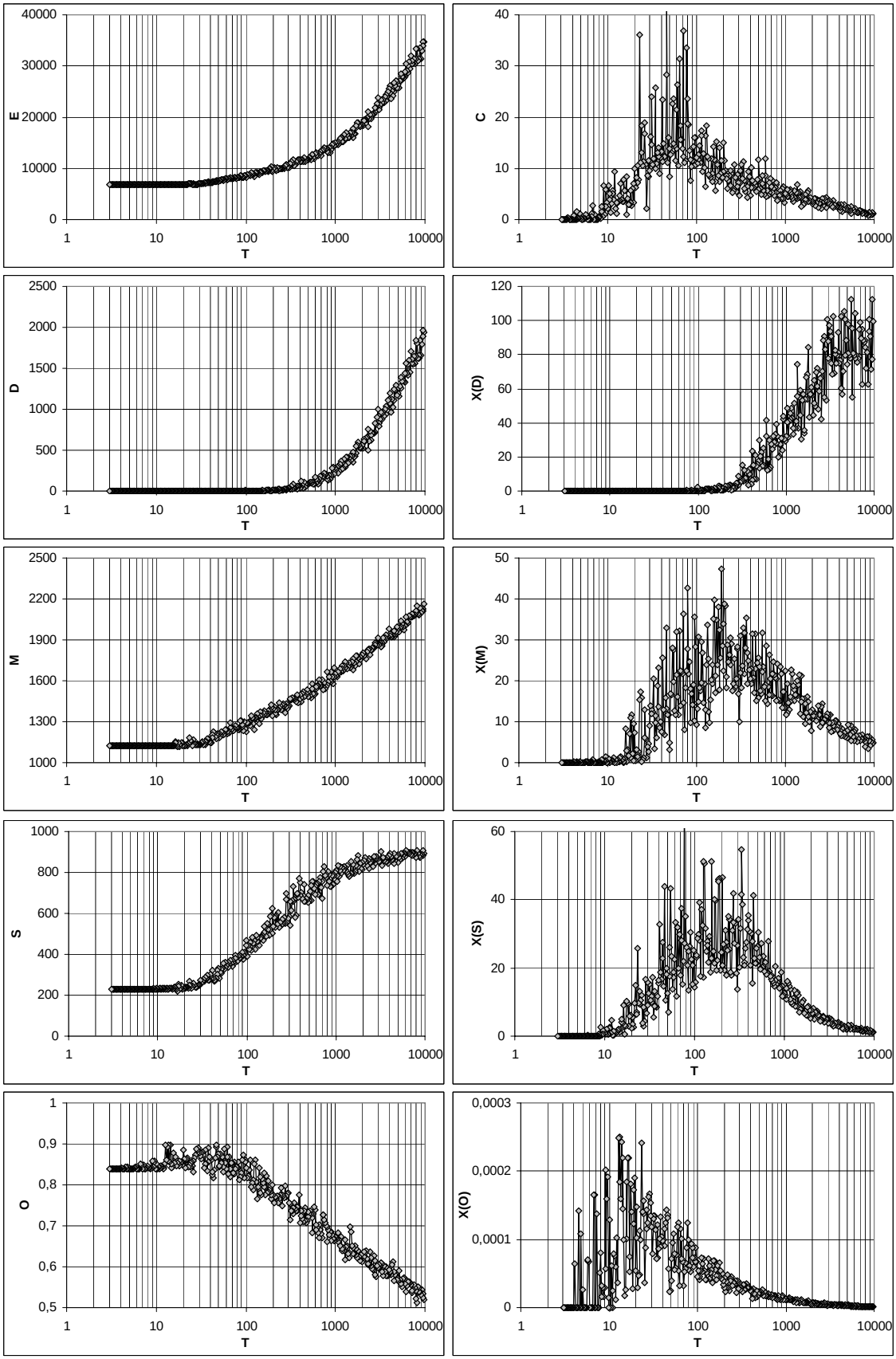


Abbildung 4.15: Zu sehen sind die Observablen der Simulation einer vereinfachten Instanz der Problemstellung Bench01. Die Energie (**E**) setzt sich zusammen aus den Verspätungen (**D**), der Durchlaufzeit (**M**) und der Rüstzeit (**S**). Neben diesen Größen sind noch der Ordnungsparameter (**O**), die spezifische Wärme (**C**) und die Suszeptibilitäten (**X**) gegen die Temperatur abgebildet.

4.3 Zusammenfassung

In diesem Kapitel wurden zwei Spinglasmodelle vorgestellt und deren Observablen mit denen der beiden Modelle der Produktionsplanung verglichen. Das übereinstimmende Verhalten der Observablen ist die Begründung für den Einsatz der statistischen Verfahren, die im Bereich der Festkörperphysik bereits erfolgreich eingesetzt wurden, im Bereich der Produktionsplanung. Bereits bei den vereinfachten Modellen des Spinglases zeigt sich, dass die Wahl der richtigen Nachbarschaft bei begrenzter Rechenzeit die Ergebnisse verbessern kann. Der einfache SingleSpinFlip war nicht in der Lage, alle von Prof. Kobe erstellten Instanzen zum SK-Modell optimal zu lösen. Mit einer speziellen Erweiterung der Nachbarschaft war dies jedoch möglich.

Zur Optimierung eines speziellen Produktionsplanungsproblems, dem **JobShop**, wurden zwei unterschiedliche Modelle miteinander verglichen. Eines basiert auf einer Spindarstellung, das andere auf einer objektorientierten direkten Darstellung. Obwohl beide Modelle sehr unterschiedlich sind, weisen beide Spinglasverhalten auf. Es kommt auch bei der Simulation von Produktionsplanungsproblemen zu einem Übergang von einem ungeordneten zu einem geordneten Zustand beim Senken der Temperatur. Die richtige Wahl der benötigten Parameter kann durch die Betrachtung der spezifischen Wärme und der Suszeptibilitäten überprüft werden. Das Spinmodell hat den Vorteil, neue Restriktionen leicht zu integrieren. Die Nachteile überwiegen jedoch. Diese liegen in der Anzahl der benötigten Parameter, so ist für jede Straffunktion ein eigener Faktor nötig, der dafür sorgt, dass das System am Ende der Optimierung in einem Zustand ohne Restriktionsverletzung einfriert, und der Größe des Suchraums. Der Suchraum ist beim Spinmodell trotz Wahl der Nachbarschaft (2) sehr viel größer als beim direkten Modell und der Hauptgrund für die, im Vergleich zur direkten Modellierung, schlechten Ergebnisse bei gleicher Rechenzeit. Der Erfolg der Optimierung beruht somit nicht auf dem Algorithmus alleine. Eine günstige Modellierung der Problemstellung ist zur erfolgreichen Lösung der Probleme der Produktionsplanung essentiell.

Kapitel 5

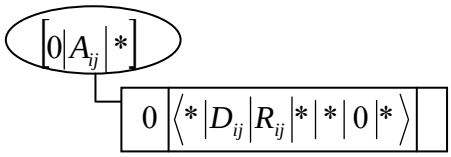
Das MT10x10-Problem

5.1 Problemstellung

Das MT10x10-Problem aus **[5.1]** ist ein **JobShop-Problem**. Es besteht aus 10 Aufträgen und 10 Ressourcen. Jeder Auftrag belegt die Ressourcen in einer bestimmten Reihenfolge. Das Ziel bei der Optimierung liegt in der Minimierung der gesamten Durchlaufzeit.

5.2 Modell

Es existieren 10 Primärressourcen. Jeder Auftrag i ($1 \le i \le 10$) besteht aus jeweils 10 Aktivitäten A_{ij} ($1 \le j \le 10$):



Zwischen den einzelnen Aktivitäten des Auftrags i werden Aktivitätslinks eingefügt, um die Ausführungsreihenfolge herzustellen.

$A_{ij}(1) \xrightarrow{ES,0} A_{ij+1}(1)$ mit $1 \le j \le 9$.

5.3 Ergebnisse

Für diese Instanz wurde bewiesen, dass eine optimale Lösung eine Durchlaufzeit von 930 besitzt.
Optimiert wird nur die Durchlaufzeit mit dem Gewicht 1. Die Energie entspricht somit der Durchlaufzeit. Die Temperatur wird dabei von 500, in 300 Schritten, mit einem logarithmischen Abkühlschema, mit einem α- Faktor von 0.98, gesenkt. Bei den Simulationen werden jeweils 100 Durchläufe mit **SA** und **TA** bei 100, 500, 1000 und 5000 Sweeps pro Temperaturschritt durchgeführt. Bei diesem Datensatz ist es möglich, auf einem Pentium mit 650MHz 100000 Lösungen in 8 Sekunden zu erzeugen und zu bewerten. Bei den Moves werden die folgenden mit jeweils gleicher Auswahlwahrscheinlichkeit verwendet: Push1, Swap, Lin2Opt.
Ein optimaler Plan mit einer Durchlaufzeit von 930 kann bei 5000 Sweeps pro Temperaturschritt (105 Sekunden) mit beiden Verfahren (**SA**, **TA**) gefunden werden.

In **Abbildung 5.1** ist der Prozentsatz der Lösungen zu sehen, deren Durchlaufzeit kleiner oder gleich der angegebenen ist. Gezeigt werden die Lösungskurven für verschiedene Anzahl an Sweeps.

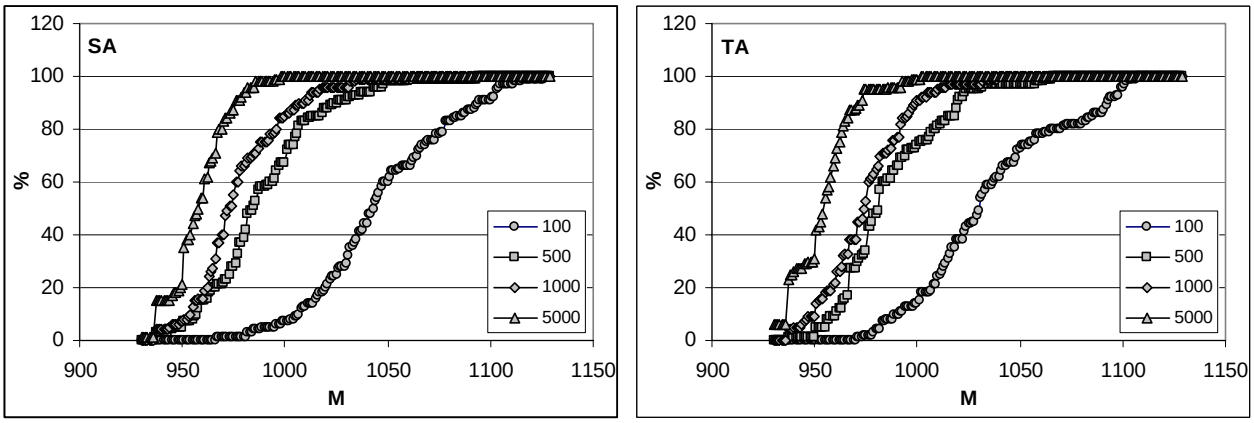


Abbildung 5.1: Es wurden jeweils 100 Simulationen mit den Verfahren **SA** und **TA** für verschiedene Anzahl Sweeps pro Temperaturschritt (100, 500, 1000, 5000) durchgeführt. Zu sehen ist jeweils der Prozentsatz (**%**) der Lösungen, die eine Durchlaufzeit $\leq$**M** haben.

Folgende Schlussfolgerungen sind anhand der Graphik für **SA** und **TA** möglich:

- Bei einer Erhöhung der Rechenzeit werden die durchschnittlichen Ergebnisse für die Durchlaufzeit besser.
- Die Wahrscheinlichkeit, eine optimale Lösung zu finden, steigt mit Erhöhung der Rechenzeit.
- Mehr Rechenzeit führt zu stabileren Ergebnissen, d.h. die Abweichung von der durchschnittlichen Durchlaufzeit nimmt ab, deutlich zu sehen am steileren Verlauf der Kurven.
- Ein merklicher Unterschied zwischen **SA** und **TA** bezüglich der Güte und der Stabilität der Lösungen ist nicht festzustellen.
- Auch bei sehr großen Rechenzeiten ist es nicht möglich, die optimale Durchlaufzeit mit 100% Wahrscheinlichkeit zu finden. Unter sehr großen Rechenzeiten wird in diesem Fall nur von endlichen, für dieses Problem vertretbaren, Rechenzeiten gesprochen. **SA** findet bei unendlich langer Rechenzeit mit Sicherheit eine optimale Lösung. Jedoch lässt sich unter dieser Voraussetzung auch mit einer Zufallssuche oder dem einfachen Absuchen des gesamten Lösungsraums eine optimale Lösung finden.

Im Folgenden (**Abbildung 5.2**) wird die Zeit für die Optimierung auf mehrere Durchläufe verteilt und dann die beste Lösung übernommen. So werden 1000 (500) Durchläufe mit 500 (1000) Sweeps pro Temperaturschritt durchgeführt und die jeweils beste Lösung von 10 (5) aufeinanderfolgenden übernommen. Die Güte und Stabilität der übernommenen Lösungen wird mit dem Ergebnis für 5000 Sweeps verglichen.

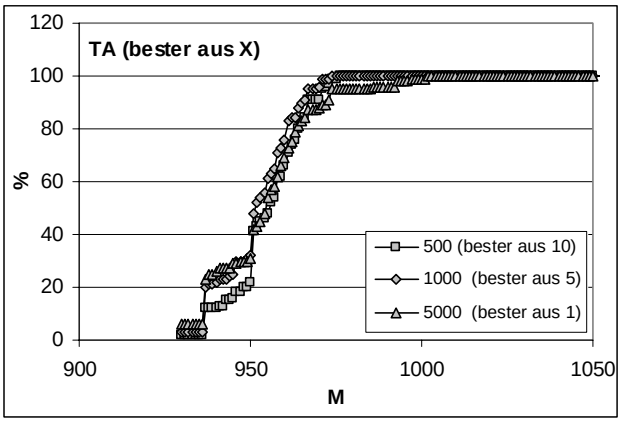


Abbildung 5.2: Simulation mit **TA**. Es wurden Simulationen mit verschiedener Anzahl Sweeps pro Temperaturschritt (500, 1000, 5000) durchgeführt. Bei den Simulationen für 500 (bzw. 1000) Sweeps wurden dabei von jeweils 10 (bzw. 5) aufeinanderfolgenden Lösungen die besten genommen, so dass jede übernommene Lösung etwa 100 Sekunden Rechenzeit zur Verfügung hatte. Zu sehen ist jeweils der Prozentsatz (**%**) aus jeweils 100 übernommenen Lösungen, die eine Durchlaufzeit $\leq$**M** haben.

Es zeigt sich, dass es keinen deutlichen Unterschied in Bezug auf die Güte und Stabilität der Lösungen gibt. Die Aufteilung der Rechenzeit auf mehrere Durchgänge hat jedoch den Vorteil, dass man nach einem Bruchteil der Rechenzeit bereits die Durchlaufzeit abschätzen kann.

Bei dem MT10x10 –Problem existieren mehrere Lösungen mit einer Durchlaufzeit von 930. Einer der optimalen Pläne ist in **Abbildung 5.3** zu sehen.

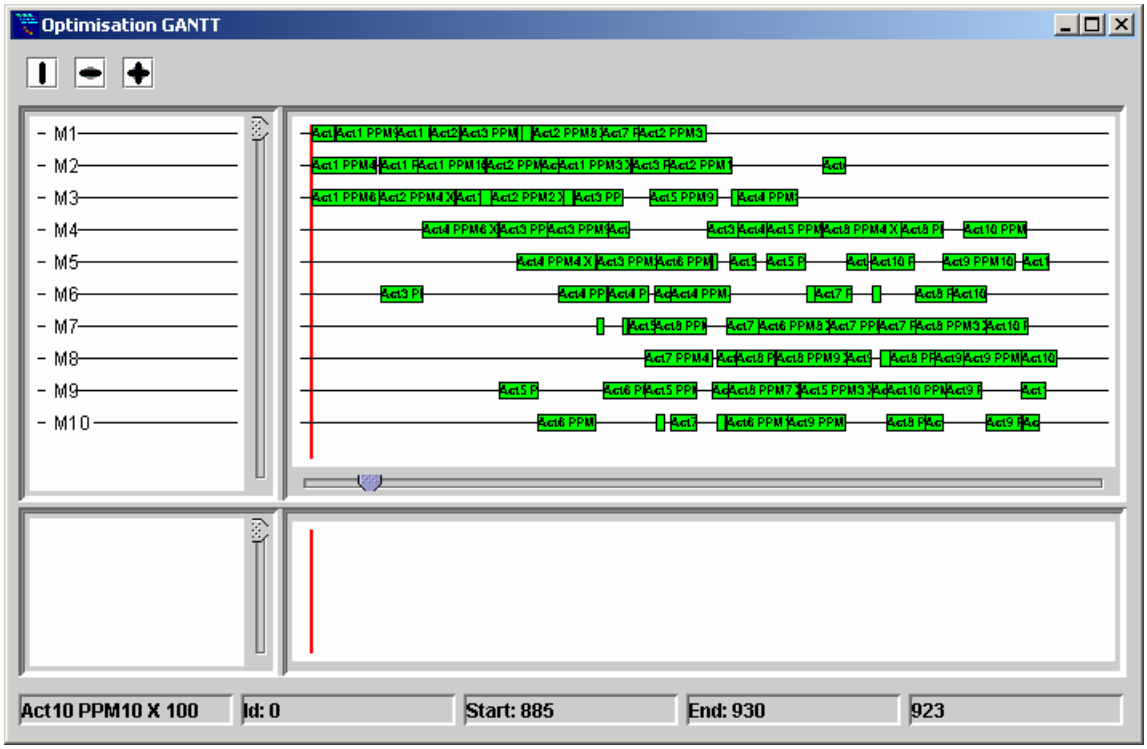


Abbildung 5.3: Ein optimaler Plan des MT10 Problems mit einer Durchlaufzeit von 930.

5.4 Erweiterung

Statt der 10 Aufträge sollen nun n*10 Aufträge optimiert werden. Dazu wird jeder Auftrag n-mal kopiert.

Untere Grenze:

Eine untere Grenze für die Durchlaufzeit D_{min}, die möglicherweise nicht erreichbar ist, kann bei diesem Problem bestimmt werden durch:

1. die Durchlaufzeit jedes einzelnen Auftrags

   $$D_{\min} = MIN_i\left(\sum_{j=1}^{10} D_{ij}\right), (1 \le i \le 10)$$

2. die Auslastung der Ressourcen

   $$D_{\min} = MIN_r\left(T_r^{\,S} + T_r^{\,D} + T_r^{\,E}\right), (1 \le r \le 10),$$

   wobei für die Primärressource r $T_r^{\,S}$ den frühestmöglichen Beginn aller Vorgänge, $T_r^{\,D}$ die Auslastung und $T_r^{\,E}$ die Mindestdauer zum Beenden der Vorgänge, die Nachfolger eines geplanten Vorganges sind, darstellt.

   $$T_r^{\,S} = MIN_i\left(\sum_{j=1}^{j'-1} D_{ij}\right), (1 \le i \le 10),$$ wobei A_{ij} die Primärressource r belegt.

   $$T_r^{\,D} = \sum_{i=1}^{10} D_{ij(i)}$$, wobei $A_{ij(i)}$ die Primärressource r belegt.

   $$T_r^{\,E} = MIN_i\left(\sum_{j=j'+1}^{10} D_{ij}\right), (1 \le i \le 10),$$ wobei A_{ij} die Primärressource r belegt.

Bei dieser Problemstellung zeigt sich, dass durch 1. die untere Grenze nicht bestimmt wird, da die Durchlaufzeiten aller Aufträge kleiner als die durch 2. bestimmte untere Grenze sind. Aus 2. erhält man für die untere Grenze des Problems mit 10 Aufträgen D_{min} =796 (=0+556+240, Primärressource 3 bestimmt hier die untere Grenze). Diese untere Grenze liegt noch weit entfernt von der Durchlaufzeit eines optimalen Plans (930). Auch bei 20 Aufträgen bestimmt Primärressource 3 die untere Grenze D_{min} =1352 (=0+2*556+240). Ab 30 Aufträgen hängt die untere Grenze jedoch von Primärressource 4 ab. D_{min} =83+n*631+0 für n*10 Aufträge mit $n \ge 3$.

Obere Grenze:

Als erreichbare obere Grenze für die Durchlaufzeit kann die optimale Lösung für das Originalproblem (930) herangezogen werden: D_{max} =n*930. Diese sollte auf jeden Fall erreicht werden, da es sich ja nur um eine Aneinanderreihung von einzelnen optimalen Plänen handelt.

Betrachtet man sich den optimalen Plan etwas genauer, so erkennt man, dass eigentlich keine Primärressource von Anfang (0) bis zum Ende (930) belegt wird. Ein Ineinanderschieben der einzelnen optimalen Pläne ist somit möglich. Dadurch wird die Durchlaufzeit pro angehängtem optimalen Einzelplan nur um 756 erhöht (statt um 930). Als erreichbare obere Grenze für die Durchlaufzeit ergibt sich dann: D_{max} =n*756+174. Analog dazu kann man die erhaltenen Pläne für n>1 aneinander reihen, um die obere Grenze weiter zu senken.

In **Tabelle 5.1** sind die Lösungen der Simulationen mit jeweils 5000 Sweeps pro Temperaturschritt für die Problemstellungen mit n*10 Aufträgen zu sehen.

n	Untere Grenze	Durchlaufzeit
1	796	(*)930
2	1352	1481
3	1976	2066
4	2608	2669
>4	n*631+83	(*)n*631+83

Tabelle 5.1: Die Tabelle zeigt die erzeugten Lösungen für die Problemstellung mit n*10 Aufträgen. Die mit (*) gekennzeichneten Lösungen sind optimal, da sie entweder der unteren Grenze entsprechen oder der optimalen Lösung des MT10X10 (930).

Bei den Simulationen kann für n*10 Aufträge mit n>4 immer ein Plan erzeugt werden, dessen Durchlaufzeit der unteren Grenze entspricht. Ein solcher Plan ist optimal. Die Suche nach einem optimalen Plan scheint für n>4 immer leichter zu werden, da dazu 5000 Sweeps pro Temperaturschritt ausreichend sind. Die Rechenzeit steigt damit bei der Lösungssuche nur linear an. Ein Grund dafür könnte in der Primärressource 4 liegen. Diese wird ab 30 Aufträgen zum Engpass (*bottleneck)*. Mit wachsenden n verstärkt sich dieser Effekt.

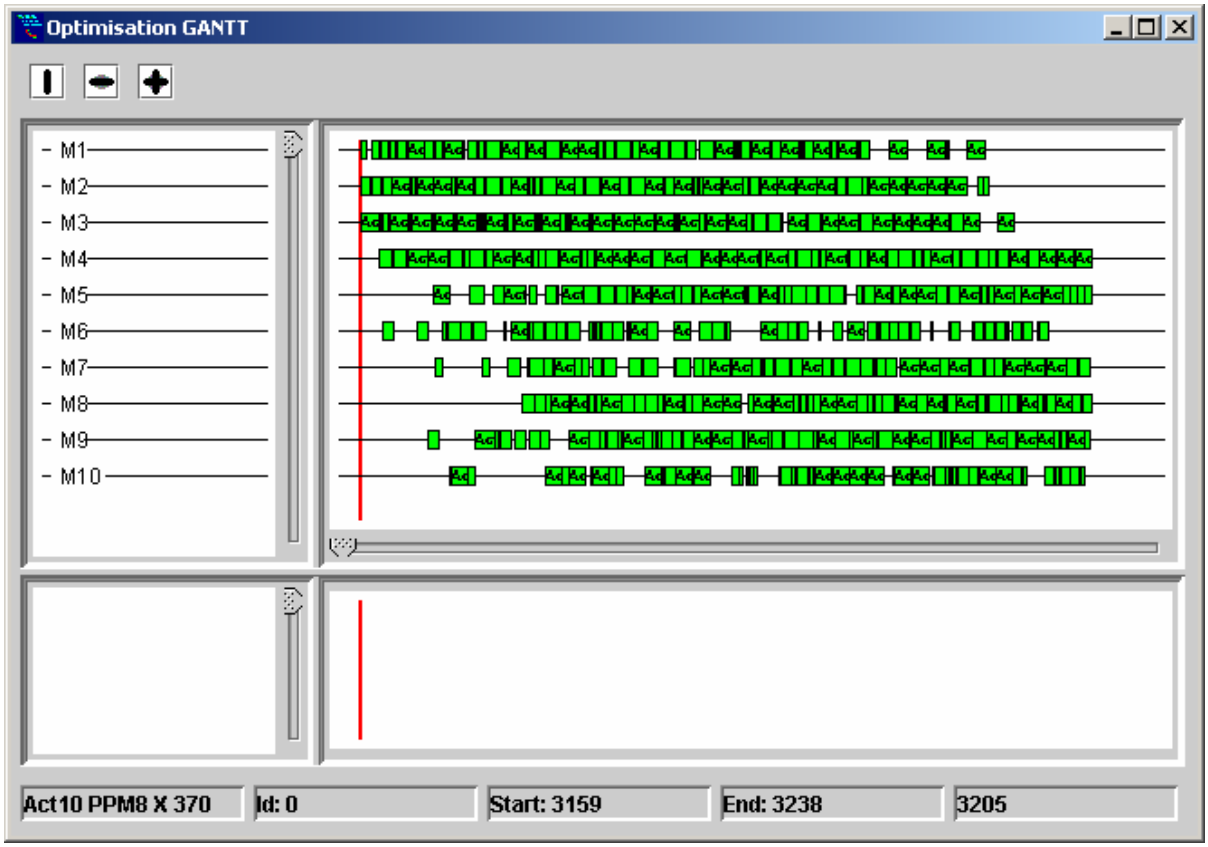


Abbildung 5.4: Ein Plan für 50 Aufträge mit einer Durchlaufzeit von 3238. Die *bottleneck* Ressource M4 ist ab einem bestimmten Zeitpunkt (83) zu 100% ausgelastet.

Bei dem optimalen Plan (**Abbildung 5.4**) kann man die vollständige Belegung der Maschine 4 von 83 bis zur gesamten Durchlaufzeit von 3238 erkennen. Diese optimalen Pläne sind kombinierbar, d.h. durch Kopieren und Ineinanderschieben des optimalen Plans für 50 Aufträge erhält man den optimalen Plan für 100 Aufträge. So lassen sich aus den optimalen Plänen für n*10 Aufträge mit n=5,...,9 alle optimalen Pläne für n'*10 Aufträge mit n'>4 erzeugen.

5.5 Weitere JobShop-Probleme

Im Folgenden werden Ergebnisse zu 40 Instanzen gegeben, die von Lawrence erstellt wurden. Zu finden sind die Instanzen auf der Seite der OR Bibliothek **[5.2]**.
Bei der Bearbeitung der verschiedenen Instanzen wird ein festes Temperaturschema verwendet. Abgekühlt wird logarithmisch in 500 Schritten von einem Anfangswert 2000 mit einem α- Faktor von 0.98. Dieser weite Temperaturbereich wird gewählt, um den für die Optimierung einer Instanz kritischen Temperaturbereich auf jeden Fall zu erfassen. Die Simulationen können deshalb beträchtlich beschleunigt werden, falls bei jeder Instanz ein eigenes Temperaturschema verwendet wird. Die Rechenzeit wird durch die Anzahl der Sweeps (1000 und 5000) pro Temperaturschritt begrenzt. Bei den Moves werden die folgenden mit jeweils gleicher Auswahlwahrscheinlichkeit verwendet: Push1, Swap, Lin2Opt.
In **Tabelle 5.3** sind die durchschnittlichen Ergebnisse für die Durchlaufzeit und die Standardabweichung aus jeweils 10 Simulationen pro Instanz gegeben. Die für jeweils 100000 Sweeps benötigte Rechenzeit der einzelnen Instanzen ist in **Tabelle 5.2** aufgelistet. Neben den originalen Instanzen werden noch gespiegelte bearbeitet. Bei diesen gespiegelten Instanzen ist die Reihenfolge der Aktivitäten innerhalb eines Auftrags gedreht.

Instanz	t
10x5	4
15X5	6
20X5	8
10x10	8
15x10	12
20x10	16
30x10	32
15x15	17

Tabelle 5.2: Für 100000 Sweeps benötigte Rechenzeit **t** in Sekunden für verschiedene JobShop **Instanzen** auf einem Pentium mit 650MHz.

Instanz	Optimum	1000 av(dev)	5000 av(dev)	5000 i av(dev)
1	666	666,0 (0,0)	666,0 (0,0)	666,0 (0,0)
2	655	661,4 (5,4)	655,0 (0,0)	655,0 (0,0)
3	597	611,3 (7,0)	603,4 (6,4)	604,4 (5,9)
4	590	594,5 (3,2)	590,0 (0,0)	590,0 (0,0)
5	593	593,0 (0,0)	593,0 (0,0)	593,0 (0,0)
6	926	926,0 (0,0)	926,0 (0,0)	926,0 (0,0)
7	890	890,0 (0,0)	890,0 (0,0)	890,0 (0,0)
8	863	863,0 (0,0)	863,0 (0,0)	863,0 (0,0)
9	951	951,0 (0,0)	951,0 (0,0)	951,0 (0,0)
10	958	958,0 (0,0)	958,0 (0,0)	958,0 (0,0)
11	1222	1.222,0 (0,0)	1.222,0 (0,0)	1.222,0 (0,0)
12	1039	1.039,0 (0,0)	1.039,0 (0,0)	1.039,0 (0,0)
13	1150	1.150,0 (0,0)	1.150,0 (0,0)	1.150,0 (0,0)
14	1292	1.292,0 (0,0)	1.292,0 (0,0)	1.292,0 (0,0)
15	1207	1.207,0 (0,0)	1.207,0 (0,0)	1.207,0 (0,0)
16	945	982,4 (14,1)	976,5 (8,8)	958,0 (16,0)
17	784	793,1 (3,1)	786,7 (2,6)	786,7 (3,0)
18	848	863,0 (11,0)	858,3 (4,2)	856,9 (5,0)
19	842	868,4 (11,5)	862,3 (5,4)	857,8 (3,3)
20	902	914,5 (8,5)	913,9 (9,3)	909,1 (3,3)
21	1046	1.092,1 (16,3)	1.065,8 (8,2)	1.067,7 (9,1)
22	927	951,7 (10,9)	942,7 (5,0)	944,7 (10,9)
23	1032	1.033,3 (3,3)	1.032,0 (0,0)	1.032,0 (0,0)
24	935	974,5 (19,1)	963,8 (9,2)	958,5 (11,8)
25	977	1.017,2 (8,7)	1.000,3 (9,2)	998,0 (11,2)
26	1218	1.225,6 (16,5)	1.220,1 (4,3)	1.218,9 (2,1)
27	1235	1.285,2 (18,3)	1.268,8 (4,8)	1.265,0 (11,3)
28	1216	1.262,1 (26,6)	1.230,9 (9,3)	1.229,2 (7,2)
29	1130-1153	1.227,1 (15,3)	1.208,7 (21,6)	1.197,5 (21,7)
30	1355	1.362,8 (15,0)	1.355,0 (0,0)	1.355,0 (0,0)
31	1784	1.784,0 (0,0)	1.784,0 (0,0)	1.784,0 (0,0)
32	1850	1.850,0 (0,0)	1.850,0 (0,0)	1.850,0 (0,0)
33	1719	1.719,0 (0,0)	1.719,0 (0,0)	1.719,0 (0,0)
34	1721	1.721,0 (0,0)	1.721,0 (0,0)	1.721,0 (0,0)
35	1888	1.889,0 (3,0)	1.888,0 (0,0)	1.888,0 (0,0)
36	1268	1.350,4 (19,8)	1.316,8 (18,5)	1.318,4 (15,8)
37	1397	1.478,1 (23,2)	1.447,9 (12,6)	1.448,8 (14,4)
38	1196	1.278,3 (19,3)	1.255,9 (13,7)	1.250,4 (25,4)
39	1233	1.310,9 (30,7)	1.269,8 (26,6)	1.260,9 (13,1)
40	1222	1.271,1 (13,7)	1.248,9 (11,7)	1.257,7 (15,3)

Tabelle 5.3: Es sind die durchschnittlichen Ergebnisse (**av**) mit ihrer Standardabweichung (**dev**) aus jeweils 10 Simulationen mit **1000** und **5000** Sweeps pro Temperaturschritt für die 40 Instanzen von Lawrence zusammen mit deren **Optima** gegeben. Zusätzlich sind noch Lösungen gegeben, die mit gespiegelten Instanzen (**5000 i**) mit 5000 Sweeps pro Temperaturschritt erhalten wurden.

Die so erzielten durchschnittlichen Ergebnisse sind bei 5000 (1000) Sweeps pro Temperaturschritt im Durchschnitt nur um ca. 1,1% (2%) schlechter als die optimalen. Bei den einzelnen Instanzen liegt die größte Abweichung bei 5% (6,9%) bei 5000 (1000) Sweeps pro Temperaturschritt. 21 (16) der 40 Instanzen konnten bei jeder einzelnen Simulation mit 5000 (1000) Sweeps pro Temperaturschritt optimal gelöst werden, wobei manche der größeren Instanzen leichter zu lösen sind als die kleineren. Der Unterschied zwischen den Lösungen zu den originalen und den gespiegelten Instanzen ist vernachlässigbar. Die Ergebnisse sind bei den gespiegelten Instanzen bei 5000 Sweeps pro Temperaturschritt im Durchschnitt nur 1% schlechter als die optimalen Lösungen, wobei die größte Abweichung bei 4,5% liegt.

Ein umfassender Vergleich verschiedener anderer Verfahren auf den einzelnen Instanzen ist z.B. in **[5.3]** oder **[2.30]** zu finden.

5.6 Zusammenfassung

In diesem Kapitel wurden neben dem originalen **MT10** Problem noch 40 Instanzen von Lawrence und solche bearbeitet, die aus dem **MT10** durch n-faches Kopieren aller Aufträge entstanden sind. Es konnten nicht alle Instanzen bei allen Simulationen bei begrenzter Rechenzeit optimal gelöst werden, obwohl im Vergleich zu den folgenden Problemstellungen nur ein Bruchteil der durch das Modell aus **Kapitel 3.2** zur Verfügung stehenden Funktionalität nötig ist. Es ist nicht festzustellen, dass größere Instanzen grundsätzlich schwieriger zu lösen sind, so sind z.B. die aus **MT10** erzeugten größeren **JobShop** Instanzen relativ leicht zu lösen, da dort eine einzelne Maschine zum Engpass (*bottleneck*) wird.

Kapitel 6

Scheduling under Labour Resource Constraints (SLRC)

6.1 Problemstellung

Zur Herstellung eines Produktes müssen die einzelnen Arbeitschritte des Auftrags nacheinander auf einer bestimmten Ressource (Typ A) durchgeführt werden. Auf dieser Ressource (Typ A) darf dabei nicht mehr als ein Arbeitschritt gleichzeitig ausgeführt werden. Ein Auftrag darf bei der Ausführung nicht durch Arbeitschritte eines anderen Auftrags, die die gleiche Ressource (Typ A) benötigen, unterbrochen werden. Bei den Instanzen wird dies bereits abgefangen durch eine festgelegte Produktionsreihenfolge der Aufträge auf dieser Ressource (Typ A). Zusätzlich zu der Ressource (Typ A) benötigen die Arbeitsschritte noch Arbeitskräfte. Diese Arbeitskräfte werden von einer Ressource (Typ B) zur Verfügung gestellt. Die benötigte Anzahl an Arbeitskräften ist während der Ausführung nicht konstant. Bei jedem Arbeitsschritt ist ein Profil hinterlegt, das angibt, welche Anzahl an Arbeitskräften zu welchem Zeitpunkt benötigt wird. Ein Arbeitsschritt kann aus mehreren unterschiedlichen Vorgängen bestehen, die direkt nacheinander auszuführen sind, so z.B. Laden, Heizen, Mischen, Entladen. Jeder dieser Vorgänge dauert eine bestimmte Zeit und benötigt eine bestimmte Anzahl an Arbeitskräften. Bei den verwendeten Instanzen ist dabei jeder Arbeitsschritt eines Auftrags gleich, d.h. er hat das gleiche Profil für die benötigten Arbeitskräfte. Die Ressource (Typ B) besitzt eine bestimmte Kapazität (Anzahl Arbeitskräfte), die in diesen Instanzen konstant ist. Denkbar ist natürlich ein zeitabhängiges Kapazitätsprofil. Auf der Ressource (Typ B) können mehrere Arbeitschritte gleichzeitig ausgeführt werden. Die von den Arbeitschritten benötigte Anzahl an Arbeitskräften darf dabei die vorhandene Kapazität der Ressource (Typ B) zu keinem Zeitpunkt übertreffen. Bei der Produktion sind nun noch evtl. Vorgänger- Nachfolgerbeziehungen der Arbeitsschritte verschiedener Aufträge einzuhalten, z.B. falls zwei Aufträge die gleiche Ressource (Typ A) belegen oder falls ein Zwischenprodukt (nach einem Arbeitschritt entstanden) eines Auftrags bei einem anderen benötigt wird. Das Ziel der Optimierung besteht darin, die gesamte Durchlaufzeit zu minimieren.

6.2 Modell

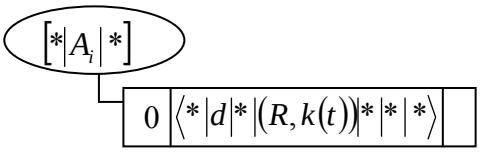
a) Ressourcen:

Auf die Abbildung der Ressource Typ A wird verzichtet, da die Aufträge, die diese Ressourcen gleichzeitig belegen, eine feste Reihenfolge einhalten müssen.

Die Ressource Typ B wird durch eine Sekundärressource *R* abgebildet. Es wird eine feste Kapazität vorgegeben: 18.

b) Aktivitäten:

Jeder Arbeitsschritt eines Auftrags wird durch eine Aktivität dargestellt. Die vom Modus von der Sekundärressource benötigte Kapazität ist zeitlich variabel und wird durch $k(t) = (k_1, k_2, ..., k_d)$ mit der Dauer des Modus d und den einzelnen benötigten Kapazitäten k_i pro Zeitpunkt angegeben.



c) Aktivitätslinks:

Die Vorgänger- Nachfolgerbeziehungen werden durch Aktivitätslinks in der folgenden Form dargestellt:

$A_V(1) \xrightarrow{ES,0} A_N(1)$

d) Beispiel

Im Folgenden ist das Modell der Instanz Ins_4o_24j_A gegeben. In diesem Datensatz existieren 4 Aufträge mit insgesamt 24 Arbeitsschritten.

Aktivitäten:

Tabelle 6.1 zeigt die 4 verschiedenen Aktivitätstypen (**Akt**), von denen jeweils **Num** Aktivitäten erzeugt werden. Zudem sind die Dauer (**d**) und das benötigte Kapazitätsprofil (**Profil**) (vgl. **Abbildung 6.1**) jedes Modus gegeben.

Akt	d	Num	Profil
A_{1i}	7	7	(12,12,6,2,2,2,2)
A_{2i}	9	4	(12,12,6,2,2,2,2,2,2)
A_{3i}	8	3	(12,12,3,3,3,3,3,3)
A_{4i}	5	10	(6,6,6,6,6)

Tabelle 6.1: Es werden 4 verschiedene Aktivitätstypen (**Akt**) benötigt, von denen jeweils **Num** Aktivitäten erzeugt werden. Jeder Aktivitätstyp ist gekennzeichnet durch seine Dauer (**d**) und das benötigte Kapazitätsprofil (**Profil**).

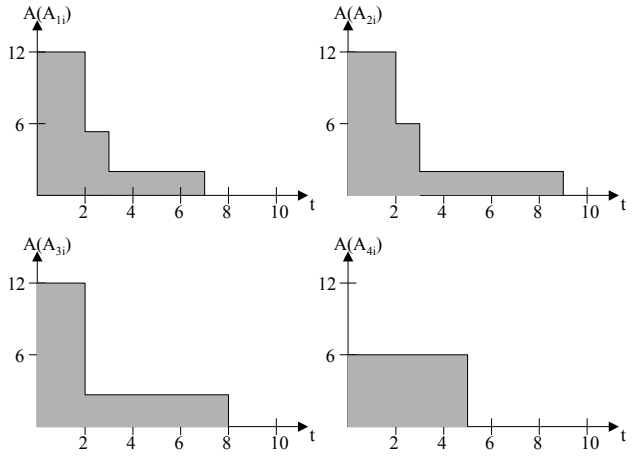


Abbildung 6.1: Profile der benötigten Arbeitskräfte der Aktivitäten A_{1i}, A_{2i}, A_{3i} und A_{4i}.

Aktivitätslinks:

Für die Vorgänger-Nachfolgerbeziehungen der Instanz (vgl. **Abbildung 6.2**) werden folgende Links benötigt:

$$A_{1i}(1) \xrightarrow{ES,0} A_{1i+1}(1) \quad (0 \le i \le 5)$$

$$A_{2i}(1) \xrightarrow{ES,0} A_{2i+1}(1) \quad (0 \le i \le 2)$$

$$A_{3i}(1) \xrightarrow{ES,0} A_{3i+1}(1) \quad (0 \le i \le 1)$$

$$A_{4i}(1) \xrightarrow{ES,0} A_{4i+1}(1) \quad (0 \le i \le 8)$$

$$A_{30}(1) \xrightarrow{ES,0} A_{40}(1)$$

$$A_{31}(1) \xrightarrow{ES,0} A_{43}(1)$$

$$A_{32}(1) \xrightarrow{ES,0} A_{46}(1)$$

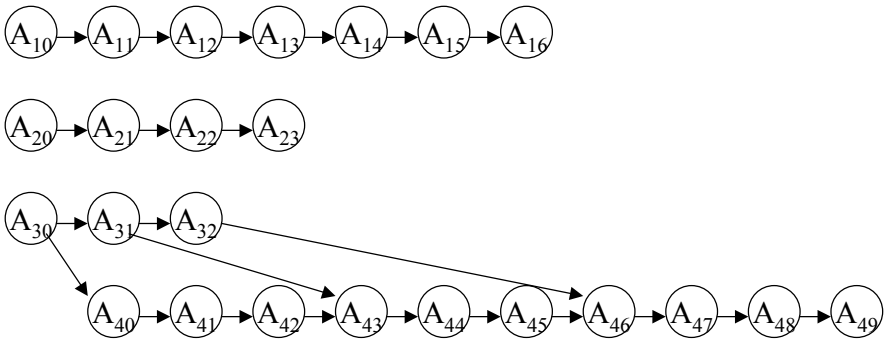


Abbildung 6.2: Vorgänger-Nachfolgerbeziehung der Aktivitäten A_{1i}, A_{2i}, A_{3i} und A_{4i}.

6.3 Ergebnisse

Bei den Instanzen gibt der Name die Anzahl der enthaltenen Aufträge und Arbeitsschritte an. So bedeutet 4o_24j_A, dass 4 Aufträge und insgesamt 24 Arbeitsschritte zu planen sind.
Bei dem Datensatz 4o_24j_A handelt es sich um einen Datensatz des Projekts PAMIPS. Die Instanz 10o_88j_A ist eine Instanz der BASF-AG. Diese wurden zuerst in **[6.1]** bearbeitet. Die anderen stammen von **[6.2]**. Es existieren insgesamt 25 Instanzen.
Bei den Ergebnissen werden manchmal Lösungen angegeben, die mit einer ‚gespiegelten' Instanz erreicht wurden. Bei einer ‚gespiegelten' Instanz sind alle Vorgänger-Nachfolgerbeziehungen und die benötigten Kapazitätsprofile im Vergleich zur ursprünglichen Instanz zeitlich gespiegelt. Die Ergebnisse für die gespiegelten Instanzen werden nur angegeben, falls sie besser sind.

Verglichen werden die Ergebnisse der Verfahren Tabu Search (**TS**), Parallel Tabu Search (**PTS**), A. Team (**AT**) (ein Agenten-Verfahren), Constrained Programming (**CP**), Lagrangian Heuristic (**LH**) und des Greedy Verfahrens (**GR**) mit denen von **SA**, für die verschiedenen Instanzen. Auf eine Darstellung der Ergebnisse von **TA** wird verzichtet, da diese bei der vorliegenden Problemstellung keinen großen Unterschied im Vergleich zu **SA** zeigten. Neben den besten Resultaten der Verfahren (**TS**, **PTS**, **AT**, **CP** und **LH**) von verschiedenen Simulationen mit unterschiedlichen Rechenzeiten und Parametersätzen ist noch der kritische Pfad (**kP**), eine untere Schranke (**LB**) und der optimale Wert (**OPT**) für einige Instanzen gegeben (**Tabelle 6.2**).
Die Rechenzeiten aus der Literatur für die einzelnen Verfahren sind aufgrund der unterschiedlichen Rechner nicht direkt miteinander vergleichbar. Der Vollständigkeit halber wurden sie jedoch angegeben (**Tabelle 6.3**).

6.3.1 Ergebnisse verschiedener Verfahren aus der Literatur

In **[6.3]** werden die Ergebnisse eines Agenten-Verfahrens (**AT**) und eines parallelen Tabu Search (**PTS**) mit denen eines sequentiellen Tabu Search (**TS**) **[6.4]** verglichen. Es wurde gezeigt, dass die parallelen Ansätze deutliche Verbesserungen der Ergebnisse erzielen. Die Simulationen wurden auf einer SUN SPARC 1000 durchgeführt. Während die gesamte Rechenzeit (**Tabelle 6.3**) bei **AT** ungefähr der Zeit entsprach, zu der die beste Lösung gefunden wurde, ist die Rechenzeit, zu der die beste Lösung beim **PTS** gefunden wurde, vor allem bei den großen Instanzen sehr viel geringer als die gesamte gegebene Rechenzeit. Bei **TS** entsprach die Rechenzeit der Zeit, in der die jeweils beste Lösung gefunden wurde.
Die Ergebnisse des Constraint Programming (**CP**) aus **[6.5]** sind in der Regel nicht so gut wie die durch **PTS** gefundenen. Die Simulationen wurden hier auf einer SUN ULTRA SPARC 2 mit 248MHz durchgeführt. Als Zeitlimit wurden 200000 Knoten des Suchbaums beim Branch&Bound vorgegeben. Die in **Tabelle 6.3** angegebenen Werte entsprechen der Zeit bis zum Beweis der Optimalität oder dem Erreichen der 200000 Knoten. Die beste Lösung wurde jedoch meist vor Erreichen des Zeitlimits gefunden.
Die Ergebnisse der **LH** sind in **[6.6]** gegeben. Die Rechenzeiten entsprechen dabei Simulationen auf einer SUN ULTRA SPARC 2 mit 200MHz. Die Ergebnisse sind zwar nicht so gut wie die der pa-

rallelen Ansätze, die Rechenzeiten sind jedoch sehr viel geringer. Zusätzlich wird noch für jede Instanz eine untere Schranke berechnet.

Instanz	cp	opt	lb	TS	PTS	AT	CP	LH
ins 4o 21 A	78	82	80	82	82	82	82	82
ins 4o 23 A	54	58	54	58	58	58	58	58
ins 4o 24 A	58	68	58	68	68	68	68	70
ins 4o 24 B	54	72	55	72	72	72	72	73
ins 4o 27 A	53	67	54	67	67	67	67	68
ins 6o 41 A	90		103	141*	140	143	152*	145
ins 6o 41 B	94		94	110*	110	111	110*	111*
ins 6o 41 C	81		87	128	126	126	134*	134*
ins 6o 44 A	75		89	117*	117	116	122*	120*
ins 6o 44 B	104		104	137	137	137	149	144*
ins 8o 63 A	174		186	261	259	259	281	276
ins 8o 63 B	196		209	316	314	316	344	344
ins 8o 63 C	227		228	296*	294	301	344	316*
ins 8o 65 A	298		299	406*	406	403	445	417*
ins 8o 65 B	230		251	384	383	382	411	409
ins 10o 84 A	270		379	636	634	641	730	699
ins 10o 84 B	200		335	556*	550	567	616*	593*
ins 10o 85 A	513		514	791	783	793	912	851
ins 10o 87 A	194		371	582*	581	585	610*	595*
ins 10o 88 A	362		362	460	450	456	473*	473*
ins 10o 100 A	352		669	1468	1468	1467	1596	1525*
ins 10o 102 A	550		622	1166	1155	1158	1239*	1206*
ins 10o 106 A	383		600	1094	1087	1098	1166*	1122*
ins 12o 108 A	520		690	1277	1271	1277	1412*	1340*
ins 12o 109 A	819		820	1343*	1324	1336	1476*	1382*

Tabelle 6.2: Kritischer Pfad (**cp**), optimales Ergebnis (**opt**) und untere Schranke (**lb**) für die einzelnen Instanzen. Zusätzlich sind noch die besten Ergebnisse verschiedener Verfahren aus der Literatur (**TS**, **PTS**, **AT**, **CP**, **LH**) aufgelistet. Wurden die Ergebnisse mit gespiegelten Instanzen erzielt sind diese mit (*) markiert.

Arbeitsschritte	TS SUN SPARC 1000	PTS SUN SPARC 1000	AT SUN SPARC 1000	CP SUN ULTRA SPARC 2 248MHZ	LH SUN ULTRA SPARC 2 200MHZ
>40	10-80	100-500	1000-2000	118-310	1-5
>60	28-182	500-1000	1000-3000	206-985	6-120
>80	123-637	2000-5000	2000-4000	325-1052	6-668
>100	87-1277	9000-14000	5000-6000	596-1352	609-2550

Tabelle 6.3: Gesamte Rechenzeiten in Sekunden der einzelnen Verfahren für verschiedene Größen der Instanzen.

6.3.2 Ergebnisse der Verfahren SA und GR

Bei **SA** wird jeweils das gleiche logarithmische Abkühlschema mit gleicher Anzahl Sweeps pro Temperaturschritt für die Simulation jeder Instanz vorgegeben. Die Temperatur wird bei jeder Simulation von einem Startwert 20 in 250 Schritten logarithmisch (α=0,98) gesenkt. Dabei werden 100, 1000 und 5000 Sweeps pro Temperaturschritt durchgeführt. Bei **GR** werden nur Simulationen mit 25000 und 250000 Sweeps durchgeführt, da **GR** normalerweise bereits nach kurzer Rechenzeit in einem lokalen Optimum einfriert und zusätzliche Rechenzeit nicht ausnutzen kann, um das Ergebnis zu verbessern. Normalerweise benötigt man mehr Sweeps pro Temperaturschritt, um bei Instanzen mit einer größeren Anzahl an Aufträgen die gleiche Lösungsqualität zu erreichen. Die Rechenzeit ist nun abhängig von dem Temperaturschema und der Anzahl an Sweeps pro Tempera-

turschritt. Zusätzlich ist die Rechenzeit noch abhängig von der Anzahl an Aktivitäten, da es mehr Rechenzeit kostet, eine einzelne Lösung für mehr Aktivitäten zu erstellen. Einen kleinen Einfluss auf die Rechenzeit haben auch die maximale Kapazität der Sekundärressource und die benötigten Kapazitätsprofile der Modi. Die Rechenzeit ist jedoch unabhängig vom gewähltem Akzeptanzkriterium (**SA** oder **GR**). Der Unterschied bei der Berechnung der Exponentialfunktion ist im Vergleich zur Erstellung einer Lösung aus der Aktivitätsreihenfolge gering. In **Abbildung 6.3** ist die benötigte Rechenzeit für insgesamt 100000 Sweeps pro Instanz zu sehen. Die Rechenzeiten beziehen sich auf Simulationen auf einem Pentium mit 650 MHz.
Gestartet wird **SA** und **GR** mit einer zufälligen Lösung. Bei den Moves werden die folgenden mit jeweils gleicher Auswahlwahrscheinlichkeit verwendet: Push1, Swap, Lin2Opt.
Bei dieser Problemstellung zeigt sich, dass der unvollständige Suchraum (siehe **Kapitel 3.2.5**) möglicherweise die optimale Lösung nicht beinhaltet (siehe dazu die beiden Beispiele). Um die Wahrscheinlichkeit zu erhöhen, dass die optimale Lösung im Suchraum enthalten ist, wird im Folgenden neben den originalen Instanzen noch mit ‚gespiegelten' Instanzen gearbeitet, bei denen alle Kapazitätsprofile und Vorgänger-Nachfolgerbeziehungen gespiegelt sind.

Beispiel 1:
Es sind 3 Aktivitäten ohne Vorgänger-Nachfolger-Beziehungen zu planen, die folgende Kapazitätsprofile besitzen: (14,10,12), (4,3,4), (4,3,4). Bei einer maximalen Kapazität 18 hat bei dem vorgegebenen Suchraum eine der besten gefundenen Lösungen die Durchlaufzeit 6 mit den Startzeitpunkten (0,0,3). Die optimale Durchlaufzeit liegt jedoch bei 4 mit den zugehörigen Startzeitpunkten (0,1,1). Benutzt man nun statt des originalen den gespiegelten Suchraum, so erhält man im gespiegelten Suchraum als beste Lösung die Startzeitpunkte (1,0,0) mit einer Durchlaufzeit 4.
Beispiel 2:
Es sind 3 Aktivitäten ohne Vorgänger-Nachfolger-Beziehungen zu planen, die folgende Kapazitätsprofile besitzen: (14,10,12,10,14), (4,3,4), (4,3,4). Nun gibt es keinen Unterschied zwischen der originalen und der gespiegelten Instanz. Bei einer maximalen Kapazität 18 hat bei dem vorgegebenen Suchraum eine der besten gefundenen Lösungen die Durchlaufzeit 6 mit den Startzeitpunkten (0,0,3). Die optimale Durchlaufzeit liegt jedoch bei 5 mit den zugehörigen Startzeitpunkten (0,1,1).

Der kritische Pfad bezieht sich auf die Vorgänger-Nachfolgerbeziehungen zwischen den Aktivitäten und deren Dauern. Bei unbegrenzter Kapazität der Sekundärressource entspricht der kritische Pfad dem optimalen Ergebnis.
Neben dem kritischen Pfad wird noch die jeweils beste untere Schranke aus **[6.4]** und **[6.6]** gegeben. Diese Schranken erweisen sich jedoch für die Instanzen mit einer größeren Anzahl an Aufträgen als nicht sehr aussagekräftig.
Bei allen Instanzen mit weniger als 30 Arbeitsschritten liegt ein optimales Ergebnis vor (**[6.4]**, **[6.5]**).

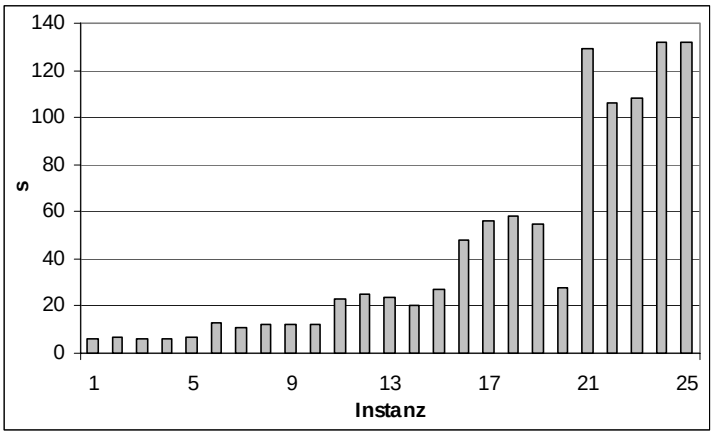


Abbildung 6.3: Benötigte Rechenzeit in Sekunden auf einem Pentium mit 650MHz für die Durchführung von 100000 Sweeps für die verschiedenen Instanzen.

6.3.2.1 Instanzen mit 20 bis 40 Aufträgen

sweeps	5000		25000	
Instanz	o	i	o	i
ins 4o 21 A	82 (0)	82 (0)	82 (0)	82 (0)
ins 4o 23 A	58 (0)	58 (0)	58 (0)	58 (0)
ins 4o 24 A	68,5 (0,5)	69 (0)	68 (0)	69 (0)
ins 4o 24 B	72 (0)	72 (0)	72 (0)	72 (0)
ins 4o 27 A	67,8 (0,4)	68 (0)	67 (0)	67,9 (0,3)

Tabelle 6.4: Zu sehen sind die durchschnittlichen Durchlaufzeiten (Standardabweichung) von jeweils 20 Simulationen mit **GR** mit **5000** bzw. **25000** Sweeps für die originalen (**o**) und die gespiegelten (**i**) Instanzen mit 20 bis 30 Aufträgen.

In **Tabelle 6.4** sind die durchschnittlichen Ergebnisse und die Standardabweichung für jeweils 20 Simulation mit **GR** mit 5000 und 25000 Sweeps für die originalen und gespiegelten Instanzen abgebildet.
Bei kurzen Laufzeiten konnte bei den meisten Instanzen (original oder gespiegelt) immer ein Optimum gefunden werden und in zwei Fällen zumindest einmal. Nur die gespiegelten Instanzen ins_4o_24_A und ins_4o_27_A konnten in keinem Fall optimal gelöst werden.
Bei der Erhöhung der Sweeps auf 25000 werden alle originalen Instanzen immer optimal gelöst. Bei der gespiegelten Instanz ins_4o_24_A kann hingegen das Optimum auch durch die Erhöhung der Rechenzeit nicht gefunden werden. Simulationen mit **SA** liefern das gleiche Resultat, weshalb davon auszugehen ist, dass das Optimum bei dieser Instanz zwar im originalen, nicht aber im gespiegelten Suchraum enthalten ist. Diese Möglichkeit wurde bereits in Beispiel 1 erwähnt. Bei der gespiegelten Instanz ins_4o_27_A ist das Optimum im Suchraum enthalten, jedoch scheint diese Instanz im gespiegelten Fall schwieriger zu lösen zu sein, da nicht jede Simulation mit erhöhter Rechenzeit das Optimum finden kann. Insgesamt sind die guten Ergebnisse des **GR** Beweis dafür, dass diese Instanzen zu klein sind, um als Kriterium für einen Vergleich verschiedener Verfahren zu dienen.

6.3.2.2 Instanzen mit mehr als 40 Aufträgen

	SA			GR	
Instanz	**100**	**1000**	**5000**	**25000**	**250000**
ins 6o 41 A	* 144,1 (1,3)	* 140,8 (0,8)	* 140,5 (0,7)	143,8 (0,6)	141,9 (0,8)
ins 6o 41 B	* 112,0 (0,8)	* 110,2 (0,4)	* 110,2 (0,4)	* 111,7 (0,9)	* 110,6 (1,2)
ins 6o 41 C	128,8 (1,3)	126,8 (0,8)	126,7 (0,6)	129,3 (0,8)	126,9 (0,7)
ins 6o 44 A	* 117,5 (0,5)	* 116,8 (0,4)	* 116,7 (0,5)	* 118,2 (1,2)	* 116,8 (0,6)
ins 6o 44 B	139,1 (0,9)	138,1 (0,8)	137,1 (0,3)	* 139,8 (0,7)	* 137,4 (0,8)
ins 8o 63 A	263,0 (2,3)	259,8 (1,1)	257,9 (0,9)	264,9 (3,3)	260,9 (2,1)
ins 8o 63 B	320,4 (1,9)	315,4 (1,1)	313,8 (0,6)	328,9 (2,8)	321,6 (3,8)
ins 8o 63 C	301,7 (3,8)	295,6 (2,5)	293,9 (1,9)	* 303,7 (2,9)	* 297,8 (1,8)
ins 8o 65 A	* 405,5 (1,9)	* 403,0 (0,0)	* 403,0 (0,0)	* 405 (2,0)	* 404,1 (1,5)
ins 8o 65 B	388,6 (2,4)	380,8 (2,0)	380,2 (1,3)	* 397,7 (3,4)	* 391,1 (2,6)
ins 10o 84 A	* 638,5 (3,3)	* 633,0 (1,7)	* 630,2 (1,1)	650,5 (4,0)	644,4 (4,0)
ins 10o 84 B	* 559,7 (4,5)	* 546,2 (1,6)	* 544,6 (1,1)	* 571,9 (3,3)	* 558,5 (4,5)
ins 10o 85 A	793,2 (7,9)	780,1 (4,0)	775,9 (2,2)	* 830,8 (12,4)	* 797 (6,7)
ins 10o 87 A	* 591,1 (4,8)	* 580,0 (2,8)	* 576,4 (2,4)	* 596,3 (4,4)	* 584,7 (3,1)
ins 10o 88 A	* 464,7 (4,0)	* 454,5 (4,3)	* 446,8 (3,2)	481,7 (10,3)	464,3 (4,1)
ins 10o 100 A	* 1470,5 (1,5)	* 1468,2 (0,6)	* 1467,8 (0,4)	1476,7 (5,5)	1468,2 (0,4)
ins 10o 102 A	1185,8 (7,5)	1166,3 (7,5)	1151,3 (6,4)	1211,9 (7,0)	1177,1 (9,0)
ins 10o 106 A	1102,5 (3,9)	1084,4 (3,6)	1074,7 (3,1)	1121,7 (7,4)	1092,6 (2,7)
ins 12o 108 A	1283,6 (5,0)	1265,7 (3,4)	1258,0 (3,3)	1312,8 (6,9)	1274,6 (4,1)
ins 12o 109 A	1339,8 (9,8)	1311,4 (6,1)	1309,1 (6.7)	* 1362,9 (16,7)	* 1322,7 (8,9)

Tabelle 6.5: Abgebildet sind die durchschnittlichen Durchlaufzeiten (Standardabweichung) von jeweils 20 Simulationen mit **SA** (**100**, **1000** und **5000** Sweeps pro Temperaturschritt) und **GR** (**25000** und **250000** Sweeps) für die originalen oder die gespiegelten Instanzen mit mehr als 40 Aufträgen. Ergebnisse von gespiegelten Instanzen sind mit (*****) markiert.

Gezeigt wird jeweils die durchschnittliche Lösung und die Standardabweichung von 20 durchgeführten Simulationen für die originalen oder die gespiegelten Instanzen mit den Verfahren **SA** und **GR** für verschiedene Anzahl Sweeps (**Tabelle 6.5**). Auf die gleichzeitige Darstellung der Ergebnisse für die originalen und gespiegelten Instanzen wird verzichtet, da sich die Mittelwerte bei fast allen Instanzen um weniger als 5 unterscheiden und nur bei der Instanz ins_10o_102_A um mehr als 10. Die besten Ergebnisse werden in **Tabelle 6.6** zusammengefasst.
Bei den Instanzen mit 40 bis 50 Aufträgen und den Instanzen ins_8o_65_A und ins_10o_100_A nähert sich der durchschnittliche Wert bei **SA** bei Erhöhung der Anzahl Sweeps pro Temperaturschritt auf 5000 den besten Ergebnissen aus der Literatur (**Tabelle 6.6**) an. Bei allen anderen Instanzen ist die durchschnittliche Durchlaufzeit bei 5000 Sweeps pro Temperaturschritt besser als die besten bekannten Ergebnisse und fast alle Simulationen liefern Ergebnisse mit einer kleineren oder zumindest gleichgroßen Durchlaufzeit.
Investiert man nur wenig Rechenzeit (100 Sweeps pro Temperaturschritt), so kann man mit **SA** trotzdem noch bei fast allen Instanzen (bis auf die Instanz ins_6o_41_B) bessere durchschnittliche Werte erhalten als die besten Durchlaufzeiten der schnellen Verfahren **CP** und **LH**.
Betrachtet man sich die besten Ergebnisse für die einzelnen Instanzen (**Tabelle 6.6**), so sieht man, dass **SA** für 7 Instanzen die gleichen Durchlaufzeiten und für 13 Instanzen kürzere Durchlaufzeiten erzielen kann. Die 7 Instanzen, für die die gleiche Durchlaufzeit gefunden wurde, scheinen einfach zu sein, da selbst **GR** bei 250000 Sweeps zumindest in einer von 20 Simulationen die gleiche Durchlaufzeit finden konnte.

GR scheint bei der gewählten Nachbarschaft für diese Problemstellung sehr erfolgreich zu sein, da bereits nach kurzen Rechenzeiten relativ gute Ergebnisse erzielt wurden. Bei einigen kleinen Instanzen konnte bei 25000 Sweeps sogar bessere durchschnittliche Durchlaufzeiten erhalten werden als bei **SA**. Insgesamt gesehen ist jedoch **SA** bei ausreichend Rechenzeit dem **GR** überlegen, vor allem bei größeren Instanzen.

Instanz	bester	sweeps	bester(Lit)
ins 6o 41 A	140 *	1000	140
ins 6o 41 B	110 *	1000	110
ins 6o 41 C	126	1000	126
ins 6o 44 A	116 *	1000	116
ins 6o 44 B	137	100	137
ins 8o 63 A	256 *	5000	259
ins 8o 63 B	312	5000	314
ins 8o 63 C	291	5000	294
ins 8o 65 A	403 *	100	403
ins 8o 65 B	376	1000	382
ins 10o 84 A	628 *	5000	634
ins 10o 84 B	542 *	5000	550
ins 10o 85 A	773	5000	783
ins 10o 87 A	573 *	5000	581
ins 10o 88 A	443	5000	450
ins 10o 100 A	1467 *	1000	1467
ins 10o 102 A	1146	5000	1155
ins 10o 106 A	1068	5000	1087
ins 12o 108 A	1252	5000	1271
ins 12o 109 A	1295	5000	1324

Tabelle 6.6: Gezeigt werden die besten Durchlaufzeiten (**bester**), die mit **SA** bei **sweeps** Sweeps pro Temperaturschritt, erzielt wurden. Zusätzlich sind noch die besten Resultate aus der Literatur (**bester(Lit)**) (siehe **Tabelle 6.2**) zu sehen. Ergebnisse von gespiegelten Instanzen sind mit (*) markiert.

6.4 PSPLib

In der **PSPLib** (**[6.7]**, **[6.8]**, **[6.9]**) sind viele Instanzen zu *Resource Constrained Project Scheduling Problems* (**RCPSP**) gegeben. Die geforderte Funktionalität unterscheidet sich bei den Instanzen nur gering von der in diesem Kapitel benötigten. So ist die erforderliche Anzahl an Arbeitskräften pro Arbeitsschritt konstant. Ein Arbeitsschritt bedarf möglicherweise nun auch verschiedener Sekundärressourcen mit einer evtl. unterschiedlichen Anzahl an Arbeitskräften.
Aus der Bibliothek wird das Paket *j120.sm* bearbeitet. Jede der 600 Instanzen dieses Pakets besteht aus 120 Arbeitsschritten und 4 Sekundärressourcen. Die anderen Pakete der Bibliothek bestehen aus Instanzen mit weniger Arbeitsschritten. Zu jeder Instanz ist eine untere und obere Schranke gegeben. Diese sind jeweils die besten verschiedener Verfahren.
Bearbeitet werden diese Instanzen hier mit **SA**, wobei das Modell dem vorhergehenden (**Kapitel 6.2**) entspricht, mit dem Unterschied der konstanten Arbeitskräfte und vier statt einer Sekundärressource. Die Temperatur wird von einem Anfangswert 15 in 250 Schritten logarithmisch (α=0.98) gesenkt. Die Nachbarschaft besteht aus **Push1** und **Swap**, wobei beide mit gleicher Wahrscheinlichkeit ausgeführt werden. Die Anzahl Sweeps pro Temperaturschritt wird auf 20, 100, 200 und

1000 festgelegt. Neben diesem **SA** wird noch eine Version mit gespiegelten Instanzen (**SA$_{inv}$**) verwendet.

Zusätzlich zu den beiden Verfahren wird die Modellierung erweitert um Lieferzeitpunkte. Jede Aktivität erhält einen propagierten Lieferzeitpunkt, der sich aus einer bestimmten Grenze L_{max}, zu der alle Aktivitäten beendet werden sollen und den Dauern der Nachfolgeaktivitäten ergibt. Die Priorität des Lieferzeitpunkts jeder Aktivität ergibt sich aus den Vorgänger-Nachfolgerbeziehungen. Der Lieferzeitpunkt L_i und die zugehörige Priorität P_i einer Aktivität i mit der Dauer D_i und den Nachfolgern j ergibt sich wie folgt:

$$L_i = \begin{cases} L_{\max} & \textit{falls Aktivität i keine Nachfolger besitzt} \\ MIN_j\left(L_j - D_j\right) & \textit{sonst} \end{cases},$$

$$P_i = \begin{cases} 1 & \textit{falls Aktivität i keine Nachfolger besitzt} \\ MAX_j\left(P_j + 1\right) & \textit{sonst} \end{cases}.$$

Die Grenze L_{max} wird durch die gegebene obere Grenze bestimmt (**SA$_{ub}$**). Das Gewicht der Durchlaufzeit wird nun auf 10 erhöht und das Gewicht der Lieferverzögerung auf 1 gesetzt. Dadurch ist eine Anpassung des Temperaturschemas erforderlich. Die Starttemperatur liegt hier bei 150.

Diese obere Grenze ist bei realistischen Instanzen eigentlich nicht vorhanden. Das Verfahren wird deshalb angepasst, so dass die obere Grenze aus der jeweils besten Lösung während der Simulation bestimmt wird. Die Lieferzeitpunkte müssen immer dann erneuert werden, falls eine Lösung mit minimaler Durchlaufzeit D_{min} gefunden wird. Der zu propagierende Zeitpunkt bestimmt sich dann durch L_{max}=D_{min}-1. Das so erstellte Verfahren wird mit **SA$_{m}$** bezeichnet.

Die Ergebnisse, die mit den vier Verfahren für verschiedene Rechenzeiten erhalten wurden, sind in **Tabelle 6.7** gegeben. Zu sehen sind dort die durchschnittlichen Ergebnisse für die 600 Instanzen und deren durchschnittliche Standardabweichung für verschiedene Verfahren und Rechenzeiten. Zusätzlich ist der durchschnittliche prozentuale Abstand zu der unteren und oberen Grenze aus **[6.7]** und dem kritischen Pfad gegeben. In den letzten Spalten sind die Anzahl der Instanzen gegeben, bei denen bei jeder Simulation und bei der besten aus 5 die beste bekannte Lösung erreicht wurde. Es konnte gegenüber den besten Lösungen (April 2004) keine neuen besten Lösungen gefunden werden, jedoch konnten im Vergleich zu den besten Lösungen vor 2002 (dieses File existiert leider nicht mehr, da es immer durch die aktuell besten Lösungen ersetzt wird) einige verbessert werden. Zu beachten ist dabei, dass pro Instanz und Rechenzeit nur 5 Simulationen durchgeführt werden.

Bei allen Verfahren wird der Abstand zur unteren und oberen Grenze und dem kritischen Pfad bei steigender Rechenzeit geringer (vgl. **Tabelle 6.7**) und es wird für mehr Instanzen die obere Grenze bei jeder Simulation erreicht. So wird z.B. bei mehr als 30% der Instanzen von **SA$_{ub}$** die obere Grenze bei 1000 Sweeps pro Temperaturschritt immer erreicht. Im Vergleich zu den neuesten (April 2004) oberen Grenzen konnte keine verbessert werden.

Der Unterschied zwischen den Verfahren **SA** und **SA$_{inv}$** ist bei den durchschnittlichen Ergebnissen nicht zu erkennen, jedoch schneiden die Verfahren bei den einzelnen Instanzen unterschiedlich ab. Eine Kombination aus beiden Planungsrichtungen innerhalb einer Simulation könnte somit auch hier von Vorteil sein.

Die zusätzliche Information der oberen Grenzen in Form von Lieferzeitpunkten (**SA$_{ub}$**) kann die Ergebnisse verbessern. Nicht nur die durchschnittlichen Ergebnisse sind besser, sondern auch die

Anzahl der Instanzen, bei denen die obere Grenze in jeder Simulation gefunden wurde. Auch die während der Optimierung festgestellte obere Grenze ($\mathbf{SA_m}$) verbessert die durchschnittlichen Ergebnisse im Vergleich zu **SA** und schneidet vergleichbar zu $\mathbf{SA_{ub}}$ ab. Allerdings muss beachtet werden, dass diese beiden Verfahren etwas mehr Rechenzeit benötigen (vgl. **Tabelle 6.8**).
Die durchschnittliche Standardabweichung der einzelnen Verfahren ist sehr gering. Dies bedeutet, dass die Verfahren sehr stabil in Bezug auf die Lösungssuche sind.

	Sweeps	av (dev)	%lb	%ub	%cp	numAll	numBest
SA	20	131,7 (1,3)	11,4	6,5	39,3	101	150
	100	128,8 (0,9)	9,0	4,3	36,2	141	184
	200	127,9 (0,8)	8,3	3,6	35,3	158	190
	1000	126,3 (0,7)	7,1	2,5	33,6	182	221
SAinv	20	131,6 (1,3)	11,3	6,4	39,2	100	158
	100	128,8 (0,9)	9,0	4,3	36,2	140	185
	200	127,9 (0,8)	8,3	3,6	35,2	159	195
	1000	126,3 (0,6)	7,0	2,4	33,5	183	218
SAub	20	132,0 (1,6)	11,4	6,5	39,6	128	178
	100	128,2 (1,0)	8,4	3,7	35,6	165	211
	200	127,3 (0,9)	7,8	3,1	34,7	177	223
	1000	125,9 (0,7)	6,6	2,0	33,1	204	246
SAm	20	131,1 (1,3)	10,8	5,9	38,6	116	163
	100	128,0 (0,9)	8,3	3,6	35,4	158	203
	200	127,2 (0,8)	7,7	3,0	34,5	170	206
	1000	125,8 (0,6)	6,6	2,0	33,0	191	241

Tabelle 6.7: Es sind die durchschnittlichen Durchlaufzeiten (**av**) und deren Standardabweichung (**dev**), gemittelt über die 600 Instanzen, zu sehen. Zusätzlich ist noch der prozentuale Abstand zur unteren Grenze (**%lb**), der oberen Grenze (**%ub**) und dem kritischen Pfad (**%cp**) gegeben. Die beiden letzten Spalten geben die Anzahl der Instanzen, bei denen bei jeder Simulation (**numAll**) und bei der besten aus 5 Simulationen (**numBest**) die obere Grenze erreicht wurden.

In **[6.10]** ist ein ausführlicher Vergleich verschiedener Verfahren gegeben, wobei dieser Vergleich immer auf dem prozentualen Abstand zum kritischen Pfad basiert. Die Rechenzeit spielt bei diesem Vergleich keine Rolle. Als Kriterium wird immer die maximale Anzahl an erzeugten Lösungen verwendet. Dies sollte dann einen Vergleich verschiedener Verfahren erlauben, ohne die verwendeten Rechner, Compiler oder Programmierkenntnisse der Entwickler der Verfahren, die einen großen Einfluss auf die Geschwindigkeit der erstellten Programme haben, beachten zu müssen. Es wird somit bei diesem Vergleich davon ausgegangen, dass das Erstellen einer Lösung bei allen Verfahren in etwa die gleiche Rechenzeit benötigt. In **Tabelle 6.8** sind die Rechenzeiten der hier verwendeten Verfahren gegeben. Diese beziehen sich auf den gleichen Rechner (Pentium mit 650 MHz), wurden vom gleichen Compiler übersetzt und vom gleichen Entwickler erstellt. Trotzdem unterscheiden sich die einzelnen Modellierungen in der benötigten Rechenzeit. Es ist also im allgemeinen Fall nicht davon auszugehen, dass die Rechenzeit bei allen Verfahren zum Erstellen einer Lösung übereinstimmt. In dieser Arbeit wird immer versucht, möglichst einfache Planungsschritte möglichst schnell durchzuführen. Es ist deshalb zu erwarten, dass in dieser Arbeit das Erstellen einer Lösung schneller erfolgt als bei anderen Verfahren. In **Kapitel 10** wird z.B. eine Heuristik zum Erstellen einer ersten Lösung vorgestellt. Dieses erste Ergebnis ist besser als eine Lösung, die mit einer einfachen lokalen Suche in mehreren Hundert Schritten erreicht werden könnte, allerdings ist die benö-

tigte Rechenzeit deutlich größer als bei der einfachen lokalen Suche. Erhöht man die Anzahl der Aufträge, so vergrößert sich der relative Rechenzeitunterschied sogar noch. In **[6.11]** wurden z.B. verschiedene Varianten eines Ameisenalgorithmus auf diesen Instanzen getestet. Dabei benötigte man zum Erstellen von 5000 Lösungen (1000 Generationen mit jeweils 5 Ameisen) bei einer Variante ~25 Sekunden auf einem Pentium III mit 500 MHz. Eine weitere Variante benötigte zum Erstellen von 50000 Lösungen ~25 Minuten auf dem gleichen Rechner! Die Anzahl der erzeugten Lösungen ist somit nicht unbedingt die aussagekräftigste Vergleichsbasis! Für einen Anwender der Software ist die Anzahl der Lösungen irrelevant, solange die Rechenzeit seinen Vorstellungen entspricht.

	SA	SAinv	SAub	SAm
av	18,7	18,7	21,9	21,9
min	10,0	10,0	14,0	14,0
max	28,0	28,0	36,0	36,0

Tabelle 6.8: Benötigte Rechenzeit für die einzelnen Verfahren für das Erstellen von 100000 Lösungen in Sekunden. Zu sehen ist der Mittelwert (**av**) aller 600 Instanzen, die minimal (**min**) und maximal (**max**) benötigte Rechenzeit.

6.5 Zusammenfassung

Insgesamt gesehen kann man sagen, dass **SA** sehr gut geeignet ist, um bei dieser Problemstellung (**SLRC**) in kurzer Zeit sehr gute Ergebnisse zu erzielen. Darüber hinaus ist das Verfahren in der Lage, bei Investition ausreichender Rechenzeit die besten Ergebnisse verschiedener anderer Verfahren aus der Literatur zu erreichen oder zu verbessern. Es wurden für 13 Instanzen neue obere Schranken gefunden. **SA** vereinigt damit die Vorteile der verschiedenen anderen Verfahren. Der Nachteil von **SA**, keine Aussage über untere Schranken geben zu können, wirkt sich in dieser Problemstellung nicht negativ aus, da selbst die unteren Schranken der vorgestellten anderen Verfahren, vor allem die der größeren Instanzen, sehr schwach waren und nur geringfügig über dem kritischen Pfad lagen.
Im Vergleich der Instanzen der **PSPLib** zeigt sich, dass die durchschnittlichen Ergebnisse bereits bei Rechenzeiten von ~50 Sekunden pro Instanz bis auf 2-2,5% die besten bisher bekannten Lösungen (2004) erreichen, wobei bei ~30% der Instanzen bei jeder Simulation die beste Durchlaufzeit gefunden wird. Im Vergleich zu den besten Lösungen, die bis 2002 bekannt waren, konnten sogar einige der besten Ergebnisse noch verbessert werden. Dies ist für ein Verfahren, das nicht speziell auf diese Problemklasse angepasst wurde, ein sehr gutes Resultat. Es ist fraglich, ob sich die Spezialisierungen der anderen Verfahren auf die allgemeine Modellierungsbasis dieser Arbeit übertragen lassen.

Kapitel 7

Production Flow Planning with Machine Assignment

7.1 Problemstellung

Betrachtet wird folgender Produktionsablauf:

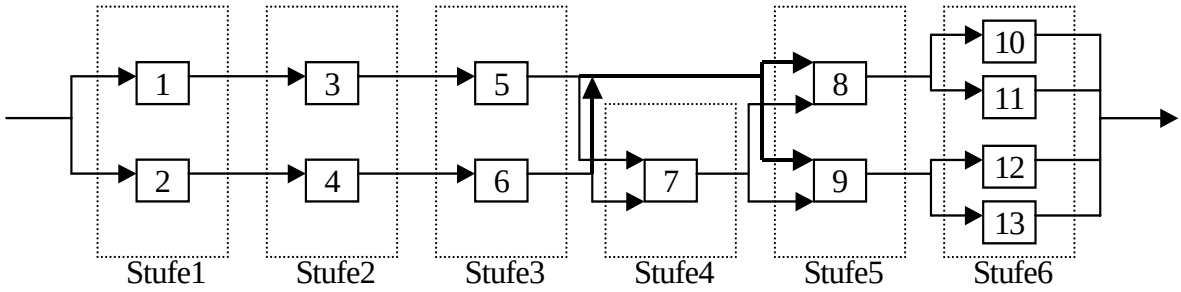


Abbildung 7.1: Struktur des Fertigungsprozesses.

Diese Problemstellung der BASF-AG lässt sich den **Flexible-FlowShop** Problemen zuordnen.
Zur Herstellung eines Produkts müssen mehrere Arbeitsschritte (verschiedene Stufen) in einer bestimmten Reihenfolge bei der Bearbeitung durchgeführt werden (Stufe 1 bis Stufe 6). Allerdings benötigt nicht jedes Produkt jeden Arbeitsschritt. Stufe 4 stellt eine Ausnahme dar, da sie nicht von allen Produkten durchlaufen werden muss. Für jeden Arbeitsschritt hat jedes Produkt mehrere Möglichkeiten bei der Wahl der Maschinen. Eine beliebige Kombination der Maschinen (1-13) ist jedoch nicht möglich. So muss ein Produkt, wird es auf Maschine 1 in Stufe 1 bearbeitet, die Maschinen 3 (Stufe 2) und 5 (Stufe 3) zur Weiterverarbeitung benutzen. Eine Lagerung zwischen den Stufen ist nicht möglich. Ist die Maschine der nachfolgenden Stufe nicht sofort nach Beendigung des Arbeitsschritts auf der vorhergehenden Stufe verfügbar, so wird die Maschine der vorhergehenden Stufe zur Lagerung des Produktes genutzt und solange gesperrt, bis mit dem Arbeitsschritt des Produkts auf der nachfolgenden Stufe begonnen werden kann. Maschine 7 (Stufe 4) bildet auch hier eine Ausnahme. Auf dieser Maschine kann nichts gelagert werden, d.h. nach dem Beenden des Arbeitsschritts auf Maschine 7 muss sofort mit dem Arbeitsschritt der Stufe 5 begonnen werden. Die Dauer der einzelnen Arbeitsschritte kann von der benutzten Maschine und dem zu fertigenden Produkt abhängen. Bei der Planung muss zusätzlich noch auf maschinenabhängige Rüstzeiten geachtet werden, die zwischen zwei Arbeitschritten auf einer Maschine eingeplant werden müssen.

Pro Tag geht eine Liste ein, die die Mengen der herzustellenden Produkte festsetzt. Mit der Ausführung des Auftrags kann frühestens am Tag des Auftrageingangs gestartet werden. Am Wochenende wird kein neuer Auftrag vergeben, mit der Ausführung der bereits eingegangenen Aufträge kann jedoch trotzdem fortgefahren werden. Das Ziel bei der Optimierung ist die Minimierung der Durchlaufzeit.
Die für die Aufgabenstellung benötigten Daten sind in **Anhang C** zu finden.

Im allgemeinen Fall kann von einer Produktionsstruktur ausgegangen werden (**Abbildung 7.2**), die n Stufen alternativer Ressourcen und alle Möglichkeiten bei der Kombination von Ressourcen aufeinanderfolgender Stufen aufweist (**Abbildung 7.3**). Auch können die Rüstzeiten in der Regel nicht nur von der Maschine, sondern auch von der Reihenfolge der geplanten Produkte abhängen.

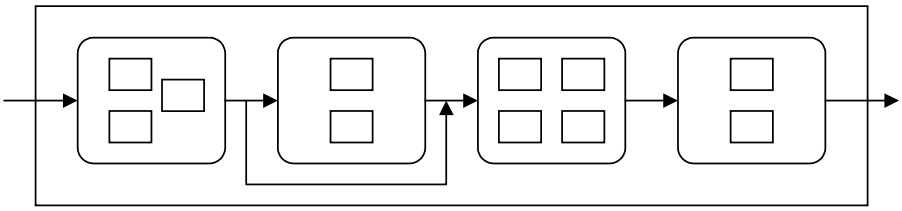


Abbildung 7.2: Beispiel eines Fertigungsprozesses mit 4 Stufen.

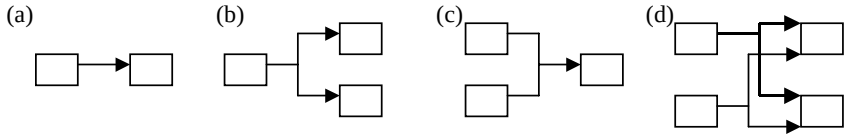


Abbildung 7.3: Verschiedene Möglichkeiten bei der Kombination von Ressourcen aufeinanderfolgender Stufen: **(a)** ein Vorgänger – ein Nachfolger, **(b)** ein Vorgänger – mehrere Nachfolger, **(c)** mehrere Vorgänger – ein Nachfolger, **(d)** mehrere Vorgänger – mehrere Nachfolger.

7.2 Modell

a) Ressourcen:

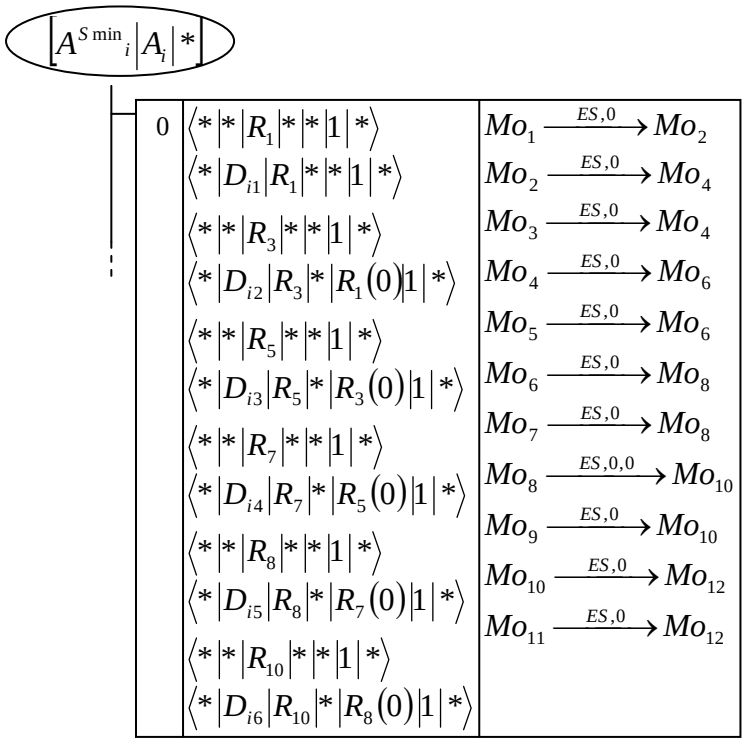
13 Primärressourcen werden benötigt ($R_i, 1 \le i \le 13$).

Es werden 6 Rüstzeitmatrizen definiert, eine für jede Stufe. Da die Rüstzeit nur von der gewählten Primärressourcen, nicht aber von einer geplanten Reihenfolge abhängt, hat jede Matrix nur einen Eintrag und die Modi alle den gleichen Rüstschlüssel.

b) Aktivitäten:

Jeder Auftrag für ein Produkt wird durch eine einzige Aktivität dargestellt. Es gibt nur 2 verschiedene ‚Typen' von Aktivitäten, eine, die Stufe 4 benutzt und eine, die diese bei der Produktion überspringt. Aktivitäten des gleichen Typs unterscheiden sich nur durch die Dauern D_{ij} der Modi. Jeder Multimodus der Aktivität gibt einen der möglichen Produktionswege an. Die Aktivitäten haben somit jeweils 8 Multimodi. Im Folgenden ist ein Beispiel für einen der 8 möglichen

Multimodi einer Aktivität gegeben, die Stufe 4 nutzt. Der Link zwischen dem 8. und 10. Modus ist bei diesen Aktivitäten mit zeitlichem Maximalabstand definiert, um die fehlende Lagermöglichkeit auf R_7 zu modellieren. Bei Aktivitäten, die Stufe 4 überspringen, wird kein Moduslink mit maximalem Abstand benötigt.
Einem Auftrag i und somit seiner Aktivität A_i kann ein frühester Start $A^{Smin}{}_i$ vorgegeben werden.



7.3 Ergebnisse

7.3.1 Ein-Tages-Pakete

Zuerst werden die Tagespakete einzeln geplant. Im Folgenden sind die Durchlaufzeiten immer in Sekunden angegeben, obwohl einige Verfahren nur minutengenau rechnen. Die sekundengenaue Darstellung entspricht den Simulationsdaten. Nicht alle Produktionszeiten sind ein Vielfaches einer Minute.

In **[7.1]** werden Simulationen mit *constraint programming* (**CP**), *mathematical programming* (**MP)** und verschiedenen kombinierten Verfahren aus **MP** und **CP** auf einem PC Pentium Pro mit 200 MHz durchgeführt. Gegeben sind für jedes Tagespaket die besten Resultate, die bei vorgegebenen maximalen Rechenzeiten von 300 und 3600 Sekunden mit den verschiedenen Verfahren erreicht wurden. In **Tabelle 7.1** sind jedoch nur die Ergebnisse von **CP** und das jeweils beste Ergebnis eines kombinierten Einsatzes von **MP** und **CP** zu sehen. Zusätzlich ist noch die Zeit gegeben, die benötigt wurde, um diese Lösungen zu finden. Falls bei Erhöhung der maximalen Rechenzeit keine bessere Lösung gefunden wurde, wird dies durch ‚-' angegeben.

Optimiert wird bei diesen Aufgabenstellungen nur mit **SA**. Die Temperatur wird dabei von dem Startwert 10000 in 300 Schritten mit einem Abkühlfaktor von 0.98 logarithmisch gesenkt. Für jedes Tagespaket werden Simulationen mit jeweils 50 und 2000 Sweeps pro Temperaturschritt durchgeführt. Dabei benötigen 100000 Sweeps im Durchschnitt 5 Sekunden Rechenzeit auf einem Pentium mit 650MHz. Die gesamte Rechenzeit einer Simulation entspricht somit weniger als 1 Sekunde (30 Sekunden) bei 50 (2000) Sweeps pro Temperaturschritt. Bei den Moves werden die folgenden mit jeweils gleicher Auswahlwahrscheinlichkeit verwendet: Push1, Swap, Lin2Opt, Change, Push-Change, Swap&Change, SwapSameResource und Lin2OptSameResource.
Gegeben werden für jedes Tagespaket und vorgegebener Rechenzeit (Anzahl Sweeps pro Temperaturschritt) die besten und durchschnittlichen Ergebnisse aus jeweils 10 Simulationen (**Tabelle 7.2**). Als Zeitraster für die Durchlaufzeit wird zuerst Minuten (**SA1**) und dann Sekunden (**SA2**) gewählt. In **[7.1]** wurde die Durchlaufzeit in Minuten angegeben, wodurch eine spätere Überprüfung aller Ergebnisse sekundengenau durchgeführt werden musste. Als Grund wird die benötigte Rechenzeit, die bei dem dort verwendeten Lösungsansatz von dem verwendeten Zeitraster abhängt, angegeben: ‚*The gain in „correctness“ of the solutions is usually negligible and outweighted by the more time consuming enumeration.*' Im Gegensatz zu den dort verwendeten Verfahren ist die Rechenzeit der Lösungssuche mit der in dieser Arbeit verwendeten Modellierung von der Wahl des Zeitrasters unabhängig.

	CP (max 300)		CP (max 3600)		comb (max 3600)	
Tag	**beste**	**t**	**beste**	**t**	**beste**	**t**
1	153840	1,5	-	-	-	-
2	123000	0,8	-	-	121800	1880,3
3	141240	0,7	-	-	-	-
4	114000	1	113700	1794,8	112680	585,7
7	126120	2,2	-	-	125700	798,2
8	110700	0,2	109800	896,5	109620	960,8
9	155100	1	-	-	-	-
10	111300	202,4	110940	820,2	110940	96
11	126000	0,2	124560	1039,4	124560	39,7

Tabelle 7.1: Die **beste**n Lösungen für **CP** bei maximalen Rechenzeiten von 300 und 3600 Sekunden und für einen kombinierten Ansatz aus **MP** und **CP** bei einer maximalen Rechenzeit von 3600 Sekunden. Zusätzlich dazu sind die Zeiten (**t**) in Sekunden angegeben, in denen diese Lösungen gefunden wurden.

	SA1				SA2			
	50 Sweeps		2000 Sweeps		50 Sweeps		2000 Sweeps	
Tag	beste	durch	beste	durch	beste	durch	beste	durch
1	141720	142902	141360	142182	140974	142909	140974	142023
2	121500	123660	119880	120702	119790	122990	119790	120797
3	137160	139206	137160	138258	137108	139237	137108	137898
4	113700	114438	112680	113334	113218	114159	112678	113173
7	125700	127596	125700	126138	125921	128117	125663	125893
8	110400	112050	109620	110652	110400	113163	108869	110328
9	143040	144876	140700	141990	142590	144446	140669	142219
10	110940	111012	110940	110940	110921	111188	110921	110921
11	124560	125712	124560	125052	124539	125595	124539	124992

Tabelle 7.2: Die **beste**n und **durch**schnittlichen Ergebnisse von **SA** für verschiedene Anzahl **Sweeps** pro Temperaturschritt bei minutengenauer (**SA1**) und sekundengenauer (**SA2**) Planung.

Betrachtet man sich die Ergebnisse von **SA1**, so sieht man, dass die durchschnittlichen Durchlaufzeiten bei kurzen Rechenzeiten höchstens 1,8% größer sind als die jeweils besten Ergebnisse. Bei erhöhter Rechenzeit werden in 5 Fällen bessere Ergebnisse gefunden und die durchschnittlichen Werte sind nur noch maximal 0,8% vom jeweils besten entfernt. Eine Erhöhung der Rechenzeit hat also bereits bei wenigen Aufträgen, es wird in diesem Fall ja nur ein einzelnes Tagespaket geplant, Auswirkungen auf die Ergebnisse und die Stabilität beim Auffinden der Lösungen.
Im Vergleich zu den besten Ergebnissen aus **[7.1]** konnte in 4 Fällen ein Ergebnis mit kleinerer Durchlaufzeit gefunden werden. In den restlichen 5 Fällen konnte ein Ergebnis mit gleicher Durchlaufzeit erreicht werden.
Ein Vergleich mit der sekundengenauen Planung **SA2** zeigt, dass **SA2** zwar die besten Ergebnisse zu den einzelnen Tagespaketen gefunden hat, jedoch bei den durchschnittlichen Werten keine großen Unterschiede zu **SA1** bestehen. Bei den nächsten Simulationen wird immer sekundengenau gearbeitet, da **SA2** nicht mehr Rechenzeit benötigt als **SA1** und damit die gewünschte Genauigkeit beachtet wird.

7.3.2 Mehr-Tages-Pakete

Neben der ursprünglichen Aufgabenstellung werden noch weitere leicht veränderte Aufgabenstellungen optimiert. Dabei unterscheiden sich die Aufgabenstellungen durch

1. den frühesten Startzeitpunkt der Aufträge. Die Aufträge können entweder alle am ersten Tag begonnen werden oder erst am Tag der Bestellung. (**Tages-Restriktion**)
2. die Mischung der einzelnen Auftragspakete. Es kann bei der Produktion möglicherweise sinnvoll sein, zuerst alle Aufträge eines Tages abzuarbeiten, bevor mit den Aufträgen des nächsten Tages begonnen wird. In diesem Fall sind die Auftragspakete nicht mischbar. (**Reihenfolge-Restriktion**)
3. die verwendeten Auftragspakete.

Folgende Aufgabenstellungen werden bearbeitet:

Strategie	Aufträge beginnen am ersten Tag	Auftragspakete mischbar
S1	ja	ja
S2	nein	ja
S3	ja	nein
S4	nein	nein

Tabelle 7.3: Die verschiedenen Strategien, die bei den Simulationen verwendet werden, unterscheiden sich im frühesten Startzeitpunkt der Aufträge und deren vorgegebener Reihenfolge.

Neben der Bearbeitung aller 9 Auftragspakete werden zusätzlich noch 2 bis 8 Tagespakete optimiert, wobei jedoch immer ‚benachbarte' Tage geplant werden. Es existieren bei diesen Simulationen keine Pausen am 5. und 6. Tag, d.h. es werden auch an diesen Tagen Aufträge angenommen und ausgeführt. Falls Aufgabenstellungen bearbeitet werden, bei denen nicht alle Aufträge am ersten Tag begonnen werden dürfen, so werden die frühesten Startzeitpunkte verschoben, so dass es immer jeweils ein Auftragspaket für den 1., 2., ... Tag gibt. So wird z.B. Auftragspaket 5, das ursprünglich als frühesten Startzeitpunkt den 7. Tag hatte, bei der Optimierung der Auftragspakete 4 bis 7 als frühesten Startzeitpunkt den 2. Tag erhalten.

Verglichen werden die Ergebnisse mit denen von **[7.1]**. Benutzt wurde dort ein **CP** Ansatz mit einem Rechenzeitlimit von 1200 Sekunden auf einem PC Pentium Pro mit 200 MHz. Meist wurden die Lösungen vor Erreichen der Rechenzeitgrenze gefunden. Die Strategie Strat1(=S4) mit den zusätzlichen Nebenbedingungen findet die Lösungen dabei deutlich schneller als Strat2(=S2) und Strat3(=S3).

Bei **SA** wird das gleiche Abkühlschema wie bei den Eintagespaketen benutzt. Es werden jeweils 10 Simulationen mit 50 Sweeps pro Temperaturschritt (**_a**) und 2000 Sweeps pro Temperaturschritt (**_b**) durchgeführt.
Die für eine bestimmte Anzahl an Tagespaketen benötigte durchschnittliche Rechenzeit für 100000 Sweeps ist in **Tabelle 7.4** zu sehen.

Anzahl Tagespakete	t
1	5
2	9
3	11
4	14
5	18
6	24
7	33
8	43
9	53

Tabelle 7.4: Durchschnittliche Rechenzeiten in Sekunden für die Durchführung von 100000 Sweeps für verschiedene Anzahl an Tagespaketen auf einem Pentium mit 650 MHz.

Im Unterschied zur **Reihenfolge-Restriktion** reduziert die **Tages-Restriktion** den Lösungsraum nicht. Die Anzahl an Lösungen mit kurzer Durchlaufzeit wird jedoch erheblich reduziert. Die **Tages-Restriktion** vergrößert die Durchlaufzeit der meisten Lösungen und lässt sie bei den restlichen gleich. Es werden dabei die Lösungen verglichen, die die gleiche Ressourcenbelegung mit Aufträgen haben. Der früheste Startzeitpunkt jedes Auftrags wird nun noch zusätzlich durch die Restriktion verändert. Er kann sich dabei nur erhöhen. So sind z.B. bei allen Lösungen, die mit einem Auftrag aus einem der letzten Tagespakete auf jeder Ressource der Stufe 1 beginnen, die ersten Tage mit Produktionslücken versehen.
Eigentlich ist zu erwarten, dass die Ergebnisse bei Strategie **S1** besser oder zumindest gleich gut sind wie die Ergebnisse der anderen Strategien. Im Vergleich zu **S3** und **S4** ist der Lösungsraum bei **S1** größer und enthält den Lösungsraum der Strategien vollständig. Die Lösungsräume von **S1** und **S2** stimmen bzgl. der Anzahl überein, **S1** enthält jedoch weitaus mehr Lösungen mit kurzen Durchlaufzeiten. Der Lösungsraum der Strategie **S2** ist sehr viel größer als der der Strategie **S4** und enthält diesen vollständig. Es ist jedoch nicht zu erwarten, dass die Einschränkung der **Tages-Restriktion** in Kombination mit der **Reihenfolge-Restriktion** (**S4**) große Auswirkungen im Vergleich zur **Reihenfolge-Restriktion** alleine (**S3**) zeigt, da nicht davon auszugehen ist, dass alle Aufträge eines Tagespaketes am gleichen Tag ausgeführt werden können. Eine Entstehung von ‚Lücken' ist also zwischen den Tagen bei **S4** nicht zu erwarten.
Bei begrenzter Rechenzeit (**_a** und **_b**) ist ein reduzierter Lösungsraum möglicherweise von Vorteil bei einer lokalen Suche. Dieser Effekt könnte sich bei den Strategien **S3** und **S4** positiv auf die Lösungsqualität im Vergleich zu **S1** und **S2** auswirken.

Strategie	S1_a	S1_b	S2_a	S2_b	S3_a	S3_b	S4_a	S4_b		Strat2(=S2)	Strat3(=S3)	Strat1(=S4)
Tag												
1_2	225342	222790	229834	227442	236008	232242	232782	232587		258840	288240	248940
2_3	221590	217811	227700	225008	223121	221021	225900	224969		254100	285240	227880
3_4	220508	217287	221363	218369	219490	217690	220124	217132		247800	273300	231840
4_7	206484	205542	212442	212184	207795	206718	212184	212063		217560	235380	219300
7_8	205421	203932	209621	208800	209363	207821	209046	207863		210960	236100	209400
8_9	225436	222600	232242	227669	227100	222269	228000	227700		243900	270000	255000
9_10	221069	219190	224063	219221	224860	218590	222000	218290		252600	241200	240420
10_11	194116	192569	210939	210939	195269	192521	210939	210939		212400	212400	212400
1_3	331990	321611	346200	335624	337242	333537	339224	332211			445140	393240
2_4	304800	301769	318000	306360	307321	304469	309000	307200			381540	368100
3_7	304990	305688	322684	311242	310421	308542	313967	309184			425400	364260
4_8	289163	285546	304395	295763	291495	290218	295242	294900			364500	328560
7_9	319800	312269	332700	325121	327984	322284	325188	321000			396000	379860
8_10	303821	298421	318060	305621	305100	302369	308021	306574			370200	342000
9_11	304690	301200	316474	305643	313169	305100	309300	306490			336000	333900
1_4	414832	405118					420195	415926				540840
2_7	397239	386069					407554	399869				513000
3_8	387190	384521					399342	393436				513000
4_9	403800	394800					413142	407963				510120
7_10	397482	385500					405300	397721				460500
8_11	390832	382690					395321	393300				485400

Tabelle 7.5: Durchlaufzeiten für die Simulation von 2-,3- und 4-Tages-Paketen. Zu sehen sind die besten Ergebnisse aus jeweils 10 Simulationen mit 50 (**_a**) bzw. 2000 (**_b**) Sweeps pro Temperaturschritt für die 4 Strategien **S1–S4**. Zum Vergleich sind noch die Ergebnisse der Strategien **Strat1**, **Strat2** und **Strat3** aus **[7.1]** zu sehen. Die Strategien **Strat1-Strat3** entsprechen dabei den Strategien **S2-S4**.

Vergleich der 2- und 3-Tagespakete (Tabelle 7.5):
Ein Vergleich der Ergebnisse der Strategien **S1** und **S2** zeigt das erwartete Resultat. Bei kurzen und langen Rechenzeiten lassen sich bei **S1** immer Lösungen mit kürzerer Durchlaufzeit finden. Bei gleicher Größe des Lösungsraums existieren bei **S1** mehr Lösungen als bei **S2** mit kleiner Durchlaufzeit. Bei begrenzter Rechenzeit findet man mit **S1** schneller bessere Lösungen.
Im Vergleich zu **S3** kann **S1** in den meisten Fällen die besseren Lösungen erzeugen. Jedoch zeigt sich, dass die Reduzierung des Lösungsraums von Vorteil ist. Die Qualität der Lösungen von **S3** ist insgesamt besser als bei **S2**.
Auch der Vergleich der Lösungen **S3** und **S4** bestätigt die Annahmen. Keine der beiden Strategien ist der anderen überlegen.
Ein Vergleich von **S2** und **S4** zeigt den positiven Effekt der Reduzierung des Lösungsraums. Bei begrenzter Rechenzeit liefert **S4** in der Regel bessere Lösungen. Die Unterschiede sind jedoch nicht besonders groß.
Insgesamt zeigt sich, dass es von Vorteil sein kann, auf die **Reihenfolge-Restriktion** und die **Tages-Restriktion** zu verzichten, um bessere Lösungen zu erhalten. Sollte es jedoch von betrieblicher Seite aus nötig sein, eine der Restriktionen zu beachten, so greift man am besten auf die **Reihenfolge-Restriktion** zurück, um den Lösungsraum zu verkleinern.
Die bei den Simulationen erhaltenen Ergebnisse haben gezeigt, dass bereits bei kurzen Rechenzeiten (**_a**) meist deutlich bessere Lösungen gefunden werden als bei den vergleichbaren Strategien aus **[7.1]**. Bei den Lösungsstrategien aus **[7.1]** haben sich die zusätzlichen Nebenbedingungen als Vorteil herausgestellt, da durch den verkleinerten Lösungsraum die Rechenzeit deutlich gesenkt werden konnte.

Aufgrund der geringen Unterschiede wird bei den Simulationen mit den 4-9-Tages-Paketen nur noch mit Strategie **S1** und **S4** gearbeitet.

Vergleich der 4-Tagespakete (Tabelle 7.5):
Alle Aufträge der 4-Tagespakete können innerhalb der ersten 6 Tage produziert werden. Eine Optimierung der Tagespakete 1-8 mit der **Tages-Restriktion** würde sich bei einer Pause also in eine Optimierung der Pakete 1-4 und 7-8 ohne Pause zerlegen, wobei die gesamte Durchlaufzeit der Durchlaufzeit bei der Optimierung der Pakete 7-8 plus 6 Tage entspricht. Der Verzicht auf die Bestellpause am 5. und 6. Tag bei den Simulationen ist somit gerechtfertigt, da sonst nur redundante Informationen erzeugt würden.
Der Unterschied zwischen den Strategien **S1** und **S4** wird bei erhöhter Anzahl an zu optimierenden Tagespaketen größer. Der größere Lösungsraum wirkt sich bei der Suche positiv auf die Lösungsqualität aus. Der Vorteil der schnelleren Suche bei kleinem Lösungsraum ist zu vernachlässigen.

Vergleich der 5-9-Tagespakete:

Strategie	S1_a	S1_b	S4_a	S4_b		Strat1(=S4)
Tag						
1_7	508011	498418	514341	513427		670980
2_8	483138	472990	488242	486190		630000
3_9	508450	497369	515595	508154		674700
4_10	483057	469115	496721	485711		589860
7_11	486863	468190	499121	486416		603900
1_8	593895	575621	609177	593363		767280
2_9	597220	581646	610454	597911		780360
3_10	585090	566621	598645	584242		728100
4_11	576562	552515	586563	574642		722160
1_9	715262	696437	731663	710634		922440
2_10	684221	664367	692013	676169		859500
3_11	676358	656006	682512	671700		871500
1_10	801000	767511	801195	785242		1005780
2_11	775911	750442	786995	766316		1002900
1_11	888217	851870	887396	881528		1149180

Tabelle 7.6: Durchlaufzeiten für die Simulationen der 5-9-Tagespakete mit den Strategien **S1** und **S2** mit 50 (**_a**) bzw. 2000 (**_b**) Sweeps pro Temperaturschritt. Zusätzlich sind die Ergebnisse der Strategie **Strat1** aus **[7.1]** gegeben.

Bei den Durchlaufzeiten kann man nun deutlich sehen (**Tabelle 7.6**), dass Strategie **S1** immer bessere durchschnittliche Ergebnisse liefert als Strategie **S4**. Mit **S4** konnte nur in einem Fall, bei kurzen Rechenzeiten, eine kürzere durchschnittliche Durchlaufzeit erreicht werden. Während sich bei der Strategie **S4** bei höheren Rechenzeiten nur maximal 2,9% kürzere Durchlaufzeiten erreicht werden, ist die Verbesserung bei **S1** bei höheren Rechenzeiten bei 4,2% im Vergleich zu den kurzen Rechenzeiten.
Im Vergleich zu den Ergebnissen aus **[7.1]** zeigt sich, dass die beiden Strategien **S1** und **S2** der Strategie **Strat1** deutlich überlegen sind. Je größer die Anzahl der bearbeiteten Tagespakete, desto größer wird die Verbesserung gegenüber den Ergebnissen der Strategie **Strat1**.

Vergleich der Strategie S5 mit einer Pause bei den Bestellungen am 5. und 6. Tag

Es soll nun untersucht werden, ob eine Berücksichtigung der Pausen bei der Planung sinnvoll ist. Dazu werden bei der Strategie **S5** die frühesten Startzeitpunkte so verschoben, dass am 1., 2., 3., 4., 7., 8., 9., 10., 11. Tag jeweils ein Auftragspaket beginnen kann. Verglichen werden die Ergebnisse mit denen einer neuen Strategie **S6**. Die Strategie **S6** entspricht der Strategie **S4**, nur die Durchlaufzeit wird um 6 Tage (=518400 Sekunden) erhöht. Die Durchlaufzeit bei der Planung von Tagespaket 7 mit **S6** kann somit mit der Durchlaufzeit bei der Planung mit Tagespaket 1_7 mit **S5** verglichen werden. In **Tabelle 7.7** sind die Durchlaufzeiten für die Planung von einigen Tagespaketen mit **S5** und **S6** zu sehen.

Strategie	S5_a	S5_b		Tag	S6_a	S6_b
Tag						
1_7	644321	644063		7	644321	644063
2_8	629069	628500		8	628800	627269
3_9	662069	660300		9	660990	659069
4_10	629321	629321		10	629321	629321
7_11	644321	642939		11	642939	642939
1_11	1019100	1006984		7_11	1017521	1004816

Tabelle 7.7: Durchlaufzeiten bei den Simulationen mit Strategien **S5** und **S6** mit 50 (**_a**) bzw. 2000 (**_b**) Sweeps pro Temperaturschritt.

In einigen Fällen wurden mit den beiden Strategien gleiche Durchlaufzeiten erzeugt, meistens waren die Durchlaufzeiten mit **S6** jedoch kürzer. Dies liegt vor allem daran, dass die zur Verfügung stehende Rechenzeit für die Planung kleinerer Auftragsmengen genutzt wurde. Der Unterschied bei den Durchlaufzeiten, bei begrenzter Rechenzeit, ist jedoch nicht besonders groß, da es immer sehr leicht möglich ist, die 4-Tagespakete innerhalb der ersten 6 Tage zu produzieren. Meist wird bei den Simulationen mit **S4** eine Durchlaufzeit für die 4-Tagespakete gefunden, die mindestens 25% kleiner ist als 6 Tage (=518400). Bei ausreichender Rechenzeit kann davon ausgegangen werden, dass sich mit **S5** immer die gleichen Durchlaufzeiten erzeugen lassen.

7.4 Zusammenfassung

In diesem Kapitel wurde eine Problemstellung aus dem Bereich der **Flexible-FlowShop**-Probleme bearbeitet. Dabei wurden verschiedene Strategien zur Erstellung der Lösung untersucht. Eine Veränderung der Planung, von minutengenau zu sekundengenau, hat bei dem verwendeten Verfahren im Unterschied zu den Verfahren aus **[7.1]** keine Auswirkung auf die Rechenzeit. Während in **[7.1]** zusätzliche Bedingungen hilfreich waren, um den Suchraum einzuschränken und damit die Suche zu beschleunigen, nutzt das in dieser Arbeit verwendete Verfahren die Freiheit aus, um noch bessere Durchlaufzeiten zu erhalten. Insgesamt gesehen ist **SA** mit der vorgestellten Modellierung deutlich besser als die in **[7.1]** verwendeten Verfahren **CP** und **MP**. Der Unterschied wird umso deutlicher, je größer die zu bearbeitenden Instanzen sind.

Kapitel 8

Das Benchmark Bench01

Der hier vorgestellte Benchmark (Bench01 **[8.1]**) wurde 1997 von der SAP AG erstellt, um verschiedene Verfahren auf ihre Praxistauglichkeit hin zu überprüfen. Der Vergleich wurde in **[8.2]** dokumentiert. Es handelt sich dabei um eine konstruierte Problemstellung, die jedoch viele der in der Prozessindustrie geforderten Funktionalitäten aufweist.

8.1 Problemstellung

Betrachtet wird folgender Produktionsablauf:

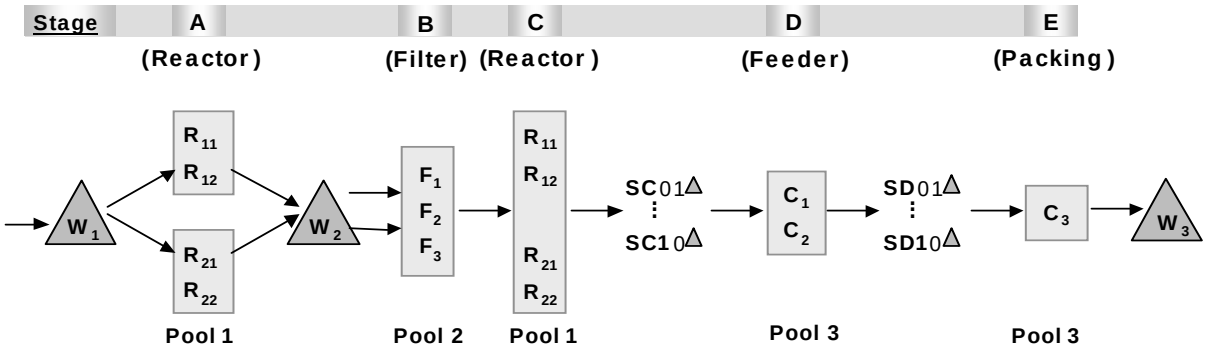


Abbildung 8.1: Struktur des Fertigungsprozesses.

a) Zur Verfügung stehende Ressourcen

- 4 *Reactors:* R_{11}, R_{12}, R_{21}, R_{22}
- 3 *Filters:* F_1, F_2, F_3
- 2 *Feeders:* C_1, C_2
- 1 *Packaging Station:* C_3
- 3 *Pools* mit Arbeitern: $Pool_1$, $Pool_2$, $Pool_3$
- 3 Lager ohne Kapazitätsbeschränkung: *Warehouses* W_1, W_2, W_3
- 20 Lager mit Kapazitätsbeschränkung (2 Einheiten): *Storages* SC_x, SD_x $(1 \le x \le 10)$

b) Rezept

Jedes Endprodukt E_x ($1 \le x \le 10$) befolgt bei seiner Herstellung ein bestimmtes „*Rezept*“ (**Abbildung 8.2**). Zur Herstellung des Endproduktes sind verschiedene Zwischenprodukte nötig. Die Produktion erfolgt in festgelegten Losgrößen.

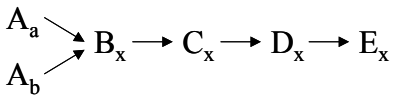


Abbildung 8.2: Rezept zur Produktion eines Endproduktes ($1 \le a \le 4$, $6 \le b \le 9$, $1 \le x \le 10$).

c) Ablauf der Produktion

Stufe A: Hier werden die Zwischenprodukte A_a und A_b ($1 \le a \le 4$, $6 \le b \le 9$) hergestellt, wobei A_a nur auf *Reactor* R_{11} oder R_{12}, A_b nur auf *Reactor* R_{21} oder R_{22} produziert werden kann. Die dazu nötigen Rohstoffe werden dem *Warehouse* W_1 entnommen, die fertigen Zwischenprodukte nimmt *Warehouse* W_2 auf.

Stufe B: Hier werden die Zwischenprodukte B_x ($1 \le x \le 10$) auf den *Filters* F_1, F_2, F_3 hergestellt. Dazu benötigt man für eine Einheit B_x eine Einheit A_a und eine Einheit A_b aus dem *Warehouse* W_2. Die B_x müssen nach ihrer Fertigstellung direkt in **Stufe C** weiterverarbeitet werden.

Stufe C: Hier werden die Zwischenprodukte C_x ($1 \le x \le 10$) auf den *Reactors* R_{11}, R_{12}, R_{21}, R_{22} hergestellt. Eine Einheit C_x benötigt eine Einheit B_x. Diese wird direkt aus **Stufe B** übernommen. Die fertigen C_x werden in den *Storages* SC_x gelagert.

Stufe D: Hier werden die Zwischenprodukte D_x ($1 \le x \le 10$) auf den *Feeders* C_1 oder C_2 hergestellt. Eine Einheit D_x benötigt eine Einheit C_x aus der *Storage* SC_x, die fertigen D_x werden in den *Storages* SD_x gelagert.

Stufe E: Hier werden die Endprodukte E_x ($1 \le x \le 10$) auf der *Packaging Station* C_3 hergestellt. Eine Einheit E_x benötigten eine Einheit D_x aus der *Storage* SD_x entnommen, die fertigen E_x kommen ins *Warehouse* W_3.

Für jedes Zwischenprodukt A_a, A_b, B_x, C_x, D_x und jedes Endprodukt E_x wird bei der Produktion jeweils eine der angegebenen Maschinen und eine bestimmte Anzahl Arbeiter aus dem dafür geeigneten *Pool* für eine bestimmte Zeitspanne benötigt.

Eine Maschine kann nicht mehrere Produkte gleichzeitig fertigen. Bei der Herstellung der Zwischenprodukte muss man noch *Rüstzeiten* beachten. Diese hängen von der Maschine und den Produktfamilien der, bei der Produktion auf dieser Maschine, aufeinanderfolgenden Zwischenprodukte ab. Die *Pools* können bei mehreren verschiedenen Maschinen gleichzeitig eingesetzt werden, solange die Gesamtkapazität des *Pools* eingehalten wird.

Die Daten der Aufgabenstellung sind in **Anhang D.2** zu finden.

Gegeben ist nun eine Liste aus Aufträgen für die Endprodukte E_x mit bestimmten Lieferzeitpunkten. Zusätzlich zu der in der Aufgabenstellung gegebenen Liste (Instanz 0) sind noch weite-

re Instanzen bearbeitet worden. Diese sind zusammen mit den Ergebnissen im **Anhang D.3** und **Anhang D.4** zu finden.
Im Folgenden sind alle Zeiten in der Einheit Minuten zu verstehen. Die Produktion beginnt zum Zeitpunkt 0.

d) Aufgabe

Zu minimieren sind bei diesem Optimierungsproblem unter Einhaltung der Nebenbedingungen der Lagerkapazitäten und der *Pool*-Arbeiter die

- **Verspätung (*delay*):**
  die Summe aus allen Lieferverzögerungen, die
- **Durchlaufzeit (*makespan*)** sowie die
- **Rüstzeiten (*setup*)**

Die Minimierung der Verspätung hat bei der Optimierung die höchste Priorität. Es wird im Folgenden davon ausgegangen, dass bei jeder Lösung die Lieferzeitpunkte eingehalten werden. Für die Prioritäten der Durchlaufzeit (P_M) und Rüstzeit (P_S) sollten drei Fälle untersucht werden:

$$P_M > P_S, \quad P_M = P_S, \quad P_M < P_S. \tag{8.1}$$

e) Nebenbedingungen

- **Lagerkapazitäten**
  Die Lagerkapazität der SC- und SD-Lager muss eingehalten werden, d.h. es darf sich nur eine begrenzte Anzahl (2 Einheiten) eines Zwischenprodukts in dem dafür vorgesehenen Lager befinden.
- **Multiressourcen**
  Die Anzahl der Arbeiter in den *Pools* ist begrenzt, d.h. ein Zwischenprodukt oder Endprodukt muss nicht nur auf eine freie Maschine warten, sondern eventuell auch noch auf die dafür benötigten Arbeitskräfte aus den *Pools*.

8.2 Modell

a) Ressourcen

Es sind 10 Primärressourcen (*Reactors*, *Filters*, *Feeders* und die *Packaging Station*), 3 Multiressourcen (*Pools*), 18 Silos ohne Kapazitätsbeschränkung (*Warehouses*) und 20 Silos mit Kapazitätsbeschränkung (*Storages*) vorhanden.
Dabei werden 3 Rüstzeitmatrizen definiert, wobei eine für alle *Reactors* gültig ist, die zweite für alle *Filters* und die dritte für die *Feeders*.
Primärressourcen: R_{11}, R_{12}, R_{21}, R_{22}, F_1, F_2, F_3, C_1, C_2, C_3
Multiressourcen: $Pool_1$, $Pool_2$, $Pool_3$
Silos: W_{Aa}, W_{Ab}, W_{Ex}, SC_x, SD_x $(1 \le x \le 10, 1 \le a \le 4,\ 6 \le b \le 9)$

b) Aktivitäten

Es existieren insgesamt 8 verschiedene Aktivitäten der **Stufe A** und jeweils 10 verschiedene der **Stufen B**, **D** und **E**, insgesamt also 38 verschiedene Aktivitäten. Pro Auftrag für ein Endprodukt E_x werden 5 Aktivitäten benötigt. Aufgrund des fehlenden Lagers zwischen **Stufe B** und **C** wird für die Produktionsvorgänge dieser beiden Stufen nur eine Aktivität benötigt, deren Modi durch einen maximalen Zeitabstand miteinander verbunden sind. In **Anhang D.1** sind die Aktivitäten für einen Auftrag gegeben.

c) Planauftrag

Es existiert eine Liste an Aufträgen für 10 verschiedene Endprodukte. Für jeden dieser Aufträge werden jeweils 5 Aktivitäten erzeugt, wobei x als Platzhalter für den Index des Endproduktes steht ($1 \leq x \leq 10$). Der Lieferzeitpunkt kann für jeden Auftrag gewählt werden und ist nicht vom Endprodukt abhängig.

Bei dieser Modellierung ist nun nicht festgelegt, welche Aktivität die benötigten Vorprodukte für welchen Auftrag herstellt. Letztendlich existiert für jedes der 38 verschiedenen Produkte eine Bestellmenge, die von Aktivitäten gedeckt wird. Man bezeichnet dies als *variables Pegging*. In **Kapitel 8.5.1** wird ein Modell mit *fixiertem Pegging* vorgestellt.

8.3 Theoretische Überlegungen zu unteren Schranken

a) Verspätungen

Die Lieferzeitpunkte sollten bei dieser Problemstellung immer eingehalten werden. Die untere und obere Schranke liegt hier bei jeder Instanz bei 0.

b) Rüstzeiten

Die untere Grenze lässt sich bestimmen durch die Betrachtung der einzelnen Produktionsstufen. Es müssen die 4 existierenden Rüstfamilien in jeder Produktionsstufe geplant werden. Eine der Lösungen mit minimaler Rüstreihenfolge ist in **Tabelle 8.1** zu sehen.

Primärressource	Rüstreihenfolge	Primärressource	Rüstreihenfolge
R_{11}	F3-F4	F_1	F1-F2-F3-F4
R_{12}	F1-F2	F_2	F4
R_{21}	F3-F4	F_3	F4
R_{22}	F1-F2	C_1	F1-F2-F3
		C_2	F4

Tabelle 8.1: Optimale Verteilung der 4 Rüstfamilien auf die Primärressourcen.

Stufe A, **C** muss aufgeteilt werden in R_{11}, R_{12} und R_{21}, R_{22}, da nicht alle Aktivitäten auf allen Primärressourcen der Produktionsstufe gefertigt werden können. **Stufe B** weist hohe Rüstzeiten

innerhalb der Rüstfamilie F4 auf. Es ist deshalb günstig, F4 auf allen Primärressourcen der **Stufe B** zu planen.

Als minimale Rüstzeit ergibt sich dann bei Instanz 0 eine Rüstzeit von 360 Minuten. Eine Lösung mit minimaler Rüstzeit ist in **Abbildung 8.3** zu sehen. Dabei werden jedoch die Lieferzeitpunkte nicht eingehalten! Bei der gewählten Zeitskalierung sind die einzelnen Rüstzeiten leider nicht zu erkennen. Allerdings sieht man sehr deutlich, dass eine solche Lösung sehr viele Lücken in der Produktion und somit eine hohe Durchlaufzeit aufweist.

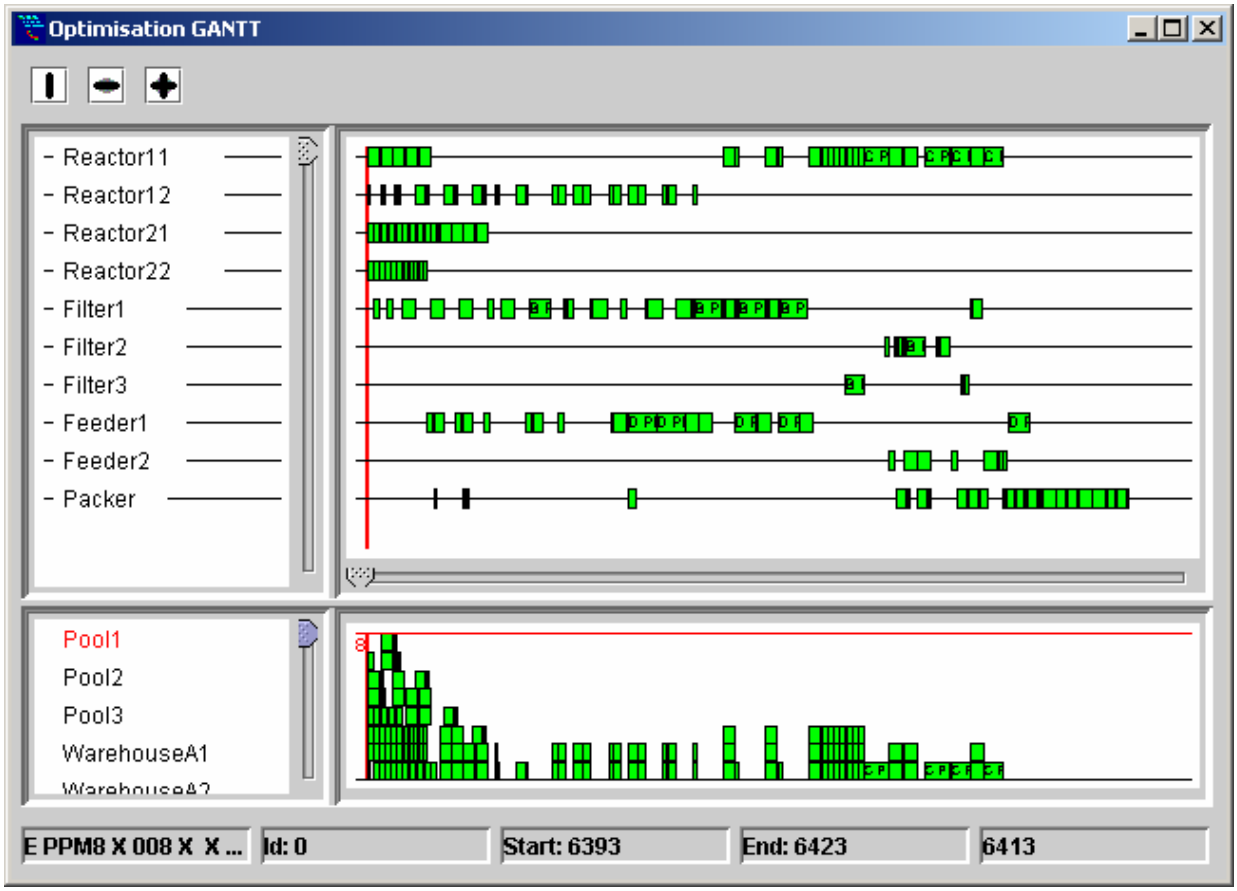


Abbildung 8.3: Zu sehen ist eine Lösung der Instanz 0 mit minimaler Rüstzeit. Rüstzeit 360 Minuten, Durchlaufzeit 6423 Minuten, Lieferzeitpunkte sind verletzt!

c) Durchlaufzeit

Bei der Durchlaufzeit lässt sich eine untere Grenze bestimmen durch die Betrachtung der Auslastung der Primärressourcen. Dazu wird jeder Produktionsstufe die minimale Produktionszeit und die minimale Rüstzeit innerhalb der Stufe aufsummiert und durch die Anzahl der Primärressourcen der Stufe geteilt. **Stufe A** und **C** werden dabei zu einer zusammengefasst, da sie aus den gleichen Primärressourcen bestehen. Zusätzlich zur Auslastung muss noch der früheste Startzeitpunkt und die minimale restliche Laufzeit zum Beenden eines Auftrags nach der Produktionsstufe beachtet werden. Es zeigt sich, dass **Stufe E** die bestimmende Größe ist. Ist eine Menge M an Aufträgen i vorhanden, deren früheste Startzeitpunkte auf einer Produktionsstufe durch S_i und

die Produktionszeit durch P_i gegeben ist, so bestimmt sich die minimale Durchlaufzeit D_{min} der **Stufe E** durch

$$D_{\min} = MIN_{N \subseteq M}\left(MIN_{i \in N}(S_i) + \sum_{i \in N} P_i \right). \qquad \textbf{(8.2)}$$

Es kann für die Instanz 0 eine untere Grenze von 1981 Minuten bestimmt werden. Im Unterschied zur minimalen Rüstzeit konnte hier jedoch keine Lösung gefunden werden, die diese untere Grenze erreicht. Es ist fraglich, ob eine solche Lösung existiert, da bisher die restlichen Bedingungen, z.B. Multiressourcen, nicht beachtet wurden. Es soll nun versucht werden, einen Plan mit der berechneten minimalen Durchlaufzeit zu finden, unter der Voraussetzung, dass alle Rüstzeiten 0 sind und keine Lieferzeitpunkte existieren. Auch mit dieser Vereinfachung konnte kein Plan mit der minimalen Durchlaufzeit gefunden werden. Betrachtet man sich einen der besten auf diese Weise erzielten Pläne (**Abbildung 8.4**), so sieht man, dass die Multiressourcen ausgelastet sind. Diese reichen zwar bei der Problemstellung mit Rüstzeiten - letztere benötigen keine Kapazität der Pools - und gleichmäßig ausgelasteten Primärressourcen aus, sind hier jedoch am Anfang der Produktion mit der Bereitstellung der Vorprodukte für die **Stufe E** überlastet. Somit entstehen Lücken am Anfang der Produktion auf **Stufe E** und verlängern die Durchlaufzeit.

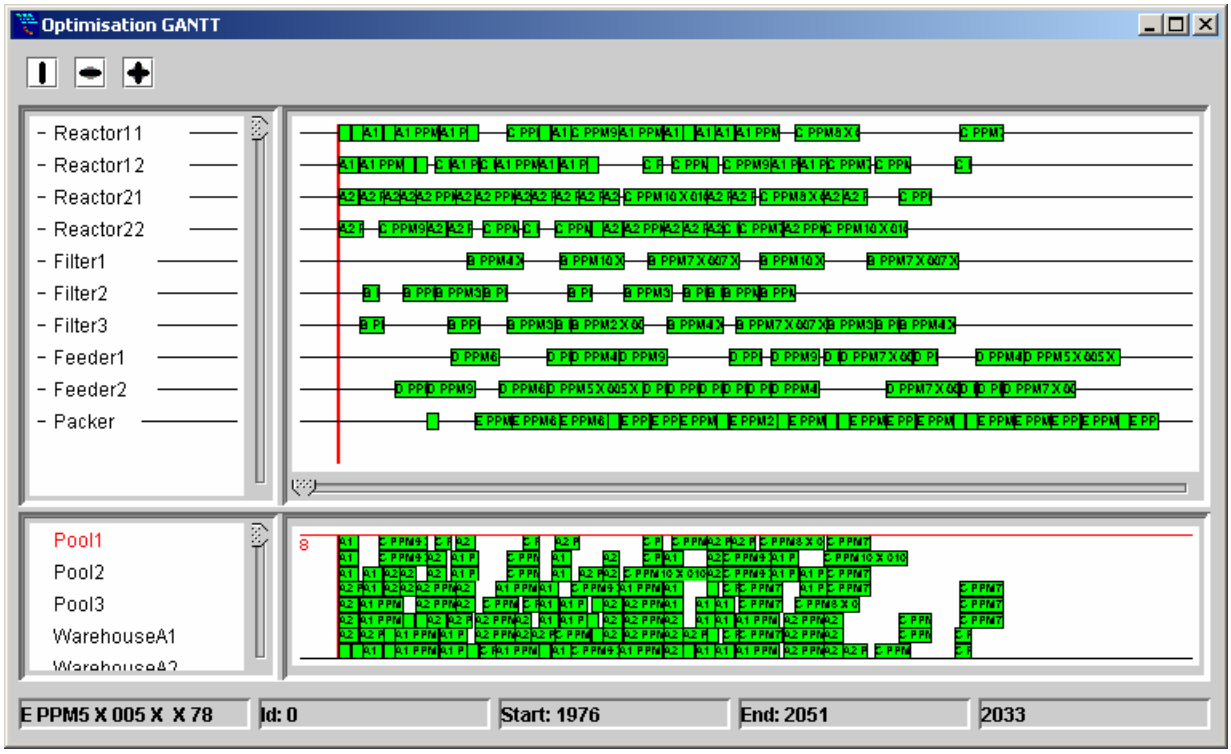


Abbildung 8.4: Zu sehen ist eine Lösung der Instanz 0 mit minimaler Durchlaufzeit (2051). Rüstzeiten und Lieferzeitpunkte wurden ignoriert.

8.4 Ergebnisse

In diesem und den folgenden Kapiteln werden die Ergebnisse zur Problemstellung **Bench01** in Abbildungen oder tabellarisch vorgestellt. Bei den Abbildungen ist meist in der Horizontalen die Rechenzeit in Anzahl Sweeps pro Temperaturschritt (100, 1000, 5000 und 10000) und in der Vertikalen die Lösungsqualität zu sehen. Bei der Lösungsqualität wird dabei unterschieden zwischen der reinen Durchlaufzeit (**M**akespan), der reinen Rüstzeit (**S**etup) und der Summe beider (**M+S**) (siehe **Kapitel 8.1 (8.1)**). Die Gewichte **M:S** wurden dabei folgendermaßen gesetzt: **M**: 10:1, **M+S**: 5:5 und **S**: 1:10. Bei den einzelnen Punkten handelt es sich jeweils um den Durchschnitt über 10 Lösungen. Werden die Ergebnisse tabellarisch gezeigt, dann ist neben dem durchschnittlichen Wert (**av**) aus 10 Simulationen noch die Standardabweichung (**dev**) zu sehen. Es wird im Folgenden die Kombination aus durchschnittlicher Lösungsqualität und Rechenzeit als Ergebnis oder Lösung bezeichnet.
Bei den Simulationen wird stets **SA** verwendet und von einer Anfangstemperatur von 2000 die Temperatur in 300 Schritten logarithmisch mit dem α–Faktor 0.98 gekühlt. Dabei können für die Instanzen mit 25 Aufträgen (z.B. Instanz 0) 100000 Sweeps in 56 Sekunden auf einem Pentium mit 650MHz erzeugt werden. Eine Simulation mit 300 Temperaturschritten und jeweils 1000 Sweeps dauert somit 168 Sekunden.

8.4.1 Vergleich verschiedener Nachbarschaften

Zuerst werden verschiedene Nachbarschaften getestet. Folgende Nachbarschaftsoperatoren werden verwendet:

(1) Change
(2) Push1
(3) PushChange
(4) Swap
(5) Lin2OptSameResource

Inwieweit diese Operatoren für die Optimierung der ursprünglichen Problemstellung günstig sind, wird anhand der folgenden Nachbarschaften untersucht:

N1 mit **(1), (2), (3), (4), (5)**
N2 mit **(1), (2), (3), (4)**
N3 mit **(1), (3), (5)**
N4 mit **(2), (4), (5)**
N5 mit **(1), (3)**

Bei einer Simulation wird bei einem Sweep jeweils ein Operator der Nachbarschaft ausgewählt, wobei jeder die gleiche Auswahlwahrscheinlichkeit besitzt, dann werden die zu verändernden Positionen zufällig bestimmt.

Die Ergebnisse werden exemplarisch an der gegebenen Instanz 0 diskutiert. 3 Aufgabenstellungen werden zu dieser Instanz untersucht (**M**, **S**, **M+S**). Die Lieferzeitpunkte müssen bei jeder Lösung eingehalten werden.
In **Abbildung 8.5** (linke Spalte) sind die durchschnittlichen Ergebnisse aus jeweils 10 Simulationen für verschiedene Anzahl an Sweeps pro Temperaturschritt für die verschiedenen Nachbarschaften und Aufgabenstellungen zu sehen.

N4 schneidet deutlich schlechter ab als die anderen. Es ist somit darauf zu achten, zumindest einen der beiden Operatoren **(1)** und **(3)** in die Nachbarschaft aufzunehmen. Dies ist eigentlich naheliegend, denn diese beiden Operatoren sind die einzigen, die für eine gleichmäßige Auslastung der Ressourcen innerhalb einer Produktionsstufe sorgen. Ohne diese Operatoren ist die Lösung stark von der anfangs gewählten Zuordnung der Aktivitäten zu den Ressourcen abhängig.
Die beiden Nachbarschaften **N3** und **N5** sind bei größeren Rechenzeiten (Anzahl Sweeps pro Temperaturschritt) schlechter als **N1** und **N2**. Es ist somit nicht ausreichend, nur die Operatoren **(1)** und **(3)** zu nutzen. Die gebildete Nachbarschaft ist nicht ausreichend. Es erscheint sinnvoll, eher Operatoren hinzuzufügen, die kleine Änderungen an der Konfiguration vornehmen, **(2)** oder **(4)**, als solche, die die Konfiguration in einem größeren Bereich umordnen, **(5)**.
Welche der beiden Nachbarschaften **N1** und **N2** die bessere ist, und damit die Aussage, ob Operator **(5)** günstig ist, hängt von der Aufgabenstellung und der zur Verfügung stehenden Rechenzeit ab.
Es kann somit nur davon ausgegangen werden, dass neben den Operatoren **(1)** und **(3)** die Operatoren **(2)** und **(4)** als günstig für diese Aufgabenstellung zu werten sind. Operator **(5)** wirkt sich nicht immer positiv auf die Optimierung aus.
Operator **(5)** ist bei anderen Optimierungsproblemen meist sehr erfolgreich, kann hier jedoch nicht vollständig überzeugen. Möglicherweise liegt es daran, dass er hier keine Systemeigenschaften ausnutzen kann. Während er bei einem **TSP** nur zwei Kanten in der ‚Tour' zerschneidet, ändert er hier den Produktionsplan evtl. zu stark. Betrachtet man sich die einzelnen Rüstmatrizen der Maschinen, so sieht man, dass diese nicht symmetrisch sind, d.h. eine Anwendung des Operators auf eine Aktivitätsreihenfolge A-B-C-D-E in der Weise, dass A-D-C-B-E entsteht, ersetzt nicht nur die Rüstzeiten A-B und D-E, sondern verändert auch die Rüstzeiten B-C und C-D in die von C-B und D-C. Bei Produktionsplanungsproblemen mit symmetrischen Rüstmatrizen erweist sich der Operator als sehr viel günstiger.
Nachbarschaften, die mit anderen, in **Kapitel 3.2.5** vorgestellten Operatoren gebildet werden, zeigen keine durchschlagenden Erfolge. Es konnten zwar ähnlich gute Ergebnisse erzielt werden, jedoch keine großen Verbesserungen. Auf diese zusätzlichen Operatoren wird deshalb verzichtet.

Bei den bisherigen Experimenten wurde nun jedoch nur festgestellt, welche Operatoren zu verwenden sind, nicht jedoch, mit welcher Auswahlwahrscheinlichkeit. Bisher wurden nur Ergebnisse gezeigt, bei denen die Auswahlwahrscheinlichkeit jedes Operators der Nachbarschaft gleich ist. Auf die Testreihen zu verschiedenen Auswahlwahrscheinlichkeiten soll hier verzichtet werden, da es nach Meinung des Autors dann zu einer viel zu problem- (bzw. instanz-) spezifischen Anpassung des Verfahrens kommen würde. Es könnte dann nicht mehr davon ausgegangen werden, dass die Ergebnisse auf andere Instanzen der Problemstellung übertragen werden können. Ein Anwender

verfügt jedoch nicht über genügend Zeit, bei der Annahme eines neuen Auftragspakets (Instanz) regelmäßig diese Forschungsarbeit durchführen zu lassen. Vielmehr soll nun eine Vorgehensweise vorgestellt werden, die die Auswahlwahrscheinlichkeiten, und damit auch die Verwendung der Operatoren, selbstständig reguliert. Der Autor geht nicht davon aus, dass dies zu besseren Resultaten führt, als eine an eine Probleminstanz speziell angepasste Auswahlwahrscheinlichkeit es tun würde. Der Vorteil liegt eher darin, dass es dann nicht mehr nötig sein wird, die Operatoren selbst auszuwählen, und eine ‚falsche' Auswahl somit verhindert wird.

Verglichen wird nun das durchschnittliche Ergebnis des adaptiven Verfahrens **AD** (vgl. **Kapitel 3.2.5.4**) aus jeweils 10 Simulationen für verschiedene Rechenzeiten und den 3 verschiedenen Aufgabenstellungen mit dem Durchschnitt aller zuvor getesteten Nachbarschaften **avA** und dem Durchschnitt der ‚besseren' Nachbarschaften **avB** (**N1**, **N2**, **N3**, **N5**). Bei den ‚besseren' Nachbarschaften wird auf **N4** verzichtet, da es eigentlich offensichtlich ist, dass die Operatoren **(1)** und/oder **(3)** benötigt werden, um gute Resultate zu erhalten. Die Ergebnisse werden in **Abbildung 8.5** (rechte Spalte) gezeigt.
Es zeigt sich, dass die Idee der adaptiven Anpassung der Auswahlwahrscheinlichkeiten der einzelnen Operatoren nicht fehlschlägt. Im Vergleich zum Durchschnitt aller zuvor getesteten Nachbarschaften schneidet das adaptive Verfahren für alle verwendeten Rechenzeiten deutlich besser ab. Vergleicht man die Ergebnisse mit **avB**, so zeigt sich, dass hier erst bei größeren Rechenzeiten mit einer Verbesserung zu rechnen ist, diese jedoch nicht bemerkenswert ist.
Verglichen mit den einzelnen Nachbarschaften kann die adaptive nicht immer Verbesserungen erzielen. Hier zeigt sich, dass die adaptive Regelung nicht zu den besten Resultaten führt. Manche Operatoren werden durchaus benötigt, um die Nachbarschaft zu vergrößern, ohne direkt zu Verbesserungen zu führen. Bei diesen Operatoren kann dann davon ausgegangen werden, dass sie im Lösungsraum verschiedene tiefe Täler über einen Höhenzug verbinden. Ohne diese Verbindung wäre es den anderen Operatoren nicht möglich, in wenigen Schritten, von einem dieser Täler in ein anderes, möglicherweise tieferliegenderes zu wandern. Lokal gesehen muss dabei jedoch zwischenzeitlich immer eine gewisse Verschlechterung in Kauf genommen werden.

Im Folgenden wird, falls nicht anders erwähnt, nur noch mit der Nachbarschaft **N2** gearbeitet.

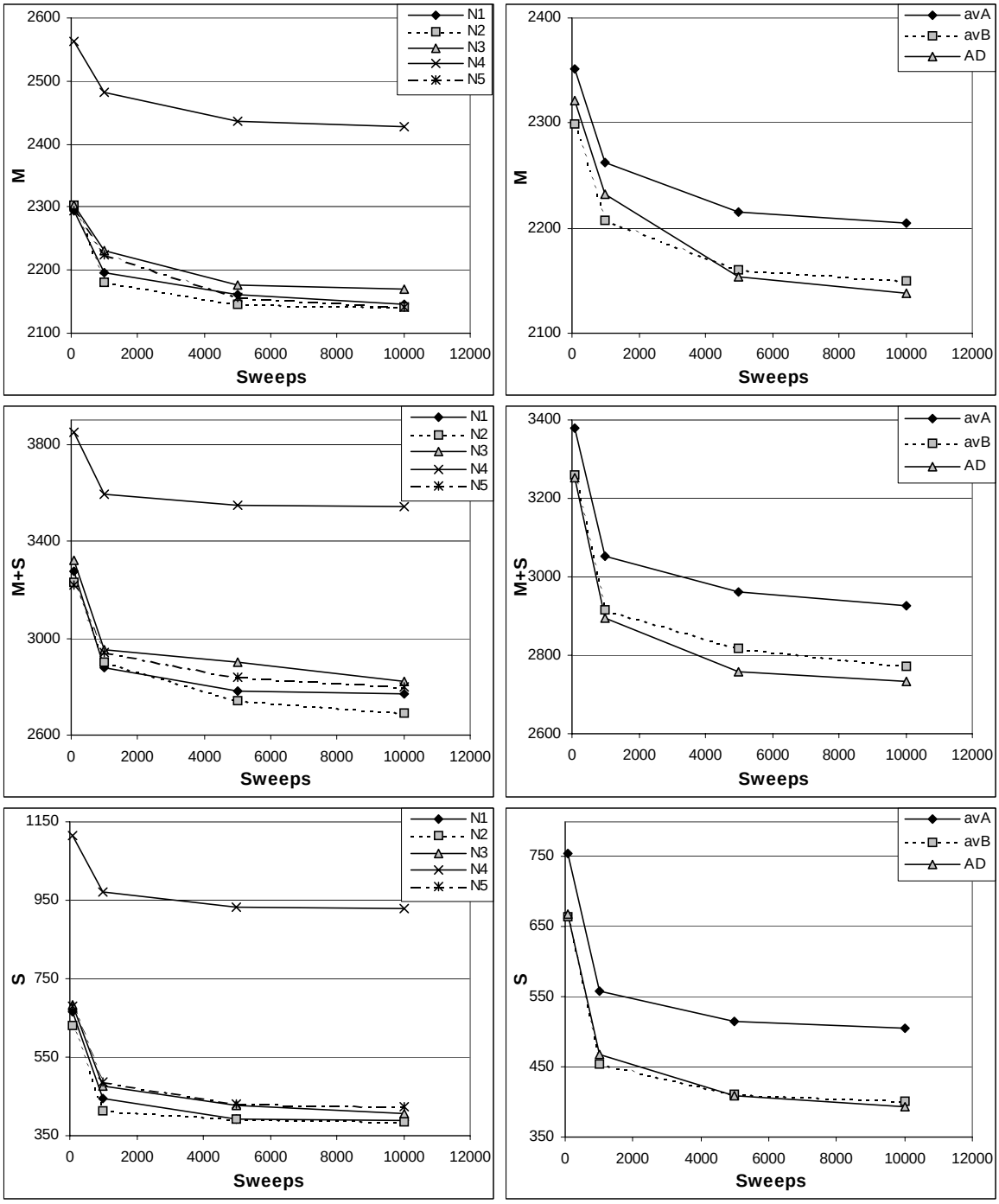


Abbildung 8.5: Es werden die durchschnittlichen Ergebnisse von Simulationen mit den 5 verschiedenen Nachbarschaften gezeigt (linke Spalte). In der rechten Spalte sind die durchschnittlichen Ergebnisse der Simulationen mit allen Nachbarschaften (**avA**), den besseren Nachbarschaften (**avB**) und der adaptiven Nachbarschaft (**AD**) zu sehen.

8.4.2 Ergebnisse zu verschiedenen Prioritäten der Kriterien

Im Folgenden soll untersucht werden, wie sich die einzelnen Kriterien aufeinander auswirken. Es werden verschiedene Instanzen erstellt, bei denen der Lieferzeitpunkt verändert wird, um den Einfluss der harten Nebenbedingung der Lieferzeitpunkte auf die anderen Optimierungskriterien zu untersuchen. Dazu wird jedem Auftrag einer Instanz der gleiche Lieferzeitpunkt gesetzt. Die so aus der Instanz 0 erzeugten Instanzen (1, 2 und 3) haben die Lieferzeitpunkte 2280, 2520 und 3960. Zu den beiden anderen Kriterien (Durchlaufzeit, Rüstzeit) wird die *Pareto-Front* der besten gefundenen Lösungen bei vorgegebener Rechenzeit für Instanz 0 aufgezeichnet.

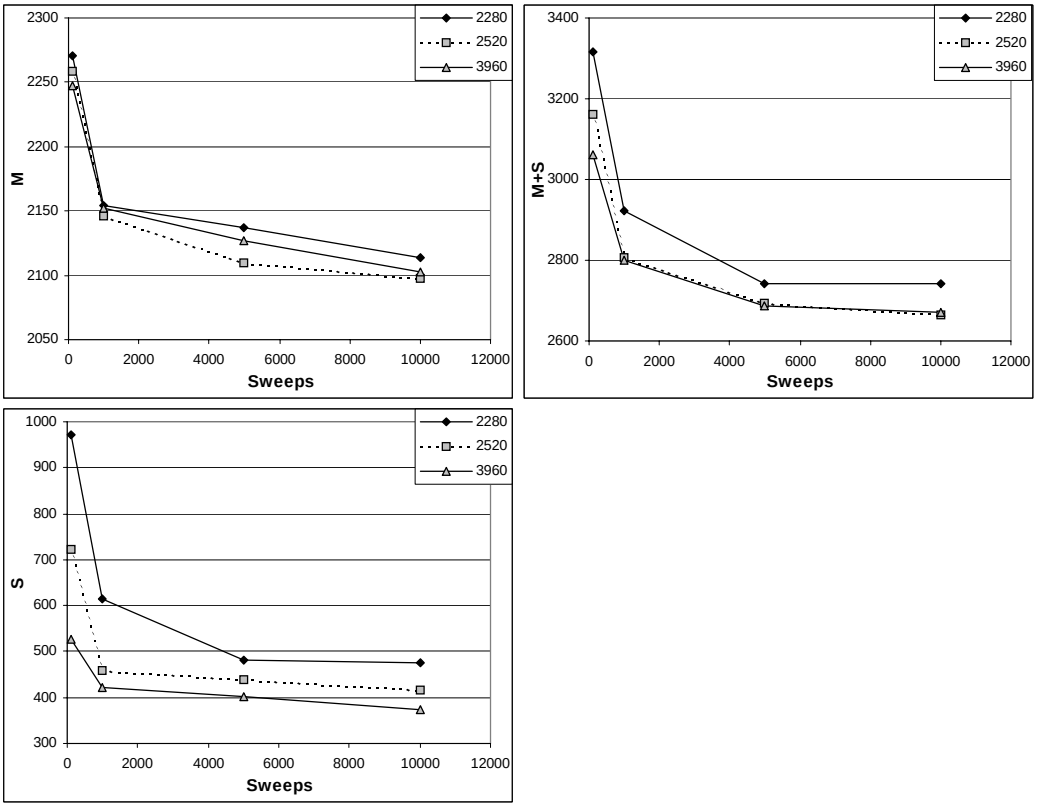


Abbildung 8.6: Zu sehen sind die durchschnittlichen Lösungen für die Optimierung mit verschiedenen Lieferzeitpunkten.

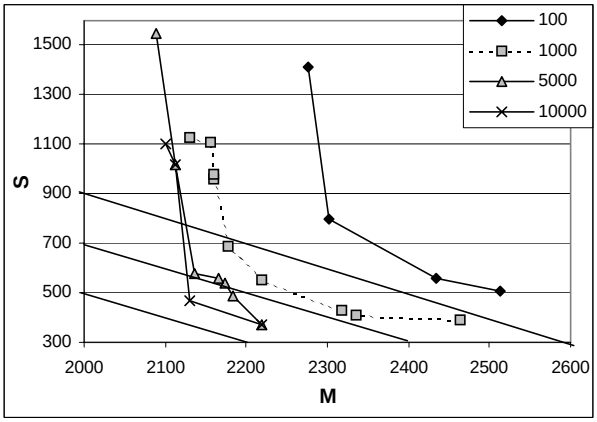


Abbildung 8.7: Pareto-Fronten verschiedener Rechenzeiten (100-10000 Sweeps pro Temperaturschritt) für Instanz 0.

Das Setzen engerer Lieferzeitpunkte (**Abbildung 8.6**) hat bei der Optimierung der Durchlaufzeit (**M**) fast keinen Einfluss auf die Qualität der Ergebnisse. Die Unterschiede bei den durchschnittlichen Lösungen sind kleiner 1%. Es fällt jedoch auf, dass das Setzen von Lieferzeitpunkten bei mittleren Rechenzeiten (5000 Sweeps) zu besseren Resultaten führen kann. Bei der Optimierung der Rüstzeiten (**S**) ergibt sich ein anderes Bild. Je enger die Grenzen gesetzt sind, umso schlechter sind die durchschnittlichen Ergebnisse für alle Rechenzeiten. Bei der gleichzeitigen Optimierung der Durchlaufzeit und der Rüstzeiten (**M+S**) sind die durchschnittlichen Ergebnisse beim Setzen eines engen Lieferzeitpunktes deutlich schlechter als beim Setzen ‚weicher' Lieferzeitpunkte. Insgesamt kann somit festgestellt werden, dass die Optimierung der Lieferzeitpunkte und der Durchlaufzeit einander nicht stören, wohingegen dies bei der Optimierung der Lieferzeitpunkte und der Rüstzeiten der Fall ist. Rüstzeiten und Lieferzeitpunkte sind zueinander konkurrierende Ziele, Durchlaufzeit und Lieferzeitpunkte dagegen konform.
Betrachtet man sich die ‚Pareto'-Fronten verschiedener Rechenzeiten (**Abbildung 8.7**), so sieht man, dass es nicht möglich ist, die beiden Optimierungskriterien Durchlaufzeit und Rüstzeiten gleichzeitig zu minimieren. Diese beiden Kriterien sind ebenfalls zueinander konkurrierende Ziele. An den Hilfslinien für gleiche **M+S** erkennt man, dass Lösungen, die **M+S** am besten erfüllen meist eine kleine Rüstzeit (**S**) aufweisen.
In **Abbildung 8.8** ist eine Lösung bei gleichzeitiger Minimierung der Durchlaufzeit und der Rüstzeit (**M+S**) zu sehen.

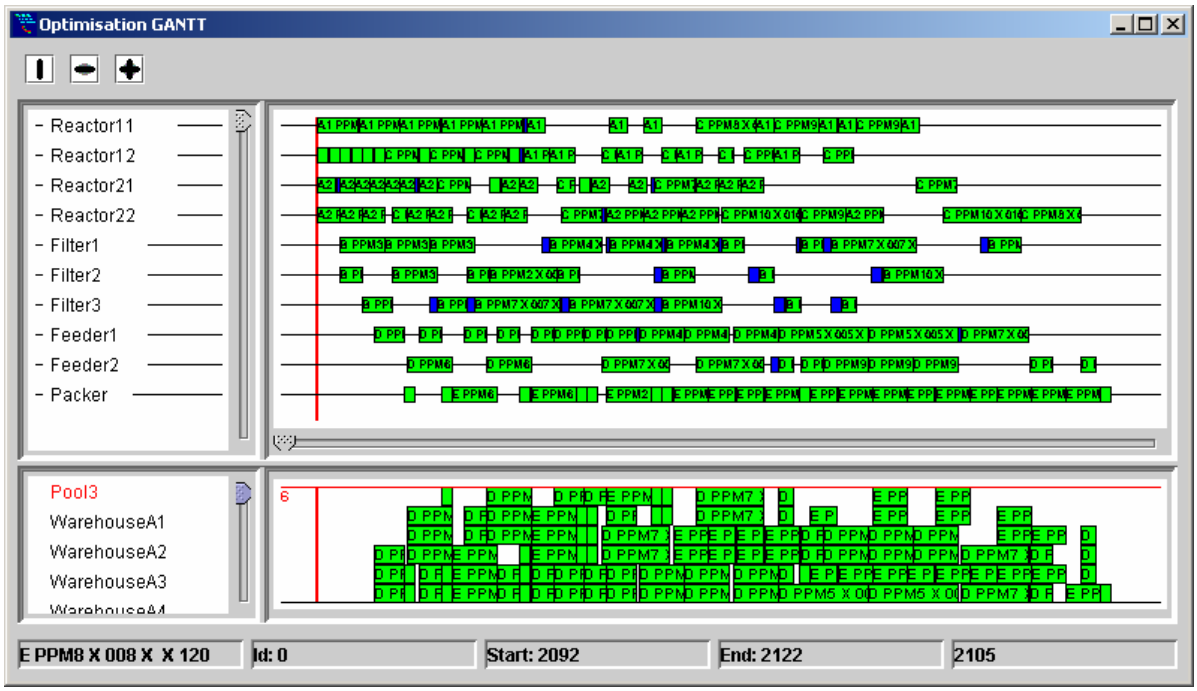


Abbildung 8.8: Abgebildet ist eine Lösung einer Optimierung der Kriterien **M+S** mit einer Durchlaufzeit von 2122 Minuten und einer Rüstzeit von 420 Minuten.

8.5 Zusätzliche Aktivitätslinks im Modell

8.5.1 Fixiertes Pegging

Bis auf die Produkte B_x werden alle anderen gelagert. Falls mehrere Modi ein bestimmtes Produkt herstellen und mehrere Modi dieses Produkt verbrauchen, ist es bisher egal, von welcher Aktivität die Modi sind, die das Produkt herstellen, solange immer genügend gelagert ist, können andere Modi dieses Produkt auch verbrauchen (Modell **M1**).
Soll stattdessen die Beziehung zwischen dem Modus, der ein Material produziert und einem Modus, der dieses Material verbraucht, während der Optimierung festgelegt sein, dann wird ein Aktivitätslink zwischen diesen Modi/Aktivitäten definiert. Meist wird dann von einem *fixierten Pegging* gesprochen (Modell **M2**). Dazu werden im Modell folgende Links pro Auftrag benötigt:

$$A_{Ax1}(2) \xrightarrow{ES,0} A_{BCx}(2)$$

$$A_{Ax2}(2) \xrightarrow{ES,0} A_{BCx}(2)$$

$$A_{BCx}(2) \xrightarrow{ES,0} A_{Dx}(2)$$

$$A_{Dx}(2) \xrightarrow{ES,0} A_{Ex}(1)$$

Im Folgenden werden Simulationen zu drei verschiedenen Instanzen vorgenommen, um die beiden Modellierungen zu vergleichen. Dabei handelt es sich um die Instanz 0 und solche, die aus dieser erzeugt wurden. Instanz 4 stellt eine Verdoppelung der Aufträge dar, wobei der Lieferzeitpunkt bei jedem kopierten Auftrag um 2000 Minuten erhöht wird. Bei Instanz 5 existieren zu jedem Auftrag aus Instanz 0 zusätzlich 4 kopierte Aufträge, wobei die Lieferzeitpunkte um n*2000 erhöht wurden. Keine Maschine benötigt mehr als 2000 Minuten zusammenhängende Produktionszeit bei Instanz 0. Die neu erzeugten Instanzen sollten deshalb ohne Verspätungen planbar sein. Die Gewichtsfaktoren aus **Kapitel 8.4** wurden beibehalten. Ebenso die zur Verfügung gestellte Rechenzeit, beachtet man nur die Anzahl Sweeps pro Temperaturschritt. Die Rechenzeit, gemessen in Sekunden, erhöhte sich natürlich bei Vergrößerung der Instanzen (vgl. **Tabelle 8.7**).
Leider war es nun nicht mehr möglich, bei jeder Simulation eine Lösung ohne Verspätung zu erzeugen. Im Folgenden sind deshalb für die Ergebnisse jeweils die durchschnittliche Energie (**avE**), deren Standardabweichung (**dev**), die durchschnittliche Verspätung (**avD**), die durchschnittliche Durchlaufzeit (**avM**) und die durchschnittliche Rüstzeit (**avS**) angegeben. Verletzen alle 10 Lösungen, über die jeweils gemittelt wird, die Lieferzeitpunkte, so ist der Wert **avD** hervorgehoben.

		M1				M2			
	Sweeps	**avE (dev)**	**avD**	**avM**	**avS**	**avE (dev)**	**avD**	**avM**	**avS**
M	100	24.401,0 (283,5)	0,0	2302,6	1375,0	25.398,0 (1.742,8)	50,7	2357,9	1312,0
	1000	22.914,0 (347,2)	0,0	2181,0	1104,0	22.842,0 (468,1)	0,0	2184,6	996,0
	5000	22.557,0 (264,7)	0,0	2145,6	1101,0	22.243,0 (317,8)	0,0	2134,6	897,0
	10000	22.452,0 (248,7)	0,0	2140,4	1048,0	22.046,0 (311,8)	0,0	2118,7	859,0
M+S	100	16.153,0 (385,1)	0,0	2360,6	870,0	16.595,0 (602,2)	16,0	2426,0	861,0
	1000	14.492,5 (444,7)	0,0	2234,5	664,0	14.258,5 (446,1)	0,0	2260,7	591,0
	5000	13.707,5 (337,4)	0,0	2194,5	547,0	13.748,5 (323,1)	0,0	2190,7	559,0
	10000	13.448,0 (212,6)	0,0	2190,6	499,0	13.353,5 (224,3)	0,0	2184,7	486,0
S	100	8.803,4 (793,5)	0,0	2513,4	629,0	9.047,4 (789,0)	0,0	2587,4	646,0
	1000	6.563,3 (234,9)	0,0	2433,3	413,0	6.681,9 (191,2)	0,0	2381,9	430,0
	5000	6.266,3 (156,7)	0,0	2346,3	392,0	6.286,8 (124,3)	0,0	2366,8	392,0
	10000	6.182,9 (138,0)	0,0	2332,9	385,0	6.141,8 (147,2)	0,0	2261,8	388,0

Tabelle 8.2: Ergebnisse für die Instanz 0 bei Modellierung mit *fixiertem Pegging* (**M2**) im Vergleich zu den Ergebnissen des Modells mit *variablem Pegging* (**M1**) aus **Kapitel 8.2**.

		M1				M2			
	Sweeps	**avE (dev)**	**avD**	**avM**	**avS**	**avE (dev)**	**avD**	**avM**	**avS**
M	100	47.630,0 (2.950,6)	117,5	4354,0	2915,0	113.017,0 (9.556,3)	**6454,7**	4545,8	3012,0
	1000	43.196,0 (442,9)	0,0	4080,9	2387,0	49.047,0 (4.437,9)	531,0	4152,4	2213,0
	5000	42.229,0 (498,9)	0,0	4020,1	2028,0	41.872,0 (404,9)	0,0	4013,3	1739,0
	10000	41.959,0 (440,8)	0,0	3991,3	2046,0	41.624,0 (573,4)	0,0	3993,5	1689,0
M+S	100	33.352,5 (1.430,5)	44,4	4376,7	2205,0	89.500,5 (7.094,8)	**5387,1**	4554,9	2571,0
	1000	28.284,5 (800,8)	2,0	4097,9	1555,0	33.882,0 (3.012,3)	456,0	4230,4	1634,0
	5000	27.036,0 (284,4)	0,0	4039,2	1368,0	26.655,5 (257,4)	0,0	4063,1	1268,0
	10000	26.397,5 (322,5)	0,0	4014,5	1265,0	25.921,0 (509,6)	0,0	4017,2	1167,0
S	100	23.709,7 (2.023,9)	34,0	4539,7	1883,0	86.584,0 (8.517,6)	**5873,3**	4611,0	2324,0
	1000	16.041,6 (779,4)	0,0	4251,6	1179,0	21.632,5 (3.111,3)	326,3	4359,5	1401,0
	5000	14.918,9 (595,1)	0,0	4198,9	1072,0	14.671,8 (776,8)	0,0	4231,8	1044,0
	10000	14.236,2 (419,8)	0,0	4176,2	1006,0	13.942,4 (424,7)	0,0	4242,4	970,0

Tabelle 8.3: Vergleich der Ergebnisse der beiden Modellierungen **M1** und **M2** für Instanz 4.

		M1				M2			
	Sweeps	**avE (dev)**	**avD**	**avM**	**avS**	**avE (dev)**	**avD**	**avM**	**avS**
M	100	1.550.108,0 (268.808,7)	**71949,7**	10237,4	8740,0	1.717.895,0 (87.354,2)	**160094,5**	10862,2	8328,0
	1000	139.691,0 (35.477,5)	1750,7	9827,6	6401,0	1.153.603,0 (59.550,0)	**104586,4**	10134,5	6394,0
	5000	102.078,0 (872,0)	0,1	9666,1	5415,0	799.135,0 (66.659,3)	**69440,3**	9880,7	5925,0
	10000	101.131,0 (982,6)	10,5	9566,8	5253,0	644.454,0 (45.325,4)	**53974,5**	9931,2	5397,0
M+S	100	812.258,0 (127.549,8)	**72080,6**	10213,4	8077,0	1.665.540,5 (109.347,8)	**157447,3**	10855,5	7358,0
	1000	87.776,5 (6.622,6)	**1178,9**	9835,5	5362,0	1.113.831,0 (63.844,8)	**103412,0**	10196,2	5746,0
	5000	69.251,5 (1.803,6)	32,0	9691,3	4095,0	776.942,5 (50.271,4)	**70169,0**	9948,5	5102,0
	10000	67.232,5 (654,5)	27,2	9607,1	3785,0	590.688,0 (93.914,2)	**51774,4**	9851,8	4737,0
S	100	1.561.925,9 (174.126,0)	**73661,7**	10251,9	7844,0	1.669.335,1 (55.085,7)	**158711,1**	10854,1	7137,0
	1000	75.665,8 (21.549,7)	**876,4**	10007,8	4813,0	1.077.997,3 (47.567,0)	**101431,4**	10233,3	5345,0
	5000	46.529,0 (2.241,2)	15,5	9899,0	3632,0	748.599,0 (69.250,1)	**69175,1**	9978,0	4687,0
	10000	43.105,5 (1.548,6)	0,0	9855,5	3325,0	580.428,7 (59.188,8)	**52409,0**	9918,7	4642,0

Tabelle 8.4: Vergleich der Ergebnisse der beiden Modellierungen **M1** und **M2** für Instanz 5.

Die Instanz 4 besteht aus zwei Auftragsmengen, die jeweils derer der Instanz 0 gleichen. Auffällig ist noch bei den Ergebnissen, dass die Durchlaufzeit bei Instanz 4 kleiner als 2*Durchlaufzeit der Instanz 0 ist, die Rüstzeiten jedoch größer als 2*Rüstzeiten der Instanz 0. Dies ist allerdings nicht verwunderlich, da die Rüstzeiten am Anfang der Produktion 0 betragen, zwischen den Produkten jedoch meist >0. Zwischen der Produktion zweier Auftragsmengen fällt eine zusätzliche Rüstzeit an. Bei der Durchlaufzeit kann davon ausgegangen werden, dass bereits die Zwischenprodukte der Auftragsmenge 2 bearbeitet werden können, obwohl Auftragsmenge 1 noch nicht vollständig fertigproduziert wurde. Die gesamte Durchlaufzeit reduziert sich somit um diese ‚Überlappung'. Entsprechendes gilt für Instanz 5 im Vergleich zu Instanz 0.
Die kleineren Rechenzeiten reichten bei den größeren Instanzen nicht mehr aus, um bei jeder Lösung die Lieferzeitpunkte einzuhalten. Dies ist jedoch leicht nachzuvollziehen, da bei den vergrößerten Instanzen der Konfigurationsraum stark zunimmt (siehe **Anhang B**). Die Lösungssuche bedarf somit einer größeren Anzahl an Schritten.
Betrachtet man sich die Ergebnisse aus **Tabelle 8.2** bis **8.4**, so sind folgende Punkte bzgl. der beiden Modellierungen **M1** und **M2** festzuhalten:

- Mit **M2** können gleichgute Ergebnisse erhalten werden, falls man ausreichend Rechenzeit investiert; bei Instanz 0 ab 1000 Sweeps, bei Instanz 4 ab 5000 Sweeps.
- Die zur Verfügung gestellte Rechenzeit reicht nicht aus, um mit **M2** die Lieferzeitpunkte bei Instanz 5 einzuhalten, während dies für **M1** ab 5000 Sweeps zum Teil der Fall ist.

Insgesamt kann festgestellt werden, dass **M1** die zusätzliche Freiheit des variablen Pegging ausnutzt, um schneller Lösungen zu erzeugen, die die Lieferzeitpunkte einhalten. Dieser Vorteil wirkt sich umso stärker aus, je mehr gleiche Produkte hergestellt werden. Grundsätzlich ist es jedoch mit **M2** ebenfalls möglich, gleich gute Ergebnisse zu erzielen, falls ausreichend Rechenzeit zur Verfügung steht. Die für eine bestimmte Anzahl Sweeps benötigte Rechenzeit ist etwas geringer bei Modell **M2**. Zwar werden zusätzliche Links angelegt und müssen evtl. im Reparaturmechanismus bearbeitet werden, jedoch fallen die Zwischenlager weg, da diese bei dieser Modellierung nicht mehr benötigt werden (**Tabelle 8.7**).

8.5.2 Reihenfolgebeziehung zwischen Aufträgen

In diesem Abschnitt soll nun eine Möglichkeit vorgestellt werden, die dazu dient, Modell **M2** zu verbessern. Gerade bei der Vergrößerung der Instanzen existieren viele Aufträge für das gleiche Endprodukt. Die Lieferzeitpunkte sind deren einziger Unterschied. Eigentlich ist es selbstverständlich, dass alle Aktivitäten eines Auftrags für ein Endprodukt E_x vor einem anderen (oder gleichzeitig, falls alternative Maschinen existieren) zu fertigen sind, falls dieser das gleiche Produkt E_x herstellt und einen größeren Lieferzeitpunkt hat. Eine festgelegte Reihenfolge kann auch zwischen Aufträgen mit gleichem Endprodukt und gleichem Lieferzeitpunkt festgelegt werden. Diese zusätzlichen Reihenfolgebeziehungen erfordern einen erhöhten Rechenzeitaufwand (**Tabelle 8.7**) bei dem Reparaturmechanismus im Vergleich zu Modell **M2**, reduzieren jedoch den Suchraum deutlich und sorgen für eine von Anfang an bessere Beachtung der Lieferzeitpunkte. Im Folgenden werden jeweils Links zwischen den Aufträgen i und j erzeugt, falls beide Aufträge das gleiche Endprodukt erzeugen und die Lieferzeitpunkte die Bedingung $L_j > L_i$ erfüllen :

$$A^{i}{}_{Ax1}(1) \xrightarrow{SS,0} A^{j}{}_{Ax1}(1)$$
$$A^{i}{}_{Ax2}(1) \xrightarrow{SS,0} A^{j}{}_{Ax2}(1)$$
$$A^{i}{}_{BCx}(1) \xrightarrow{SS,0} A^{j}{}_{BCx}(1)$$
$$A^{i}{}_{Dx}(1) \xrightarrow{SS,0} A^{j}{}_{Dx}(1)$$
$$A^{i}{}_{Ex}(1) \xrightarrow{SS,0} A^{j}{}_{Ex}(1)$$

Die Ergebnisse werden in **Tabelle 8.5** und **8.6** denen des Modells **M2** gegenübergestellt, wobei das erweiterte Modell **M2b** genannt wird.

		M2				M2b			
	Sweeps	**avE (dev)**	**avD**	**avM**	**avS**	**avE (dev)**	**avD**	**avM**	**avS**
M	100	113.017,0 (9.556,3)	**6454,7**	4545,8	3012,0	64.299,0 (3.535,2)	**1715,8**	4435,8	2783,0
	1000	49.047,0 (4.437,9)	531,0	4152,4	2213,0	43.091,0 (276,9)	0,1	4095,8	2132,0
	5000	41.872,0 (404,9)	0,0	4013,3	1739,0	41.885,0 (463,7)	0,0	3998,4	1901,0
	10000	41.624,0 (573,4)	0,0	3993,5	1689,0	41.157,0 (523,5)	0,0	3940,6	1751,0
M+S	100	89.500,5 (7.094,8)	**5387,1**	4554,9	2571,0	51.439,0 (5.745,4)	**1784,9**	4474,0	2244,0
	1000	33.882,0 (3.012,3)	456,0	4230,4	1634,0	28.176,5 (627,7)	0,0	4125,3	1510,0
	5000	26.655,5 (257,4)	0,0	4063,1	1268,0	26.121,5 (392,6)	0,0	4015,3	1209,0
	10000	25.921,0 (509,6)	0,0	4017,2	1167,0	25.823,0 (469,6)	0,0	3981,6	1183,0
S	100	86.584,0 (8.517,6)	**5873,3**	4611,0	2324,0	39.783,5 (3.304,8)	**1562,7**	4626,5	1953,0
	1000	21.632,5 (3.111,3)	326,3	4359,5	1401,0	16.182,6 (750,4)	0,0	4262,6	1192,0
	5000	14.671,8 (776,8)	0,0	4231,8	1044,0	14.609,6 (516,3)	0,0	4219,6	1039,0
	10000	13.942,4 (424,7)	0,0	4242,4	970,0	13.755,7 (424,4)	0,0	4145,7	961,0

Tabelle 8.5: Ergebnisse für die Instanz 4 bei Modellierung mit *fixiertem Pegging* (**M2**) im Vergleich zu den Ergebnissen bei Erweiterung um zusätzliche Links zwischen Aktivitäten verschiedener Aufträge (**M2b**).

		M2				M2b			
	Sweeps	**avE (dev)**	**avD**	**avM**	**avS**	**avE (dev)**	**avD**	**avM**	**avS**
M	100	1.717.895,0 (87.354,2)	**160094,5**	10862,2	8328,0	673.309,0 (28.111,7)	**56392,6**	10288,4	6499,0
	1000	1.153.603,0 (59.550,0)	**104586,4**	10134,5	6394,0	328.249,0 (20.395,2)	**22514,4**	9736,3	5742,0
	5000	799.135,0 (66.659,3)	**69440,3**	9880,7	5925,0	126.234,0 (11.162,8)	**2660,1**	9464,6	4987,0
	10000	644.454,0 (45.325,4)	**53974,5**	9931,2	5397,0	98.375,0 (561,2)	0,0	9381,6	4559,0
M+S	100	1.665.540,5 (109.347,8)	**157447,3**	10855,5	7358,0	628.266,0 (30.133,8)	**54906,0**	10250,2	5591,0
	1000	1.113.831,0 (63.844,8)	**103412,0**	10196,2	5746,0	249.537,0 (21.482,9)	**17697,1**	9681,2	4832,0
	5000	776.942,5 (50.271,4)	**70169,0**	9948,5	5102,0	91.619,0 (14.234,0)	**2471,3**	9549,2	3832,0
	10000	590.688,0 (93.914,2)	**51774,4**	9851,8	4737,0	63.775,5 (588,2)	0,0	9423,1	3332,0
S	100	1.669.335,1 (55.085,7)	**158711,1**	10854,1	7137,0	600.885,2 (26.976,5)	**53798,7**	10298,2	5260,0
	1000	1.077.997,3 (47.567,0)	**101431,4**	10233,3	5345,0	234.196,1 (17.852,3)	**17871,6**	9890,1	4559,0
	5000	748.599,0 (69.250,1)	**69175,1**	9978,0	4687,0	57.107,3 (7.363,2)	**1471,7**	9700,3	3269,0
	10000	580.428,7 (59.188,8)	**52409,0**	9918,7	4642,0	38.260,9 (935,1)	0,0	9660,9	2860,0

Tabelle 8.6: Vergleich der Ergebnisse der beiden Modellierungen **M2** und **M2b** für Instanz 5.

Bei der Erweiterung der Modellierung um zusätzliche Reihenfolgebeziehungen zwischen Aktivitäten verschiedener Aufträge des gleichen Endprodukts (**M2b**) können vor allem bzgl. der Einhaltung der Lieferzeitpunkte deutliche Verbesserungen erzielt werden. Steht allerdings ausreichend Rechen-

zeit zur Verfügung (z.B. 10000 Sweeps bei Instanz 4), innerhalb derer selbst die Modellierung **M2** die Lieferzeitpunkte einhält, sind die Unterschiede nur noch gering.

	M1	**M2**	**M2b**
Instanz 0	56	48	50
Instanz 4	236	188	208
Instanz 5	830	643	670

Tabelle 8.7: Benötigte Rechenzeit in Sekunden für 100000 Sweeps auf einem Pentium mit 650MHz für die verschiedenen Instanzen und Modellierungen.

8.6 Veränderung der Produktionskapazitäten

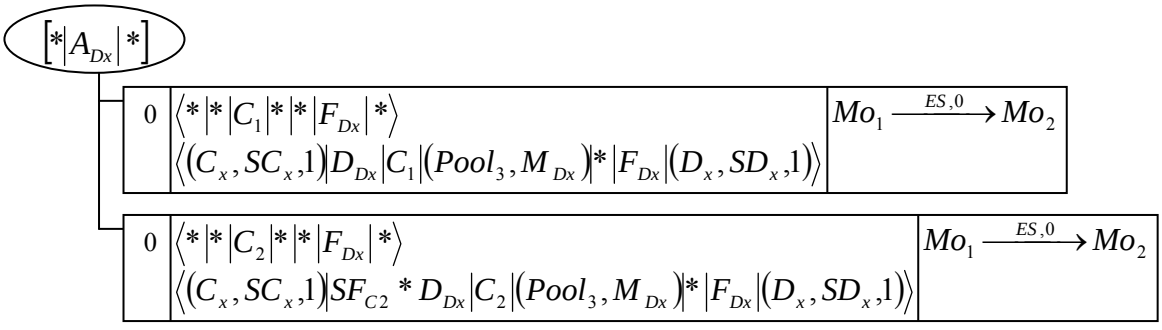
Nachdem festgestellt wurde, dass **Stufe D** und **E** den Engpass bei der Produktion (*bottleneck*) darstellen und die zur Verfügung gestellte Kapazität der Silo und der Multiressourcen meist nicht vollständig ausgenutzt wurde, kann nun die Ausgangssituation verändert werden, um mit Hilfe von Simulationen festzustellen, ob sich diese Veränderungen positiv auf die Optimierung auswirken. Es muss dabei natürlich immer der Nutzen den durch die Veränderungen erzeugten Kosten gegenübergestellt werden.
Kosteneinsparpotentiale ergeben sich durch Reduzierung der Kapazitäten. Verbesserte Optimierungsergebnisse können sich durch eine Erweiterung der Produktionskapazität einstellen.

Dem Benchmark werden drei Neuerungen hinzugefügt, um die Produktionskapazität zu erhöhen:

- *speed factor*
- *continuous mode*
- alternative Maschine in **Stufe E**

a) speed factor

Die Produktionszeit eines Zwischenprodukts kann sich auf alternativen Maschinen unterscheiden. So ist nun Maschine C_2 neuerer Bauart und somit effektiver als C_1. Es müssen also alle Produktionszeiten der Zwischenprodukte D_x, die auf C_2 hergestellt werden, mit dem *speed factor* (SF_{C2}=0.8) multipliziert werden. Im Modell werden die Aktivitäten A_{Dx} angepasst.



b) continuous mode

Bisher liefen alle Maschinen C_1, C_2, C_3 im *batch mode*, d.h. die zur Produktion nötigen Rohstoffe werden komplett in die Maschine übernommen, dann bearbeitet und schließlich in ein Lager weitergeleitet. Nun arbeiten sie im *continuous mode*, d.h. die Rohstoffe werden kontinuierlich aus dem Lager entnommen (C_3) oder bei der Bearbeitung kontinuierlich in ein Lager abgegeben (C_1,C_2) (vgl. **Abbildung 8.9**).

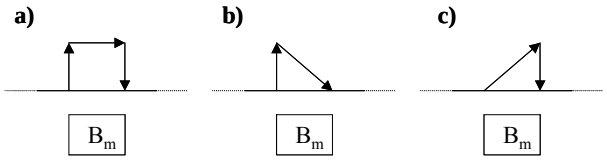


Abbildung 8.9: Während bisher die Maschinen C_1, C_2, C_3 im *batch mode* (**a)**) arbeiteten, wird jetzt im *continuous mode* das Material abgegeben (**b)** ; Maschine C_1 und C_2) oder aufgenommen (**c)** ; Maschine C_3).

Dadurch kann es bei der Produktion zwischen **Stufe D** und **Stufe E** zu Überlappungen kommen, d.h. mit der Herstellung des Endprodukts E_x kann unter Umständen begonnen werden, obwohl D_x noch nicht komplett fertiggestellt wurde.
Im Modell werden die Aktivitäten A_{Dx} und A_{Ex} angeglichen:

$\left[*\middle|A_{Dx}\middle|*\right]$

0	$\langle *\|*\|C_1\|*\|*\|F_{Dx}\|*\rangle$ $\langle (C_x, SC_x, 1)\|D_{Dx}\|C_1\|(Pool_3, M_{Dx})\|*\|F_{Dx}\|(D_x, SD_x, 0)\rangle$	$Mo_1 \xrightarrow{ES,0} Mo_2$
0	$\langle *\|*\|C_2\|*\|*\|F_{Dx}\|*\rangle$ $\langle (C_x, SC_x, 1)\|D_{Dx}\|C_2\|(Pool_3, M_{Dx})\|*\|F_{Dx}\|(D_x, SD_x, 0)\rangle$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$\left[*\middle|A_{Ex}\middle|L_i\right]$

0	$\langle (D_x, SD_x, 0)\|D_{Ex}\|C_3\|(Pool_3, M_{Ex})\|*\|*\|(E_x, W_{Ex}, 1)\rangle$	

und evtl. der Typ des Aktivitätslink zwischen A_{Dx} und A_{Ex} verändert, falls er existiert (Modell **M2**):

$A_{Dx}(2) \xrightarrow{SS,0} A_{Ex}(1)$

c) alternative Maschine

Produktionsstufe E erhält einen zusätzlichen Packer.

$\left[*\middle|A_{Ex}\middle|L_i\right]$

0	$\langle (D_x, SD_x, 1)\|D_{Ex}\|C_3\|(Pool_3, M_{Ex})\|*\|*\|(E_x, W_{Ex}, 1)\rangle$	
0	$\langle (D_x, SD_x, 1)\|D_{Ex}\|C_{3b}\|(Pool_3, M_{Ex})\|*\|*\|(E_x, W_{Ex}, 1)\rangle$	

8.6.1 Ergebnisse zur Kosteneinsparung

Zuerst wird untersucht, wie stark sich eine Kosteneinsparung durch Reduzierung der Kapazität der Silos oder der Multiressourcen auf die Optimierungsergebnisse auswirkt (**Tabelle 8.8**).
Bei den bisherigen Lösungen (Modell **M1**) erkennt man, dass die Silos mit Kapazitätsbeschränkung meist gar nicht oder nur mit einer Einheit belegt waren. Waren sie mit zwei Einheiten belegt, dann nur für sehr kurze Zeit. Es werden deshalb Simulationen durchgeführt, bei denen die Kapazität der Silos auf 1 gesetzt wird, wobei diese reduzierte Kapazität zuerst als harte Nebenbedingung (**M3**) gesetzt wird und anschließend als weiche Grenze (**M4**).
Die Multiressourcen waren bisher ebenfalls nicht jederzeit vollständig belegt. Es wird im Folgenden ein Ergebnis gezeigt, bei dem die Kapazität der Multiressourcen ($Pool_1$, $Pool_2$, $Pool_3$) auf (6,5,5) reduziert wird (**M5**), statt bisher (8,6,6).
Die Reduzierung der Kapazität der Silos auf 1 kann, bei größeren Rechenzeiten, ohne große Verschlechterungen bei den Optimierungsergebnissen eingeführt werden. Zwischen der Optimierung mit **M3** und derer mit **M4** ergibt sich für den Fall der Optimierung der Durchlaufzeit (**M**) fast kein Unterschied. Muss allerdings auf die Rüstkosten geachtet werden (**M+S**, **S**), so schneidet die Version mit harten Nebenbedingung umso besser ab, je wichtiger die Rüstkosten werden. Eigentlich hat die Veränderung der Kapazität keinen Einfluss auf die Optimierung der Rüstzeiten. Der Unterschied der Modellierung wirkt sich wahrscheinlich durch folgenden Effekt aus: während bei der Optimierung mit harten Nebenbedingungen die Lösung ‚repariert' wird, wodurch sich eine bzgl. der Nebenbedingung immer gültige Lösung ergibt, wird bei der Optimierung mit weichen Nebenbedingungen in Kauf genommen, dass die Lösungen die Nebenbedingung nicht immer einhalten. Der Nachteil der weichen Nebenbedingung, viele bzgl. der Nebenbedingung ungültigen Lösungen zu erzeugen, wiegt in diesem Fall schwerer als der Vorteil, den Suchraum nicht in Inseln zerfallen zu lassen.
Eine Reduzierung der Kapazität der Multiressourcen wirkt sich sehr viel stärker auf die Optimierungsergebnisse aus. Bei der Optimierung der Durchlaufzeit (**M**, **M+S**) sind die Ergebnisse deutlich schlechter als die Ergebnisse des Modells **M1**. Eigentlich sollte sich die Reduzierung der Kapazität nicht auf die Optimierung der Rüstzeiten (**S**) auswirken. Es müssen jedoch immer die Lieferzeitpunkte beachtet werden, wodurch sich die Ergebnisse, insgesamt gesehen, bei großen Rechenzeiten doch leicht verschlechtern.
Aufgrund solcher Simulationen ist es nun für eine Produktionsabteilung möglich, verschiedene Veränderungen bei der Produktion bereits vorab zu testen. Scheinbar kostensparende Maßnahmen, wie die Reduzierung der Kapazität der Multiressourcen, die sich deutlich in den Optimierungsergebnissen auswirken und somit evtl. mehr Kosten verursachen als sie sparen, können so vermieden werden.

	Sweeps	M1 av (dev)	M3 av (dev)	M4 av (dev)	M5 av (dev)
M	100	2.302,6 (29,1)	2.289,3 (49,0)	2.290,7 (30,5)	2.537,8 (32,4)
	1000	2.181,0 (31,1)	2.196,3 (33,4)	2.203,0 (29,1)	2.406,8 (30,6)
	5000	2.145,6 (32,8)	2.152,9 (26,2)	2.158,8 (29,3)	2.363,7 (30,0)
	10000	2.140,4 (31,5)	2.150,3 (25,3)	2.135,5 (29,1)	2.342,2 (37,3)
M+S	100	3.230,6 (77,0)	3.208,0 (88,8)	3.154,5 (98,8)	3.468,3 (124,0)
	1000	2.898,5 (88,9)	2.904,2 (101,3)	2.899,9 (132,4)	3.081,6 (101,3)
	5000	2.741,5 (67,5)	2.786,0 (76,7)	2.764,6 (74,6)	2.948,6 (56,8)
	10000	2.689,6 (42,5)	2.759,1 (65,7)	2.701,4 (53,9)	2.917,6 (66,7)
S	100	629,0 (79,9)	645,0 (86,6)	669,0 (51,3)	650,0 (67,7)
	1000	413,0 (20,5)	440,0 (31,3)	441,0 (28,8)	443,0 (34,9)
	5000	392,0 (8,7)	400,0 (20,0)	395,0 (20,6)	407,0 (15,5)
	10000	385,0 (8,1)	383,0 (11,9)	380,0 (10,0)	394,0 (12,8)

Tabelle 8.8: Durchschnittliche Ergebnisse zu den verschiedenen Kosteneinsparversuchen (**M3**, **M4**, **M5**) gegenüber der ursprünglichen Problemdefinition (**M1**).

8.6.2 Ergebnisse zur erweiterten Produktionskapazität

Als nächstes wird untersucht, ob sich eine erweiterte Produktionskapazität positiv auswirkt. Es werden Simulationen durchgeführt, wobei zuerst nur der *continuous mode* eingesetzt wird (Modell **M6**), dann eine Kombination aus *continuous mode* und dem *speed factor* (**M7**) und zuletzt die Erweiterung um eine zusätzliche Maschine in **Stufe E** (**M8**).
Die in **Kapitel 8.3** berechneten unteren Schranken für die Durchlaufzeit gelten nun nicht mehr. Die neuen unteren Schranken für die Durchlaufzeit sind in **Tabelle 8.9** zu sehen.

Modell	untere Schranke M
M1	1981
M6	1881
M7	1881
M8	1635

Tabelle 8.9: Untere Schranken für die Durchlaufzeit für die verschiedenen Erweiterungen der Produktionskapazität.

Die Ergebnisse zu den verschiedenen Modellen sind in **Tabelle 8.10** zu sehen. Alle drei Veränderungen haben auf die Rüstzeitoptimierung (**S**) keine besondere Auswirkung, bewirken jedoch eine deutliche Verbesserung bei der Optimierung der Durchlaufzeit (**M**, **M+S**). Zwar hat der *speed-factor* (**M7**) zusätzlich zum *continuous mode* (**M6**) keinen Einfluss auf die untere Grenze (diese wird in diesen beiden Fällen vor allem durch **Stufe E** bestimmt), wirkt sich aber deutlich auf die Optimierung aus. Die Kombination beider Veränderungen führt zu deutlich besseren Ergebnissen, wodurch die untere Grenze bis auf 5% erreicht wird.
Die alternative Ressource in **Stufe E** (**M8**) wirkt sich sehr stark auf die untere Schranke aus. Statt **Stufe E** ist nun **D** die bestimmende Größe für die untere Grenze. Die Optimierungsergebnisse werden etwas besser als bei der Kombination der anderen beiden Veränderungen (**M7**), sind jedoch von ihrer unteren Grenze sehr viel weiter entfernt. Betrachtet man die Ergebnisse der Optimierung der

Durchlaufzeit (**M**, **M+S**), so sieht man, dass die Auslastung der Ressourcen nun sehr viel gleichmäßiger ist. Der vorhandene Engpass (*bottleneck*) wurde beseitigt.
Aufgrund einer Simulation könnte so entschieden werden, ob sich eine Erweiterung der Produktionskapazität langfristig lohnt.

	Sweeps	M1 av (dev)	M6 av (dev)	M7 av (dev)	M8 av (dev)
M	100	2.302,6 (29,1)	2.196,3 (27,2)	2.148,9 (38,6)	2.158,5 (36,2)
	1000	2.181,0 (31,1)	2.108,9 (33,2)	2.030,8 (37,8)	2.039,2 (40,1)
	5000	2.145,6 (32,8)	2.050,3 (38,1)	2.009,3 (18,8)	1.997,4 (35,8)
	10000	2.140,4 (31,5)	2.042,7 (23,9)	1.994,4 (17,9)	1.964,4 (18,1)
M+S	100	3.230,6 (77,0)	3.123,1 (55,0)	3.086,6 (83,4)	3.122,4 (94,9)
	1000	2.898,5 (88,9)	2.749,4 (111,4)	2.688,1 (46,1)	2.751,7 (71,9)
	5000	2.741,5 (67,5)	2.652,4 (46,0)	2.607,9 (44,8)	2.603,5 (28,5)
	10000	2.689,6 (42,5)	2.596,0 (58,8)	2.586,6 (73,1)	2.555,4 (58,6)
S	100	629,0 (79,9)	620,0 (55,0)	677,0 (45,6)	666,0 (83,0)
	1000	413,0 (20,5)	425,0 (46,1)	434,0 (25,4)	466,0 (35,6)
	5000	392,0 (8,7)	392,0 (19,4)	394,0 (21,5)	391,0 (12,2)
	10000	385,0 (8,1)	388,0 (16,6)	383,0 (16,2)	379,0 (10,4)

Tabelle 8.10: Durchschnittliche Ergebnisse zu den verschiedenen erweiterten Produktionskapazitäten (**M6**, **M7**, **M8**) gegenüber der ursprünglichen Aufgabenstellung (**M1**).

8.7 Hinzufügen erforderlicher Wartungspausen

Bei den nicht *bottleneck* Ressourcen zeigen sich immer wieder zeitliche Lücken im Produktionsplan. Aus Kostenersparnis werden nun auf den nicht zu 100% ausgelasteten Ressourcen Pausen hinzugefügt (**Tabelle 8.11**).
Diese werden z.B. benötigt, um Wartungsmaßnahmen an einzelnen Maschinen durchzuführen. Jede Wartungsmaßnahme hat pro Tag eine Dauer von 1 Stunde. Die Wartungsmaßnahmen sind über den Tag verteilt, um dem durchführenden Personal genügend Zeit zur Vor- und Nachbreitung zu geben. Nicht alle Produktionsaktivitäten können pausiert werden (**Tabelle 8.12**). In diesem Kapitel soll nun untersucht werden, wie stark eine Reduzierung der Kapazität in Form von regelmäßigen Pausen die Durchlaufzeit erhöht (Modell **M9**).

Primärressource	Pausenzeit (Uhrzeit)	Pausenzeit (Minuten)
R_{11}	5:00-6:00	300+n*1440 – 360+n*1440
R_{12}	11:00-12:00	660+n*1440 – 720+n*1440
R_{21}	17:00-18:00	1020+n*1440 – 1080+n*1440
R_{22}	23:00-24:00	1380+n*1440 – 1440+n*1440
C_1	8:00-9:00	480+n*1440 – 540+n*1440
C_2	20:00-21:00	1200+n*1440 – 1260+n*1440

Tabelle 8.11: Für die einzelnen Maschinen werden Pausen zur Durchführung von Wartungsarbeiten definiert.

Aktivität	kann pausiert werden
A_{Ax1}	ja
A_{Ax2}	ja
A_{BCx}	nein
A_{Dx}	nein

Tabelle 8.12: Je nach Produktionsstufe können Aktivitäten pausiert werden oder nicht.

Die Einführung von Pausen zur Durchführung von Wartungsarbeiten verschlechtert die Ergebnisse, obwohl sie nur auf nicht zu 100% ausgelasteten Ressourcen eingeführt werden. Die Verschlechterungen sind allerdings kleiner als die gesparte Zeit. Während die Pausen 4% der Zeit sperren, verschlechtern sich die Ergebnisse bei größeren Rechenzeiten nur um 2-3%.

	Sweeps	M1 av (dev)	M9 av (dev)
M	100	2.302,6 (29,1)	2.423,0 (51,3)
	1000	2.181,0 (31,1)	2.288,0 (31,5)
	5000	2.145,6 (32,8)	2.222,9 (32,7)
	10000	2.140,4 (31,5)	2.220,3 (27,7)
M+S	100	3.230,6 (77,0)	3.348,0 (45,8)
	1000	2.898,5 (88,9)	2.955,4 (72,3)
	5000	2.741,5 (67,5)	2.848,6 (62,0)
	10000	2.689,6 (42,5)	2.767,6 (40,6)
S	100	629,0 (79,9)	617,0 (95,3)
	1000	413,0 (20,5)	466,0 (43,2)
	5000	392,0 (8,7)	399,0 (25,1)
	10000	385,0 (8,1)	394,0 (18,0)

Tabelle 8.13: Durchschnittliche Ergebnisse bei Einführung von Wartungspausen (**M9**) gegenüber der ursprünglichen Problemdefinition (**M1**).

8.8 Vergleich mit anderen Verfahren

Zuerst werden einige Ergebnisse von Verfahren vorgestellt, die von der SAP-AG für die Lösung des Benchmarks getestet wurden. Dazu wird auf Kapitel 15.3 in **[8.2]** verwiesen. Bei den Ergebnissen handelt es sich um die jeweils besten gefundenen bei spezieller Parameteranpassung an die Instanz 0. Als Rechenzeitlimit waren 10 Minuten auf einem Rechner mit 200 MHz vorgegeben. Es sind Ergebnisse zu einem *Simulated Annealing* Ansatz (**TFP_SA**), einem auf einer Bibliothek für *Constraint Propagation* aufgesetztem *Branch&Bound* Verfahren (**CP**), einem *Branch&Bound* Verfahren von Schwindt (**B&B**) und einem Ansatz basierend auf einem genetischen Algorithmus **GA** gegeben. Meist waren diese Verfahren noch Prototypen, speziell auf den Benchmark angepasst. Jedoch steckten in den Grundlagen der Verfahren oft jahrelange Entwicklungsarbeiten. So war

TFP_SA bereits im Einsatz bei Kunden, ebenso die Bibliothek für *Constraint Propagation* von **CP**. **B&B** wurde bereits in **[8.3]** vorgestellt, jedoch noch um einige Funktionalitäten für das Benchmark erweitert. Details bzgl. der einzelnen Verfahren sind dem Autor leider nicht bekannt.
Der größte Unterschied zwischen **TFP_SA** zu dem in dieser Arbeit vorgestellten Verfahren (**SA_N2**) scheint in der Modellierung zu liegen. Während der in dieser Arbeit vorgestellte Ansatz die einzelnen Aktivitäten seriell nacheinander bestmöglich plant, nutzt **TFP_SA** einen parallelen Aufbau des Plans auf einer direkten Repräsentation, soll bedeuten, dass evtl. mehrere Aktivitäten gleichzeitig zu vorher festgelegten Zeitpunkten geplant werden und anschließend der Plan korrigiert wird, damit einige Nebenbedingungen eingehalten werden. Meist sind die Nebenbedingungen bei **TFP_SA** weich beachtet worden.
CP basiert auf einer Bibliothek für *Constraint Propagation*. Darauf aufgesetzt wurde ein Verfahren für Scheduling Probleme, das zuerst eine gültige Initiallösung erstellt und dann mittels einer Fenstertechnik die Lösung schrittweise verbessert. Grundlage dafür ist ein *Branch&Bound*-Verfahren, das eine Suche mit Vorausschau verwendet.
Unter **GA** versteht man ein auf einem genetischen Algorithmus basierendes Verfahren. In **[8.2]** wurden verschiedene Initialisierungen und Nachbarschaften getestet. Die in **Tabelle 8.14** angegebenen Ergebnisse sind die besten aus einer Menge von Lösungen, die mithilfe eines Parametertunings auf Instanz 0 erreicht wurden. **GA** schafft die Planung von 1000 Aktivitäten pro Sekunde. Zum Aufbau eines gesamten Plans benötigt der Algorithmus somit 150/1000 Sekunden, oder anders ausgedrückt: der Algorithmus vermag in 15 Sekunden 100 Lösungen zu erzeugen (Rechner mit 200 MHz).

TFP_SA		CP		B&B		GA		GA (av)	
M	S	M	S	M	S	M	S	M	S
2767	760	2361	780	2804	1412	2254	1410	2261	1451
2729	680	2368	650	2714	1270	2288	920	2409	631
2609	660	2510	530	2674	850	2346	640	2442	572
		2725	500	2338	--	2385	540		
		2758	430						
		2798	390						

Tabelle 8.14: Die besten Ergebnisse der Verfahren **TFP_SA**, **CP**, **B&B** und **GA**. Zusätzlich sind für **GA** noch durchschnittliche Ergebnisse angegeben **GA(av)**. Es wird dabei ein Rechenzeitlimit von 10 Minuten auf einem Pentium mit 200 MHz eingehalten.

Bei den Ergebnissen aus **Tabelle 8.14** ist jedoch noch zu beachten, dass

- beim 3. Ergebnis von **TFP_SA** mit erhöhter Lagerkapazität gerechnet wurde,
- beim 2. Ergebnis von **B&B** zusätzlich noch ein Nachoptimierungsschritt speziell für Rüstzeiten zum Einsatz kam und beim letzten Ergebnis die Umrüstzeiten ignoriert wurden,
- beim **GA** zuerst nur die besten Ergebnisse mit angepasster Parametereinstellung und zusätzlich noch durchschnittliche Werte angegeben sind **GA(av)**.

In **Tabelle 8.15** werden die Ergebnisse aus **[8.4]** gezeigt. Diese sind die besten bei einer Rechenzeit von 4-5 Stunden auf einer UltraSparc mit 296 MHz mit den Verfahren *Threshold Accepting* (**Vogl_TA**) und *Simulated Annealing* (**Vogl_SA**). Allerdings sind diese Ergebnisse möglicherweise

nicht korrekt. In der Arbeit wurde ein Gantt einer Lösung gezeigt, in dem nur 27 Aktivitäten vorhanden waren, die die Produkte A_x produzieren. Eigentlich müssten es jedoch 50 sein, da bei jedem der 25 Aufträge jeweils 2 benötigt werden. Leider fehlen auch Ergebnisse zu veränderten Prioritäten bzgl. Durchlaufzeit und Rüstzeiten.
Diese Verfahren (**Vogl_SA** und **Vogl_TA**) erzeugt 1000 Moves pro Sekunde (UltraSparc mit 296 MHz). Im Unterschied zu **SA_N2** wird bei **Vogl_SA** auf weiche Nebenbedingungen viel Wert gelegt. Viele der Nebenbedingungen sind deshalb weich modelliert und mit hohen Strafkosten belegt, wodurch nach Aussagen des Autors **[8.4] ,**eine typische Simulation 30 Minuten dauert und man nach dieser Zeit eine erste Lösung vorliegen hat'.

Vogl_SA		Vogl_TA	
M	S	M	S
2109	650	2106	670

Tabelle 8.15: Die besten Ergebnisse der Verfahren **Vogl_SA** und **Vogl_TA** nach 350 Temperaturschritten mit je 50000 Moves. Dies entspricht 4-5 Stunden Rechenzeit auf einer UltraSparc mit 296 MHz.

In **[2.7]** wurden vom Autor bereits verschiedene Verfahren zur Lösung dieser Problemstellung vorgestellt. Neben einem *Simulated Annealing* Ansatz (**SA_N1**) finden sich dort auch ein Ansatz basierend auf *Threshold Accepting* (**TA_N1**) und Ergebnisse zu einem *Temperature Bouncing* (**Bounce_N1**). Der Unterschied zu dem in dieser Arbeit vorgestellten **SA_N2** Ansatz liegt in der Modellierung. Es werden hier die meisten Nebenbedingungen hart modelliert, so z.B. die Lagerkapazitäten, was den Vorteil hat, dass sehr früh Lösungen gefunden werden, die alle Nebenbedingungen einhalten. Neben der Betrachtung als harte Nebenbedingung könnte man in der Modellierung natürlich auch auf eine weiche wechseln, was jedoch zu keinen nennenswerten Unterschieden in der Qualität der Ergebnisse nach Erreichen des Rechenzeitlimits führt. Die größte Veränderung der Modellierung liegt in der Nachbarschaft. Während in **[2.7]** nur eine Nachbarschaft auf den Primärressourcen eingeführt wurde und die Reihenfolge auf den Multiressourcen bei Überschneidung von mehreren Primärressourcen, die diese benötigen, durch ein deterministisches Verfahren bestimmt wurde, erlaubt die Nachbarschaft in dieser Arbeit nun eine Veränderung der Reihenfolge bei der Planung der Multiressourcen. Ebenso werden die Veränderungen nun gleichmäßiger auf alle Ressourcen und Aktivitäten verteilt, da nicht zuerst wie in **[2.7]** eine Primärressource und erst dann ein Nachbarschaftsoperator gewählt wird, was zu einer unterdurchschnittlichen Veränderung pro Aktivität auf einer stark ausgelasteten Primärressource führt, sondern zuerst die Operatoren und die zu verändernden Aktivitäten ausgewählt werden. Spezielle Operatoren, die nur Veränderungen auf einer ausgewählten Ressource vornehmen, wurden zwar erstellt, in dieser Problemstellung jedoch nicht angewendet, da die Auswertung der Nachbarschaft (siehe **Kapitel 8.4.1**) offenlegt, dass hier mit der einfachsten Nachbarschaft die besten Ergebnisse zu erzielen sind. Kein großer Unterschied ergab sich in der Geschwindigkeit beim Erstellen einer Lösung. In **[2.7]** benötigten 10000 Lösungen 16 Sekunden auf einem Pentium mit 200 MHz, hier ist eine Rechenzeit von 5,6 Sekunden für 10000 Lösungen auf einem Pentium mit 650 MHz zu erwarten. In **Tabelle 8.16** und **8.17** sind die besten Ergebnisse aus **[2.7]** für verschiedene Rechenzeiten gegeben.

SA_N1		TA_N1		Bounce_N1	
M	S	M	S	M	S
2522	660	2514	550	2283	540
2845	540	2643	440	2540	420
2302	1900	2254	1950	2139	1280

Tabelle 8.16: Die besten Ergebnisse der Verfahren **SA_N1**, **TA_N1** und **Bounce_N1** bei einer Rechenzeitgrenze von 524 Sekunden auf einem Pentium mit 200MHz.

SA_N1		TA_N1		Bounce_N1	
M	S	M	S	M	S
2357	370	2284	430	2273	380
2074	1000	2465	410	2061	1130
		2171	1270		

Tabelle 8.17: Die besten Ergebnisse der Verfahren **SA_N1**, **TA_N1** und **Bounce_N1**, erreicht innerhalb der Rechenzeitgrenze von 22 Stunden (**SA_N1**), 7 Stunden (**TA_N1**) und 16 Stunden (**Bounce_N1**) auf einem Pentium mit 200MHz.

Es werden nun die aus der Literatur entnommenen Ergebnisse mit denen aus **Kapitel 8.4.2 Abbildung 8.7** verglichen. Dazu werden die Pareto-Fronten für 2 Rechenzeiten (1000 und 10000 Sweeps) genommen (**Tabelle 8.18**). Eine Simulation mit je 1000 Sweeps pro Temperaturschritt benötigt 168 Sekunden auf Pentium mit 650MHZ.

SA_N2(1000)		SA_N2(10000)	
M	S	M	S
2464	390	2220	370
2335	410	2130	470
2317	430	2113	1020
2219	550	2101	1100
2177	690	2089	1550
2159	960		
2159	980		
2155	1110		
2131	1130		

Tabelle 8.18: Die besten Ergebnisse von **SA_N2** mit einer Rechenzeitschranke von 1000 und 10000 Sweeps (entspricht 168 und 1680 Sekunden auf Pentium mit 650MHz).

In **Abbildung 8.10** sind die besten Ergebnisse der verschiedenen Verfahren für kurze Rechenzeiten (<10 Minuten auf Pentium mit 200 MHz) zu sehen. **B&B** schneidet von den vorgestellten Verfahren am schlechtesten ab. **SA_N2** am besten. Im Vergleich zu **TFP_SA** schneiden die beiden anderen *Simulated Annealing* Ansätze **SA_N2** und **SA_N1** vor allem bei Betrachtung der Durchlaufzeit (**M**) sehr viel besser ab. Es ist deshalb davon auszugehen, dass der parallele Planaufbau mit anschließendem Reparaturschritt für diese Problemstellung nicht zu empfehlen ist. Betrachtet man sich die verschiedenen statistischen Versionen aus **[2.7]**, erkennt man, dass die hier gewählte Nachbarschaft in den meisten Fällen deutlich bessere Ergebnisse liefert. Selbst der bisher beste Ansatz des *Temperature Bouncing* schneidet schlechter ab als **SA_N2**. Der in **[8.2]** vorgestellte Ansatz mit einem **GA** ist zwar besser als die schlechteren *Simulated Annealing* Versionen, liefert jedoch deutlich schlechtere Ergebnisse als **Bounce_N1** und **SA_N2**.

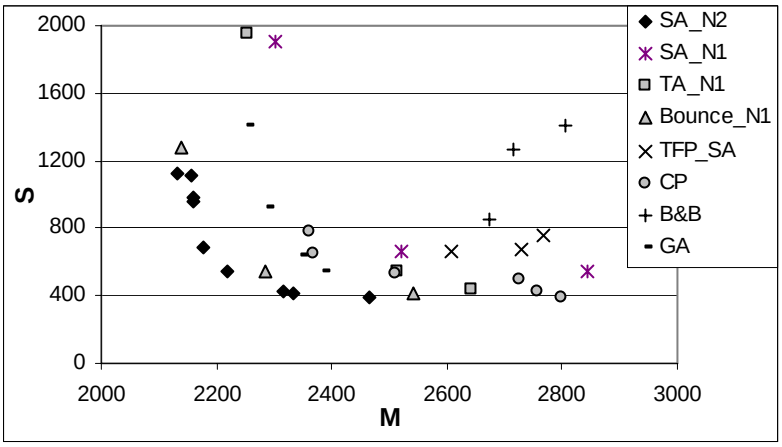


Abbildung 8.10: Die besten Ergebnisse der verschiedenen Verfahren bei kurzen Rechenzeiten (<=10 Minuten auf einem Pentium mit 200 MHz).

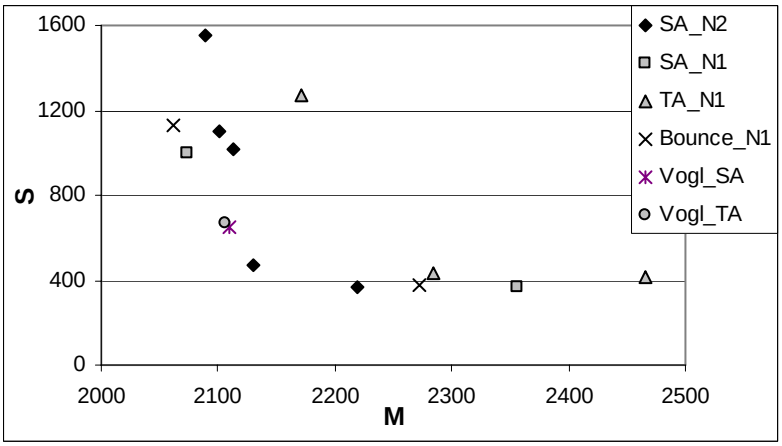


Abbildung 8.11: Die besten Ergebnisse der verschiedenen Verfahren bei langen Rechenzeiten (>10 Minuten auf einem Pentium mit 200 MHz).

Bei den besten Ergebnissen zu den längeren Rechenzeiten (**Abbildung 8.11**) zeigt sich ein ähnliches Bild. **SA_N2** liefert bis auf die Optimierung der Durchlaufzeit die besten Ergebnisse. Jedoch ist anzumerken, dass **SA_N2** sehr viel weniger Rechenzeit (Minuten) zur Verfügung hatte als **SA_N1** und **Bounce_N1** (mehrere Stunden). Bei **Vogel_SA** und **Vogl_TA** ist noch zu beachten, dass weniger Aktivitäten als benötigt geplant wurden und die Rechenzeiten im Bereich von Stunden lagen.

In **[8.5]** sind Ergebnisse eines genetischen Algorithmus (**OmeGA**) zu einer vereinfachten Problemstellung gegeben. Dort werden keine Lagerbeschränkungen vorgegeben. In den Lösungen im Anhang des Buches sind einige der besten Ergebnisse des dort vorgestellten Verfahrens zu sehen. Es fällt jedoch auf, dass die Rüstzeiten in **Stufe B** und **D** nicht den vorgegebenen entsprechen. Nach Rücksprache mit dem Autor konnte festgestellt werden, dass statt den vorgegebenen Rüstmatrizen die Rüstmatrix der **Stufe A** verwendet wurde. Im Folgenden wurde deshalb eine Simulation mit angepassten Daten (**[8.5]**) vorgenommen. In **Tabelle 8.19** sind einige der besten Ergebnisse aus

jeweils 10 Simulationen mit **SA_N2** mit einer Rechenzeit von 1000 Sweeps pro Temperaturschritt zu sehen.

OmeGA		SA_N2(1000)	
M	S	M	S
2225	360	2298	90
		2199	250
		2105	720

Tabelle 8.19: Es werden die jeweils besten Ergebnisse des **OmeGA** und **SA_N2** für eine veränderte Problemstellung gezeigt. Die Rechenzeit war bei **SA_N2** auf 1000 Sweeps pro Temperaturschritt beschränkt.

In **[8.6]** wurde mit einem parallelen Ansatz basierend auf dem **GA** aus **[8.2]** die Problemstellung bearbeitet. Es wurden dabei Instanzen verwendet, die dem 50fachen der ursprünglichen Aufgabenstellung entsprechen. Leider können keine Vergleiche zu den dort erstellten Lösungen gegeben werden, da sich der Autor nur mit Aufgabenstellungen mit zusätzlichen Funktionalitäten befasst hat. Es wurde ein Schichtkalender eingeführt, der an die Lösung die Bedingung stellt, dass bestimmte Aktivitäten nur innerhalb einer Schicht bearbeitet werden dürfen. Neben dieser von dem in dieser Arbeit vorgestellten Verfahren nicht erfüllbaren Funktionalität werden in **[8.6]** noch Pausen eingeführt, die Kapazitäten der *Pools* zeitabhängig verändert und ein Kalender für zeitabhängige Produktivität eingeführt. Die zeitabhängige Produktivität führt zu Produktionsdauern, die nun nicht mehr nur von dem Produktionsvorgang und der verwendeten Maschine, sondern auch noch vom Startzeitpunkt des Produktionsvorgangs abhängig sind.

8.9 Reduzierung der benötigten Parameter

Zur Lösung eines Optimierungsproblems mit dem vorgestellten *Simulated Annealing* Ansatz bedarf es neben der Modellierung der Problemstellung noch der Einstellung der Nachbarschaft und der Wahl der benötigten Parameter. Nachdem das Modell von einem Optimierungsexperten festgelegt wurde, stellt sich die Frage, ob auch ein nicht besonders versierter Anwender die tägliche Planung mit dem vorgestellten Ansatz durchführen kann. Dazu sollten ihm möglichst viele Entscheidungen abgenommen werden, so dass er im Idealfall nur noch die aktuelle Auftragsliste pflegen muss. In **Kapitel 8.4.1** wurde bereits eine Möglichkeit vorgestellt, die Nachbarschaft adaptiv zu regeln. Im Anschluss daran soll nun untersucht werden, wie sich die verschiedenen Parameter des **SA** vermeiden lassen.
Die folgenden Parameter werden für eine Simulation mit **SA** gefordert:

- Starttemperatur T_S,
- Abkühlschema: linear oder geometrisch,
- Steigung des linearen Schemas oder α-Faktor des geometrischen,
- Einfriertemperatur T_E und
- Anzahl Sweeps pro Temperaturschritt.

Einige dieser Parameter sind bei der Gruppe der gegebenen Problemstellungen (Scheduling) unabhängig von der Modellierung einzustellen. So hat sich das geometrische Abkühlschema mit einem α-Faktor von 0.98 bewährt. Die Wahl von α im Bereich [0.90,0.99] hatte bei keiner der in dieser Arbeit beachteten Problemstellungen besonderen Einfluss auf die erzielten Ergebnisse.
Die Wahl des Temperaturbereichs $[T_S,T_E]$ hängt jedoch sehr stark von der Problemstellung und deren Modellierung ab.

Im Folgenden werden Ergebnisse vorgestellt, die ohne ‚*annealing*' bei verschiedenen konstanten Temperaturen erzielt werden konnten (**Tabelle 8.20**).

	Sweeps	Greedy av (dev)	T=1 av (dev)	T=10 av (dev)	T=50 av (dev)	T=100 av (dev)	T=1000 av (dev)	SA-Cool av (dev)
M	100	2.332,5 (30,5)	2.339,8 (28,9)	2.338,2 (44,0)	2.337,3 (42,5)	2.388,3 (34,7)	2.742,1 (56,0)	2.302,6 (29,1)
	1000	2.251,1 (28,4)	2.229,1 (28,2)	2.215,6 (44,7)	2.222,8 (44,9)	2.296,9 (41,0)	2.631,2 (43,5)	2.181,0 (31,1)
	5000	2.224,5 (31,6)	2.173,2 (31,6)	2.146,8 (38,1)	2.150,8 (30,4)	2.249,2 (16,2)	2.597,9 (17,2)	2.145,6 (32,8)
	10000	2.222,2 (30,6)	2.149,9 (27,6)	2.122,6 (30,9)	2.143,5 (30,7)	2.244,8 (18,2)	2.585,7 (32,7)	2.140,4 (31,5)
M+S	100	3.291,3 (79,6)	3.306,3 (95,5)	3.303,4 (81,1)	3.332,3 (84,9)	3.609,6 (169,4	4.789,1 (131,3	3.230,6 (77,0)
	1000	3.115,9 (38,1)	3.071,2 (51,5)	3.049,3 (43,1)	2.996,5 (56,8)	3.266,7 (48,0)	4.584,5 (61,4)	2.898,5 (88,9)
	5000	3.075,4 (39,1)	2.861,4 (41,9)	2.830,0 (81,4)	2.845,1 (48,8)	3.219,9 (39,8)	4.475,3 (58,0)	2.741,5 (67,5)
	10000	3.066,7 (43,5)	2.744,1 (58,5)	2.772,2 (78,1)	2.803,8 (38,2)	3.197,6 (40,8)	4.450,6 (97,4)	2.689,6 (42,5)
S	100	792,0 (26,8)	734,0 (59,9)	696,0 (67,9)	706,0 (65,0)	760,0 (59,0)	1.739,0 (68,0)	629,0 (79,9)
	1000	685,0 (27,3)	575,0 (45,2)	543,0 (74,8)	427,0 (24,1)	550,0 (28,6)	1.634,0 (54,1)	413,0 (20,5)
	5000	644,0 (47,4)	453,0 (23,3)	432,0 (60,0)	384,0 (6,6)	530,0 (21,0)	1.548,0 (60,6)	392,0 (8,7)
	10000	636,0 (56,6)	421,0 (16,4)	413,0 (40,8)	379,0 (3,0)	508,0 (21,8)	1.525,0 (42,0)	385,0 (8,1)

Tabelle 8.20: Gezeigt werden die Ergebnisse der Simulationen mit einem *annealing* (**SA-Cool**) und mehreren konstanten Temperaturen (**T=...**). Zusätzlich sind noch die Ergebnisse eines **Greedy** Verfahren gegeben.

Die Ergebnisse zeigen, dass

- die einzelnen Kriterien **M**, **M+S**, **S** in verschiedenen Temperaturbereichen optimiert werden. Es werden die besten Ergebnisse erzielt, falls bei (**M**, **M+S**, **S**) die Temperaturen (10,1-10,50) zugeordnet werden.
- die Ergebnisse sich im Temperaturbereich [1,50] nicht sehr stark unterscheiden.
- der Unterschied zu einem *annealing* bei längeren Laufzeiten meist kleiner als 5% ist.
- man bei längeren Laufzeiten bei **M** und bei **S** bei der richtigen Wahl der konstanten Temperatur bessere Ergebnisse erhalten kann als bei einem *annealing*.

Insgesamt muss jedoch erwähnt werden, dass es einfacher ist, ein Schema für ein *annealing* zu erzeugen, als die richtige konstante Temperatur für eine optimale Simulation zu finden. Die Wahl eines Temperaturbereichs hat immer den Vorteil, diese optimale Temperatur zu enthalten und nur den kleinen Nachteil, dass evtl. zu viel Rechenzeit bei zu großen und/oder zu kleinen Temperaturen investiert wird. Ein Fehler, wie z.B. die Wahl der konstanten Temperatur >100, der auch bei langen Rechenzeiten zu keiner Verbesserung der Ergebnisse führt, wird so vermieden.
Bei allen bisherigen Simulationen wurde festgestellt, dass die Temperatur sehr stark von der Problemstellung, manchmal sogar noch von den einzelnen Instanzen (entspricht hier dem Auftragspaket) abhängt. Es wird deshalb im Folgenden versucht, diesen problemspezifischen Parameter durch ei-

nen solchen zu ersetzen, der allgemein definierbar ist. Dadurch soll dann die Temperatur aus der Simulation heraus selbst bestimmt werden. Es wird nun versucht, als Parameter die Akzeptanzrate zu verwenden. Diese sollte während einer **SA** Simulation von einem anfangs hohen Wert bis auf 0 sinken.
Im Folgenden wird im Unterschied zur der in **Kapitel 2.4.4** erwähnten Methode, aus der Akzeptanzrate die Anfangstemperatur zu bestimmen, nicht der Maximalwert über die Beträge aller Energieänderungen von *N* getesteten Moves beachtet, sondern nur der Mittelwert des Betrags der letzten *N* negativen Energieänderungen (Lösungsverbesserungen). Mit der so erstellten Temperatur wird festgelegt, mit welcher Wahrscheinlichkeit man in der Energielandschaft einen Schritt aufwärts gehen darf, dessen Energieänderung dem Mittelwert der letzen *N* Verbesserungen entspricht. Die Akzeptanzrate kann bei einem festen Wert fixiert sein oder während der Simulation gesenkt werden (vgl. **Kapitel 2.4.6**).
Bei der Simulation des **SA-Acc** wird die Akzeptanzrate von 50% in 150 Schritten logarithmisch mit einem α-Faktor von 0.98 gesenkt. Die Ergebnisse (**Tabelle 8.21**) sind dabei vergleichbar mit denen des **SA**. Mit Hilfe dieses Parameters lassen sich nun auch Simulationen von in Optimierungsproblemen unerfahrenen Benutzern durchführen, nachdem die Modellierung von einem Experten vorgenommen wurde.

	Sweeps	SA-Cool av (dev)	SA-Akz av (dev)
M	100	2.302,6 (29,1)	2.272,0 (22,4)
	1000	2.181,0 (31,1)	2.189,1 (30,2)
	5000	2.145,6 (32,8)	2.139,1 (34,7)
	10000	2.140,4 (31,5)	2.131,9 (36,0)
M+S	100	3.230,6 (77,0)	3.198,6 (76,8)
	1000	2.898,5 (88,9)	2.814,4 (73,1)
	5000	2.741,5 (67,5)	2.668,9 (40,2)
	10000	2.689,6 (42,5)	2.648,4 (33,8)
S	100	629,0 (79,9)	625,0 (72,4)
	1000	413,0 (20,5)	459,0 (42,1)
	5000	392,0 (8,7)	418,0 (23,2)
	10000	385,0 (8,1)	405,0 (20,1)

Tabelle 8.21: Vergleich der Ergebnisse des bisher verwendeten SA (**SA-Cool**) mit denen der Simulation mit der neuen Parameterdefinition (**SA-Acc**).

8.10 Zusammenfassung

In diesem Kapitel wurde zuerst versucht, eine gute Nachbarschaft für eine Modellierung der Problemstellung zu finden. Eine solche Suche bedarf allerdings einigen Aufwands und bedeutet nicht, dass eine einmal gefundene optimale Einstellung der Nachbarschaft auf allen Instanzen des gleichen Optimierungsproblems bei den verschiedenen Optimierungsprioritäten (Durchlaufzeit : Rüstzeit) ebenso optimal ist. Es wurde deshalb versucht, eine adaptive Nachbarschaft zu definieren, die sich während der Simulation selbst einstellt. Im Vergleich zu den per Hand erstellten Nachbarschaften kann diese adaptive Version nicht immer die besten Ergebnisse erzielen, jedoch im Vergleich zum Durchschnitt über alle getesteten Nachbarschaften durchaus ähnlich gute Ergebnisse erzeugen.
Im Anschluss daran wurde eine neue Modellierung für das gleiche Optimierungsproblem erstellt, die das *variable Pegging* durch ein *fixiertes* ersetzt. Die Unterschiede bei den Ergebnissen werden vor allem bei den Instanzen deutlich, bei denen viele Aufträge für das gleiche Endprodukt existieren und die zur Verfügung gestellte Rechenzeit relativ klein ist. Ein Modell mit *variablem Pegging* ist dann sehr viel erfolgreicher als ein Modell mit *fixiertem Pegging*. Zusätzlich wurde noch eine Verbesserung der Modellierung mit *fixiertem Pegging* vorgestellt, bei der zusätzliche Aktivitätslinks definiert werden, um die Aktivitäten, die für gleiche Endprodukte benötigt werden, nach ihren Lieferzeitpunkten zu sortieren. Diese Verbesserung der Modellierung war vor allem bei den großen Instanzen sehr hilfreich, um Lösungen zu finden, die die Lieferzeitpunkte einhalten.
Neben der ursprünglich gegebenen Problemstellung wurden dann noch solche bearbeitet, die kleine Veränderungen aufwiesen. Diese Simulationen sollten dazu dienen, eine ‚Was wäre wenn' Frage zu beantworten. Auf diese Weise können Entscheidungen zur Umstrukturierung des Produktionsprozesses simulativ getestet werden, bevor hohe Kosten z.B. bei der Anschaffung neuer Maschinen entstehen.
Im Vergleich zu verschiedenen anderen Verfahren aus der Literatur konnte sich die Modellierung mit *variablen Pegging* mit einem **SA** Ansatz behaupten. Die aus der Literatur bereits bekannten besten Ergebnisse konnten stets erreicht, in den meisten Fällen sogar verbessert werden.
Zum Schluss wurde erfolgreich versucht, die einzelnen problemabhängigen Steuerungsparameter des **SA** Algorithmus durch solche zu ersetzen, die, einmal definiert, für alle Problemstellungen anwendbar sind. Zusammen mit der adaptiven Nachbarschaft ergibt sich damit die Möglichkeit, ein einmal durch einen Experten erstelltes Modell zu benutzen, um Simulationen von einem nicht speziell in Optimierungsfragen ausgebildeten Benutzer durchführen zu lassen.

Kapitel 9

Der Westenberger-Kallrath-Benchmark

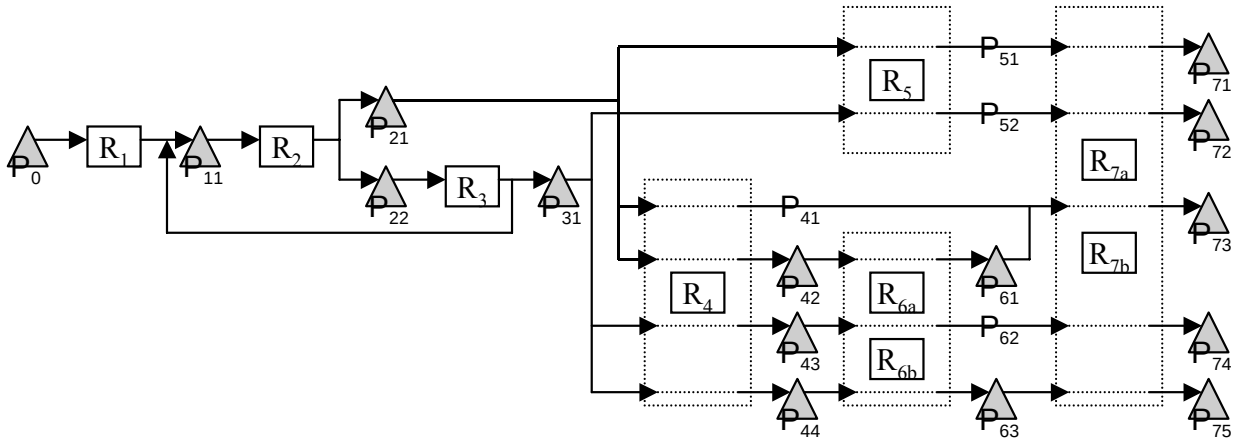


Abbildung 9.1: Graphische Darstellung der Problemstellung.

9.1 Problemstellung (siehe [9.1] oder [9.2])

a) Zur Verfügung stehende Ressourcen

- 9 Maschinen: R_1, R_2, R_3, R_4, R_5, R_{6a}, R_{6b}, R_{7a}, R_{7b}
- 10 Lager für die Zwischenprodukte mit Kapazitätsbeschränkung: S_{11}, S_{21}, S_{22}, S_{31}, S_{42}, S_{43}, S_{44}, S_{61}, S_{63}.
- 5 Lager für die Endprodukte ohne Kapazitätsbeschränkung: S_{71}, S_{72}, S_{73}, S_{74}, S_{75}.

b) Rezept

Jedes Endprodukt befolgt bei seiner Herstellung ein bestimmtes „*Rezept*“. Zur Herstellung des Endproduktes sind verschiedene Zwischenprodukte nötig. Die Losgrößen bei der Produktion sind veränderbar innerhalb bestimmter Grenzen. Diese Grenzen hängen dabei von der verwendeten Maschine ab. Unter der produzierten Losgröße versteht man die Summe der Mengen der einzelnen hergestellten Produkte. Die Produktionszeiten für die einzelnen Produkte sind von der gewählten Maschine (bei Stufe 6 und 7 existieren alternative Ressourcen) und dem Produkt abhängig, jedoch unabhängig von der Losgröße.

c) Ablauf der Produktion

Stufe 1: Aus dem Rohstoff P_0 wird das Zwischenprodukt P_{11} auf der Maschine R_1 hergestellt. Da der Rohstoff P_0 immer in ausreichender Menge vorhanden ist und das zugehörige Lager keine Kapazitätsrestriktion besitzt, wird im Folgenden auf dieses Produkt (und das Lager) verzichtet.

Stufe 2: Aus P_{11} werden P_{21} und P_{22} auf der Maschine R_2 hergestellt. Die Aufteilung in P_{21} und P_{22} ist dabei in einem bestimmten Bereich frei wählbar. So muss die Produktion von P_{21} zwischen 20% und 70% (der Menge von P_{11}) liegen.

Stufe 3: Aus P_{22} wird P_{31} auf der Maschine R_3 hergestellt. Dabei entsteht jedoch als Nebenprodukt noch P_{11}. Die Summe der Mengen von P_{31} und P_{11} entspricht dabei der Menge von P_{22}. Die Aufteilung auf P_{31} und P_{11} ist hier im Unterschied zur Stufe 2 fest. Je nach Aufgabenstellung wird 100% oder 69,23% von P_{22} in P_{31} verwandelt.

Stufe 4: Auf dieser Stufe können verschiedene Produkte auf der Maschine R_4 hergestellt werden. Aus P_{21} kann P_{41} oder P_{42} produziert werden, aus P_{31} kann P_{43} oder P_{44} entstehen. P_{41} ist nicht lagerfähig und muss sofort in Stufe 7 weiterverarbeitet werden.

Stufe 5: Es werden P_{51} (aus P_{21}) und P_{52} (aus P_{31}) auf der Maschine R_5 hergestellt. Da diese Produkte nicht lagerfähig sind, müssen sie sofort in Stufe 7 weiterverarbeitet werden.

Stufe 6: Es werden P_{61} (aus P_{42}), P_{62} (aus P_{43}) und P_{63} (aus P_{44}) auf den alternativen Maschinen R_{6a} und R_{6b} hergestellt. P_{62} ist nicht lagerfähig und muss sofort in Stufe 7 weiterverarbeitet werden. Die Produktionszeiten für die einzelnen Produkte hängen nun zusätzlich noch von der gewählten Maschine ab.

Stufe 7: Es werden die Endprodukte P_{71} (aus P_{51}), P_{72} (aus P_{52}), P_{73} (aus P_{41} und P_{61}), P_{74} (aus P_{62}), P_{75} (aus P_{63}) auf den alternativen Maschinen R_{7a} und R_{7b} hergestellt. Bei der Produktion von P_{73} werden P_{41} und P_{61} zu gleichen Anteilen verbraucht. Produkt P_{73} bildet zusätzlich noch eine Ausnahme zu den anderen Endprodukten, da es nur auf Maschine R_{7a} herstellbar ist. Die Produktionszeiten für die einzelnen Produkte hängen nun zusätzlich noch von der gewählten Maschine ab.

Bei den einzelnen Maschinen sind noch Rüstzeiten bei der Produktion zu beachten. Die Definition der Rüstzeiten weist hier einige Besonderheiten auf. So ist nach der Produktion für jede Maschine grundsätzlich ein Reinigungsvorgang einzuplanen, dessen Dauer der halben Produktionszeit entspricht. Dieser Reinigungsvorgang entfällt nur, falls sofort nach dem Beenden der Herstellung eines Produkts mit der Herstellung eines Produkts mit kleinerer Produktnummer begonnen wird. Am Ende der gesamten Produktion auf einer Maschine ist immer eine Reinigung einzuplanen.

Gegeben ist nun noch für jedes Material die zu produzierende Menge. Die Lager können anfangs bereits teilweise gefüllt sein.

d) Aufgabe

Es werden 4 verschiedene Aufgabenstellungen untersucht. Zu minimieren ist bei diesen Aufgabenstellungen unter Einhaltung der Nebenbedingungen der Lagerkapazitäten und Losgrößen nur die Durchlaufzeit.

(1) Es sind keine Rüstzeiten zu beachten. Stufe 3 wandelt P_{22} zu 100% in P_{31}.
(2) Die Rüstzeiten sind zu beachten. Stufe 3 wandelt P_{22} zu 100% in P_{31}.
(3) Es sind keine Rüstzeiten zu beachten. Stufe 3 wandelt P_{22} zu 69,23% in P_{31}.
(4) Die Rüstzeiten sind zu beachten. Stufe 3 wandelt P_{22} zu 69,23% in P_{31}.

Bei den 4 Aufgabenstellungen werden jeweils 3 verschiedene Initialzustände (I_1, I_2, I_3) der Lager vorgegeben:

	P_{11}	P_{21}	P_{22}	P_{31}	P_{42}	P_{43}	P_{44}	P_{61}	P_{63}
I_1	0	0	0	0	0	0	0	0	0
I_2	20	20	0	20	0	0	0	0	0
I_3	10	10	0	10	0	0	0	0	0

Tabelle 9.1: Initialzustände der Lager.

Zusätzlich zu der in **[9.1]**,**[9.2]** angegebenen Bestellmenge B_0 der Endprodukte P_{71}, P_{72}, P_{73}, P_{74}, P_{75} wird noch die Bestellmenge $B_1 - B_{22}$ aus **[9.3]**, **[9.4]** und die frei gewählte B_{23} bearbeitet (**Tabelle 9.2**).
Aufgabenstellung (4) entspricht nicht ganz der Aufgabenstellung 4 aus **[9.1]**,**[9.2]**. In Aufgabenstellung 4 aus **[9.1]**,**[9.2]** können die Initialzustände der Lager bei der Optimierung frei gewählt werden. Zu beachten ist dort allerdings, dass die Lager am Ende der Produktion (zum Zeitpunkt der minimalen Durchlaufzeit) wieder ihren Initialzustand besitzen müssen. Die zu fertigenden Endprodukte sollten dabei in der originalen Aufgabenstellung 4 aus **[9.1]**,**[9.2]** genau eingehalten werden.

	P_{71}	P_{72}	P_{73}	P_{74}	P_{75}
B_0	30	30	40	20	40
B_1	20	20	20	0	0
B_2	20	20	0	20	0
B_3	20	20	0	0	20
B_4	20	0	20	20	0
B_5	20	0	20	0	20
B_6	20	0	0	20	20
B_7	0	20	20	20	0
B_8	0	20	20	0	20
B_9	0	20	0	20	20
B_{10}	0	0	20	20	20
B_{11}	10	10	20	20	30

	P_{71}	P_{72}	P_{73}	P_{74}	P_{75}
B_{12}	30	20	20	10	10
B_{13}	10	20	30	20	10
B_{14}	18	18	18	18	18
B_{15}	15	15	30	30	45
B_{16}	45	30	30	15	15
B_{17}	15	30	45	30	15
B_{18}	27	27	27	27	27
B_{19}	20	20	40	40	60
B_{20}	60	40	40	20	20
B_{21}	20	40	60	40	20
B_{22}	36	36	36	36	36
B_{23}	120	90	130	80	80

Tabelle 9.2: Bestellmengen der Endprodukte.

9.2 Modell

9.2.1 Veränderungen an der Modellierungsfunktionalität

Aufgrund der veränderten Bedingung für die Rüstzeiten und der neuen Funktionalität der variablen Losgröße ist eine Anpassung der Modellierung (vgl. **Kapitel 3.2**) erforderlich.

a) Rüstzeiten

Zur Abbildung der Rüstzeiten muss die Funktionalität der Primärressource leicht verändert werden:

3.2.2.1 Primärressource

c) Funktionalität Umrüstung

Die Produktionsdauer P^D_i der Umrüstung mit Rüstschlüssel RS_i, dem direkten Nachfolger P_k und dem direkten Vorgänger auf der Primärressource P_j ist in diesem Fall gegeben durch

$$P^D_i = \begin{cases} 0 & P^S_k = P^E_j \wedge RS_i \leq RS_j \\ M^{RZ}_{i,j} & sonst \end{cases}$$

Es ist bei den Rüstzeiten noch darauf zu achten, dass am Ende der Produktion auf jeder Primärressource Umrüstungen geplant werden. Dies wird durch eine speziell für diese Problemstellung definierte Funktion am Ende jeder Planung (vgl. **Kapitel 3.2.6**) erfüllt.

b) Variable Losgröße

Die Funktionalität der variablen Losgröße bedarf einer Veränderung im Modus und der Planung.

3.2.1 Modus

f) Losgröße

Der Modus erhält nun den Freiheitsgrad, die Menge des produzierten Materials (seine Losgröße) zu verändern. Die Losgröße Mo^L_i muss dabei im Intervall [min,max] liegen. Bei der Planung kann die Losgröße nur diskrete Werte annehmen. Sind M^{in}_i die vom Modus verbrauchten und M^{out}_j die produzierten Mengen der Materialien M_i und M_j, so ist die Losgröße definiert durch $Mo^L_i = \sum_j M^{out}_j$.

Die Losgröße wird jedoch bei der Planung nicht direkt vorgegeben, vielmehr existiert eine Liste aus vordefinierten Einstellungen für die Materialien. Jede Einstellung legt die M^{in}_i und M^{out}_j fest. Eine dieser Einstellungen wird anfangs vorgegeben. Diese ist meist eine Einstellung mit minimaler Losgröße. Bei gleicher Losgröße können verschiedene Einstellungen existieren. Diese variable Losgrößenaufteilung ist in diesem Benchmark bei **Stufe 2** vorhanden. Mit dieser Modellierung lassen sich die in **Abbildung 9.2** gezeigten Materialflüsse durch den Modus abbilden.

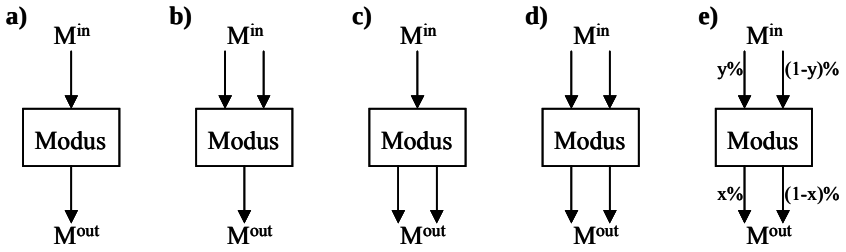


Abbildung 9.2: Mögliche Materialflüsse durch einen Modus: **a)** single input - single output, **b)** multi input - single output, **c)** single input - multi output, **d)** multi input - multi output, **e)** multi variable input - multi variable output. Bei multi input/output können statt den gezeigten 2 Materialien auch mehrere verbraucht/produziert werden.

Bei der Definition des Modus durch $\langle (B,S,b) | D | PR | (SR, P^{MK}(t)) | GR(t) | RS | (P,S,b) \rangle$ muss nun noch die Information aus den variablen Losgrößen einfließen. Dazu wird eine Erweiterung hinzugefügt:

$$\langle (B,S,b) | D | PR | (SR, P^{MK}(t)) | GR(t) | RS | (P,S,b) \rangle \rightarrow \{ (i \rightarrow j) | F_1 | F_2 \},$$

dabei wird durch $(i \rightarrow j)$ eine mögliche Einstellung für die Losgröße festgelegt, wobei i den Mengen der verbrauchten Materialien entspricht und j den Mengen der produzierten. Existieren mehrere Materialien, so sind diese durch ‚;' voneinander getrennt in der gleichen Reihenfolge, in der sie beim Modus auftauchen. Die Bedingung F_1 definiert die Liste aller möglichen Einstellungen der Losgröße und zusammen mit Bedingung F_2 die anfangs vorgegebene Einstellung. Ein ‚*' vor den Materialien **B** und **P** weist auf die variable Losgröße hin.

3.2.6.2 Planung

Zu beachten ist nun, dass nicht mehr alle Aktivitäten der Konfiguration geplant werden müssen. Aus der vorliegenden Bestellmenge wird ersichtlich, wie viele Zwischenprodukte zur Herstellung der Endprodukte benötigt werden.

Die folgenden Schritte werden ergänzt bzw. neu hinzugefügt:

1.: Die Prüfung, ob eine Aktivität A_i^k planbar ist, wird mit der anfänglichen Einstellung für die Losgröße aller Modi der Aktivität durchgeführt. Auf die Planung der Aktivität kann verzichtet werden, falls die durch A_i^k hergestellten Zwischen- bzw. Endprodukte bereits in ausreichender Menge vorhanden sind.

Zwischen Schritt 4 und Schritt 5 werden die Losgrößen angepasst:

4b: Es wird versucht, die Losgröße aller Modi der Aktivität zu erhöhen, ohne dabei die vorgegebenen Startzeitpunkte zu verändern. Dazu werden für jeden Modus aus der Liste aller Einstellungen diejenigen in eine Liste aus möglichen Einstellungen kopiert, die zu dem bestimmten Startzeitpunkt möglich sind. Ob eine Einstellung möglich ist, hängt von den Losgrößen der einzelnen Produkte und den freien bzw. belegten Kapazitäten der zugehörigen Silos ab. Die Liste der möglichen Einstellungen ist

niemals leer, da sie immer die anfängliche Einstellung des Modus enthält. Aus der Liste der möglichen Einstellungen wird nun diejenige ausgewählt, die am ehesten den noch zu produzierenden Mengen der Produkte entspricht. Sind $B^{out}{}_j$ die benötigten Mengen der Produkte, dann wird die Einstellung mit minimalem f ausgewählt, wobei $f = \sum_j \left| B^{out}{}_j - M^{out}{}_j \right|$

9.2.2 Modellierung

a) Produkte:
Es existieren 9 lagerfähige, 4 nicht lagerfähige Zwischenprodukte und 5 Endprodukte.

b) Ressourcen:
Es werden 9 Primärressourcen für die Produktion und 14 Silos für die Lagerung der Zwischen- und Endprodukte benötigt. Bei den Silos wird die maximale Kapazität auf 200% der vorhandenen eingestellt. Das Einhalten der durch die Aufgabenstellung vorgegebenen Kapazität wird durch eine weiche Grenze im Silo realisiert. In der Kostenfunktion wird die Nebenbedingung der weichen Silogrenzen mit einem Faktor 10 gewichtet.

c) Aktivitäten:
Es werden 14 verschiedene Aktivitäten definiert (**Anhang E.1**). Aus diesen Schablonen werden später bei der Umsetzung der Produktionsaufträge die zu planenden Aktivitäten erzeugt.
Produkt P_{74} kann auf zwei verschiedene Arten hergestellt werden. Bei der ersten Aktivität A_{74} wird die Stufe 7 von einer Maschine der Stufe 6 befüllt. Die maximale Menge des Produkts liegt dann bei 7 Einheiten. Bei der zweiten Aktivität A_{74b} wird Stufe 7 von beiden Primärressourcen R_{6a}, R_{6b} befüllt. Es ist damit möglich P_{74} mit einer erhöhten Losgröße zu planen. Es können nun 12 Einheiten P_{74} produziert werden. Bei der folgenden Planung kann zwischen den beiden Varianten während der Planung getauscht werden, falls von beiden Varianten genügend Aktivitäten aus den Produktionsaufträgen für die benötigten Produkte erzeugt werden.

d) Produktionsauftrag:
Es existiert eine Liste an Bestellmengen für die Endprodukte P_{71}, P_{72}, P_{73}, P_{74}, P_{75}. Diese wird in Aktivitäten umgesetzt. Da die Losgrößen nicht festgelegt sind, müssen genug Aktivitäten erzeugt werden, um die benötigten Mengen mit minimaler Losgröße pro Aktivität herstellen zu können.

9.3 Ergebnisse

Ein Vergleich mit anderen Verfahren ist nicht bei allen Aufgaben und Kombinationen aus den Initialzuständen (I_1 – I_3) und den Bestellmengen (B_0 – B_{23}) möglich. Es ist darauf zu achten, dass die Rüstzeiten in **[9.3]** und **[9.7]** anders definiert sind als in Aufgabe 2 und 4 aus **[9.1],[9.2]**. Ein Vergleich mit den angegebenen Verfahren in **[9.3]** und **[9.7]** machte daher eine Anpassung der Rüstzeiten erforderlich (-> Aufgabe (4b)).

Der **SA** Ansatz wird in allen Aufgabenstellungen angewendet. Zusätzlich werden die Ergebnisse der Aufgabe (3) und Aufgabe (4b) mit Initialzuständen I_3 denen der Verfahren **LPH** (LP-Heuristiken) aus **[9.3]**, und dem Verfahren **B+BS** (Batching+BatchScheduling) aus **[9.7]** gegenübergestellt.

Die für jeweils 100000 Sweeps des Verfahrens **SA** benötigte Rechenzeit (PC Pentium mit 650MHz) ist für jede Aufgabenstellung in **Tabelle 9.8** zu sehen. Die Rechenzeiten aus der Literatur für die einzelnen Verfahren sind aufgrund der unterschiedlichen Rechner nicht direkt miteinander vergleichbar. Der Vollständigkeit halber wurden sie jedoch angegeben (**Tabelle 9.4, Tabelle 9.6, Tabelle 9.7**).

9.3.1 Ergebnisse verschiedener Verfahren aus der Literatur

In **[9.3]** werden verschiedene LP-Heuristiken vorgestellt. Die Heuristiken werden durch Hinzufügen eines anschließenden Verbesserungsverfahrens erweitert. Im Folgenden werden immer nur die Ergebnisse der Zeitrasterheuristik (**TGB** – time grid based heuristic) angegeben, da diese die besten Ergebnisse erzielte. Ergebnisse zu den verschiedenen LP-Heuristiken sind noch in **[9.4]**, **[9.5]** und **[9.6]** gegeben. Jedoch ist zu beachten, dass die Definition der Rüstzeiten in **[9.4]** nicht mit derjenigen aus **[9.3]** übereinstimmt. Für **TGB** gilt ein Zeitlimit von 3600 Sekunden (PC Pentium II mit 266MHz). Für die maximale Anzahl der Knoten ist 50000 vorgegeben.
In **[9.7]** wird das Westenberger-Kallrath Problem unterteilt in ein *Batching*- und ein *Batch-Scheduling*-Problem. Das *Batching*-Problem wird mit Hilfe einer linearen Formulierung und CPLEX 6.0 gelöst. Dabei wurden für Aufgabe 3 aus **[9.1]**,**[9.2]** mit Rüstzeiten (entspricht Aufgabe (4) mit I_2 und B_0) nur 4 Sekunden (PC Pentium mit 800MHz) für eine optimale Lösung benötigt. Das folgende *Batch-Scheduling*-Problem wird mit einem Branch&Bound Algorithmus bearbeitet. Für Aufgabenstellung 3 aus **[9.1]**,**[9.2]** mit Rüstzeiten (entspricht Aufgabe (4) mit I_2 und B_0) konnte in 56 Sekunden (PC Pentium mit 800MHz) eine zulässige Lösung mit einer Durchlaufzeit von 88 Zeiteinheiten (entspricht 3168 Minuten; 1 Zeiteinheit=36 Minuten) gefunden werden. Insgesamt wurde zum Auffinden der Lösung somit nur 1 Minute Rechenzeit benötigt. Dies ist die bis dahin beste bekannte Lösung: *‚best known thus far for the WK example'* **[9.7]**.
Zusätzlich sind noch Ergebnisse gegeben, die Aufgabe (3) mit I_3 mit B_1–B_{22} und Aufgabe (4b) mit I_3 mit B_1–B_{22} entsprechen. Diese Ergebnisse werden noch dem besten Verfahren (**TGB**) aus **[9.3]** gegenübergestellt. Als Zeitlimit wird bei **B+BS** generell 60 Sekunden (PC Pentium mit 800MHz) vorgegeben. Bessere Ergebnisse für dieses Planungsverfahren sind in **[9.8]** gegeben.

Im Folgenden werden die Ergebnisse für die Aufgaben (3) (**Tabelle 9.3**) und (4b) (**Tabelle 9.5**) mit I_3 und den Bestellmengen B_1–B_{22} für die Verfahren **TGB** und **B+BS** tabellarisch zusammengestellt. Bei **TGB** wird weiterhin zwischen der ‚reinen' Heuristik und der Kombination mit einem anschließenden Verbesserungsverfahrens unterschieden. Es existiert in der angegebenen Literatur nur ein Ergebnis zur Aufgabe (4) (**Tabelle 9.7**) mit I_2 und der Bestellmenge B_0.
Die Durchlaufzeit wird immer in Minuten angegeben. In den zitierten Veröffentlichungen wird eine andere Zeiteinheit gewählt. Diese entspricht 36 Minuten (z.B. 88 = 3168 Minuten).
Die benötigten Rechenzeiten sind in Sekunden angegeben und befinden sich in **Tabelle 9.4** (Aufgabe (3)), **Tabelle 9.6** (Aufgabe (4b)) und **Tabelle 9.7** (Aufgabe (4)). Diese beziehen sich auf Simulationen auf einem PC Pentium mit 266MHz (**TGB**) und einem PC Pentium mit 800MHz (**B+BS**). Für Aufgabe (3) sind die Rechenzeiten für **TGB [9.6]** entnommen. Für diese Aufgabe sind leider keine Rechenzeiten für **B+BS** gegeben. Die Rechenzeiten für das Verbesserungsverfahren für **TGB** Aufgabe (4b) sind ebenfalls nicht gegeben (**[9.3]**).
Zusätzlich zu den Ergebnissen der Verfahren sind teilweise Ergebnisse angegeben, für die bewiesen werden konnte, dass sie optimal sind (**[9.3]**). Sind keine optimalen Ergebnisse bekannt, so werden die bisher besten Ergebnisse aus **[9.3]** angegeben.

Bestell menge	beste # opt	TGB H	TGB VV	B+BS	Bestell menge	beste # opt	TGB H	TGB VV	B+BS
B_1	1008#	1080	1080	1080	B_{12}	1296#	1512	1512	1440
B_2	1080#	1224	1224	1260	B_{13}	1404	1692	1620	1656
B_3	1080#	1224	1224	1224	B_{14}	1404	1728	1656	1728
B_4	1008#	1260	1188	1188	B_{15}	2268	2808	2592	2592
B_5	1008#	1224	1080	1152	B_{16}	1872	2088	1944	2232
B_6	1260#	1476	1404	1512	B_{17}	2124	2556	2268	2484
B_7	1080#	1260	1188	1296	B_{18}	2088	2592	2448	2628
B_8	1080#	1296	1152	1296	B_{19}	2916	3600	3600	3312
B_9	1332#	1584	1512	1620	B_{20}	2304	2520	2520	2664
B_{10}	1368	1476	1476	1512	B_{21}	2592	2952	2844	3060
B_{11}	1620	1728	1692	1908	B_{22}	2556	3168	2952	3168

Tabelle 9.3: Ergebnisse der Verfahren **TGB** und **B+BS** für Aufgabe (3), I_3 und B_1-B_{22}. Bei **TGB** sind zusätzlich zu den Ergebnissen der ‚reinen' Heuristik (**H**) noch jene angegeben, die durch Hinzufügen eines Verbesserungsverfahrens (**VV**) erhalten wurden. Die Spalte (**beste / # opt**) zeigt die besten bekannten Ergebnisse aus **[9.3]**. Sind optimale Werte bewiesen worden, so wird dies durch (#) gekennzeichnet. Die Ergebnisse sind immer in der Zeiteinheit Minuten zu sehen.

Bestell menge	TGB H	TGB VV
B_1-B_{10}	87.28	0.34
B_{11}-B_{14}	589.29	0.63
B_{15}-B_{18}	1418.71	2.47
B_{19}-B_{22}	3141.47	913.67

Tabelle 9.4: Benötigte Rechenzeit (PC Pentium II mit 266MHz) in Sekunden für **TGB** (**H**) und das Verbesserungsverfahren (**VV**) für die Lösung der Aufgabe (3), I_3 mit B_1–B_{22}. Die Rechenzeiten sind dabei Mittelwerte der in Paketen zusammengefassten Bestellmengen (aus **[9.6]**).

Bestell menge	beste	TGB H	TGB VV	B+BS
B_1	1296	1512	1512	1296
B_2	1368	1512	1512	1368
B_3	1368	1512	1512	1368
B_4	1368	1728	1728	1404
B_5	1296	1584	1512	1476
B_6	1728	1728	1728	1548
B_7	1512	1944	1800	1368
B_8	1512	1728	1728	1404
B_9	1944	1944	1944	1908
B_{10}	2016	2160	2016	1800
B_{11}	2376	2448	2376	2376

Bestell menge	beste # opt	TGB H	TGB VV	B+BS
B_{12}	1872	2160	1944	1872
B_{13}	2196	2340	2196	1800
B_{14}	2376	2376	2376	2052
B_{15}	4032	5328		4104
B_{16}	2736	4464		2880
B_{17}	3168	4032		3276
B_{18}	3168	4464		3276
B_{19}	7488	7488		4860
B_{20}	6624	6624		3600
B_{21}	4464	6624		4032
B_{22}	6192	7704		4824

Tabelle 9.5: Ergebnisse der Verfahren **TGB** und **B+BS** für Aufgabe (4b), I_3 und B_1-B_{22}. Bei **TGB** sind zusätzlich zu den Ergebnissen der ‚reinen' Heuristik (**H**) noch jene angegeben, die durch Hinzufügen eines Verbesserungsverfahrens (**VV**) erhalten wurden. Die Spalte (**beste**) zeigt die besten bekannten Ergebnisse aus **[9.3]**. Die Ergebnisse sind immer in der Zeiteinheit Minuten zu sehen.

Bestell menge	TGB - H	B+BS
B_1-B_{10}	~1500	45.3
B_{11}-B_{14}	~2700	60
B_{15}-B_{18}	~3600	60
B_{19}-B_{22}	~3600	60

Tabelle 9.6: Benötigte Rechenzeit in Sekunden für **TGB** (PC Pentium II mit 266MHz) und **B+BS** (PC Pentium mit 800MHz) für die Lösung der Aufgabe (4b), I_3 mit B_1–B_{22}. Die Rechenzeiten sind dabei Mittelwerte der in Paketen zusammengefassten Bestellmengen.

Bestell menge	B+BS	Zeit
B_0	3168	60

Tabelle 9.7: Ergebnis des Verfahrens **B+BS** für Aufgabe (4), I_2 und B_0. Benötigte Rechenzeit (**Zeit**) in Sekunden für **B+BS** (PC Pentium mit 800MHz).

9.3.2 Ergebnisse des Verfahrens SA

In den Tabellen im **Anhang E.2** sind die durchschnittlichen Durchlaufzeiten und deren Standardabweichung von jeweils 10 Simulationen des Verfahrens **SA** mit den beiden Strategien **S1** und **S2** für die verschiedenen Aufgabenstellungen (Aufgabe, Bestellmenge und Initialzustand der Silos) für verschiedene Anzahl Sweeps pro Temperaturschritt zu sehen. Die besten Ergebnisse werden in **Tabelle 9.8** zusammengefasst. Die für 100000 Sweeps aufgewendeten Rechenzeiten für jede Aufgabenstellung finden sich in **Tabelle 9.9**, gemessen auf einem Pentium mit 650 MHz.

Die beiden Strategien $\mathbf{S_1}$ und $\mathbf{S_2}$ unterscheiden sich dabei in der Losgröße auf der zweiten Produktionsstufe. Während bei Strategie **S1** mit einer minimalen Losgröße 10 gearbeitet wird, ist die minimale Losgröße bei Strategie **S2** 20. Dies führt dazu, dass die Produktion auf Stufe 2 bei **S2** evtl.

etwas verzögert wird, um dann jedoch umso mehr Material P_{21} und/oder P_{22} zur Verfügung zu haben.

Bei **SA** wird jeweils das gleiche logarithmische Abkühlschema für die Simulation vorgegeben. Die Temperatur wird bei der Simulation jeder Instanz von einem Startwert 2000 in 200 Schritten mit einem α-Faktor von 0.98 gesenkt. Es werden Simulationen für verschiedene Anzahl Sweeps pro Temperaturschritt durchgeführt (20, 100 und 1000). Bei den Moves werden die folgenden mit jeweils gleicher Auswahlwahrscheinlichkeit verwendet: Push1, Swap, Lin2Opt, Change, PushChange, Swap&Change, SwapSameResource und Lin2OptSameResource.

Als Ergebnis zählt bei den Aufgabenstellungen immer nur die Durchlaufzeit. Jedoch werden bei Aufgabenstellungen mit Rüstzeiten diese zur Bewertung herangezogen, da sie die Durchlaufzeit beeinflussen. In der gewichteten Summe für die Energieberechnung werden die Gewichte 5 für die Durchlaufzeit und 1 für die Summe der Rüstzeiten verwendet.

Bestell	(1)			(2)			(3)			(4)			(4b)
menge	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3	I_3
B_0	2376	1944	2016	2592	2304	2304	2376	2016	2088	2772	2376	2448	2556
B_1	1296	1008	1008	1476	1080	1188	1296	1008	1008	1476	1080	1188	1296
B_2	1296	1008	1008	1476	1080	1188	1368	1008	1080	1512	1188	1224	1296
B_3	1296	1008	1008	1440	1080	1188	1368	1008	1080	1476	1188	1224	1296
B_4	1296	972	1008	1512	1116	1152	1296	972	1008	1548	1116	1152	1224
B_5	1296	1008	1008	1440	1224	1152	1296	1008	1008	1440	1224	1152	1224
B_6	1584	1224	1296	1890	1530	1548	1656	1224	1368	1926	1530	1566	1620
B_7	1296	972	1008	1512	1116	1224	1368	1008	1080	1566	1152	1260	1296
B_8	1296	972	1008	1458	1116	1224	1368	1008	1080	1530	1152	1224	1332
B_9	1584	1224	1296	1890	1530	1548	1656	1224	1368	1944	1530	1584	1584
B_{10}	1584	1224	1368	1962	1548	1548	1656	1296	1368	1944	1548	1548	1692
B_{11}	1800	1440	1512	2088	1728	1728	1872	1476	1584	2160	1728	1728	1872
B_{12}	1512	1224	1224	1692	1404	1404	1584	1224	1296	1764	1404	1440	1512
B_{13}	1584	1296	1296	1908	1512	1584	1728	1296	1368	1998	1512	1584	1728
B_{14}	1872	1440	1584	2088	1692	1764	1908	1440	1584	2160	1728	1836	2016
B_{15}	2376	2052	2160	2808	2448	2484	2592	2124	2232	2916	2484	2556	2700
B_{16}	2160	1872	1944	2448	2160	2160	2232	1872	1944	2448	2160	2196	2304
B_{17}	2232	1872	1872	2592	2232	2232	2268	1872	2016	2664	2304	2322	2520
B_{18}	2268	1908	1980	2628	2304	2304	2376	1944	2052	2736	2340	2394	2592
B_{19}	3168	2700	2736	3582	3204	3276	3204	2772	2772	3798	3294	3276	3744
B_{20}	2592	2304	2304	2880	2592	2592	2592	2304	2304	2844	2484	2592	2700
B_{21}	2520	2160	2232	3096	2520	2610	2592	2160	2304	3096	2592	2718	2952
B_{22}	2520	2232	2376	3096	2628	2808	2736	2268	2448	3240	2844	2844	3060
B_{23}	5976	5544	5472	7470	7038	6786	6300	5724	6012	7650	7182	7056	8604

Tabelle 9.8: Zu sehen sind die besten Durchlaufzeiten für die verschiedenen Aufgabenstellungen (Aufgabe, Bestellmengen, Initialzustand der Silos), die mit dem Verfahren **SA** und den beiden Strategien innerhalb der Rechenzeitgrenze erhalten wurden.

Meist konnte die beste gefundene Lösung bereits bei kleinen (100 Sweeps pro Temperaturschritt) oder sehr kleinen Rechenzeiten gefunden werden. Nur bei den Aufgaben mit größeren Bestellmengen war eine Erhöhung der Anzahl Sweeps auf 1000 pro Temperaturschritt erforderlich. Es zeigt sich, dass die Planung zum frühest möglichen Startzeitpunkt (Strategie **S1**) fast immer schlechtere Ergebnisse erzielt als eine Planung, bei der der Startzeitpunkt auf Produktionsstufe 2 absichtlich verzögert wird, um mit maximaler Losgröße zu arbeiten (Strategie **S2**). Die beiden Strategien lie-

fern für I_2 die gleichen Ergebnisse, da hier bereits zu Beginn auf Stufe 2 mit maximaler Losgröße gearbeitet werden kann.

Bestell menge	Aufgabe (1)	(2)	(3)	(4),(4b)	Bestell menge	Aufgabe (1)	(2)	(3)	(4),(4b)
B_0	240	257	355	401	B_{12}	69	75	98	103
B_1	31	34	37	42	B_{13}	84	88	117	130
B_2	38	41	59	63	B_{14}	74	81	113	123
B_3	47	50	72	73	B_{15}	191	210	295	330
B_4	37	39	46	48	B_{16}	140	141	191	223
B_5	51	53	63	67	B_{17}	147	157	219	239
B_6	53	54	77	79	B_{18}	150	166	242	260
B_7	44	46	67	71	B_{19}	350	397	586	627
B_8	56	57	79	81	B_{20}	245	277	345	367
B_9	65	71	126	127	B_{21}	270	283	404	437
B_{10}	60	61	84	89	B_{22}	235	263	401	420
B_{11}	105	108	163	177	B_{23}	1820	1921	2703	2720

Tabelle 9.9: Benötigte Rechenzeit (PC Pentium mit 650MHz) in Sekunden für 100000 Sweeps mit dem Verfahren **SA** für die verschiedenen Aufgaben und Bestellmengen. Eine typische Simulation mit 200 Temperaturschritten und jeweils 1000 Sweeps benötigt also z.B. für Aufgabe (1) und Bestellmenge B_0 480 Sekunden.

9.3.3 Gegenüberstellung der Ergebnisse der verschiedenen Verfahren

Aufgabe (3) mit I_3:

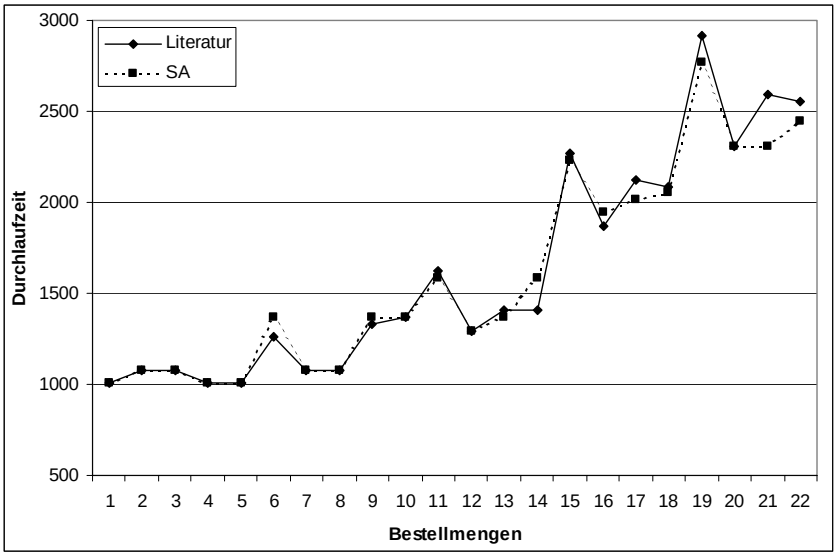


Abbildung 9.3: Zu sehen sind die jeweils besten Durchlaufzeiten aus den Literaturergebnissen und den beiden Strategien bei der Planung mit **SA**.

Von den 10 bewiesenen Optima konnten 8 erreicht werden, jedoch nicht immer mit beiden Strategien gleichzeitig. Meist wurde dieses Optimum dann bei geringer Rechenzeit (20 Sweeps pro Temperaturschritt) in allen Simulationen gefunden. Bei den restlichen 12 besten bekannten Ergebnissen konnten 2 erreicht und 8 verbessert werden (vgl. **Abbildung 9.3**). Die Verfahren aus der Literatur benötigten jedoch überwiegend erheblich mehr Rechenzeit, um diese besten bekannten (optimalen) Ergebnisse zu finden.

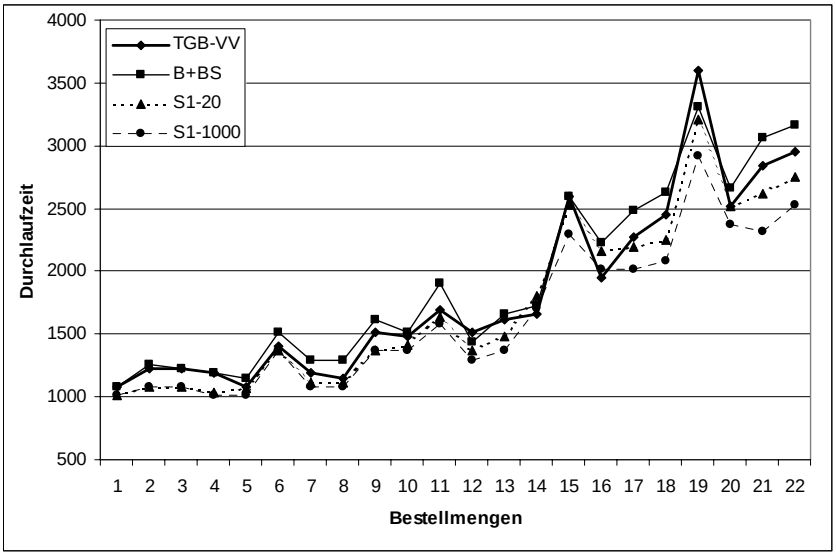


Abbildung 9.4: Zu sehen sind die jeweils besten Durchlaufzeiten aus der Literatur für die Verfahren **TGB-VV** und **B+BS** (bei begrenzter Rechenzeit) und die durchschnittlichen Durchlaufzeiten des Verfahrens **SA** mit Strategie **S1** mit 20 und 1000 Sweeps pro Temperaturschritt.

Im Vergleich zu den Ergebnissen der Verfahren aus der Literatur mit begrenzter Rechenzeit (vgl. **Abbildung 9.4**) schneidet **SA** erheblich besser ab. Bereits bei den kleinsten Rechenzeiten (20 Sweeps pro Temperaturschritt, entspricht 2 bis 24 Sekunden) erzielt **SA** mit Strategie **S1** in 20 der 22 Fälle ein besseres durchschnittliches Ergebnis als die beiden Verfahren aus der Literatur, mit Strategie **S2** in 19 Fällen. Eine Erhöhung der Rechenzeit verbessert die Durchlaufzeiten vor allem bei großen Bestellmengen zum Teil erheblich.

Aufgabe (4) mit I_2:

Bestell menge	B+BS	SA-20	SA-1000	SA-best
B_0	3168	2736,0	2419,2	2376

Tabelle 9.10: Zu sehen ist ein Vergleich der Durchlaufzeiten der Verfahren **SA** und **B+BS**. Bei **SA** ist die durchschnittliche Durchlaufzeit bei 20 und 1000 Sweeps pro Temperaturschritt und die beste gefundene Durchlaufzeit gegeben.

Bereits bei kleinen Rechenzeiten (20 Sweeps pro Temperaturschritt; entspricht 16 Sekunden Rechenzeit auf einem PC Pentium mit 650MHz) ist die durchschnittliche Durchlaufzeit deutlich besser als die in **[9.8]** gegebene beste Durchlaufzeit (bei 60 Sekunden mit 800MHz) (**Tabelle 9.10**).
In **Abbildung 9.5** ist die für diese Aufgabenstellung beste bekannte Lösung gegeben. Diese wurde durch die Vorgehensweise aus **Kapitel 9.3.4** erzielt.

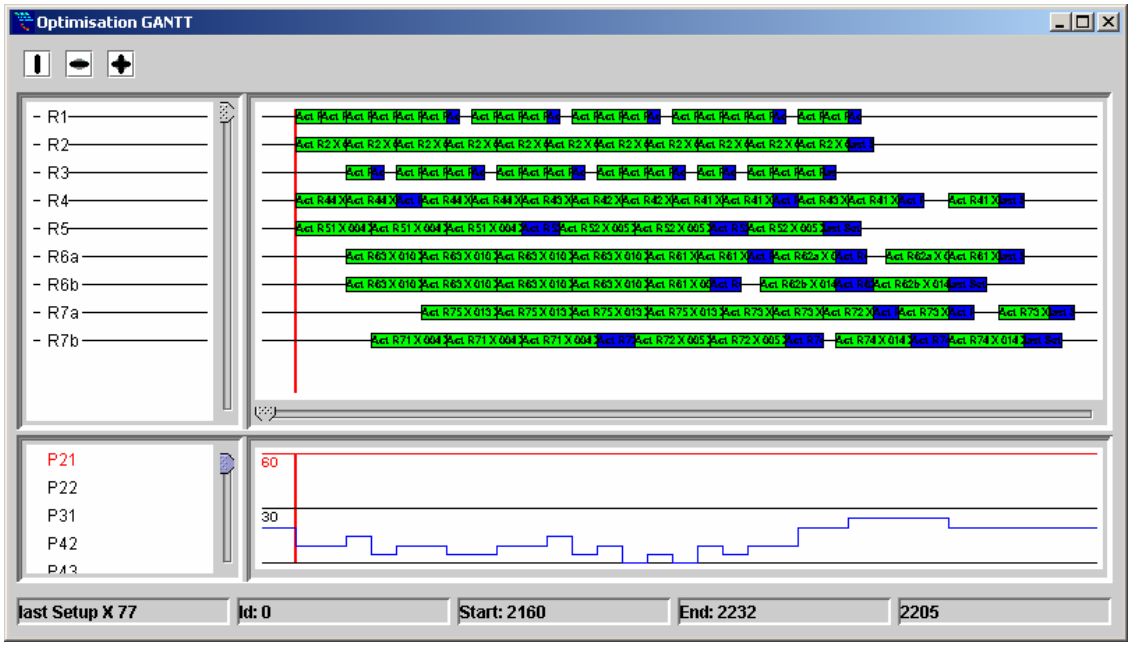


Abbildung 9.5: Zu sehen ist die beste Lösung für Aufgabe (4) mit Initialzuständen der Silos I_2 mit einer Durchlaufzeit von 2232 Minuten.

Aufgabe (4b) mit I_3:

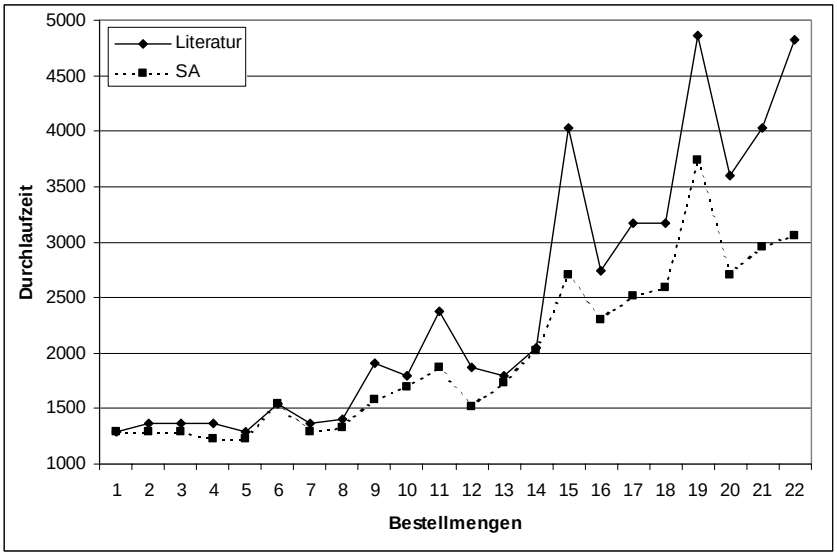


Abbildung 9.6: Zu sehen sind die jeweils besten Durchlaufzeiten aus den Literaturergebnissen und den beiden Strategien bei der Planung mit **SA**.

Hier zeigt sich ein noch viel deutlicheres Ergebnis (vgl. **Abbildung 9.6**). Nur in 2 Fällen erhält **SA** die gleiche beste Durchlaufzeit, in allen anderen Fällen (20) schneidet **SA** besser ab.

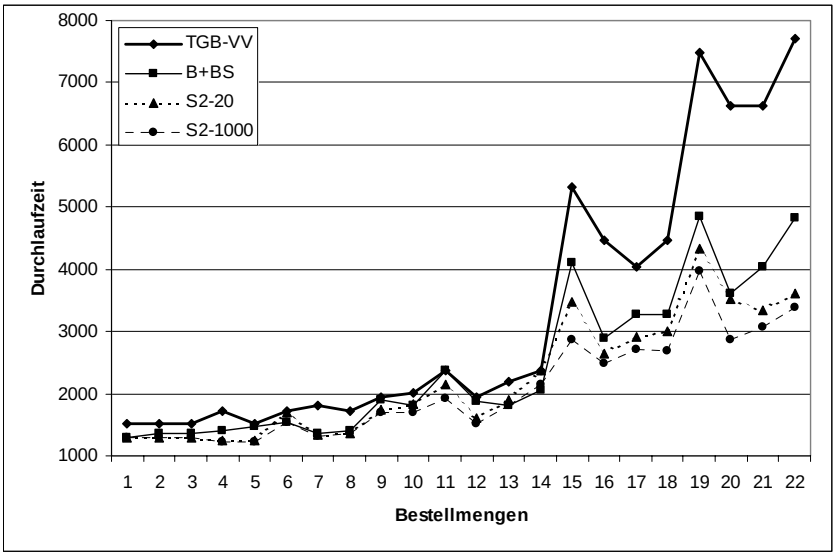


Abbildung 9.7: Abgebildet sind die jeweils besten Durchlaufzeiten aus der Literatur für die Verfahren **TGB-VV** und **B+BS** (bei begrenzter Rechenzeit) und die durchschnittlichen Durchlaufzeiten des Verfahrens **SA** mit Strategie **S2** mit 20 und 1000 Sweeps pro Temperaturschritt.

Im Vergleich der besten Ergebnisse für die beiden Verfahren aus der Literatur zeigt sich, dass die durchschnittlichen Durchlaufzeiten von **SA** mit Strategie **S2** bereits bei kleinen Rechenzeiten (20 Sweeps pro Temperaturschritt) meist deutlich besser sind als von **TGB-VV**. Im Vergleich zu den besten Durchlaufzeiten von **B+BS** erreicht **SA** bei kurzen Rechenzeiten in 18 Fällen kürzere durchschnittliche Durchlaufzeiten, bei längeren Rechenzeiten in 19 Fällen (vgl. **Abbildung 9.7**).

9.3.4 Vergleich der Ergebnisse bei fixierter Losgröße

In **Tabelle 9.11** sind die besten Ergebnisse aus der Optimierung mit Strategie **S2** mit 1000 Sweeps pro Temperaturschritt auf ‚vereinfachten' Problemstellungen gegeben. Die Vereinfachung bezieht sich auf Produktionsstufe 1. Das Silo für P_{11} ist hier ohne Kapazitätsbeschränkung. Die Aktivitäten, die P_{11} herstellen, werden fixiert, so dass Produkt P_{11} stufenweise zur Verfügung steht, ohne dass die Aktivitäten der Produktionsstufe geplant werden müssen. Die so erzeugten Produktionspläne verletzen meist die Kapazitätsbeschränkung des Silos S_{11} und sind deshalb nicht gültig. Sie können jedoch immer in gültige Produktionspläne umgesetzt werden, falls die Aktivitäten der Produktionsstufe 1 nach der Optimierung zeitlich verzögert werden. Es konnten so einige der besten Ergebnisse weiter verbessert werden. Im vorhergehenden Kapitel wurden diese Ergebnisse jedoch nicht zu einem Vergleich herangezogen.

Meist ergaben sich die Verbesserungen bei den größeren Instanzen. Bei diesen Instanzen ist die Standardabweichung sehr groß und die Rechenzeit reicht nicht aus, um sehr nah ans Optimum zu

gelangen. Die Vereinfachung bei der Planung führt zu einem kleineren Suchraum, wodurch die Ergebnisse bei begrenzter Rechenzeit durchaus verbessert werden können.

Bestell	(1)			(2)			(3)			(4)			(4b)
menge	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3	I_3
B_0	**2268**	1944	2016	**2556**	**2268**	**2232**	2376	**1944**	2088	**2700**	**2232**	**2358**	**2412**
B_1	1296	1008	1008	**1440**	1080	1188	1296	1008	1008	1476	1080	1188	1296
B_2	1296	1008	1008	1476	1080	1188	1368	1008	1080	1512	1188	1224	1296
B_3	1296	1008	1008	1440	1080	1188	1368	1008	1080	1476	1188	1224	1296
B_4	1296	972	1008	1512	1116	1152	1296	972	1008	1548	1116	1152	1224
B_5	1296	**972**	1008	1440	**1080**	1152	1296	**972**	1008	1440	**1080**	1152	1224
B_6	1584	1224	1296	1890	1530	1548	1656	1224	1368	1926	1530	1566	1620
B_7	1296	972	1008	1512	1116	1224	1368	1008	1080	1566	1152	1260	1296
B_8	1296	972	1008	1458	1116	1224	1368	1008	1080	1530	1152	1224	**1296**
B_9	1584	1224	1296	1890	1530	1548	1656	1224	1368	1944	1530	1584	1584
B_{10}	1584	1224	1368	1962	1548	1548	1656	1296	1368	1944	1548	1548	1692
B_{11}	1800	1440	1512	2088	1728	1728	1872	1476	1584	2160	1728	1728	1872
B_{12}	1512	1224	1224	1692	1404	1404	1584	1224	1296	1764	1404	1440	1512
B_{13}	1584	1296	1296	**1872**	1512	1584	**1656**	1296	1368	**1908**	1512	1584	**1656**
B_{14}	**1692**	**1368**	**1404**	**1980**	**1620**	**1656**	**1728**	**1368**	**1440**	**2070**	**1620**	**1710**	**1764**
B_{15}	2376	2052	2160	2808	2448	2484	2592	2124	2232	2916	2484	2556	2700
B_{16}	2160	1872	**1872**	**2340**	**2088**	**2052**	2232	1872	1944	2448	**2088**	**2160**	**2268**
B_{17}	2232	1872	1872	**2520**	2232	2232	2268	1872	2016	2664	2304	**2304**	2520
B_{18}	**2232**	**1872**	**1908**	**2556**	**2196**	**2196**	**2268**	**1872**	**2016**	**2664**	**2232**	**2268**	**2448**
B_{19}	**3132**	2700	2736	**3564**	3204	**3222**	3204	2772	2772	**3708**	3294	**3222**	**3564**
B_{20}	2592	2304	2304	**2772**	**2484**	**2484**	2592	2304	2304	**2772**	2484	**2484**	**2592**
B_{21}	2520	2160	2232	**2916**	2520	2610	2592	2160	2304	**3024**	2592	2718	2952
B_{22}	2520	2232	2376	3096	2628	2808	2736	2268	2448	3240	2844	2844	3060
B_{23}	**5760**	**5328**	5472	**6642**	**6372**	**6372**	**5760**	**5364**	**5472**	**6840**	**6372**	**6660**	**7308**

Tabelle 9.11: Bei einer vereinfachten Planung der Problemstellung, die auf die Planung der Produktionsstufe 1 verzichtet, allerdings eine ‚Reparatur' des ungültigen optimierten Produktionsplans erforderte, wurden einige Ergebnisse noch verbessert. Diese sind hervorgehoben.

Im Folgenden soll nun die Losgröße für alle Produkte festgelegt werden. Dazu wird bei den Produkten P_{71}, P_{72}, P_{73}, P_{74}, P_{75}, P_{42}, P_{43}, P_{44}, P_{31} mit einer Losgröße von 10 gearbeitet. Bei P_{74} muss deshalb die 2. Variante verwendet werden, bei der R_{6a} und R_{6b} die Produktionsstufe 7 befüllen. Die Produktion von P_{31} verbraucht immer 10 P_{22}, erstellt jedoch je nach Aufgabenstellung ≤ 10 P_{31}. Die Produkte P_{61}, P_{63} werden mit 5 Einheiten geplant. Die Losgröße auf Produktionsstufe 2 wird auf 20 festgelegt. Die Aufteilung in P_{21} und P_{22} wird dabei am Anfang je nach Bedarf der beiden Produkte fixiert. Die Produktion von P_{11} wird wieder fixiert und nach der Planung repariert. Die bei jeweils 10 Simulationen gefundenen besten Ergebnisse sind in **Tabelle 9.12** zu sehen.

Trotz Fixierung der Losgröße während der Planung konnten bei manchen Instanzen Lösungen gefunden werden, die den besten bisherigen entsprechen. Die restliche Lösungen waren allerdings alle von schlechterer Qualität. Es fällt auf, dass die meisten der gleich guten Lösungen bei Problemstellungen mit einem anfänglichen Füllstand der Silos >0 gefunden wurden.

Der Grund für die relativ guten Lösungen liegt in der richtigen Wahl der Losgrößen. So scheint es bei den erfolgreich gelösten Instanzen richtig zu sein, die Lösgrößen auf ein Vielfaches von 5 festzusetzen. Dadurch wird das Material gleichmäßig durch die Produktionsstufen geschleust und die Lagerkapazitäten beachtet. Meist stellt Produktionsstufe 2 das Problem bei der Planung dar. Es

scheint hier nicht sinnvoll, eine feste Aufteilung der Produkte P_{21} und P_{22} zu wählen. Bei den Aufgabestellungen 3 und 4 und den Bestellmengen 1, 4, 5 erzeugen beide Verfahren (fixierte Losgröße und variabel geplante) Lose mit jeweils 10 P_{21} und 10 P_{22}. Daher auch die gleichen Resultate. Gefüllte Lager können eine nicht optimal eingestellte Losgröße bei kleinen Instanzen ausgleichen, da die gelagerten Produkte P_{21} und P_{22} ausreichen, um die weiterverarbeitenden Schritte zu beginnen. Die so gewonnene Zeit reicht aus, um genügend Produkte P_{21} und P_{22} herzustellen, ohne dabei auf die Nachfolgerstufe achten zu müssen.
Insgesamt gesehen haben die mit einer festen Losgröße erzielten Ergebnisse jedoch durchschnittlich eine um 3,3% längere Durchlaufzeit bei den jeweils besten Lösungen.

Bestell	(1)			(2)			(3)			(4)			(4b)
menge	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3	I_3
B_0	2340	2052	**2016**	2628	2322	2304	2448	2016	**2088**	2736	2340	2448	2520
B_1	**1296**	**1008**	**1008**	1476	**1080**	**1188**	**1296**	**1008**	**1008**	**1476**	**1080**	**1188**	**1296**
B_2	1368	**1008**	1080	1548	1188	1224	1440	**1008**	1152	1620	**1188**	1332	1368
B_3	1368	**1008**	1080	1548	1188	**1188**	1440	**1008**	1152	1620	**1188**	1332	1332
B_4	1368	**972**	**1008**	1548	1170	**1152**	**1296**	**972**	**1008**	**1548**	1170	**1152**	**1224**
B_5	1368	**972**	**1008**	1548	**1080**	**1152**	**1296**	**972**	**1008**	**1440**	**1080**	**1152**	**1224**
B_6	**1584**	**1224**	**1296**	1908	1548	**1548**	1728	**1224**	1440	2052	1548	1656	1728
B_7	1368	**972**	**1008**	**1512**	1170	**1224**	1440	**1008**	**1080**	1656	**1152**	**1260**	**1296**
B_8	1368	**972**	**1008**	1512	**1116**	**1224**	1440	**1008**	**1080**	1602	**1152**	**1224**	**1296**
B_9	**1584**	**1224**	**1296**	1908	1548	**1548**	1728	**1224**	1440	2052	1548	1656	1728
B_{10}	1728	1440	1404	**1962**	1710	1638	1836	1404	1440	2106	1854	1656	1764
B_{11}	1944	1620	1620	2178	1944	1854	2052	1584	1656	2322	1836	1872	1980
B_{12}	1584	**1224**	1296	1800	**1404**	1440	1656	**1224**	1368	1836	**1404**	1548	1656
B_{13}	1656	**1296**	1368	**1872**	1584	**1584**	1728	**1296**	1440	2016	**1512**	1656	1728
B_{14}	1728	1404	1440	**1980**	1710	**1656**	1836	**1368**	1512	2124	**1620**	1764	1872
B_{15}	2664	2340	2340	2970	2682	2664	2772	2304	2376	3114	2628	2664	2880
B_{16}	2232	**1872**	1944	2448	**2088**	2160	2304	**1872**	2016	2592	**2088**	2196	2376
B_{17}	2304	2016	2016	**2520**	2304	2304	2448	2016	**2016**	2736	2340	2376	2592
B_{18}	**2232**	**1872**	1944	2592	**2196**	2232	2376	**1872**	**2016**	2700	2268	2376	**2448**
B_{19}	**3132**	2808	2844	3618	3456	**3240**	3276	2808	2880	**3708**	3348	**3222**	**3564**
B_{20}	**2592**	**2304**	**2304**	**2772**	**2484**	**2484**	**2592**	**2304**	**2304**	**2772**	**2484**	**2484**	**2592**
B_{21}	2664	2304	2304	3006	2736	2826	2736	2448	**2304**	3168	2916	2808	**2952**
B_{22}	2772	2520	2448	3204	3024	2880	2880	2448	2592	3276	2916	2970	3204
B_{23}	5796	5400	**5472**	6660	6426	6390	5904	5400	5616	7056	6534	**6660**	**7308**

Tabelle 9.12: Die besten Ergebnisse aus jeweils 10 Simulationen für die verschiedenen Aufgabenstellungen. Hier wurde mit einer fixierten Losgröße geplant. Die markierten Ergebnisse sind solche, die die gleiche Qualität aufweisen wie die besten aus **Tabelle 9.11**.

9.4 Bearbeitung einer neuen Aufgabenstellung

Es soll nun noch Aufgabe (5) bearbeitet werden. Diese lehnt sich an Aufgabe 5 aus **[9.1]**, **[9.2]** an. Hier ist das Ziel der Optimierung, innerhalb einer vorgegebenen maximalen Produktionszeit einen Mix aus Endprodukten mit vorgegebenen Verkaufspreisen zu produzieren und die gesamten Verkaufserlöse zu maximieren. Die Initialzustände der Lager sind vorgegeben (I_2). Der in Aufgabe 5 (**[9.1]**, **[9.2]**) gegebene Einkaufpreis wird hier nicht berücksichtigt.

Als maximale Produktionszeit sind 1 Tag (1440 Minuten), 2 Tage (2880 Minuten) und 5 Tage (7200 Minuten) gegeben. Die Verkaufspreise sind in **Tabelle 9.13** aufgelistet.
Die Rahmenbedingungen, wie z.B. Rüstzeiten, werden aus den Aufgaben (1) bis (4) übernommen.
Bei der Bewertung einer Lösung wird nun statt der Durchlaufzeit die gewichtete Summe aller bis zum Produktionsende hergestellten Produkte, der Erlös E, herangezogen. Zu beachten sind bei den Aufgabenstellungen noch die Rüstzeiten am Ende der Produktion. Diese sollten noch innerhalb der gegebenen Produktionszeit durchgeführt werden.

$$E = \sum_{i=1}^{5} M(P_{7i}) * P(P_{7i}),$$

wobei $M(P_{7i})$ der hergestellten Menge des Produkts P_{7i} entspricht und $P(P_{7i})$ dem Verkaufspreis. E muss dabei maximiert werden.

Produkt	**P_{71}**	**P_{72}**	**P_{73}**	**P_{74}**	**P_{75}**
Preis	10	10	30	20	15

Tabelle 9.13: Verkaufspreise der einzelnen Produkte.

Die für die verschiedenen Rahmenbedingungen erzielten Ergebnisse sind in den **Tabellen 9.14**, **9.15** und **9.16** gegeben.

Aufgabe	**Erlös**	**Lösung (P_{71},P_{72},P_{73},P_{74},P_{75})**
(1)	2500	(0,50,60,10,0)
(2)	2300	(0,50,60,0,0)
(3)	2500	(0,50,60,10,0)
(4)	2300	(0,50,60,0,0)

Tabelle 9.14: Erlös und Lösung bei den einzelnen Rahmenbedingungen für 1 Tag.

Aufgabe	**Erlös**	**Lösung (P_{71},P_{72},P_{73},P_{74},P_{75})**
(1)	5400	(0,120,140,0,0)
(2)	4900	(0,110,120,10,0)
(3)	5400	(0,120,140,0,0)
(4)	4900	(0,110,120,10,0)

Tabelle 9.15: Erlös und Lösung bei den einzelnen Rahmenbedingungen für 2 Tage.

Aufgabe	**Erlös**	**Lösung (P_{71},P_{72},P_{73},P_{74},P_{75})**
(1)	14000	(0,310,360,10,0)
(2)	12700	(0,310,320,0,0)
(3)	14000	(0,310,360,10,0)
(4)	12700	(0,310,320,0,0)

Tabelle 9.16: Erlös und Lösung bei den einzelnen Rahmenbedingungen für 5 Tage.

9.5 Zusammenfassung

Bei der Bearbeitung dieser Problemstellung wurde absichtlich auf ein Anpassen der einzelnen Parameter des Optimierungsverfahrens auf die einzelnen Aufgabenstellungen verzichtet. Es wird immer das gleiche Abkühlschema für die Temperatur verwendet. Eine Anpassung ist hier nicht unbedingt erforderlich, da sich die Produktionszeiten der einzelnen Produkte nicht verändert haben, sondern nur die zu produzierende Mengen. Die Auswahl und die Durchführung der einzelnen Moves wurde nicht angeglichen. Es wird aus der Liste der verwendeten Moves jeder mit gleicher Wahrscheinlichkeit ausgewählt. Trotz dieser einmal festgelegten Parameter konnten mit **SA** sehr gute Ergebnisse für die einzelnen Aufgabenstellungen erhalten werden, die, soweit vergleichbar, meist besser als die in der Literatur angegebenen waren. Eine Verbesserung der Ergebnisse könnte noch erreicht werden durch eine Anpassung des Abkühlschemas und der Auswahlwahrscheinlichkeit der Moves an jede Aufgabenstellung und Instanz. Unter einer Verbesserung der Ergebnisse ist in diesem Zusammenhang ein Erreichen einer kleineren Durchlaufzeit oder einer gleichen Durchlaufzeit in einer kürzeren Rechenzeit zu verstehen.
Die Bestimmung der Losgröße stellt bei diesem Benchmark einen wichtigen Teil bei der Planung dar. Im Unterschied zum Verfahren **B+BS**, bei dem die Losgröße in einem Schritt (**batching**) für alle Lose bestimmt wird und erst in einem zweiten Schritt die Einplanung der Lose zu bestimmten Zeitpunkten (**BatchScheduling**) erfolgt, wird bei **SA** die endgültige Losgröße erst bei der Planung des Loses festgelegt. Es kommt vor, dass sich die Losgrößen verschiedener Lose des gleichen Produkts unterscheiden. Auch die Aufteilung der Produktion von P_{21} und P_{22} bei Stufe 2 ist bei verschiedenen Losen möglicherweise unterschiedlich. Es ist in den Ergebnistabellen deutlich zu sehen, dass eine fixierte Losgröße zwar bei einigen Instanzen gleiche Resultate liefert, jedoch insgesamt gesehen der variablen Planung unterlegen ist. Die globalere Sicht auf das Optimierungsproblem scheint hier, trotz der erschwerten Planung durch zusätzliche Freiheitsgrade, erfolgreicher zu sein als eine Trennung der Planungsebenen der Losgrößen und der Maschinenzuordnung mit Zeitpunkten. Trotz der Freiheiten bei der Planung der Losgröße bleibt dennoch die Einschränkung, dass zuerst der Startzeitpunkt des Loses mit minimaler Losgröße gesucht und erst danach die Losgröße angepasst wird. Mögliche Auswirkungen dieser Einschränkung sind beim Vergleich der beiden Strategien **S1** und **S2** zu sehen. Eine Strategie, die absichtlich den Startzeitpunkt verspätet, um dann mit erhöhter Losgröße zu produzieren, ist in gewissen Fällen erfolgreicher als eine Strategie, die immer mit der minimalen Losgröße den Startzeitpunkt bestimmt. Somit lässt sich eine Verbesserung des ursprünglichen Verfahrens zur Bestimmung der Startzeitpunkte und Losgrößen finden. Eine Möglichkeit wäre es, zusätzlich zum bisherigen Verfahren zu bestimmen, wie lange der Startzeitpunkt eines Loses verzögert werden müsste, um mit höherer Losgröße zu produzieren und dann für jedes Los einzeln zu entscheiden, ob dieses mit einem verzögerten Startzeitpunkt geplant wird.

Kapitel 10

Das Fließbandproblem

10.1 Problemstellung

Auf einem Fließband werden Produkte bearbeitet, bei denen die Herstellung in klar definierte Schritte unterteilt werden kann, die nacheinander auszuführen sind. Diese Schritte benötigen meist ein Vielfaches einer bestimmten Zeiteinheit, der Taktzeit. An jeder Position des Fließbandes, die einen bestimmten Produktionsschritt darstellt, muss nun ausreichend Kapazität vorhanden sein, so dass im Durchschnitt ein Produkt pro Takt bearbeitet werden kann. Die Produktionsschritte, die bei jedem Produkt erforderlich sind, stellen meist kein Problem der Planung dar. Neben diesen existieren jedoch noch solche Produktionsschritte, die nicht bei jedem Produkt nötig sind. Sie integrieren z.B. Sonderausstattungen in die Produkte. Die Produktionsschritte für die Sonderausstattungen belegen eigene Positionen des Fließbandes. Deren Kapazität ist aus Kostensparmaßnahmen in der Regel so ausgelegt, dass sie pro Zeiteinheit (>Taktzeit) eine bestimmte Anzahl an Produkten mit den Sonderausstattungen versehen kann. Diese Anzahl ergibt sich aus der durchschnittlichen Ausstattung der Produkte mit der Sonderausstattung. Um die Positionen der Sonderausstattungen nicht zu überlasten, ist es deshalb erforderlich, die Produkte mit der Sonderausstattung gleichmäßig zu verteilen. Neben dieser Gleichverteilung existieren noch weitere Nebenbedingungen für die Sonderausstattungen, z.B. minimaler Abstand, maximaler Pulk, etc.. Die Nebenbedingungen (Restriktionen) können dabei von unterschiedlicher Priorität sein, so existieren Nebenbedingungen, die in einer gültigen Lösung des zugehörigen Produktionsplanungsproblems unbedingt eingehalten werden müssen und solche, bei denen eine möglichst gute Einhaltung erwünscht, aber nicht erforderlich ist. Sind sehr viele Nebenbedingungen vorhanden, so existiert womöglich keine Lösung, die alle Nebenbedingungen erfüllt.
Eine solche Fließbandproduktion wird z.B. in der Automobilindustrie verwendet. Bei seiner Bestellung kann ein Kunde aus einer Vielzahl an Sonderausstattungen wählen, so z.B. Klimaanlage, Schiebedach, Rechtslenker, etc.. In der Automobilindustrie existieren meist sehr viele Nebenbedingungen unterschiedlicher Priorität, die im Normalfall nicht alle gleichzeitig erfüllt werden können. Die Planungsprobleme sind *overconstrained*.

10.2 Modell

Die Darstellung der Lösung erfolgt in dieser Problemstellung mittels einer Konfiguration aus Produkten. Im Unterschied zur Erstellung des Produktionsplans (**Kapitel 3.2.6**) müssen hier keine Zeitpunkte festgelegt werden. Die Position eines Produkts in der Konfiguration gibt den Takt, und damit den Zeitpunkt, an, zu dem das Produkt gestartet wird. Die Definition der Konfiguration (**Kapitel 3.2.4**) und der Nachbarschaft (**Kapitel 3.2.5**) kann teilweise übernommen werden. Sie wurde bereits in **[2.2]** verwendet, um *car sequencing problems* zu bearbeiten. Bei der Optimierung sollen die durch eine Kostenfunktion definierten Kosten minimiert werden. Die Kosten ergeben sich dabei aus der Verletzung der Nebenbedingungen.

10.2.1 Konfiguration

Unter einer Konfiguration $\langle K \rangle$ soll im Folgenden eine Sequenz aus Produkten P_i verstanden werden. Für diese Konfiguration werden nun einige Definitionen gegeben:

- $|\langle K \rangle|$ gibt die Anzahl der Produkte in der Konfiguration $\langle K \rangle$ an.
- Jede Konfiguration, die P_i enthält, lässt sich darstellen als $\langle K_1 \rangle \bullet P_i \bullet \langle K_2 \rangle$ mit zwei möglicherweise leeren Konfigurationen $\langle K_1 \rangle$ und $\langle K_2 \rangle$ ($|\langle K_1 \rangle| \geq 0, |\langle K_2 \rangle| \geq 0$). Man spricht von einer leeren Konfiguration $\langle K \rangle$, falls $|\langle K \rangle| = 0$.
- $K(i)$ gibt das Produkt an der Position i an, also $P_j = K(i)$ falls $\langle K \rangle = \langle K_1 \rangle \bullet P_j$ und $|\langle K_1 \rangle| = i - 1$.
- Unter $\overline{\langle K \rangle}$ versteht man eine gespiegelte Konfiguration: $\overline{\langle K_1 \rangle \bullet P_i} = P_i \bullet \overline{\langle K_1 \rangle}$.
- $\langle K_2 \rangle \subseteq \langle K \rangle$ stellt eine Teilkonfiguration von $\langle K \rangle$ dar. Es gilt: $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle$. Insbesondere ist $P_i \subseteq \langle K \rangle$, falls $\langle K \rangle = \langle K_1 \rangle \bullet P_i \bullet \langle K_3 \rangle$.
- Zwei Konfigurationen $\langle K_1 \rangle = P_i \bullet \langle K_3 \rangle$ und $\langle K_2 \rangle = P_j \bullet \langle K_4 \rangle$ sind identisch ($\langle K_1 \rangle = \langle K_2 \rangle$), falls sie aus identischen Produkten bestehen, also falls $|\langle K_1 \rangle| = |\langle K_2 \rangle| \wedge P_i = P_j \wedge \langle K_3 \rangle = \langle K_4 \rangle$.

10.2.2 Kostenfunktion

Die Berechnung der Kosten, die sich aus einer Verletzung einer Nebenbedingungen/Restriktion ergeben, kann auf verschiedene Arten erfolgen. So wird in **[2.7]** und **[10.1]** eine Restriktionsverletzung linear bestraft, wobei in **[2.2]** und **[10.2]** die Restriktionsverletzungen quadratisch bestraft werden.

Die *N* Sonderausstattungen werden durch *A(i,j)* den *M* Produkten zugeordnet, wobei

$$A(i,j)=\begin{cases}1 & \textit{falls Produkt i Sonderausstattung j benötigt}\\ 0 & \textit{falls } i=0\\ 0 & \textit{sonst}\end{cases}$$

Mit $A^G(i,j)$ wird die Anzahl der gemeinsamen Sonderausstattungen bezeichnet und mit $N(\langle K\rangle, j)$ die Anzahl der Produkte mit der Sonderausstattung *j* in der Konfiguration $\langle K\rangle$:

$$A^G(i,j)=\sum_{k=1}^{N}\left(A(i,k)+A(j,k)-1\right)^2, \qquad \textbf{(10.1)}$$

$$N(\langle K\rangle, j)=\sum_{i=1}^{|\langle K\rangle|}A(K(i),j). \qquad \textbf{(10.2)}$$

Die Funktion *block(i,j,k)* liefert die Länge eines Blocks aus in der Konfiguration benachbarten Produkten, der bei *i* startet und dessen Produkte die Sonderausstattung *j* besitzen (k=1) oder nicht (k=0).

```
ret=0
while( A(K(i),j) = k & i <= M) {
    ++i;
    ++ret;
}
```

Der Wert für *block(i,j,k)* ist in *ret* gespeichert.

Im Folgenden sind für die einzelnen Nebenbedingungen die Kostenfunktionen gegeben. Die Gewichte G^X_i der Nebenbedingungen sind in die Kostenfunktionen integriert. Die lineare und quadratische Version unterscheiden sich hinsichtlich der Potenz α.

- **Abstand**
  Zwischen zwei Produkten mit einer Sonderausstattung *i* sollen mindestens A^{min}_i Produkte und maximal A^{max}_i ohne diese Sonderausstattung in der Konfiguration geplant sein. Die zugehörigen Kostenfunktionen E_{Amin} und E_{Amax} sind durch **(10.3)** und **(10.4)** gegeben.

$$E_1=E_{A\min}=\sum_{i=1}^{N}\left\{G^{A\min_i}\bullet\left(\sum_{j=1}^{M-1}A(K(j),i)\bullet MIN(1,M-j+block(j+1,i,0))\bullet \left[MAX(0,A^{\min_i}-block(j+1,i,0))\right]^{\alpha}\right)\right\} \qquad \textbf{(10.3)}$$

$$E_2=E_{A\max}=\sum_{i=1}^{N}\left\{G^{A\max_i}\bullet\left(\sum_{j=1}^{M-1}A(K(j),i)\bullet MIN(1,M-j+block(j+1,i,0))\bullet \left[MAX(0,block(j+1,i,0)-A^{\max_i})\right]^{\alpha}\right)\right\} \qquad \textbf{(10.4)}$$

- **Block**
  Es sollen mindestens $B^{min}{}_i$ und maximal $B^{max}{}_i$ Produkte mit einer Sonderausstattung in der Konfiguration benachbart geplant sein. Die zugehörigen Kostenfunktionen E_{Bmin} und E_{Bmax} sind durch **(10.5)** und **(10.6)** gegeben.

$$E_3 = E_{B\min} = \sum_{i=1}^{N}\left\{ G^{B\min}{}_i \bullet \left(\sum_{j=1}^{M} (1 - A(K(j-1),i)) \bullet A(K(j),i) \bullet \left[MAX\left(0, B^{\min}{}_i - block(j,i,1)\right)\right]^{\alpha} \right)\right\} \tag{10.5}$$

$$E_4 = E_{B\max} = \sum_{i=1}^{N}\left\{ G^{B\max}{}_i \bullet \left(\sum_{j=1}^{M} (1 - A(K(j-1),i)) \bullet A(K(j),i) \bullet \left[MAX\left(0, block(j,i,1) - B^{\max}{}_i\right)\right]^{\alpha} \right)\right\} \tag{10.6}$$

- **Menge**
  In einem gegebenen Intervall $I_{i,k}$ sollen minimal $M^{min}{}_{i,k}$ und maximal $M^{max}{}_{i,k}$ Produkte mit einer Sonderausstattung i geplant werden. Die zugehörigen Kostenfunktionen E_{Mmin} und E_{Mmax} sind durch **(10.7)** und **(10.8)** gegeben. $I^S{}_{i,k}$ ist dabei der Start und $I^E{}_{i,k}$ das Ende des Intervalls.

$$E_5 = E_{M\min} = \sum_{i=1}^{N}\sum_{k}\left\{ G^{M\min}{}_i \bullet \left[MAX\left(0, M^{\min}{}_{i,k} - \sum_{j=I^S{}_{i,k}}^{I^E{}_{i,k}} A(K(j),i)\right)\right]^{\alpha}\right\} \tag{10.7}$$

$$E_6 = E_{M\max} = \sum_{i=1}^{N}\sum_{k}\left\{ G^{M\max}{}_i \bullet \left[MAX\left(0, \sum_{j=I^S{}_{i,k}}^{I^E{}_{i,k}} A(K(j),i) - M^{\max}{}_{i,k}\right)\right]^{\alpha}\right\} \tag{10.8}$$

- **KausL**
  Diese Nebenbedingung stellt eine Erweiterung der Mengenrestriktion dar. Bei der **KausL**-Restriktion wird nun erwartet, dass in jedem Intervall der Größe L die Mengenrestriktion der Menge $K^{min}{}_i$, $K^{max}{}_i$ eingehalten wird. Die zugehörigen Kostenfunktionen E_{Kmin} und E_{Kmax} sind durch **(10.9)** und **(10.10)** gegeben.

$$E_7 = E_{K\min} = \sum_{i=1}^{N}\left\{ G^{K\min}{}_i \bullet \sum_{k=1}^{M-L+1}\left[MAX\left(0, K^{\min}{}_i - \sum_{j=0}^{L-1} A(K(k+j),i)\right)\right]^{\alpha}\right\} \tag{10.9}$$

$$E_8 = E_{K\max} = \sum_{i=1}^{N}\left\{ G^{K\max}{}_i \bullet \sum_{k=1}^{M-L+1}\left[MAX\left(0, \sum_{j=0}^{L-1} A(K(k+j),i) - K^{\max}{}_i\right)\right]^{\alpha}\right\} \tag{10.10}$$

- **Gleichverteilung**
  Bei der Bestimmung der Gleichverteilung können verschiedene Ansätze gewählt werden. So wurde in **[2.7]** die Gleichverteilung durch eine lineare Kostenfunktion **(10.12),** bestehend aus dem Skalarprodukt *D(i,j)* **(10.11)** der Sonderausstattungsvektoren der beiden Endprodukte *i* und *j*, verwendet.

$$D(i,j)=\sum_{k=1}^{N} A(i,k)\cdot A(j,k) \qquad \textbf{(10.11)}$$

$$E_9 = E_G = G^G \bullet \sum_{j=1}^{M-1} D(K(j),K(j+1)) \qquad \textbf{(10.12)}$$

  Eine zu den quadratischen Kostenfunktionen (α=2) passende Kostenfunktion für die Gleichverteilung wurde z.B. in **[10.2]** definiert:

$$E_{9'} = E_G = \sum_{i=1}^{N} G^{G_i} \bullet \sum_{j=1}^{M} \left(\frac{j}{M} - \frac{Num(i,j)}{Num(i,M)} \right)^{\alpha}, \qquad \textbf{(10.13)}$$

  wobei durch *Num(i,j)* die Anzahl der Produkte mit der Sonderausstattung *i* ist, die in den ersten *j* Positionen der Konfiguration geplant sind:

$$Num(i,j)=\sum_{k=1}^{j} A(K(k),i). \qquad \textbf{(10.14)}$$

Die gesamte Kostenfunktion ergibt sich dann als Summe der gewichteten Kostenfunktionen:

$$E=\sum_{i=1}^{9} E_i \,. \qquad \textbf{(10.15)}$$

Bisher sind die Restriktionen und damit die zugehörigen Kostenfunktionen auf den gesamten Planungsbereich der Konfiguration gültig. Dies ist jedoch nicht unbedingt erforderlich. So existieren womöglich Restriktionen, die z.B. nur in der ersten Produktionsschicht am Montag gelten. In **[10.2]** wurden die Kostenfunktionen bereits auf Teilintervalle der Konfiguration eingeschränkt.
Ist davon auszugehen, dass die Restriktionen der einzelnen Sonderausstattungen eingehalten werden können, so unterscheiden sich die optimalen Lösungen mit den Kosten 0 nicht voneinander. Liegt jedoch eine Problemstellung vor, bei der nicht davon auszugehen ist, dass alle Restriktionen erfüllt werden können, so führt eine quadratische Definition der Kostenfunktion dazu, dass statt weniger großer Restriktionsverletzungen viele kleine in der optimalen Lösung existieren. Bei einer linearen Kostenfunktion existiert bzgl. der Kosten kein Unterschied zwischen vielen kleinen Restriktionsverletzungen und wenigen großen. Welche der beiden Definitionen der Kostenfunktion verwendet wird, hängt also stark von den Bedürfnissen des Anwenders ab. In der Automobilindustrie existieren bei der Produktion meist viele Nebenbedingungen verschiedener Prioritäten, die nicht alle gleichzeitig erfüllt werden können. Eine kurzfristige Überlastung der Arbeiter an einer Position des Bandes, also eine kleine Restriktionsverletzung, kann durch evtl. vorhandene Puffer zwischen den Positionen ausgeglichen werden und ist deshalb erlaubt, wohingegen eine starke Überlastung produktionstechnisch keinesfalls möglich ist. Die quadratische Definition der Kostenfunktion ist deshalb die Voraussetzung bei der Suche nach einer möglichst guten Lösung.

10.2.3 Nachbarschaft

Die Nachbarschaft wird mit Hilfe von Operatoren (*Moves*) definiert, die die Konfiguration verändern. Im Folgenden sind $\langle K \rangle$ und $\langle K' \rangle$ benachbarte Konfigurationen.

- **Push1** (bzw. **Insert):**
  $\langle K \rangle = \langle K_1 \rangle \bullet P_i \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet P_i \bullet \langle K_3 \rangle$
  oder
  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet P_i \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet P_i \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle,$
  wobei
  $|\langle K_2 \rangle| > 0$.

- **Swap**
  $\langle K \rangle = \langle K_1 \rangle \bullet P_i \bullet \langle K_2 \rangle \bullet P_j \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet P_j \bullet \langle K_2 \rangle \bullet P_i \bullet \langle K_3 \rangle,$
  wobei
  $|\langle K_2 \rangle| \geq 0$.

- **Lin2Opt**
  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \overline{\langle K_2 \rangle} \bullet \langle K_3 \rangle,$
  wobei
  $|\langle K_2 \rangle| > 1$.

- **Transposition**
  $\langle K \rangle = \langle K_1 \rangle \bullet P_i \bullet P_j \bullet \langle K_2 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet P_j \bullet P_i \bullet \langle K_2 \rangle$.

- **Random**
  Ein Teilbereich der Konfiguration wird zufällig erstellt:
  $\langle K \rangle = \langle K_1 \rangle \bullet \langle K_2 \rangle \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet \langle K_2' \rangle \bullet \langle K_3 \rangle$.

- **SwapS**
  Es werden 2 Produkte getauscht, die bzgl. ihrer Nebenbedingungen ähnlich, jedoch nicht identisch sind. Die beiden Produkte *i* und *j* unterscheiden sich in mindestens einer Sonderausstattung und maximal in 2 ($1 \leq A^G(i, j) \leq 2$).
  $\langle K \rangle = \langle K_1 \rangle \bullet P_i \bullet \langle K_2 \rangle \bullet P_j \bullet \langle K_3 \rangle \Rightarrow \langle K' \rangle = \langle K_1 \rangle \bullet P_j \bullet \langle K_2 \rangle \bullet P_i \bullet \langle K_3 \rangle$.

Ähnlich zu der in **Kapitel 3.2.5** definierten Nachbarschaft wird hier von jedem Operator gefordert, die Konfiguration zu verändern. Die beiden Konfigurationen $\langle K \rangle$ und $\langle K' \rangle$ dürfen nicht identisch sein: $\langle K \rangle \neq \langle K' \rangle$.
Die Nachbarschaft besteht somit aus Standardoperatoren für Sequenzierungsprobleme und auf diese Problemstellung spezialisierte Operatoren. Die **Transposition** und der **SwapS** stellen Spezialisierungen des **Swap** dar, werden jedoch eigens definiert, da sie sehr effizient durchführbar sind. Bei der Berechnung der Kostenfunktion wird meist, bis auf die Bewertung der Initialsequenz, eine lokale Berechnung verwendet, wobei nur der Teil der Sequenz neu bewertet wird, der durch den Operator verändert wurde. Wird also nur ein sehr kleiner Bereich verändert, z.B. verändert die **Transposition** nur zwei benachbarte Positionen, oder führt die Anwendung eines Operators nur zu einer Veränderung der Sequenz bzgl. weniger Restriktionen, z.B. bei der Ausführung des **SwapS**, so ist die Bewertung der veränderten Sequenz deutlich schneller durchzuführen als bei den anderen Operatoren. Die Veränderung eines kleinen Bereichs oder nur bzgl. weniger Nebenbedingungen hat zudem den Vorteil, dass sich die Kosten benachbarter Konfigurationen nur wenig unterscheiden und somit selbst bei niedrigen Temperaturen eine ausreichende Wahrscheinlichkeit besteht, solche Veränderungen zu akzeptieren. Aufgrund der effizienten lokalen Bewertung einer Konfiguration werden des weiteren auch Nachbarschaften verwendet, die die bei den Operatoren veränderten Bereiche auf einen Bruchteil der gesamten Konfiguration einschränken, z.B. dürfen die veränderten Positionen bei einem **Swap** dann maximal 25% der Anzahl der Positionen der Konfiguration voneinander entfernt liegen. Dadurch wird die Nachbarschaft, vor allem bei **Random**, erheblich reduziert. Die benötigte Rechenzeit für die Ausführung der Operatoren ist in **Tabelle 10.6** für eine Probleminstanz exemplarisch gegeben.

10.3 Spinglasverhalten

Im Folgenden werden Ergebnisse aus **[2.7]** vorgestellt. Dort wurde eine Simulation zu einer Problemstellung durchgeführt, die auf einer linearen Definition der Kostenfunktion basiert und nur die Gleichverteilung, den maximalen Block und den minimalen Abstand als Restriktion aufwies. Die Simulationen wurden mit **SA** und **TA** durchgeführt, wobei die Temperatur jeweils von dem Anfangswert 12 in 100 Schritten logarithmisch (α=0.95) gesenkt wurde. Die Nachbarschaft bestand aus dem **Lin2Opt** und dem **Swap**.
Die Energie und Wärmekapazität sind für **SA** und **TA** in **Abbildung 10.1** zu sehen.

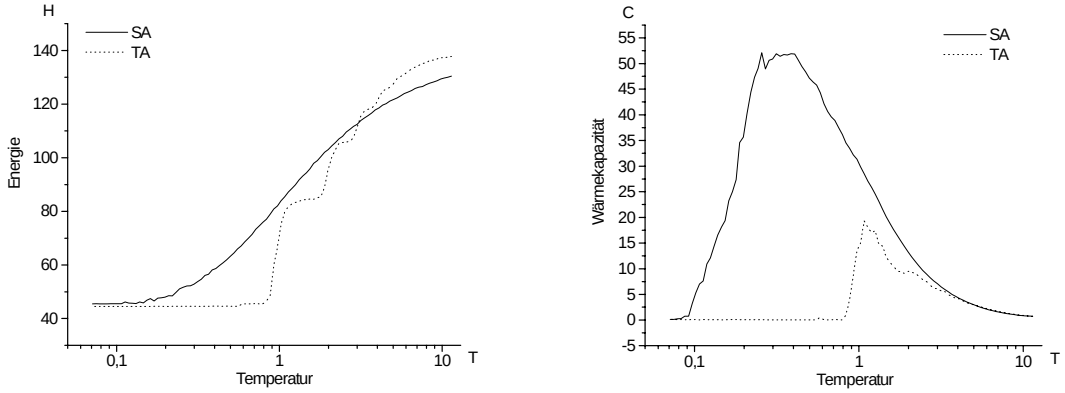


Abbildung 10.1: Energie und Wärmekapazität der Problemstellung mit linearer Definition der Kostenfunktion bei Simulationen mit **SA** und **TA**.

Bei der Energie zeigt sich ein deutlicher Unterschied zwischen den Verfahren. Während bei **SA** der Energieverlauf denen der Simulationen anderer Modelle (vgl. **Kapitel 4.2**) ähnelt, verläuft die Energie bei der Simulation mit **TA** in Stufenform. Der Grund dafür liegt in den diskreten Energie- bzw. Kostenwerten der einzelnen Konfigurationen. Beim **TA** werden z.B. beim Temperaturübergang von Temperaturen nahe 2 von $T>2$ nach $T<2$ alle *Moves* verboten, die einen Kostenunterschied >1 zur Folge haben, während beim **SA** nur die Wahrscheinlichkeit, diese zu akzeptieren, etwas gesenkt wird. Das Maximum der Wärmekapazität der Simulationen mit **TA** liegt deshalb bereits bei höheren Temperaturen als bei **SA**, in der Nähe der Temperatur $T=1$, unterhalb derer sich **TA** wie ein **GR** verhält und keine schlechteren Lösungen mehr akzeptiert. Die Wärmekapazität verschwindet bei beiden Verfahren bei tiefen Temperaturen. Das System friert hier bei einem Energiewert ein.
Ähnlich zur Gesamtenergie verlaufen auch die summierten Kosten der Nebenbedingungen (vgl. **Abbildung 10.2**).

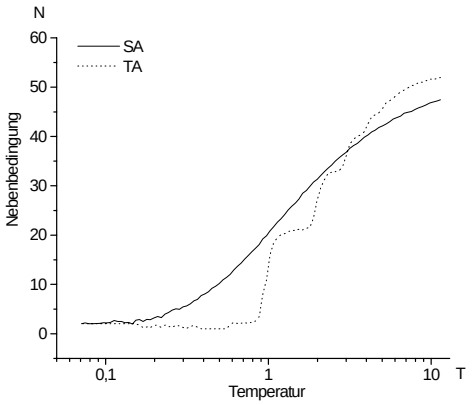


Abbildung 10.2: Es ist der temperaturabhängige Verlauf der summierten Kostenfunktionen der Nebenbedingungen zu sehen.

Ähnlich zum Ordnungsparameter aus **Kapitel 4.2** lässt sich auch hier ein Ordnungsparameter definieren. Dieser ergibt sich aus der Übereinstimmung (*overlap*) zweier Konfigurationen K_1 und K_2:

$$O = \frac{1}{M} \cdot \sum_{i=1}^{M} \delta\left(K_1(i), K_2(i)\right). \qquad \textbf{(10.16)}$$

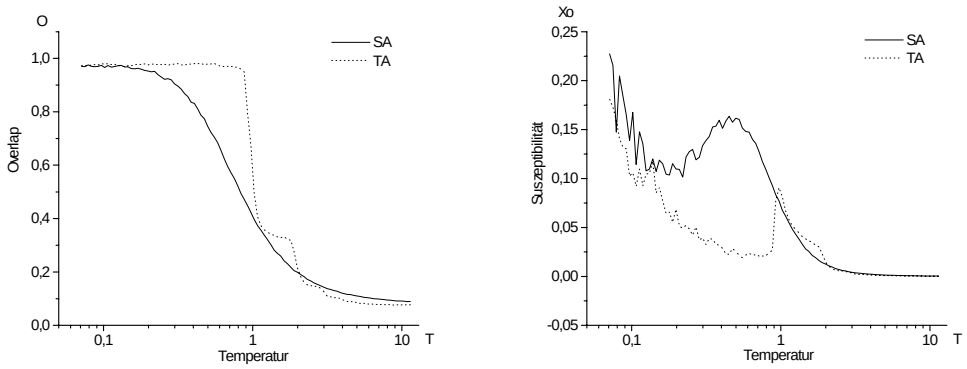


Abbildung 10.3: Ordnungsparameter (**Overlap**) und Suszeptibilität der Problemstellung mit linearer Definition der Kostenfunktion bei Simulationen mit **SA** und **TA**.

Auch hier (vgl. **Kapitel 4.2**) ist der Verlauf des Ordnungsparameters und der zugehörigen Suszeptibilität ein Beweis für die Entartung des Systems, wobei Konfigurationen gleicher Energie benachbart und durch sog. *triviale moves* miteinander verbunden sind. Das System friert bei kleinen Temperaturen zwar in einem Zustand bestimmter Energie ein, verfügt jedoch noch über die Freiheit verschiedene Konfigurationen abzusuchen. Dies ist an dem Ordnungsparameter <1 und der ansteigenden Suszeptibilität bei kleinen Temperaturen zu sehen.

10.4 Benchmarkinstanzen CSPLib

Die **CSPLib** (**[10.3]**) beinhaltet einen Satz Instanzen für die in diesem Kapitel vorgestellte Problemstellung. Bei allen Instanzen wird nur ein Restriktionstyp verwendet: **KausL**, wobei in jedem Intervall der Länge l_i maximal k_i Produkte mit der Sonderausstattung i geplant werden sollen. Aus den gegebenen Restriktionen und den Produkten lässt sich die Auslastung (*utilization rate*) *AL* der Konfiguration $\langle K \rangle$ definieren:

$$AL = \frac{1}{N} \cdot \sum_{i=1}^{N} AL(i), \qquad \textbf{(10.17)}$$

wobei *AL(i)* die Auslastung einer Sonderausstattung definiert:

$$AL(i) = \frac{N\left(\langle K \rangle, i\right)}{M} \bullet \frac{l_i}{k_i}. \qquad \textbf{(10.18)}$$

Instanzen mit einer Auslastung größer 100% sind nicht lösbar, die optimale Lösung wird mindestens eine Restriktion verletzen. Je kleiner die Auslastung, desto leichter sollte eine Instanz gelöst werden können.

Die gegebenen Instanzen können in 2 Gruppen unterteilt werden. Die erste Gruppe besteht aus 70 leichten Instanzen aus jeweils 200 Produkten und 5 Sonderausstattungen, die wiederum in 7 Gruppen zu je 10 Instanzen zusammengefasst sind. Diese leichten Instanzen haben eine Auslastung von 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9 und sind alle lösbar (**[10.4]**, **[10.5]**, **[10.6]**). Die Gruppe mit einer Auslastung vom z.B. 0.6 wird dabei als Gruppe **60-*** bezeichnet. Die zweite Gruppe besteht aus 9 schwierigen Instanzen aus jeweils 100 Produkten und 5 Sonderausstattungen. Bei der zweiten Gruppe sind die Auslastungen nahe 1 und nicht alle Instanzen sind lösbar. Teilweise wurden diese Instanzen in **[10.7]** verwendet.
Aufgrund der einfachen Lösbarkeit der ersten Gruppe der Instanzen wird eine Heuristik (**DED**) erstellt, auf der dann in einem zweiten Schritt ein einfaches Greedy-Verfahren aufbaut, um die erhaltene Lösung lokal zu verbessern. Die Heuristik wird mit ähnlichen Heuristiken verglichen und die Ergebnisse der Kombination mit einer lokalen Suche einem Ameisenalgorithmus gegenübergestellt, der die vorgestellten Heuristiken zum Aufbau einzelner Lösungen benutzt (vgl. **[2.2]**).

10.4.1 Heuristiken

Die Heuristiken erstellen eine Konfiguration $\langle K \rangle$ (bzw. eine Lösung) zu einer Instanz schrittweise durch Hinzufügen benötigter Produkte ans Ende der unvollständigen Konfiguration $\langle K' \rangle$. Bei jedem Schritt wird dabei ein Produkt P_i ausgewählt, das zu den wenigsten neuen Restriktionsverletzungen führt, also folgende Funktion minimiert:

$$f(\langle K' \rangle \bullet P_i) = \sum_{j=1}^{N} A(P_i, j) \bullet MAX\left[0, \left(\sum_{m=MAX(1, |\langle K' \rangle| - l_i)}^{|\langle K' \rangle|} A(K'(m), j) \right) + A(P_i, j) - k_i \right]. \qquad \textbf{(10.19)}$$

Im Normalfall existieren mehrere Produkte, die die Funktion **(10.19)** minimieren. Die verschiedenen in **[2.2]** vorgestellten Heuristiken unterscheiden sich in der Auswahl eines Produkts aus der Gruppe mit minimalem *f*. Dazu werden zusätzliche Bedingungen festgelegt.

- **Rand**: Das zu planende Produkt wird zufällig gewählt.
- **SHU** (Static Highest Utilization Rates): Es wird ein Produkt aus der Gruppe gewählt, das die Sonderausstattung mit der größten Auslastung besitzt (**[10.8]**). Existieren mehrere solcher Produkte, wird die Sonderausstattung mit der nächsthöheren Auslastung beachtet. Dies wird wiederholt, bis nur noch ein Produkt existiert. Es wird also dasjenige Produkt i gewählt, das die folgende Funktion maximiert:

  $$g = \sum_{j=1}^{N} A(i, j) \cdot 2^{k_j}, \qquad \textbf{(10.20)}$$

  wobei die Sonderausstattung *j* die k_j-te Auslastung besitzt.
- **DHU** (Dynamic Highest Utilization Rates): Während die Auslastung bei **SHU** anfangs festgelegt ist, wird sie nun während jedes Schritts neu angepasst an die bereits teilweise erstellte Konfiguration. Die Auslastung jeder Sonderausstattung i wird bestimmt durch:

  $$AL_{dyn}(i) = \frac{N(\langle K \rangle, i) - N(\langle K' \rangle, i)}{M - |\langle K' \rangle|} \bullet \frac{l_i}{k_i}. \qquad \textbf{(10.21)}$$

- **SSU** (Static Sum of Utilization Rates): Im Unterschied zu **SHU** wird nun statt einer wohldefinierten Ordnung der Auslastung der Sonderausstattung deren Summe verwendet. Es wird also dasjenige Produkt *i* gewählt, das die folgende Funktion maximiert:

$$g = \sum_{j=1}^{N} A(i,j) \cdot AL(j). \qquad \textbf{(10.22)}$$

- **DSU** (Dynamic Sum of Utilization Rates): Diese Heuristik verwendet die dynamisch angepasste Auslastung:

$$g = \sum_{j=1}^{N} A(i,j) \cdot AL_{dyn}(j). \qquad \textbf{(10.23)}$$

- **DED** (Dynamic Even Distribution): Die vier zuletzt definierten Heuristiken verwenden die Auslastung der Sonderausstattungen zur Bestimmung der Produkte. Dies führt bei mehrmaliger Anwendung der Heuristiken meist zu gleichen oder zumindest ähnlichen Konfigurationen. Bei dieser Heuristik soll nun eine gleichmäßige Verteilung der Produkte mit den Sonderausstattungen erfolgen. Ist die durchschnittliche Anzahl der Produkte mit einer bestimmten Sonderausstattung in der Teilkonfiguration kleiner als in der gesamten Konfiguration, so werden Produkte mit dieser Sonderausstattung bevorzugt geplant. Es wird das Produkt als nächstes geplant, das die folgende Funktion maximiert:

$$g = \sum_{j=1}^{N} \left[\left(A(i,j) = 0\right) XOR \left(\frac{N(\langle K \rangle, j)}{M} > \frac{N(\langle K' \rangle, j)}{|\langle K' \rangle|} \right) \right] \qquad \textbf{(10.24)}$$

  Das erste Produkt der Konfiguration wird aus denjenigen mit maximaler Anzahl an Sonderausstattungen zufällig ausgewählt.

10.4.2 Lokale Suche

Zur Lösung der Instanzen der **CSPLib** wird nur eine einfache lokale Suche (**GR**) verwendet, wobei nur Veränderungen akzeptiert werden, die keine Verschlechterungen erzeugen. Während der Suche werden die Restriktionsverletzungen quadratisch bestraft. Die beste Lösung entspricht dann allerdings bei dieser speziellen Aufgabenstellung mit meist lösbaren Instanzen der besten Lösung einer linearen Modellierung. Falls Restriktionen verletzt werden, so ist im Folgenden immer deren Anzahl und nicht die Kostenfunktion gegeben.

Als Initiallösung wird der lokalen Suche eine zufällige Konfiguration übergeben (**LSRand**) oder eine durch **DED** erzeugte (**LSDED**).

Die Nachbarschaft besteht bei der lokalen Suche aus den in **Kapitel 10.2.3** definierten Operatoren, wobei jeder mit gleicher Wahrscheinlichkeit ausgewählt wird. Die maximale Größe des veränderten Bereichs wird auf 25% der gesamten Konfiguration festgelegt.

10.4.3 Ant Colony Optimization (ACO)

Die Ergebnisse der einfachen lokalen Suche werden denen eines Ameisenalgorithmus von Solnon gegenübergestellt. Ameisenalgorithmen imitieren die Vorgehensweise einer Ameisengesellschaft auf der Suche nach Futter, um möglichst gute Lösungen für Optimierungsprobleme zu erstellen. Sie wurden bereits auf eine Vielzahl von Optimierungsproblemen eingesetzt **[10.9]**. Der in **[2.2]** verwendete Ansatz ist eine verbesserte Version des **ACO** aus **[10.6]**. Die Verbesserungen betreffen vor allem eine Elitist-Strategie, bei der nur Produkte gewählt werden, die die wenigsten neuen Restriktionsverletzungen zur Folge haben, zusätzliche Ideen aus **[10.10]**, um stärker *exploration* durchzuführen, und die bereits definierten Heuristiken zur Lösungssuche.
Eine Konfiguration wird hier ähnlich zur Vorgehensweise der Heuristiken erstellt. Es wird der Konfiguration schrittweise ein Produkt hinzugefügt, wobei bei jedem Schritt ein Produkt i aus der Menge derjenigen mit minimalen neuen Restriktionsverletzungen (vgl. **(10.17)**) mit einer bestimmten Wahrscheinlichkeit p_i gewählt wird:

$$p_i = \frac{{\tau_{j,i}}^{\alpha} \cdot g^{\beta}}{\sum_i {\tau_{j,i}}^{\alpha} \cdot g^{\beta}}. \qquad \textbf{(10.25)}$$

g ist eine beliebige der in **Kapitel 10.4.1** definierten Funktionen und τ die Pheromonmatrix abhängig vom zuletzt geplanten Produkt j. Über die beiden Parameter α und β steuert man den Einfluss der Pheromone gegenüber der Heuristikfunktion. Die Pheromonmatrix wird nach jeder erzeugten Generation an Lösungen verändert und zwar derart, dass die bestehenden Einträge reduziert werden und die besten gefundenen Lösungen die zugehörigen Einträge in der Pheromonmatrix vergrößern. Die Matrix hat dabei nur Einträge im Bereich $[\tau_{min},\tau_{max}]$. Eine Generation wird durch die Anzahl der Ameisen bestimmt, die nach einer Lösung suchen. Für die Simulationen ist die Anzahl der Ameisen stets auf 15 festgelegt.
Bei **ACO** werden zwei Varianten mit unterschiedlichen Parametern getestet. Bei der ersten Version (**Ant**) wählt man α=1, β=6, τ_{min}=0.001 und τ_{max}=4. Die zweite Version (**Iter**) wird ohne Pheromonmatrix gestartet, α=0, β=6. τ_{min}=τ_{max}=1, und entspricht somit einem wiederholten Anwenden der Heuristikfunktion. Kombiniert werden diese beiden Versionen mit den Heuristikfunktionen von **DHU**, **DSU** und **Rand**.

10.4.4 Ergebnisse

Die folgenden Tabellen und Abbildungen sind **[2.2]** entnommen.

Zuerst werden die Ergebnisse der einzelnen Heuristiken miteinander verglichen (vgl. **Tabelle 10.1**). Die Standardabweichungen der Ergebnisse der Heuristiken **SHU** und **DHU** sind 0. Es wird nur eine Lösung erzeugt. Bei diesen beiden Heuristiken existiert bei jedem Schritt jeweils nur ein Produkt, das die Funktion g maximiert. Bei **SSU** und **DSU** existieren bei jedem Schritt nur wenige Produkte, die gleiche Funktionswerte g aufweisen. Die Standardabweichung ist gering. Im Unterschied zu den bereits genannten Heuristiken existieren bei **DED** nur wenige Werte für g, d.h., es existieren bei jedem Schritt mehrere mögliche Produkte, die gewählt werden können. Die Standardabweichung ist deshalb sehr viel größer.

Betrachtet man die durchschnittliche Verletzung der Restriktionen, so sieht man, dass die auf einer dynamischen Auslastung definierten Verfahren **DHU** und **DSU** sehr ähnliche Resultate erzielen und meist besser sind als ihre statischen Versionen **SHU** und **SSU**. **DED** erzielt bessere durchschnittliche Ergebnisse als **DHU** und **DSU** für die Instanzen mit kleiner Auslastung (60-* bis 85-*), ist jedoch schlechter für 90-* und die Instanzen der zweiten Gruppe. Dies ist auch sehr deutlich an der Erfolgsrate (vgl. **Tabelle 10.2**) zu sehen.
Während die Instanzen der ersten Gruppe von zumindest einer Heuristik gelöst werden können, werden die optimalen Werte der Instanzen der zweiten Gruppe von keiner Heuristiken erreicht. Die erste Gruppe besteht somit aus leichter zu lösenden Instanzen als die zweite Gruppe. Dies wurde bereits bei der Definition der Auslastung erwartet.

Suite	No.	Rand	SSU	DSU	SHU	DHU	DED
1	60-*	20,4 (5,1)	10,2 (0,5)	1,6 (0,0)	10,5 (0,0)	1,4 (0,0)	0,8 (0,9)
	65-*	17,0 (5,0)	13,9 (0,3)	2,3 (0,0)	16,4 (0,0)	2,1 (0,0)	0,8 (1,0)
	70-*	13,8 (4,9)	17,4 (0,4)	3,0 (0,4)	18,6 (0,0)	3,6 (0,0)	0,7 (0,9)
	75-*	12,0 (5,0)	18,2 (0,2)	5,6 (0,4)	18,5 (0,0)	6,7 (0,0)	0,9 (1,0)
	80-*	17,8 (6,3)	16,6 (1,1)	4,3 (0,6)	17,2 (0,0)	5,0 (0,0)	1,0 (1,1)
	85-*	29,4 (7,3)	11,5 (0,6)	4,2 (0,4)	12,4 (0,0)	3,3 (0,0)	1,8 (1,2)
	90-*	54,2 (8,3)	5,9 (0,6)	1,2 (0,3)	6,1 (0,0)	1,6 (0,0)	4,3 (1,7)
2	10-93	59,1 (6,5)	11,0 (0,0)	10,4 (1,6)	11,0 (0,0)	12,0 (0,0)	14,1 (2,5)
	16-81	43,3 (5,1)	9,7 (1,2)	7,7 (0,9)	17,0 (0,0)	11,0 (0,0)	9,5 (2,7)
	19-71	56,0 (6,7)	13,6 (1,7)	8,5 (1,1)	13,0 (0,0)	7,0 (0,0)	10,3 (2,4)
	21-90	49,1 (5,6)	8,0 (0,0)	5,5 (0,7)	9,0 (0,0)	7,0 (0,0)	9,1 (1,9)
	26-82	40,6 (5,9)	6,0 (0,0)	3,6 (0,9)	7,0 (0,0)	3,0 (0,0)	8,8 (2,2)
	36-92	50,4 (6,2)	5,0 (0,0)	8,5 (0,5)	8,0 (0,0)	7,0 (0,0)	12,7 (2,3)
	41-66	34,1 (5,8)	4,7 (1,4)	3,7 (0,7)	6,0 (0,0)	2,0 (0,0)	6,5 (2,0)
	4-72	46,6 (5,8)	4,7 (0,6)	5,0 (0,0)	4,0 (0,0)	2,0 (0,0)	13,3 (3,2)
	6-76	28,1 (4,8)	6,0 (0,0)	6,0 (0,0)	6,0 (0,0)	6,0 (0,0)	9,1 (1,4)

Tabelle 10.1: Zu sehen sind die Ergebnisse der verschiedenen Heuristiken zu den verschiedenen Instanzen. Dabei sind die Mittelwerte aus jeweils 500 Simulationen und deren Standardabweichung gegeben.

No.	Rand	SSU	DSU	SHU	DHU	DED
60-*	0,002	0,002	0,400	0,000	0,500	0,547
65-*	0,002	0,000	0,100	0,100	0,200	0,536
70-*	0,004	0,000	0,000	0,000	0,000	0,580
75-*	0,005	0,000	0,200	0,000	0,100	0,507
80-*	0,001	0,000	0,116	0,000	0,100	0,487
85-*	0,000	0,200	0,218	0,100	0,200	0,251
90-*	0,000	0,300	0,500	0,300	0,500	0,114

Tabelle 10.2: Erfolgsraten der Heuristiken für die Instanzen der ersten Gruppe. Darunter wird der Prozentsatz der Lösungen (von 500) verstanden, die alle Restriktionen erfüllt.

Als nächstes werden die Ergebnisse der Verfahren **LS** und **ACO** miteinander verglichen. Die zur Verfügung gestellte Rechenzeit sollte für beide Verfahren in etwa gleich sein. Die Anzahl der erzeugten Lösungen unterscheidet sich dabei sehr stark. Bei der Instanz 10-93 benötigt **ACO** für die Erstellung von 15000 Lösungen, also 1000 Generationen, 20 Sekunden auf einem Rechner mit 300MHz, während **LS** in dieser Zeit 200000 Lösungen bewerten kann.

Bei der Bearbeitung der ersten Gruppe (vgl. **Tabelle 10.3**) wird die Anzahl der Generationen bei den **ACS** Varianten auf 100 und die Anzahl der Lösungen der **LS** Verfahren auf 20000 begrenzt. Der Einfluss der Pheromone macht sich hier nicht so stark bemerkbar, da die Anzahl der Generationen zu gering ist. Die Ergebnisse der **Iter** und der **Ant** Versionen unterscheiden sich deshalb nicht voneinander. Diese Gruppe besteht aus einfachen Instanzen, weshalb sie sehr schnell von allen Verfahren gelöst werden können, vorausgesetzt, man verwendet eine der Heuristiken (z.B. **DSU**, **DHU**, **DED**). Verwendet man keine Heuristik (**Rand**), kann **LS** bei allen Instanzen optimale Lösungen bei beinahe jeder Simulation finden. Die beiden **ACS** Verfahren hingegen sind ohne gute Heuristik vor allem bei den Instanzen mit hoher Auslastung **LS** deutlich unterlegen. Eine Erhöhung der Generationen auf 1000 verbesserte zwar die Ergebnisse (auf 23,9 für **IterRand** und 12,9 für **AntRand** bei 90-*) und zeigt, dass sich die Pheromone positiv auf die Suche auswirken, konnte jedoch nicht die Lösungsqualität von **LS** erreichen. Der Einfluss der Heuristik auf die Lösungssuche ist bei den **ACS** Verfahren deutlich größer als bei **LS**. Dies liegt daran, dass die Heuristik bei **LS** nur einmal verwendet wird, um die anfängliche Lösung zu erzeugen. Trotzdem hat die Heuristik auch bei **LS** einen merklichen Effekt. Ein Grund dafür ist das gute Abschneiden von **DED** bei den Instanzen. **Iter** ist hinsichtlich der Instanzen deutlich besser als die Heuristiken. Dies liegt daran, dass **Iter** aus einer Menge an generierten Lösungen die beste auswählt. Zusätzlich wird die Anzahl der möglichen Lösungen im Vergleich zu den Heuristiken erweitert, da die einzelnen Produkte bei der Suche mit einer bestimmten Wahrscheinlichkeit gewählt werden und nicht mehr deterministisch. Werden viele Lösungen generiert, kann dies von Vorteil sein.

No.	IterDSU	IterDHU	IterRand	AntDSU	AntDHU	AntRand	LSRand	LSDED
60-*	0,0 (0,0)	0,0 (0,0)	5,4 (1,0)	0,0 (0,0)	0,0 (0,0)	5,2 (0,9)	0,0 (0,1)	0,0 (0,0)
65-*	0,0 (0,0)	0,0 (0,0)	3,0 (0,7)	0,0 (0,0)	0,0 (0,0)	3,0 (0,7)	0,1 (0,1)	0,0 (0,0)
70-*	0,0 (0,0)	0,0 (0,0)	1,4 (0,5)	0,0 (0,0)	0,0 (0,0)	1,4 (0,6)	0,0 (0,1)	0,0 (0,0)
75-*	0,0 (0,0)	0,0 (0,0)	0,6 (0,4)	0,0 (0,0)	0,0 (0,0)	0,6 (0,4)	0,1 (0,2)	0,0 (0,0)
80-*	0,0 (0,0)	0,0 (0,0)	2,0 (0,9)	0,0 (0,0)	0,0 (0,0)	2,0 (0,8)	0,2 (0,3)	0,0 (0,0)
85-*	0,0 (0,0)	0,0 (0,0)	8,3 (1,7)	0,0 (0,0)	0,0 (0,0)	8,2 (1,6)	0,4 (0,6)	0,0 (0,1)
90-*	0,0 (0,0)	0,0 (0,0)	28,1 (2,5)	0,0 (0,0)	0,0 (0,0)	28,1 (2,4)	2,5 (1,2)	0,6 (0,6)

Tabelle 10.3: Ergebnisse der Verfahren **Iter**, **Ant** und **LS** für die Instanzen der ersten Gruppe. Zu sehen sind jeweils die durchschnittlichen Ergebnisse von jeweils 100 Simulationen und deren Standardabweichung.

Bei den Instanzen der zweiten Gruppe wird die Rechenzeit erhöht. Diese Instanzen sind zwar kleiner, jedoch schwieriger zu lösen. Bei den **ACS** Verfahren werden maximal 1000 Generationen erzeugt, bei **LS** maximal 200000 Lösungen.
An den Lösungen (vgl. **Tabelle 10.4**) ist der positive Effekt der Pheromone im **ACS** deutlich zu sehen, vor allem in Kombination mit **Rand**. Jedoch ist eine geeignete Heuristik für die Suche mit **Ant** essentiell. Zwischen den beiden Heuristiken **DSU** und **DHU** sind weder bei **Iter** noch bei **Ant** deutliche Unterschiede in der Lösungsqualität zu erkennen. Die gute Lösungsqualität der **Iter** Varianten **IterDSU**, **IterDHU** ist bemerkenswert. Zwar sind die Lösungen etwas schlechter als die **Ant** Varianten, jedoch vergleichbar mit **LS**. **LS** profitiert ebenfalls von einer guten Heuristik. Die Gründe wurden bereits bei den einfachen Instanzen erläutert. **LS** zeigt zwar gute Resultate, ist jedoch **AntDSU** und **AntDHU** unterlegen. Dies liegt an der einfachen Definition von **LS**, die es nicht ermöglicht, lokalen Optima zu entkommen.

No.	IterDSU	IterDHU	IterRand	AntDSU	AntDHU	AntRand	LSRand	LSDED
10-93	5,4 (0,7)	6,5 (0,7)	35,4 (1,8)	4,7 (0,5)	4,7 (0,5)	21,9 (1,7)	5,9 (1,1)	5,5 (0,9)
16-81	0,8 (0,5)	2,2 (0,5)	23,0 (1,0)	0,2 (0,4)	0,9 (0,4)	13,0 (1,2)	2,0 (0,8)	1,6 (0,8)
19-71	2,8 (0,5)	3,0 (0,5)	30,0 (1,7)	2,8 (0,4)	2,7 (0,4)	18,1 (1,8)	2,2 (0,4)	2,1 (0,3)
21-90	2,6 (0,0)	2,0 (0,0)	26,9 (1,6)	2,3 (0,5)	2,0 (0,1)	13,4 (1,5)	2,2 (0,4)	2,2 (0,4)
26-82	0,9 (0,5)	0,4 (0,5)	19,3 (1,3)	0,0 (0,0)	0,0 (0,0)	8,9 (1,1)	0,4 (0,5)	0,3 (0,5)
36-92	3,4 (0,3)	2,9 (0,3)	26,0 (1,8)	2,1 (0,3)	2,0 (0,0)	15,7 (1,2)	3,0 (0,5)	2,9 (0,6)
41-66	0,0 (0,0)	0,0 (0,0)	13,2 (1,5)	0,0 (0,0)	0,0 (0,0)	5,2 (0,9)	0,0 (0,2)	0,0 (0,1)
4-72	0,2 (0,5)	1,0 (0,5)	24,1 (1,6)	0,0 (0,0)	0,0 (0,0)	16,3 (1,4)	0,8 (0,6)	0,8 (0,6)
6-76	6,0 (0,0)	6,0 (0,0)	13,2 (0,8)	6,0 (0,0)	6,0 (0,0)	7,6 (0,6)	6,0 (0,0)	6,0 (0,0)

Tabelle 10.4: Ergebnisse der Verfahren **Iter**, **Ant** und **LS** für die Instanzen der zweiten Gruppe. Zu sehen sind jeweils die durchschnittlichen Ergebnisse von jeweils 100 Simulationen und deren Standardabweichung.

Es soll noch gezeigt werden, wie sich zusätzliche Rechenzeit auf die einzelnen Verfahren auswirkt. Die maximale Anzahl der Generationen wird bei **IterDHU** und **AntDHU** auf jeweils 100, 500, 1000, 2500, 10000 und die maximale Anzahl der Lösungen bei **LSDED** auf 20000, 100000, 200000, 500000 und 2000000 festgelegt. Die Verbesserung der Lösungsqualität bei steigender Rechenzeit ist für die einzelnen Verfahren exemplarisch für einige Instanzen in **Abbildung 10.4** zu sehen.

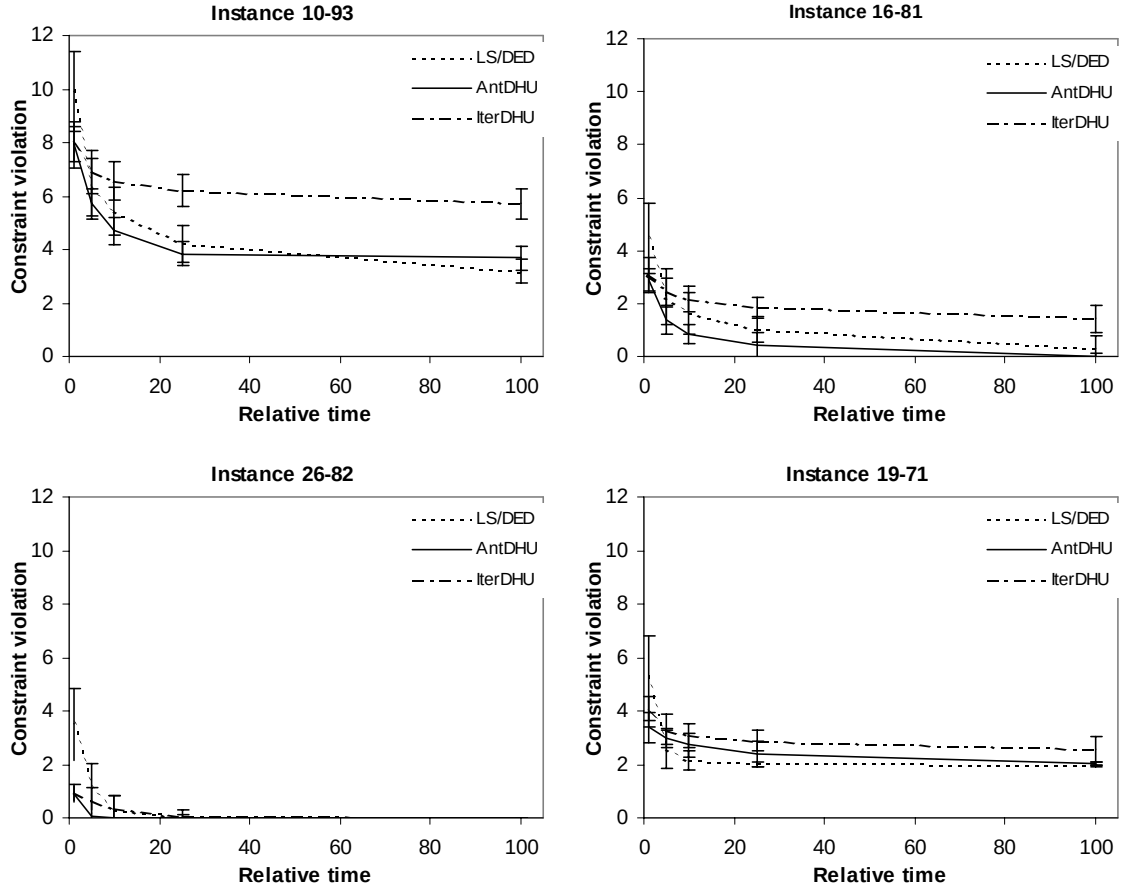


Abbildung 10.4: Ergebnisse der Verfahren **LSDED**, **IterDHU** und **AntDHU** für vier Instanzen der zweiten Gruppe.

Bei kleinen Rechenzeiten ist **LSDED** den anderen Verfahren (**IterDHU**, **AntDHU**) unterlegen. Bei großen Rechenzeiten ergeben die Verfahren **LSDED** und **AntDHU** vergleichbare Ergebnisse. **IterDHU** ist im Vergleich zu **LSDED** und **AntDHU** bei großen Rechenzeiten nicht sehr erfolgreich. Insgesamt ist festzustellen, dass einfache Heuristiken ausreichen, um die Instanzen der ersten Gruppe der Instanzen zu lösen. Bei den etwas schwierigeren Instanzen der zweiten Gruppe reicht der Einsatz der Heuristiken nicht mehr aus, um die optimalen Werte zu erhalten. In **Tabelle 10.5** ist ein Vergleich zu anderen aus der Literatur gegebenen Ergebnissen gegeben. Mit den Versionen des **ACO** und **LS** konnten für die Instanzen, bei denen eine gültige Lösung existiert, diese auch gefunden werden. Zusätzlich wurde bei einer Instanz eine gültige Lösung gefunden, bei der keine bekannt war und die in **[10.11]** bewiesene untere Schranke 2 für die Instanz 19-71 konnte erreicht werden.

Instance	10-93	16-81	19-71	21-90	26-82	36-92	41-66	4-72	6-76
Satisfiable?	no [10.7]	yes [10.3]	no [10.7], [10.11]	?	?	?	yes [10.3]	yes [10.7]	no [10.7]
Best solution	3,00	0,00	2,00	2,00	0,00	2,00	0,00	0,00	6,00
SR for IterDHU	0,00	0,00	0,43	1,00	1,00	0,70	1,00	0,65	1,00
SR for AntDHU	0,32	0,99	0,99	1,00	1,00	1,00	1,00	1,00	1,00
SR for LSDED	0,85	0,68	1,00	1,00	1,00	1,00	1,00	1,00	1,00

Tabelle 10.5: Vergleich der Ergebnisse mit denen aus **[10.3]**, **[10.7]**, **[10.11]**. **Best solution** gibt die besten Ergebnisse der in diesem Kapitel vorgestellten Verfahren an. Die Erfolgsrate (**SR**) gibt für jedes Verfahren den Prozentsatz der Lösungen an, die den besten Zielfunktionswert erreicht haben.

10.5 Realistische Problemstellung

Neben den Instanzen der **CSPLib** werden nun noch realistische Problemstellungen bearbeitet. Dabei handelt es sich um Daten eines deutschen Automobilproduzenten, mit einem Auftragsvolumen von jeweils 1000 Fahrzeugen und 25-30 Sonderausstattungen. Es werden die Restriktionstypen **Abstand**, **Block**, **Menge** und **KausL** benötigt. Zusätzlich sollen etwa 80% der Sonderausstattungen eine Gleichverteilung erfüllen. Jede Restriktion besitzt eine bestimmte Priorität. So sind die Gleichverteilungsrestriktionen immer von geringerer Priorität im Vergleich zu den anderen Restriktionen zur gleichen Sonderausstattung. So sollte z.B. eine Sonderausstattung mit einer Blockrestriktion zusammen mit einer Gleichverteilungsrestriktion zu gleichmäßig verteilten Blöcken führen. Die Instanzen sind deutlich *overconstrained*, d.h., es wird nicht erwartet, dass alle Restriktionen erfüllt werden. Bei einer Instanz zeigte sich, dass es z.B. zwei Sonderausstattungen gibt, wobei für die erste eine Blockrestriktion mit einem Minimalwert 5 und für die andere eine Abstandsrestriktion mit einem Mimimalwert 3 vorgegeben ist. Diese Restriktionen sind nicht gleichzeitig erfüllbar, da 50% der Fahrzeuge mit Sonderausstattung 1 auch Sonderausstattung 2 besitzen. Erfüllbar sind beide Restriktionen nur dann gleichzeitig, falls maximal 40% der Fahrzeuge mit Sonderausstattung 1 auch Sonderausstattung 2 besitzen. Ziel ist es, möglichst kleine Restriktionsverletzungen zu erzeugen, weshalb die quadratische Definition der Kostenfunktion verwendet wird.
Zur Lösung dieser Instanzen werden verschiedene lokale Suchverfahren verwendet: *Greedy*, *Threshold Accepting*, *Temperature Bouncing*. Die Nachbarschaft besteht bei allen Verfahren aus den in

Kapitel 10.2.3 beschriebenen Operatoren, wobei jeder mit gleicher Wahrscheinlichkeit ausgewählt wird. Die Operatoren verändern bei jeder Ausführung maximal 5% der Konfiguration, also nicht mehr als 50 zusammenhängende Positionen. Die für die Ausführung eines Operators für diese Problemstellung benötigte Rechenzeit ist in **Tabelle 10.6** gegeben. Die anfängliche Lösung für die Verfahren besteht aus einer zufälligen Konfiguration. Die benötigten Einstellungen des Temperaturschemas des **TA** wird anfangs festgelegt, wobei 0,5% der Rechenzeit aufgewendet wird um Nachbarschaftsoperatoren auszuführen und festzustellen, welche Energieänderungen auftreten. Die Starttemperatur T_S des **TA** wird dann so festgesetzt, dass 75% der durchgeführten Moves akzeptiert worden wären. Die Endtemperatur T_E entspricht der kleinsten Energieveränderung der Moves. Als Abkühlfaktor wird α=0,98 gewählt. Das Temperaturschema des **TA** wird für **TBouncing** übernommen, wobei 5 *bouncing* Schritte durchgeführt werden und die zugehörige Starttemperatur T_B durch $T_B=T_S*\alpha^{0,4*N}$ bestimmt wird, wobei N die Anzahl der Temperaturschritte des **TA** ist. Die Rechenzeit ist bei den Verfahren auf 10, 30, 60, 120, 300 und 600 Sekunden auf einem Pentium mit 600Mhz festgelegt. Im Kundeneinsatz ist meist die Vorgabe, die Planung in 5 Minuten ausführen zu können, zu berücksichtigen.
Die Ergebnisse der Simulationen sind beispielhaft anhand zweier Instanzen dargestellt. In **Abbildung 10.5** sind die durchschnittlichen Ergebnisse und deren Standardabweichung von jeweils 30 Simulationen pro Verfahren und Rechenzeitlimit zu sehen.
Die folgenden Tabellen und Abbildungen sind **[10.2]** entnommen.

Operator	Insert	Swap	Transposition	SwapS	Lin2Opt	Random
Moves / 100 s	128542	127121	514532	1971348	127652	126592

Tabelle 10.6: Anzahl der Moves die für die verschiedenen Nachbarschaftsoperatoren in 100 Sekunden auf einem Pentium mit 600MHz ausgeführt werden können.

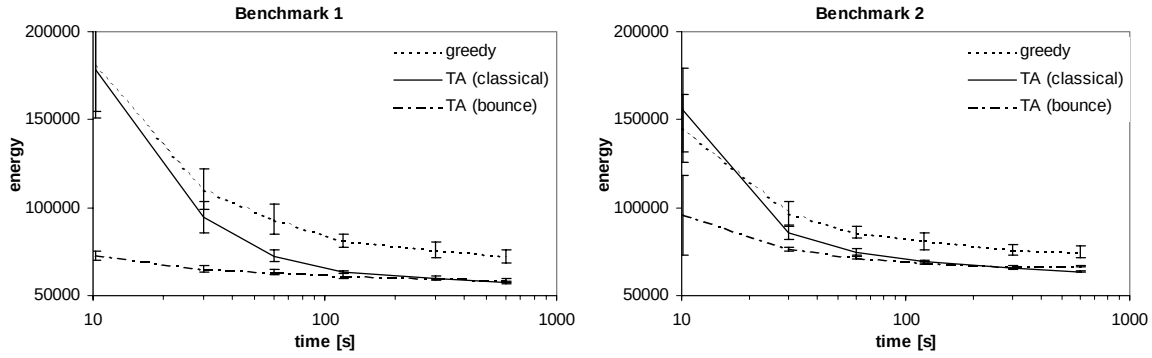


Abbildung 10.5: Durchschnittliche Ergebnisse für die verschiedenen Verfahren für verschiedene Rechenzeitgrenzen. Die Rechenzeit bezieht sich dabei auf einen Pentium mit 600MHz.

Eigentlich erwartet man, dass das **Greedy**-Verfahren schnell in einem strikten lokalen Optimum einfriert, da es keine Veränderungen akzeptiert, die das Ergebnis verschlechtern. Aufgrund der großen Nachbarschaft sind strikte lokale Optima jedoch selten, so dass selbst nach relativ langer Rechenzeit noch bessere Lösungen gefunden werden. Die Lösungsqualität ist dabei jedoch stark von der anfänglichen Lösung abhängig, wodurch sich eine große Standardabweichung ergibt.

TA ist dem **Greedy** überlegen, investiert man ausreichend Rechenzeit (>10 Sekunden). Nicht nur die durchschnittlichen Werte sind besser, sondern auch die schlechtesten Lösungen des **TA** sind besser als die besten des **Greedy**. Die Lösungssuche ist dabei sehr viel stabiler, die Standardabweichung ist immer geringer als beim **Greedy**. Die Gründe liegen in der positiven Eigenschaft des **TA** zwischenzeitlich auch Verschlechterungen zu akzeptieren.
Bei Rechenzeiten kleiner 120 Sekunden ist die **TBouncing**-Strategie den beiden anderen Verfahren deutlich überlegen. Neben den durchschnittlichen Werten sind auch deren Standardabweichungen deutlich geringer. **TBouncing** hat die Möglichkeit, zwischenzeitlich größere Verschlechterungen zu akzeptieren und kann so sehr viel leichter als **TA** aus lokalen Optima entkommen. Bei größeren Rechenzeiten ist die Lösungsqualität der beiden Verfahren jedoch ähnlich.

10.6 Zusammenfassung

In diesem Kapitel wurde ein anderes Produktionsplanungsproblem, das Fließbandproblem, vorgestellt, bei dem jedes Produkt auf einem Fließband gefertigt wird. Die Aufgabe bei dieser Problemstellung besteht darin, eine Sequenz zu erstellen, wobei die Produktionssequenz bestimmte Restriktionen erfüllen muss.
Zur Lösung der Problemstellung musste eine neues Modell entwickelt werden. Dieses basiert auf einer Konfiguration und einer Kostenfunktion, wobei diese linear oder quadratisch definiert werden kann. Bei realistischen Problemstellungen aus der Automobilindustrie, bei denen nicht davon auszugehen ist, dass alle Restriktionen eingehalten werden können, ist es meist besser, viele kleine Restriktionsverletzungen zu erzeugen als wenige, aber dafür große. Die quadratische Definition der Kostenfunktion wird deshalb dort zur Planung verwendet.
In diesem Kapitel konnte gezeigt werden, dass auch das hier verwendete Modell mit einer linearen Kostenfunktion Spinglasverhalten aufweist und den Einsatz lokaler Suchverfahren ähnlich dem *Simulated Annealing* begründet. Der definierte Ordnungsparameter weist auf eine hohe Entartung hin, da es in der erstellten Nachbarschaft auch am Ende der Simulation bei eingefrorener Energie zu Veränderungen der Konfiguration kommt. Es ist deshalb bei dieser Problemstellung essentiell, bei Greedy-Verfahren Veränderungen der Konfiguration zu erlauben, die die Energie nicht verändern, um so über Plateaus in der Energielandschaft zu wandern.
Als nächstes wurde der Einsatz einfacher Heuristiken zum Erstellen der Lösung für einfache Instanzen der **CSPLib** getestet. Diese einfachen Heuristiken sind jedoch nicht in der Lage, die schwierigeren Instanzen der Bibliothek zu lösen. Zusätzlich zur Heuristik wurde deshalb ein darauf aufbauendes lokales Sucheverfahren definiert. Zur Lösung war jedoch ein einfaches Greedy-Verfahren ausreichend. Die Ansätze zur Lösung der Instanzen sind sehr einfach, und das ist in diesem Fall ihr Vorteil. Es werden für diese Problemstellung viele verschiedene Verfahren vorgestellt, die oft ein Vielfaches an Implementationsaufwand und Parametertuning benötigen, z.B. eine *Large Neighbourhood Search* (**LNS**) **[10.12]**.
Im nächsten Abschnitt wurde dann eine realistische Problemstellung eines deutschen Automobilproduzenten vorgestellt. Im Unterschied zu den Instanzen der **CSPLib** existieren bei diesen Instan-

zen sehr viel mehr Produkte und Sonderausstattungen. Die Restriktionen sind dabei in der Regel nicht alle gleichzeitig erfüllbar, weshalb Prioritäten definiert werden, um wichtige Restriktionen stärker zu beachten. Der einfache **Greedy** ist bei diesen Instanzen den anderen lokalen Suchverfahren **TA** und **TA** mit einer ***bouncing*** Strategie deutlich unterlegen. Steht nur wenig Rechenzeit zur Verfügung, so ist die ***bouncing*** Strategie dem einfachen **TA** vorzuziehen. Jedoch benötigt diese die Einstellung zusätzlicher Parameter.
Das in **[2.7]** vorgestellte Verfahren zur Lösung der Fließbandprobleme wurde in einem Wettbewerb namens „Erstellung von Montage-Reihenfolgen“ eines deutschen Automobilproduzenten eingesetzt. Die Auswertung der dort abgegebenen Ergebnisse wurde folgendermaßen bewertet: „Als besonders bemerkenswert ist die Tatsache zu werten, dass Herr Puchta als Einzelperson in der Lage war, in kurzer Zeit qualitativ gleichwertige Ergebnisse zu liefern wie namhafte Anbieter von kommerzieller Optimierungssoftware“.
In den letzten Jahren wurde aufgrund der gesammelten Erfahrungen in diesem Bereich ein genetischer Algorithmus als Teil einer Softwarelösung (**SAP APO SEQ**) zur Sequenzplanung in der diskreten Industrie erstellt. Diese Software ist nun bereits bei mehreren Kunden bei der täglichen Planung produktiv im Einsatz. Die Funktionalität der Softwarelösung geht dabei weit über die in diesem Kapitel vorgestellte Modellierung hinaus. So sind z.B. die einzelnen Restriktionen auf bestimmte Bereiche der Konfiguration definierbar. Die umfangreichste Erweiterung der Funktionalität bezieht sich auf die Planung alternativer Fließbänder, wobei nicht jedes Produkt auf jeder Produktionslinie gefertigt werden kann und evtl. Kosten bei der Bearbeitung eines Produkts auf einer speziellen Linie anfallen.

Zusammenfassung

Zur Anwendung kamen in dieser Arbeit ausschließlich lokale Suchverfahren. Neben dem in der Festkörperphysik, zum Auffinden des Grundzustandes in Spinglasmodellen, erfolgreich eingesetzten Simulated Annealing (**SA**) wurden auch Threshold Accepting (**TA**), Temperature Bouncing (**TBouncing**) und eine Version des **SA** basierend auf einer adaptiven Steuerung der Parameter verwendet.

In dieser Arbeit wurden zwei grundsätzlich verschiedene Modelle zur Produktionsplanung vorgestellt (**Kapitel 3**). Das erste, ein Spinmodell, basiert dabei auf Spins, die nur die Werte 0,1 annehmen können und auf einem dreidimensionalen Gitter verteilt sind. Die Nebenbedingungen sind in diesem Modell alle weich abgebildet, d.h. Konfigurationen, die Nebenbedingungen verletzen, sind während der Optimierung erlaubt. Die Wechselwirkung der Spins wird durch zusätzliche Terme in der Energie- bzw. Kostenfunktion, die Straffunktionen, festgelegt. Diese sorgen dafür, dass die Nebenbedingungen am Ende der Optimierung eingehalten werden. Das zweite Modell besteht aus einer sehr viel direkteren Repräsentation der Problemstellung. Dort werden die einzelnen Produktionsvorgänge direkt als Modi abgebildet. Die meisten Nebenbedingungen des zweiten Modells werden von jeder Konfiguration eingehalten. Dafür sorgt ein definierter Reparaturmechanismus. Im Vergleich der beiden Modelle zeigte sich sehr schnell, dass die Darstellung der Konfiguration, der Nachbarschaft und der Nebenbedingungen im ersten Modell sehr viel einfacher ist, wodurch sich leicht neue Nebenbedingungen hinzufügen lassen, die erzielten Ergebnisse des zweiten Modells jedoch sehr viel besser sind. Ein Ergebnis wird dabei als besser angesehen, falls die Kostenfunktion kleinere Werte aufweist, oder eine Lösung mit gleichen Kosten in kürzerer Zeit gefunden wurde.

Im Vergleich zu den Observablen zweier Modelle zur Simulation eines Spinglases (**Kapitel 4**) zeigt sich, dass nicht nur das Spinmodell der Produktionsplanung Spinglasverhalten aufweist, sondern auch das Modell mit direkter Repräsentation. So existiert bei beiden Modellen der Produktionsplanung Unordnung und Frustration als Folge von konkurrierenden Wechselwirkungen (Konkurrenz). Während des Temperatursenkens beim **SA** findet ein Übergang von einem ungeordneten in einen geordneten Zustand statt, bis das System schließlich in einem lokalen Optimum einfriert. Mit Hilfe des definierten Ordnungsparameters für Produktionsplanungsprobleme lässt sich leicht die Entartung der Zustände nachweisen, da dieser selbst bei eingefrorener Energie (Kosten) bei kleinen Temperaturen keine verschwindende Suszeptibilität aufweist.

Auf die Darstellung der Simulationsergebnisse von **TA** wurde im überwiegenden Teil dieser Arbeit verzichtet, da sie keinen großen Unterschied zu denen der Simulation mit **SA** zeigten (vgl. **Kapitel 5**). Eine solch kleine Veränderung im Algorithmus führte bei den gegebenen Aufgabenstellungen

nicht zu einer Beschleunigung der Simulation, da die Berechnung der Exponentialfunktion im Vergleich zu der Erstellung des Produktionsplans aus der Konfiguration immer zu vernachlässigen war.

Anhand der verschiedenen Problemstellungen (**Kapitel 5** bis **Kapitel 9**) wurden verschiedene Modellierungen getestet.
Verschiedene Nachbarschaften wurden am Benchmark Bench01 (**Kapitel 8**) verglichen. Des weiteren wurde an diesem Beispiel untersucht, wie sich zusätzliche Aktivitätslinks auf die Ergebnisse auswirken. Die Ergebnisse des so erzeugten fixierten Peggings waren nur für lange Rechenzeiten ebenso erfolgreich wie das zuvor modellierte variable Pegging. Zusätzliche Links zwischen Aktivitäten verschiedener Aufträge, die gleiche Produkte herstellen, konnten die entstandenen Nachteile bei kurzen Rechenzeiten teilweise ausgleichen. Beim variablen Pegging wurde intuitiv die Vorgehensweise durchgeführt, die durch die zusätzlichen Links zwischen Aktivitäten verschiedener Aufträge erzwungen wurde: Aufträge mit kleinem Lieferzeitpunkt werden zuerst mit Produkten versorgt. Anhand der Problemstellung wurde des weiteren noch untersucht, wie sich zusätzliche Nebenbedingungen in ein bereits bestehendes Modell mit nur geringem Aufwand integrieren lassen, um ‚Was wäre wenn'-Fragen simulativ zu beantworten, ohne kostspielige Veränderungen an der realen Prozessstruktur, z.B. den vorhandenen Maschinen, durchzuführen. Es zeigte sich, dass das verwendete Verfahren in Kombination mit dem Modell sehr robust auf solche Veränderungen reagiert und sich weder in der Lösungsqualität noch in der benötigten Rechenzeit unangenehm verschlechtert. Dies ist unter anderem ein wichtiges Kriterium für den Einsatz als Planungssoftware bei real existierenden Problemstellungen.
Bei einer Problemstellung, die neben dem reinen Scheduling noch eine Losgrößenplanung erforderte (**Kapitel 9**), konnte gezeigt werden, dass eine globale Sicht auf das Optimierungsproblem erfolgreicher ist als eine Trennung der beiden Aufgabenstellungen. Mit dem vorhandenen Modell konnte diese Problemstellung auf mehrere verschiedene Arten modelliert werden. So zeigte sich, dass ein Modell mit Losgrößen, die anfangs auf einen bestimmten Wert fixiert wurden, zwar in manchen Instanzen die gleichen besten Durchlaufzeiten erzielen konnten, insgesamt jedoch deutlich schlechtere beste Ergebnisse lieferten als eine Modellierung, bei der die Losgrößen während des Scheduling festgelegt wurden. Ein Nachteil der vorgestellten Vorgehensweise zur Wahl der Losgröße, bei der zuerst ein Zeitpunkt gesucht wird, bei der ein Modus mit minimaler Losgröße geplant wird und dann die endgültige Losgröße deterministisch festgelegt wird, wurde anhand zweier Strategien untersucht. Die beiden Strategien unterschieden sich in der minimalen Losgröße einer Produktionsstufe. So war es in einigen Fällen durchaus erfolgreicher, den frühesten Startzeitpunkt zu verzögern, um dann mit erhöhter Losgröße zu produzieren.

Anhand der vorgestellten Ergebnisse zu den einzelnen Problemstellungen (**Kapitel 5** bis **Kapitel 9**) wurde die verwendete Kombination aus Optimierungsverfahren und Modell mit verschiedenen Verfahren aus der Literatur verglichen. Nicht bei jeder Problemstellung, die bearbeitet werden kann, erweist sich das hier vorgestellte Verfahren als überlegen im Hinblick auf die Lösungsqualität und die benötigte Rechenzeit. Dies trifft vor allem auf Problemstellungen zu, die nur wenig der durch das Modell zur Verfügung gestellten Funktionalität benötigen (**Kapitel 5**). Es zeigte sich jedoch, dass bei Problemstellungen, die eine komplexe Produktionsstruktur aufweisen (**Kapitel 6** bis **Kapi-**

tel 9), die meisten der in der Literatur vorhandenen besten Ergebnisse verbessert werden konnten, wobei meist bereits bei kleinen Rechenzeiten die erwartete Lösungsqualität (Mittelwert aus mehreren Simulationen) sehr nahe an der besten bisher bekannten Lösungen liegt oder sogar noch besser ist.

In **Kapitel 8** wurden außerdem verschiedene Möglichkeiten diskutiert, die für eine Simulation benötigten Parametereinstellungen zu erleichtern, um es auch einem in Optimierungsfragen nicht speziell geschulten Produktionsplaner zu ermöglichen, die Software einzusetzen, nachdem ein Optimierungsexperte das Modell erstellt hat. Zu diesen Vereinfachungen zählte eine adaptive Nachbarschaft, die sich während der Optimierung selbstständig an die gegebene Problemstellung anpasst. Dabei konnten ähnlich gute Ergebnisse wie bei den fest eingestellten Nachbarschaften erzielt werden. Als wichtigste Vereinfachung ist aber die Nutzung der Akzeptanzrate als Steuerung der Temperatur beim **SA** zu sehen. Der zeitliche Verlauf der Akzeptanzrate während der Simulation ist unabhängig von der gewählten Problemstellung festzulegen und erspart es dem Anwender, das Temperaturschema des **SA** auf jede Instanz einer Problemstellung anzupassen. Zu beachten ist nun, dass während der Simulation, aufgrund der Berechnung der Temperatur aus der Akzeptanzrate, die Temperatur zwischenzeitlich auch erhöht werden kann. Das Verfahren **SA-Acc** erwies sich dem **SA** Ansatz als ebenbürtig.

Eine Problemstellung aus der diskreten Industrie, das Fließbandproblem, erforderte eine eigene Modellierung. Dazu wurden in **Kapitel 10** verschiedene Modelle vorgestellt, die auf einer Kostenfunktion für Restriktionsverletzungen aufbauen. Die Restriktionsverletzungen können dabei linear oder quadratisch bestraft werden. Ein Unterschied ergibt sich dabei vor allem bei Instanzen, die *overconstrained* sind, d.h., bei denen nicht alle Restriktionen gleichzeitig erfüllt werden können. Bei einem Wettbewerb eines deutschen Automobilherstellers wurden die erzielten Resultate folgendermaßen eingestuft: „Als besonders bemerkenswert ist die Tatsache zu werten, dass Herr Puchta als Einzelperson in der Lage war, in kurzer Zeit qualitativ gleichwertige Ergebnisse zu liefern wie namhafte Anbieter von kommerzieller Optimierungssoftware“. Aufgrund der gesammelten Erfahrungen in diesem Bereich wurde ein genetischer Algorithmus als Teil einer Softwarelösung (**SAP APO SEQ**) zur Sequenzplanung in der diskreten Industrie erstellt. Diese Software ist nun bereits bei mehreren Kunden bei der täglichen Planung produktiv im Einsatz. Die Funktionalität der Softwarelösung geht dabei weit über die in **Kapitel 10** vorgestellte Modellierung hinaus.

Insgesamt ist zusammenfassend zu sagen, dass nicht nur das Optimierungsverfahren alleine über eine erfolgreiche Lösungssuche entscheidet, sondern vielmehr eine richtige Kombination aus Lösungsverfahren und Modellierung der Problemstellung erforderlich ist. Der in dieser Arbeit gezeigte Ansatz eines Modells mit direkter Repräsentation, auf dem ein lokales Suchverfahren arbeitet, erweist sich als sehr robust in der Lösungssuche bei verschiedensten Problemstellungen, die nicht nur das reine Scheduling, sondern auch zum Teil eine im Scheduling integrierte Losgrößenplanung erfordern. Die Robustheit zusammen mit der bemerkenswert guten Lösungsqualität ermöglichen den Einsatz bei real existierenden Problemstellungen. Die Reduzierung der für einen Optimierungslauf

erforderlichen Parameter erlaubt es auch einem Optimierungslaien, ein durch einen Experten erstelltes Modell zu verwenden, um die tägliche Produktionsplanung durchzuführen.

Neben den realistischen Benchmarks aus der Prozessindustrie wurde die in dieser Arbeit vorgestellte Kombination aus Modell und Optimierungsverfahren noch auf eine reale Problemstellung bei der Infineon AG Regensburg mit dem Titel *Erprobung einer Optimierungsmethode zur Verbesserung der Feinsteuerung in der Fertigung* angewendet. Dabei konnte nachgewiesen werden, dass dieses Planungsverfahren durchaus einsetzbar ist. Bei der Problemstellung waren als Optimierungsziele die Reduzierung der Rüstzeiten, Rüstkosten und der Durchlaufzeit sowie eine Einhaltung der Liefertermine gewünscht. Die Nebenbedingungen umfassten alternative Maschinen auf mehreren Produktionsstufen, unterschiedliche Produktionsgeschwindigkeiten der Maschinen und Pausen, in denen spezielle Umrüstvorgänge nicht durchgeführt werden durften.

Es kann wohl davon ausgegangen werden, dass das hier vorgestellte Verfahren aufgrund seiner großen Flexibilität bei der Definition neuer Nebenbedingungen und der erzielbaren Lösungsqualität in vertretbaren Rechenzeiten in Zukunft von noch größerer Bedeutung im kommerziellen Einsatz sein wird.

Anhang A - Flussdiagramme

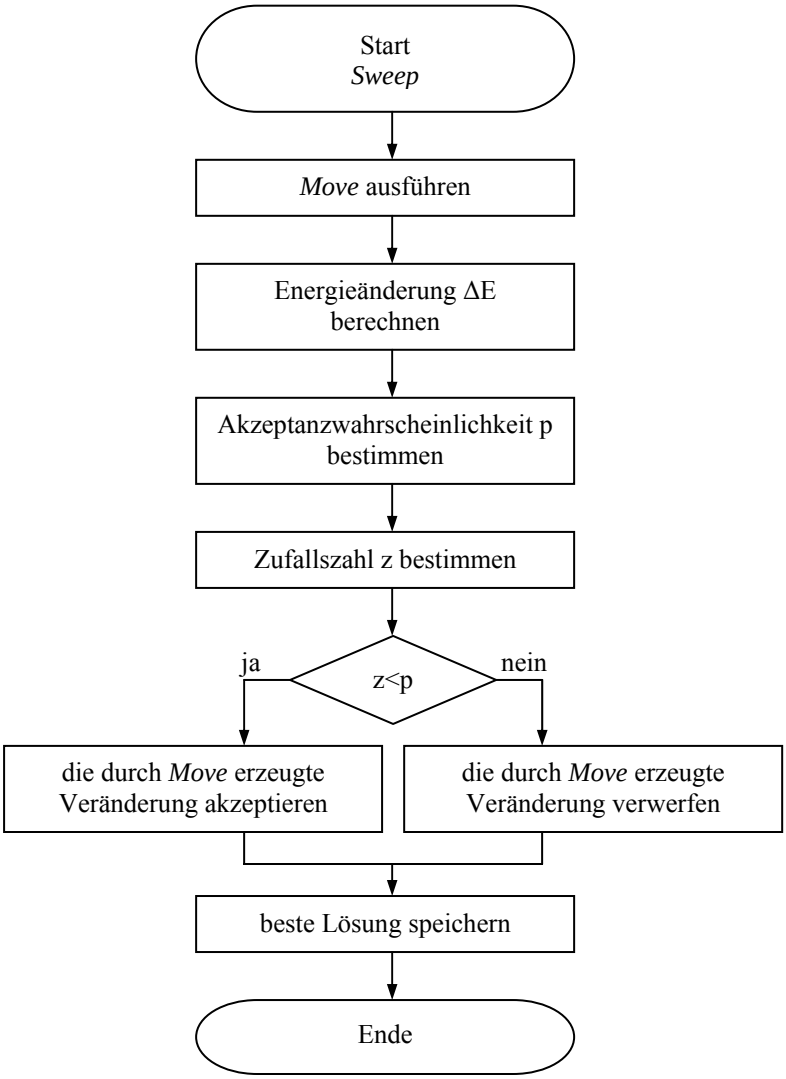


Abbildung A.1: Flussdiagramm eines *Sweeps*.

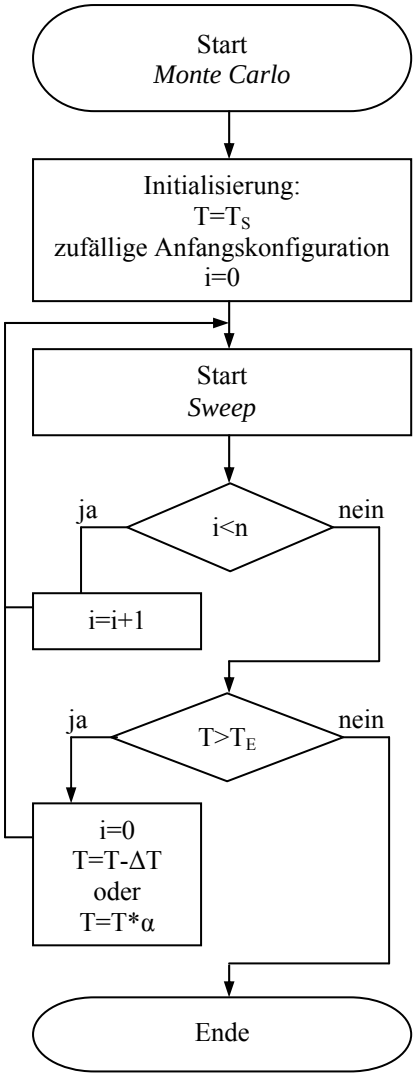


Abbildung A.2: Flussdiagramm eines *Monte Carlo* Verfahrens.

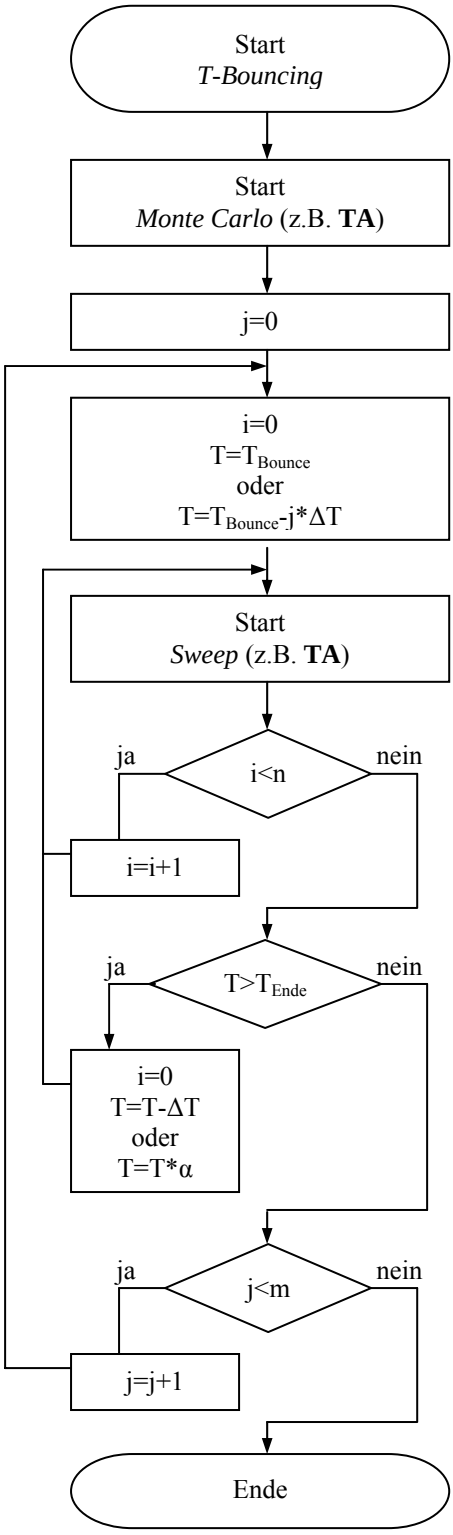


Abbildung A.3: Flussdiagramm des *T-Bouncing*.

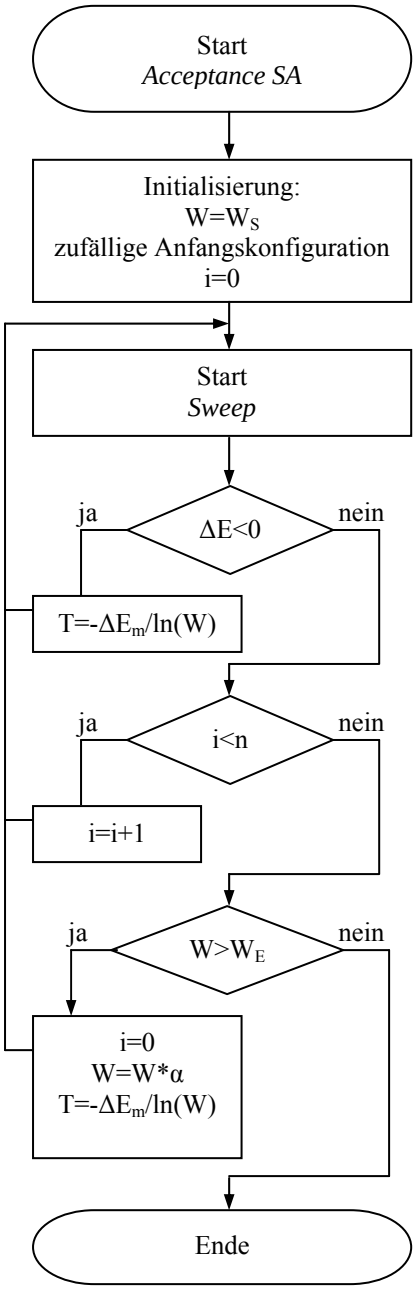


Abbildung A.4: Flussdiagramm des *Acceptance SA* (SA-Acc).

Anhang B - Suchraum

Suchraum und Nachbarschaften einiger Problemstellungen

Die Darstellung einer Lösung erfolgt in der Modellierung aus **Kapitel 3.2** mittels einer Konfiguration aus Aktivitäten (vgl. **Kapitel 3.2.4**). Diese Konfiguration dient zur Definition der Nachbarschaft (vgl. **Kapitel 3.2.5**). Eine Lösung, der Produktionsplan, wird deterministisch aus der Konfiguration erstellt (vgl. **Kapitel 3.2.6**).

B.1 Suchraum und Nachbarschaft der Konfiguration

Betrachtet man sich nur eine Konfiguration aus N Aktivitäten A_i, wobei jede Aktivität M_i Multimodi zur Auswahl hat, so besteht der Suchraum aus maximal $|S|$ Konfigurationen, wobei

$$|S| = N! \bullet \prod_{i=1}^{N} M_i \ . \qquad \textbf{(B.1)}$$

Die maximale Größe der Nachbarschaft der einzelnen Operatoren $|O_i|$ kann aus der Größe der Konfiguration N und der Anzahl der Multimodi einer Aktivität M_i bestimmt werden. Die Bedingung an die Operatoren, nicht nur die Konfiguration, sondern auch den daraus resultierenden Produktionsplan zu verändern (vgl. **Kapitel 3.2.5.3**) schränken diese Nachbarschaft und damit den Suchraum ein. In den folgenden Beispielen (**B.2** bis **B.8**) zur Größe des Suchraums ist auf diesen Unterschied zu achten.

Push1: $$|O_1| = (N-1) \bullet (N-1) \qquad \textbf{(B.2)}$$

Swap: $$|O_2| = \frac{1}{2} \bullet N \bullet (N-1) \qquad \textbf{(B.3)}$$

Lin2Opt: $$|O_3| = \frac{1}{2} \bullet N \bullet (N-1) \qquad \textbf{(B.4)}$$

Change: $$|O_4| = \sum_{i=1}^{N} M_i \qquad \textbf{(B.5)}$$

PushChange: $$|O_5| = \sum_{i=1}^{N} (N-1) \bullet (M_i - 1) \qquad \textbf{(B.6)}$$

Swap&Change: $$|O_6| \le \frac{1}{2} \bullet N \bullet (N-1) \qquad \textbf{(B.7)}$$

SwapSameRessource: $$|O_7| \le \frac{1}{2} \bullet N \bullet (N-1) \qquad \textbf{(B.8)}$$

Lin2OptSameRessource: $$|O_8| \le \frac{1}{2} \bullet N \bullet (N-1) \qquad \textbf{(B.9)}$$

Die gesamte Nachbarschaftsgröße $|O_g|$ ist jedoch kleiner als die Summe der einzelnen Operatoren, da deren erzeugte Nachbarschaften sich überschneiden, so ist z.B. die Nachbarschaft des **SwapSameRessource** vollständig in der Nachbarschaft des **Swap** enthalten:

$$\left|O_g\right| \le \left(\frac{3}{2} \cdot N - 3\right) \bullet (N-1) + \sum_{i=1}^{N} N \cdot M_i \ . \qquad \textbf{(B.10)}$$

Die folgenden Beispiele dienen zur Veranschaulichung der unterschiedlichen Größen der Suchräume verschiedener Problemstellungen.

B.2 Eine Maschine

Es sollen N Aktivitäten auf einer Maschine geplant werden. Die Anzahl der Lösungen im Suchraum ist:

$$\left|S^1\right| = N! . \qquad \textbf{(B.11)}$$

Sind jedoch identische Aktivitäten vorhanden, so wird der Suchraum reduziert. Sind A_i die K unterscheidbaren Aktivitäten und $K_i>0$ die Anzahl der mit A_i identischen Aktivitäten, dann ist die Größe des Suchraums gegeben durch $|S^2|$. Folgende Bedingungen gelten in diesem Fall:

$$N = \sum_{i=1}^{K} K_i \ , \qquad \textbf{(B.12)}$$

$$\left|S^2\right| = \left|S^1\right| \bullet \prod_{i=1}^{N^U} \frac{1}{K_i!} . \qquad \textbf{(B.13)}$$

Ist durch Aktivitätslinks eine Reihenfolge einer oder mehrerer Teilmengen an Aktivitäten festgelegt, so verkleinert dies den Suchraum. Sind L solcher Teilmengen vorhanden und L_i deren Anzahl an Aktivitäten, dann ist die Größe des Suchraums gegeben durch $|S^3|$:

$$\left|S^3\right| = \left|S^1\right| \bullet \prod_{i=1}^{L} \frac{1}{L_i!} . \qquad \textbf{(B.14)}$$

B.3 Mehrere unabhängige Maschinen

Sind M unabhängige Maschinen zu planen, d.h. jede zu planende Aktivität kann nur auf einer dieser Maschinen ausgeführt werden, und ist $|S^1_i|$ die Größe des Suchraums der Maschine *i*, so ergibt sich der gesamte Suchraum durch $|S^4|$:

$$\left|S^4\right| = \prod_{i=1}^{M} \left|S^1{}_i\right| . \qquad \textbf{(B.15)}$$

In solch einem Fall sind jedoch *M* Planungsvorgänge mit jeweils einer Maschine der gleichzeitigen Planung der *M* Maschinen vorzuziehen.

B.4 Mehrere alternative Maschinen

Sind M alternative **unterscheidbare** Maschinen zu planen, d.h. jede der N Aktivitäten kann auf jeder der Maschinen ausgeführt werden, so besteht der Suchraum aus $|S^5|$ Konfigurationen: **(B.16)**. Die Maschinen sind z.B. unterscheidbar aufgrund der unterschiedlichen Produktionsdauer der Aktivitäten auf den verschiedenen Maschinen.

$$\left|S^5\right| = N! \bullet g(N, M), \qquad \textbf{(B.16)}$$

wobei durch *g(N,M)* die Verteilung auf die einzelnen Maschinen geregelt wird:

$$g(N, M) = \sum_{i=0}^{N} g(i, M-1), \qquad \textbf{(B.17)}$$

wobei *g(0,M)*=1 und *g(N,1)*=1. Dies ergibt z.B.

$$g(N,2) = N + 1,$$
$$g(N,3) = \frac{1}{2} \cdot \left(N^2 + 3 \cdot N + 2\right),$$
$$g(N,4) = \frac{1}{6} \cdot N^3 + N^2 + \frac{11}{6} \cdot N + 1.$$

B.5 FlowShop 1 (Sonderfall eine Maschinen)

Es wird auf jeder Maschine die gleiche Reihenfolge der Aufträge produziert. Existieren pro Maschine *N* Aktivitäten, dann ist die Größe des Suchraum gegeben durch:

$$\left|S^6\right| = N! \ . \qquad \textbf{(B.18)}$$

B.6 FlowShop 2 (Sonderfall mehrerer unabhängiger Maschinen)

Die *M* Maschinen haben eine bestimmte Reihenfolge. Es kann auf jeder Maschine eine unterschiedliche Reihenfolge der Aufträge produziert werden. Bei *N* Aktivitäten pro Maschine existieren $|S^7|$ Konfigurationen im Suchraum:

$$\left|S^7\right| = (N!)^{M} \ . \qquad \textbf{(B.19)}$$

B.7 OpenShop

Neben der Reihenfolge der *N* Aktivitäten pro Maschine sind noch die Reihenfolgen der *M* Aktivitäten pro Auftrag festzulegen. Der Suchraum hat die Größe:

$$\left|S^8\right| = (N!)^{M} \bullet (M!)^{N} \ . \qquad \textbf{(B.20)}$$

B.8 JobShop

Es existieren *N* Aufträge, die auf *M* Maschinen geplant werden sollen. Der Suchraum ist hier im Unterschied zu der Planung *M* unabhängiger Maschinen eingeschränkt durch die Reihenfolgebedingung innerhalb eines Auftrags. Die Reihenfolge, in der die einzelnen Aufträge die Maschinen benötigen kann zwischen den einzelnen Aufträgen unterschiedlich sein. Einige der möglichen Sequenzen auf den Maschinen sind deshalb nicht gültig. Der Suchraum besteht aus $|S^8|$ Konfigurationen:

$$\left|S^8\right| \le (N!)^{M} \ . \qquad \textbf{(B.21)}$$

Anhang C – Production Flow Planning

Die Simulationsdaten

Im folgenden sind die Daten für die Problemstellung des *Production Flow Planning with Machine Assignment* aus **Kapitel 7** gegeben.
Es existieren Aufträge für 9 Tage, wobei der 5. und 6. Tag das Wochenende darstellt, für die keine Aufträge existieren (**Tabelle C.1**). Bei dieser Problemstellung existieren 24 verschiedene Produkte. Zusätzlich zu den ressourcenabhängigen Produktionszeiten (und Rüstzeiten) (**Tabelle C.2**) müssen auf den Ressourcen der 1. und 2. Stufe noch produktabhängige Produktionszeiten addiert werden (**Tabelle C.3**).

Datensatz	Tag	Aufträge
1	1	P1, P2, P2, P3, P4, P6, P7, P8, P9, P9, P11, P12
2	2	P1, P2, P4, P5, P9, P11, P14, P15, P16, P18
3	3	P1, P2 P3, P4, P6, P7, P8, P9, P10, P17, P24
4	4	P2, P2, P3, P4, P8, P9, P9, P11, P12, P21
5	7	P4, P6, P7, P9, P9, P11, P16, P20, P22
6	8	P1, P2, P3, P4, P8, P9, P11, P13, P18, P23
7	9	P1, P2, P3, P4, P6, P9, P9 ,P12, P14, P15, P16, P17, P24
8	10	P1, P2, P4, P7, P8, P9, P9, P11
9	11	P2, P3, P4, P5, P6, P7, P9, P11, P21

Tabelle C.1: Auftragsliste (**Aufträge**) der einzelnen Tagespakete (**Tag**).

Ressource	Stufe	Produktionszeit	Rüstzeit
1	1	10	75
2	1	10	75
3	2	30	75
4	2	30	75
5	3	60	190
6	3	60	190
7	4	0	195
8	5	0	260
9	5	0	260
10	6	380	150
11	6	380	150
12	6	380	150
13	6	380	150

Tabelle C.2: Ressourcenabhängige Produktions- und Rüstzeiten in Minuten.

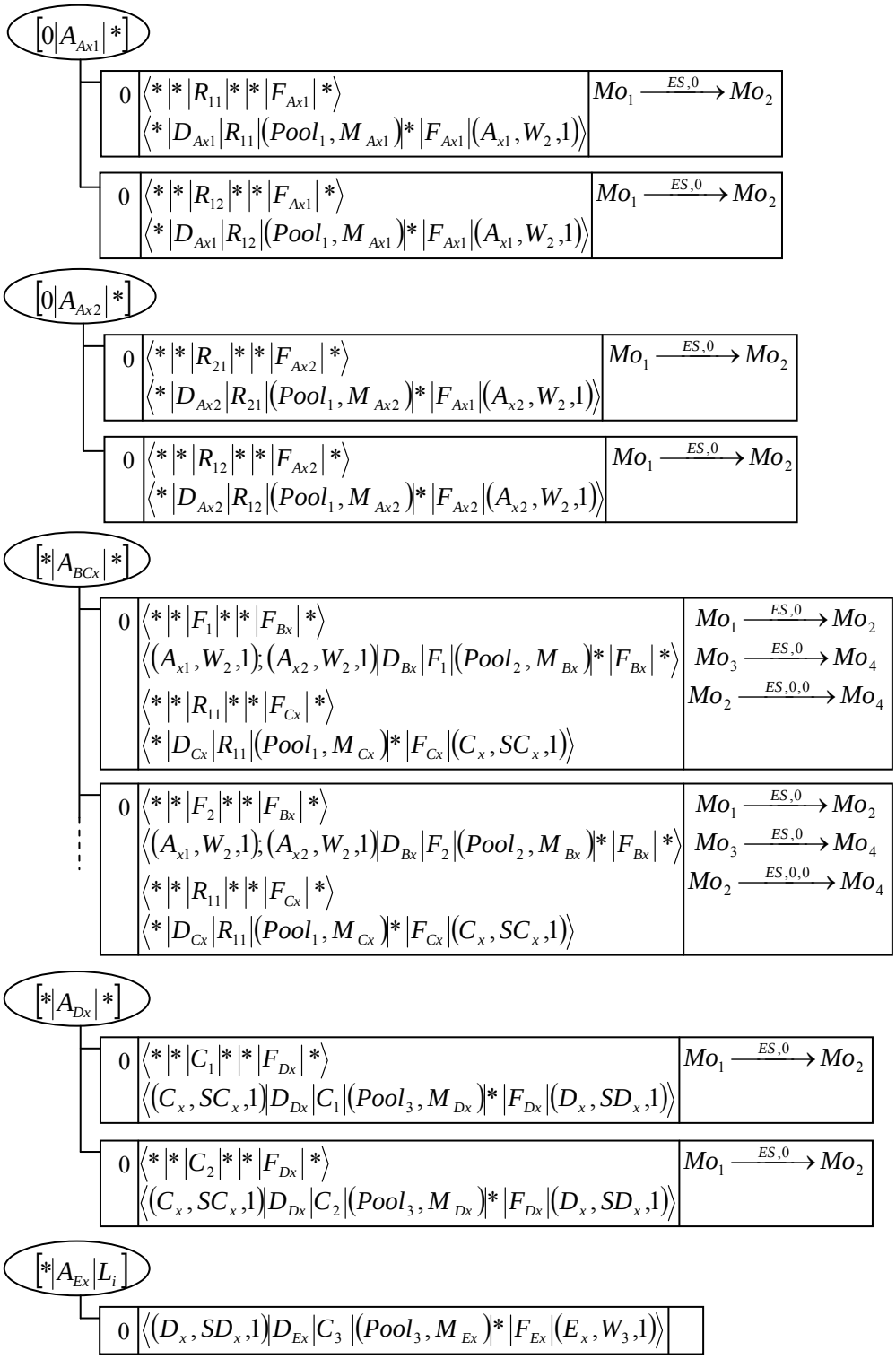
Produkt	Produktionszeit Stufe 1	Produktionszeit Stufe 2	überspringt Stufe4
P1	2:04:29	1:30:00	ja
P2	2:59:34	2:30:00	ja
P3	2:11:58	2:15:00	ja
P4	4:00:00	3:00:00	ja
P5	2:00:00	3:00:00	ja
P6	1:55:42	3:00:00	nein
P7	7:23:41	7:15:00	nein
P8	2:41:00	2:30:00	ja
P9	2:00:00	3:00:00	ja
P10	2:00:00	3:00:00	ja
P11	2:00:00	3:00:00	ja
P12	3:50:18	0:30:00	nein
P13	1:30:00	2:00:00	nein
P14	1:30:00	3:00:00	nein
P15	1:30:00	6:45:00	nein
P16	2:47:01	1:30:00	nein
P17	2:00:00	3:00:00	nein
P18	2:00:00	3:00:00	ja
P19	2:00:00	2:00:00	ja
P20	4:00:00	4:25:00	ja
P21	3:00:00	2:30:00	nein
P22	4:04:06	3:10:00	ja
P23	2:00:00	3:30:00	nein
P24	2:00:00	3:00:00	ja

Tabelle C.3: Produktabhängige Produktionszeiten für die 1. und 2. Stufe und die Information, ob ein Produkt die 4. Stufe bei der Produktion überspringt.

Anhang D – Bench01

D.1 Aktivitäten

Es existieren 10 unterschiedliche Aufträge, wobei jeder aus 6 Aktivitäten besteht. Die Unterschiede liegen dabei in den Dauern der Modi (D_{Yx}), den Rüstfamilien (F_{Yx}), den benötigten Kapazitäten der Multiressourcen (M_{Yx}) und der Nutzung der Silos. Von den zwölf möglichen Multimodi der Aktivität A_{BCx} sind nur zwei gegeben.



D.2 Simulationsdaten

In den folgenden Tabellen sind die Daten für die Problemstellung Bench01 gegeben.

product ID	(alternative) machines	T_O [h]	Product Family	required pool	required number of pool objects	required preproducts
A01	R11, R12	00:30	F1	1	1	
A02	R11, R12	01:10	F2	1	2	
A03	R11, R12	01:50	F3	1	2	
A04	R11, R12	00:50	F4	1	3	
A06	R21, R22	00:50	F1	1	3	
A07	R21, R22	00:40	F2	1	2	
A08	R21, R22	01:00	F3	1	1	
A09	R21, R22	01:40	F4	1	2	
B01	F1, F2, F3	01:00	F1	2	2	A01, A06
B02	F1, F2, F3	03:00	F1	2	3	A01, A07
B03	F1, F2, F3	02:00	F1	2	3	A01, A08
B04	F1, F2, F3	02:20	F2	2	1	A02, A06
B05	F1, F2, F3	01:00	F2	2	3	A02, A09
B06	F1, F2, F3	01:20	F3	2	2	A03, A07
B07	F1, F2, F3	03:50	F3	2	2	A03, A08
B08	F1, F2, F3	01:30	F4	2	1	A04, A07
B09	F1, F2, F3	00:40	F4	2	3	A04, A08
B10	F1, F2, F3	02:40	F4	2	1	A04, A09
C01	R11, R12,R21, R22	00:30	F1	1	2	B01
C02	R11, R12,R21, R22	00:50	F1	1	2	B02
C03	R11, R12,R21, R22	01:30	F1	1	1	B03
C04	R11, R12,R21, R22	00:40	F2	1	2	B04
C05	R11, R12,R21, R22	01:20	F2	1	2	B05
C06	R11, R12,R21, R22	00:40	F3	1	1	B06
C07	R11, R12,R21, R22	01:50	F3	1	3	B07
C08	R11, R12,R21, R22	02:40	F4	1	1	B08
C09	R11, R12,R21, R22	02:00	F4	1	2	B09
C10	R11, R12,R21, R22	03:30	F4	1	1	B10
D01	C1, C2	01:20	F1	3	3	C01
D02	C1, C2	01:00	F1	3	3	C02
D03	C1, C2	01:00	F1	3	2	C03
D04	C1, C2	02:00	F2	3	2	C04
D05	C1, C2	04:00	F2	3	1	C05
D06	C1, C2	02:00	F3	3	3	C06
D07	C1, C2	03:03	F3	3	2	C07
D08	C1, C2	00:40	F4	3	3	C08
D09	C1, C2	02:00	F4	3	2	C09
D10	C1, C2	01:00	F4	3	2	C10
E01	C3	00:30	F1	3	2	D01
E02	C3	02:00	F1	3	1	D02
E03	C3	00:30	F1	3	3	D03
E04	C3	01:15	F2	3	2	D04
E05	C3	01:15	F2	3	1	D05
E06	C3	02:00	F3	3	3	D06
E07	C3	01:32	F3	3	2	D07
E08	C3	00:30	F4	3	1	D08
E09	C3	01:32	F4	3	3	D09
E10	C3	01:32	F4	3	1	D10

Tabelle D.1: Liste der Produkte (**product ID**) mit ihren alternativen Maschinen (**machines**), ihren Produktionszeiten (**T_0 [h]**), den benötigten Kapazitäten (**number pool objects**) der Sekundärressourcen (**pool**), den benötigten Rohstoffen (**required preproducts**) und der zugehörigen Produktfamilie für die Rüstzeiten (**Family**).

Pool	Number of Pool Objects
1	8
2	6
3	6

Tabelle D.2: Kapazität der Pools.

Storage	Location after production stage	Capacity	Dedicated to product
SC01	C	2	C01
SC02	C	2	C02
SC03	C	2	C03
SC04	C	2	C04
SC05	C	2	C05
SC06	C	2	C06
SC07	C	2	C07
SC08	C	2	C08
SC09	C	2	C09
SC10	C	2	C10
SD01	D	2	D01
SD02	D	2	D02
SD03	D	2	D03
SD04	D	2	D04
SD05	D	2	D05
SD06	D	2	D06
SD07	D	2	D07
SD08	D	2	D08
SD09	D	2	D09
SD10	D	2	D10

Tabelle D.3: Kapazität der speziellen Lager für bestimmte Produkte.

Reactor				
P \ S	F1	F2	F3	F4
F1	0	10	10	20
F2	10	0	40	20
F3	30	20	0	10
F4	60	30	10	0

Feeder				
P \ S	F1	F2	F3	F4
F1	0	20	20	20
F2	30	10	20	20
F3	60	30	20	20
F4	70	50	40	30

Filter				
P \ S	F1	F2	F3	F4
F1	0	10	30	50
F2	20	0	10	20
F3	30	30	0	20
F4	80	60	40	0

Tabelle D.4: Rüstzeiten der Maschinen (**Reactor**, **Feeder**, **Filter**) zwischen dem Vorgänger (**P**) und dem Nachfolger (**S**) aus den Produktfamilien (**F1**, **F2**, **F3**, **F4**).

D.3 Probleminstanzen

Instanz_0	
Produkt	L
E01	2160
E01	2640
E01	3600
E02	3360
E03	1200
E03	1920
E03	2520
E03	3720
E04	2160
E04	3240
E04	3600
E05	2160
E05	3600
E06	960
E06	3660
E07	1140
E07	2700
E07	3420
E08	1380
E08	3660
E09	1800
E09	1800
E09	3840
E10	2640
E10	3720

Instanz_1	
Produkt	L
E01	2280
E01	2280
E01	2280
E02	2280
E03	2280
E03	2280
E03	2280
E03	2280
E04	2280
E04	2280
E04	2280
E05	2280
E05	2280
E06	2280
E06	2280
E07	2280
E07	2280
E07	2280
E08	2280
E08	2280
E09	2280
E09	2280
E09	2280
E10	2280
E10	2280

Instanz_2	
Produkt	L
E01	2520
E01	2520
E01	2520
E02	2520
E03	2520
E03	2520
E03	2520
E03	2520
E04	2520
E04	2520
E04	2520
E05	2520
E05	2520
E06	2520
E06	2520
E07	2520
E07	2520
E07	2520
E08	2520
E08	2520
E09	2520
E09	2520
E09	2520
E10	2520
E10	2520

Instanz_3	
Produkt	L
E01	3960
E01	3960
E01	3960
E02	3960
E03	3960
E03	3960
E03	3960
E03	3960
E04	3960
E04	3960
E04	3960
E05	3960
E05	3960
E06	3960
E06	3960
E07	3960
E07	3960
E07	3960
E08	3960
E08	3960
E09	3960
E09	3960
E09	3960
E10	3960
E10	3960

Instanz_4		
Produkt	L_1	L_2
E01	2160	4160
E01	2640	4640
E01	3600	5600
E02	3360	5360
E03	1200	3200
E03	1920	3920
E03	2520	4520
E03	3720	5720
E04	2160	4160
E04	3240	5240
E04	3600	5600
E05	2160	4160
E05	3600	5600
E06	960	2960
E06	3660	5660
E07	1140	3140
E07	2700	4700
E07	3420	5420
E08	1380	3380
E08	3660	5660
E09	1800	3800
E09	1800	3800
E09	3840	5840
E10	2640	4640
E10	3720	5720

Instanz_5					
Produkt	L_1	L_2	L_3	L_4	L_5
E01	2160	4160	6160	8160	10160
E01	2640	4640	6640	8640	10640
E01	3600	5600	7600	9600	11600
E02	3360	5360	7360	9360	11360
E03	1200	3200	5200	7200	9200
E03	1920	3920	5920	7920	9920
E03	2520	4520	6520	8520	10520
E03	3720	5720	7720	9720	11720
E04	2160	4160	6160	8160	10160
E04	3240	5240	7240	9240	11240
E04	3600	5600	7600	9600	11600
E05	2160	4160	6160	8160	10160
E05	3600	5600	7600	9600	11600
E06	960	2960	4960	6960	8960
E06	3660	5660	7660	9660	11660
E07	1140	3140	5140	7140	9140
E07	2700	4700	6700	8700	10700
E07	3420	5420	7420	9420	11420
E08	1380	3380	5380	7380	9380
E08	3660	5660	7660	9660	11660
E09	1800	3800	5800	7800	9800
E09	1800	3800	5800	7800	9800
E09	3840	5840	7840	9840	11840
E10	2640	4640	6640	8640	10640
E10	3720	5720	7720	9720	11720

Tabelle D.5: Verschiedene Instanzen der Problemstellung Bench01. Es ist jeweils eine Liste an zu fertigenden Produkten mit den zugehörigen Lieferzeitpunkten (in Minuten nach dem Produktionsstart) gegeben. **Instanz_0** ist die in der ursprünglichen Aufgabenstellung gegebene Instanz.

D.3 Lösungen

Hier sind die besten Lösungen zu sehen, die mit den verschiedenen Modellen für verschiedene Instanzen und Prioritäten erhalten wurden.

Modell	Instanz	Prioritäten	Makespan	Setup
M1	0	**M**	2081	980
		M+S	2122	420
		S	2220	370
M1	1	**M**	2086	1150
		M+S	2171	470
		S	2239	440
M1	2	**M**	2052	1190
		M+S	2221	370
		S	2221	370
M1	3	**M**	2079	1120
		M+S	2209	370
		S	2209	370
M1	4	**M**	3908	1420
		M+S	4103	970
		S	4285	940
M1	5	**M**	9474	4920
		M+S	9734	3020
		S	9734	3020
M2	0	**M**	2061	850
		M+S	2075	440
		S	2238	370
M2	4	**M**	3905	1570
		M+S	3979	1040
		S	4188	1040
M2b	4	**M**	3847	1830
		M+S	4084	920
		S	4150	910
M2b	5	**M**	9288	4590
		M+S	9540	2800
		S	9632	2720

Modell	Instanz	Prioritäten	Makespan	Setup
M3	0'	**M**	2097	1030
		M+S	2221	370
		S	2221	370
M4	0'	**M**	2087	780
		M+S	2207	370
		S	2207	370
M5	0'	**M**	2291	1230
		M+S	2412	390
		S	2574	380
M6	0'	**M**	1990	1330
		M+S	2037	460
		S	2134	370
M7	0'	**M**	1955	550
		M+S	1987	490
		S	2117	370
M8	0'	**M**	1908	1100
		M+S	2052	380
		S	2065	370
M9	0'	**M**	2157	560
		M+S	2293	370
		S	2293	370

Tabelle D.6: Zu sehen sind die besten Lösungen für die verschiedenen Modelle und Instanzen, die während der Simulationen bei begrenzter Rechenzeit (siehe **Kapitel 8**) erhalten wurden.

Anhang E – Westenberger-Kallrath

E.1 Aktivitäten

Es werden die folgenden Aktivitätsschablonen für die Planung des Westenberger-Kallrath-Benchmark benötigt:

$[0|A_1|*]$

0	$\langle *\|*\|R_1\|*\|*\|1\|* \rangle$ $\langle *\|72\|R_1\|*\|*\|1\|(*P_{11},S_{11},1) \rangle$ $\rightarrow \{(* \rightarrow i) \mid i \in N;\ 3 \le i \le 10 \mid i = 3\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$[0|A_2|*]$

0	$\langle *\|*\|R_2\|*\|*\|1\|* \rangle$ $\langle (*P_{11},S_{11},1)\|144\|R_2\|*\|*\|1\|(*P_{21},S_{21},1);(*P_{22},S_{22},1) \rangle$ $\rightarrow \{(i \rightarrow j;k) \mid i,j,k \in N; i = j+k \wedge 10 \le i \le 20 \wedge 0.2*i \le j \le 0.7*i \mid i = 10 \wedge j = 5\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$[0|A_3|*]$

0	$\langle *\|*\|R_3\|*\|*\|1\|* \rangle$ $\langle (*P_{22},S_{22},1)\|72\|R_3\|*\|*\|1\|(*P_{31},S_{31},1);(*P_{11},S_{11},1) \rangle$ $\rightarrow \{(i \rightarrow j;k) \mid i \in N;\ j,k \in R; 5 \le i \le 10 \wedge i = j+k \wedge j = F_{31}*i \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$[0|A_{42}|*]$

0	$\langle *\|*\|R_4\|*\|*\|2\|* \rangle$ $\langle (*P_{21},S_{21},1)\|144\|R_4\|*\|*\|2\|(*P_{42},S_{42},1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i,j \in N; 4 \le i \le 10 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$[0|A_{43}|*]$

0	$\langle *\|*\|R_4\|*\|*\|3\|* \rangle$ $\langle (*P_{31},S_{31},1)\|144\|R_4\|*\|*\|3\|(*P_{43},S_{43},1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i,j \in N; 4 \le i \le 10 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$[0|A_{44}|*]$

0	$\langle *\|*\|R_4\|*\|*\|4\|* \rangle$ $\langle (*P_{31},S_{31},1)\|144\|R_4\|*\|*\|4\|(*P_{44},S_{44},1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i,j \in N; 4 \le i \le 10 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$\left[0\middle|A_{61}\middle|*\right]$

0	$\left\langle *\middle\|*\middle\|R_{6a}\middle\|*\middle\|*\middle\|1\middle\|*\right\rangle$ $\left\langle (*P_{42},S_{42},1)\middle\|144\middle\|R_{6a}\middle\|*\middle\|*\middle\|1\middle\|(*P_{61},S_{61},1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 3 \le i \le 7\middle\| i=3\right\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$
0	$\left\langle *\middle\|*\middle\|R_{6b}\middle\|*\middle\|*\middle\|1\middle\|*\right\rangle$ $\left\langle (*P_{42},S_{42},1)\middle\|180\middle\|R_{6b}\middle\|*\middle\|*\middle\|1\middle\|(*P_{61},S_{61},1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 3 \le i \le 7\middle\| i=3\right\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$\left[0\middle|A_{63}\middle|*\right]$

0	$\left\langle *\middle\|*\middle\|R_{6a}\middle\|*\middle\|*\middle\|3\middle\|*\right\rangle$ $\left\langle (*P_{44},S_{44},1)\middle\|216\middle\|R_{6a}\middle\|*\middle\|*\middle\|3\middle\|(*P_{63},S_{63},1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 3 \le i \le 7\middle\| i=3\right\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$
0	$\left\langle *\middle\|*\middle\|R_{6b}\middle\|*\middle\|*\middle\|3\middle\|*\right\rangle$ $\left\langle (*P_{44},S_{44},1)\middle\|216\middle\|R_{6b}\middle\|*\middle\|*\middle\|3\middle\|(*P_{63},S_{63},1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 3 \le i \le 7\middle\| i=3\right\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$\left[0\middle|A_{71}\middle|*\right]$

0	$\left\langle *\middle\|*\middle\|R_{5}\middle\|*\middle\|*\middle\|1\middle\|*\right\rangle$ $\left\langle (*P_{21},S_{21},1)\middle\|216\middle\|R_{5}\middle\|*\middle\|*\middle\|1\middle\|(*P_{51},*,1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 4 \le i \le 10\middle\| i=4\right\}$ $\left\langle *\middle\|*\middle\|R_{7a}\middle\|*\middle\|*\middle\|1\middle\|*\right\rangle$ $\left\langle (*P_{51},*,1)\middle\|144\middle\|R_{7a}\middle\|*\middle\|*\middle\|1\middle\|(*P_{71},S_{71},1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 4 \le i \le 10\middle\| i=4\right\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$
0	$\left\langle *\middle\|*\middle\|R_{5}\middle\|*\middle\|*\middle\|1\middle\|*\right\rangle$ $\left\langle (*P_{21},S_{21},1)\middle\|216\middle\|R_{5}\middle\|*\middle\|*\middle\|1\middle\|(*P_{51},*,1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 4 \le i \le 10\middle\| i=4\right\}$ $\left\langle *\middle\|*\middle\|R_{7b}\middle\|*\middle\|*\middle\|1\middle\|*\right\rangle$ $\left\langle (*P_{51},*,1)\middle\|216\middle\|R_{7b}\middle\|*\middle\|*\middle\|1\middle\|(*P_{71},S_{71},1)\right\rangle$ $\rightarrow \left\{(i \rightarrow j)\middle\| i,j \in N; 4 \le i \le 10\middle\| i=4\right\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$

$\left[0\middle|A_{72}\middle|*\right]$

0	$\langle *\|*\|R_5\|*\|*\|2\|*\rangle$ $\langle (*P_{31},S_{31},1)\|216\|R_5\|*\|*\|2\|(*P_{52},*,1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 10\mid i=4\}$ $\langle *\|*\|R_{7a}\|*\|*\|2\|*\rangle$ $\langle (*P_{52},*,1)\|144\|R_{7a}\|*\|*\|2\|(*P_{72},S_{72},1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 10\mid i=4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$
0	$\langle *\|*\|R_5\|*\|*\|2\|*\rangle$ $\langle (*P_{31},S_{31},1)\|216\|R_5\|*\|*\|2\|(*P_{52},*,1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 10\mid i=4\}$ $\langle *\|*\|R_{7b}\|*\|*\|2\|*\rangle$ $\langle (*P_{52},*,1)\|216\|R_{7b}\|*\|*\|2\|(*P_{72},S_{72},1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 10\mid i=4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$

$\left[0\middle|A_{73}\middle|*\right]$

0	$\langle *\|*\|R_4\|*\|*\|1\|*\rangle$ $\langle (*P_{21},S_{21},1)\|144\|R_4\|*\|*\|1\|(*P_{41},*,1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 6\mid i=4\}$ $\langle *\|*\|R_{7a}\|*\|*\|3\|*\rangle$ $\langle (*P_{41},*,1);(*P_{61},S_{61},1)\|144\|R_{7a}\|*\|*\|3\|(*P_{73},S_{73},1)\rangle$ $\rightarrow \{(i,j\rightarrow k)\mid i,j,k\in N;8\le k\le 12\wedge i+j=k\wedge i=j\mid i=4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$

$\left[0\middle|A_{75}\middle|*\right]$

0	$\langle *\|*\|R_{7a}\|*\|*\|5\|*\rangle$ $\langle (*P_{63},S_{63},1)\|216\|R_{7a}\|*\|*\|5\|(*P_{75},S_{75},1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 12\mid i=4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$
0	$\langle *\|*\|R_{7b}\|*\|*\|5\|*\rangle$ $\langle (*P_{63},S_{63},1)\|216\|R_{7b}\|*\|*\|5\|(*P_{75},S_{75},1)\rangle$ $\rightarrow \{(i\rightarrow j)\mid i,j\in N;4\le i\le 12\mid i=4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$

$[0|A_{74}|*]$

0	$\langle * \mid * \mid R_{6a} \mid * \mid * \mid 2 \mid * \rangle$ $\langle (* P_{43}, S_{43}, 1) \mid 180 \mid R_{6a} \mid * \mid * \mid 2 \mid (* P_{62}, *, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$ $\langle * \mid * \mid R_{7a} \mid * \mid * \mid 4 \mid * \rangle$ $\langle (* P_{62}, *, 1) \mid 216 \mid R_{7a} \mid * \mid * \mid 4 \mid (* P_{74}, S_{74}, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$
0	$\langle * \mid * \mid R_{6a} \mid * \mid * \mid 2 \mid * \rangle$ $\langle (* P_{43}, S_{43}, 1) \mid 180 \mid R_{6a} \mid * \mid * \mid 2 \mid (* P_{62}, *, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$ $\langle * \mid * \mid R_{7b} \mid * \mid * \mid 4 \mid * \rangle$ $\langle (* P_{62}, *, 1) \mid 216 \mid R_{7b} \mid * \mid * \mid 4 \mid (* P_{74}, S_{74}, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$
0	$\langle * \mid * \mid R_{6b} \mid * \mid * \mid 2 \mid * \rangle$ $\langle (* P_{43}, S_{43}, 1) \mid 216 \mid R_{6b} \mid * \mid * \mid 2 \mid (* P_{62}, *, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$ $\langle * \mid * \mid R_{7a} \mid * \mid * \mid 4 \mid * \rangle$ $\langle (* P_{62}, *, 1) \mid 216 \mid R_{7a} \mid * \mid * \mid 4 \mid (* P_{74}, S_{74}, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$
0	$\langle * \mid * \mid R_{6b} \mid * \mid * \mid 2 \mid * \rangle$ $\langle (* P_{43}, S_{43}, 1) \mid 216 \mid R_{6b} \mid * \mid * \mid 2 \mid (* P_{62}, *, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$ $\langle * \mid * \mid R_{7b} \mid * \mid * \mid 4 \mid * \rangle$ $\langle (* P_{62}, *, 1) \mid 216 \mid R_{7b} \mid * \mid * \mid 4 \mid (* P_{74}, S_{74}, 1) \rangle$ $\rightarrow \{(i \rightarrow j) \mid i, j \in N; 4 \le i \le 7 \mid i = 4\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_2 \xrightarrow{ES,0,0} Mo_4$

$[0|A_{74b}|*]$

0	$\langle *\|*\|R_{6a}\|*\|*\|2\|*\rangle$ $\langle (*P_{43},S_{43},1)\|180\|R_{6a}\|*\|*\|2\|(*P_{62},*,1)\rangle$ $\rightarrow \{(i \rightarrow j)\| i,j \in N; 5 \le i \le 6 \| i=5\}$ $\langle *\|*\|R_{6b}\|*\|*\|2\|*\rangle$ $\langle (*P_{43},S_{43},1)\|216\|R_{6b}\|*\|*\|2\|(*P_{62},*,1)\rangle$ $\rightarrow \{(i \rightarrow j)\| i,j \in N; 5 \le i \le 6 \| i=5\}$ $\langle *\|*\|R_{7a}\|*\|*\|4\|*\rangle$ $\langle (*P_{62},*,1)\|216\|R_{7a}\|*\|*\|4\|(*P_{74},S_{74},1)\rangle$ $\rightarrow \{(i \rightarrow j)\| i,j \in N; 10 \le i \le 12 \| i=10\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_5 \xrightarrow{ES,0} Mo_6$ $Mo_2 \xrightarrow{ES,0,0} Mo_6$ $Mo_4 \xrightarrow{ES,0,0} Mo_6$
0	$\langle *\|*\|R_{6a}\|*\|*\|2\|*\rangle$ $\langle (*P_{43},S_{43},1)\|180\|R_{6a}\|*\|*\|2\|(*P_{62},*,1)\rangle$ $\rightarrow \{(i \rightarrow j)\| i,j \in N; 5 \le i \le 6 \| i=5\}$ $\langle *\|*\|R_{6b}\|*\|*\|2\|*\rangle$ $\langle (*P_{43},S_{43},1)\|216\|R_{6b}\|*\|*\|2\|(*P_{62},*,1)\rangle$ $\rightarrow \{(i \rightarrow j)\| i,j \in N; 5 \le i \le 6 \| i=5\}$ $\langle *\|*\|R_{7b}\|*\|*\|4\|*\rangle$ $\langle (*P_{62},*,1)\|216\|R_{7b}\|*\|*\|4\|(*P_{74},S_{74},1)\rangle$ $\rightarrow \{(i \rightarrow j)\| i,j \in N; 10 \le i \le 12 \| i=10\}$	$Mo_1 \xrightarrow{ES,0} Mo_2$ $Mo_3 \xrightarrow{ES,0} Mo_4$ $Mo_5 \xrightarrow{ES,0} Mo_6$ $Mo_2 \xrightarrow{ES,0,0} Mo_6$ $Mo_4 \xrightarrow{ES,0,0} Mo_6$

E.2 Lösungen

In den folgenden Tabellen werden die Ergebnisse der Simulationen gezeigt. Zur Begrenzung der Rechenzeit wurde die Anzahl der Sweeps pro Temperaturschritt festgelegt (**20**, **100**, **1000**). Bei jeweils 200 Temperaturschritten führt dies zu insgesamt (4000, 20000 und 200000) durchgeführten Sweeps. Es wurden jeweils 24 verschiedene Bestellmengen ($\mathbf{B_0 - B_{23}}$) bearbeitet. Jede Tabelle enthält dabei für eine bestimmte Kombination aus Aufgabe (**(1)**, **(2)**, **(3)**, **(4)** und **(4b)**) und Initialzustand der Silos ($\mathbf{I_1}$, $\mathbf{I_2}$ und $\mathbf{I_3}$) die durchschnittlichen Ergebnisse (**av**) und die Standardabweichung (**dev**) aus jeweils 10 Simulation mit den beiden Strategien (**S1**, **S2**). Zusätzlich ist die Durchlaufzeit der besten gefundenen Lösung (**best**) zu sehen.

(1) - I_1	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2656,8 (52,9)	2484,0 (32,2)	2390,4 (17,6)	2584,8 (57,6)	2462,4 (43,2)	2390,4 (28,8)	2376
B_1	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_2	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_3	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1310,4 (28,8)	1296,0 (0,0)	1296,0 (0,0)	1296
B_4	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_5	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_6	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584
B_7	1360,8 (14,4)	1332,0 (0,0)	1332,0 (0,0)	1332,0 (32,2)	1296,0 (0,0)	1296,0 (0,0)	1296
B_8	1360,8 (14,4)	1332,0 (0,0)	1332,0 (0,0)	1368,0 (0,0)	1353,6 (28,8)	1296,0 (0,0)	1296
B_9	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584
B_{10}	1749,6 (48,8)	1720,8 (35,3)	1656,0 (0,0)	1692,0 (39,4)	1641,6 (48,8)	1584,0 (0,0)	1584
B_{11}	1936,8 (35,3)	1908,0 (39,4)	1872,0 (0,0)	1908,0 (45,5)	1843,2 (42,0)	1800,0 (0,0)	1800
B_{12}	1641,6 (28,8)	1612,8 (35,3)	1584,0 (0,0)	1569,6 (28,8)	1555,2 (35,3)	1526,4 (28,8)	1512
B_{13}	1807,2 (73,4)	1749,6 (36,7)	1670,4 (17,6)	1728,0 (60,2)	1656,0 (0,0)	1656,0 (45,5)	1584
B_{14}	2052,0 (29,4)	2023,2 (61,9)	2008,8 (121,3)	2052,0 (72,0)	2023,2 (69,8)	1958,4 (28,8)	1872
B_{15}	2743,2 (80,2)	2620,8 (42,0)	2577,6 (80,8)	2763,0 (112,0)	2592,0 (82,1)	2440,8 (57,6)	2376
B_{16}	2332,8 (35,3)	2304,0 (0,0)	2289,6 (28,8)	2289,6 (53,9)	2259,0 (29,8)	2217,6 (28,8)	2160
B_{17}	2541,6 (66,8)	2349,0 (77,9)	2296,8 (35,3)	2448,0 (82,1)	2311,2 (61,9)	2246,4 (17,6)	2232
B_{18}	2505,6 (74,1)	2419,2 (26,9)	2354,4 (28,8)	2412,0 (39,4)	2394,0 (74,2)	2311,2 (35,3)	2268
B_{19}	3402,0 (40,2)	3321,0 (53,2)	3211,2 (35,3)	3412,8 (107,8)	3276,0 (67,3)	3182,4 (17,6)	3168
B_{20}	2829,6 (84,0)	2736,0 (45,5)	2664,0 (0,0)	2727,0 (53,2)	2649,6 (28,8)	2606,4 (28,8)	2592
B_{21}	2835,0 (103,0)	2700,0 (45,5)	2577,6 (28,8)	2772,0 (50,9)	2671,2 (52,9)	2563,2 (35,3)	2520
B_{22}	2952,0 (80,5)	2836,8 (52,9)	2685,6 (53,9)	2923,2 (80,2)	2764,8 (57,6)	2642,4 (66,8)	2520
B_{23}	6804,0 (210,0)	6528,0 (189,0)	6120,0 (120,5)	6480,0 (178,2)	6609,6 (123,9)	6213,6 (165,1)	5976

(1) - I_2	Strategie - S1 (=S2)			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2232,0 (68,3)	2080,8 (66,0)	1987,2 (35,3)	1944
B_1	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_2	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_3	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_4	972,0 (0,0)	972,0 (0,0)	972,0 (0,0)	972
B_5	1123,2 (69,8)	1022,4 (28,8)	1036,8 (35,3)	1008
B_6	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1224
B_7	979,2 (14,4)	972,0 (0,0)	972,0 (0,0)	972
B_8	979,2 (14,4)	972,0 (0,0)	972,0 (0,0)	972
B_9	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1224
B_{10}	1353,6 (66,8)	1260,0 (45,5)	1224,0 (0,0)	1224
B_{11}	1584,0 (45,5)	1483,2 (14,4)	1447,2 (14,4)	1440
B_{12}	1267,2 (35,3)	1224,0 (0,0)	1224,0 (0,0)	1224
B_{13}	1375,2 (52,9)	1303,2 (14,4)	1296,0 (0,0)	1296
B_{14}	1701,0 (29,8)	1591,2 (61,9)	1530,0 (64,9)	1440
B_{15}	2361,6 (110,6)	2210,4 (66,8)	2102,4 (36,7)	2052
B_{16}	2008,8 (57,6)	1926,0 (31,2)	1872,0 (0,0)	1872
B_{17}	2131,2 (69,8)	1951,2 (26,9)	1879,2 (14,4)	1872
B_{18}	2102,4 (58,5)	2001,6 (36,7)	1922,4 (17,6)	1908
B_{19}	3012,0 (33,9)	2908,8 (26,9)	2764,8 (52,9)	2700
B_{20}	2361,6 (28,8)	2332,8 (26,9)	2304,0 (0,0)	2304
B_{21}	2440,8 (42,0)	2289,6 (48,8)	2167,2 (14,4)	2160
B_{22}	2520,0 (109,2)	2325,6 (92,8)	2296,8 (52,9)	2232
B_{23}	6768,0 (45,5)	5940,0 (36,0)	5760,0 (128,8)	5544

(1) - I_3	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2340,0 (91,8)	2174,4 (17,6)	2080,8 (35,3)	2296,8 (80,2)	2160,0 (22,8)	2124,0 (45,5)	2016
B_1	1015,2 (14,4)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_2	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_3	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1022,4 (28,8)	1008,0 (0,0)	1008,0 (0,0)	1008
B_4	1044,0 (32,2)	1029,6 (28,8)	1008,0 (0,0)	1022,4 (28,8)	1008,0 (0,0)	1008,0 (0,0)	1008
B_5	1080,0 (0,0)	1051,2 (35,3)	1008,0 (0,0)	1080,0 (45,5)	1022,4 (28,8)	1008,0 (0,0)	1008
B_6	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_7	1022,4 (28,8)	1008,0 (0,0)	1008,0 (0,0)	1036,8 (35,3)	1015,2 (14,4)	1008,0 (0,0)	1008
B_8	1051,2 (35,3)	1008,0 (0,0)	1008,0 (0,0)	1051,2 (35,3)	1008,0 (0,0)	1008,0 (0,0)	1008
B_9	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_{10}	1418,4 (36,7)	1396,8 (14,4)	1382,4 (17,6)	1404,0 (0,0)	1389,6 (17,6)	1368,0 (0,0)	1368
B_{11}	1656,0 (50,9)	1605,6 (17,6)	1584,0 (0,0)	1648,8 (35,3)	1555,2 (52,9)	1512,0 (0,0)	1512
B_{12}	1353,6 (28,8)	1310,4 (28,8)	1296,0 (0,0)	1296,0 (45,5)	1267,2 (35,3)	1224,0 (0,0)	1224
B_{13}	1418,4 (28,8)	1404,0 (32,2)	1382,4 (28,8)	1440,0 (39,4)	1375,2 (26,9)	1310,4 (28,8)	1296
B_{14}	1764,0 (91,8)	1771,2 (42,0)	1677,6 (58,5)	1809,0 (59,0)	1755,0 (53,2)	1683,0 (59,0)	1584
B_{15}	2462,4 (84,0)	2282,4 (36,7)	2181,6 (28,8)	2469,6 (66,8)	2268,0 (76,4)	2174,4 (17,6)	2160
B_{16}	2073,6 (53,9)	2023,2 (14,4)	1965,6 (28,8)	2008,8 (57,6)	2023,2 (47,8)	1944,0 (0,0)	1944
B_{17}	2203,2 (52,9)	2088,0 (22,8)	1965,6 (28,8)	2124,0 (50,9)	1994,4 (43,2)	1936,8 (47,8)	1872
B_{18}	2167,2 (73,4)	2131,2 (42,0)	2052,0 (32,2)	2217,6 (48,8)	2102,4 (48,8)	2008,8 (14,4)	1980
B_{19}	3015,0 (29,8)	2973,6 (66,8)	2764,8 (26,9)	3036,0 (89,8)	2908,8 (69,8)	2826,0 (82,5)	2736
B_{20}	2498,4 (95,5)	2376,0 (0,0)	2376,0 (0,0)	2462,4 (53,9)	2390,4 (53,9)	2318,4 (28,8)	2304
B_{21}	2484,0 (64,4)	2397,6 (66,8)	2268,0 (32,2)	2469,6 (62,8)	2412,0 (50,9)	2275,2 (26,9)	2232
B_{22}	2685,6 (62,8)	2534,4 (74,1)	2426,4 (28,8)	2682,0 (64,9)	2469,6 (62,8)	2404,8 (14,4)	2376
B_{23}	7164,0 (180,0)	6348,0 (61,2)	5832,0 (78,9)	6336,0 (84,0)	6219,0 (64,3)	5608,8 (73,4)	5472

(2) - I_1	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	3073,5 (64,1)	2898,0 (38,9)	2739,6 (109,4)	3192,0 (139,7)	2862,0 (112,1)	2674,8 (78,5)	2592
B_1	1512,0 (0,0)	1512,0 (0,0)	1512,0 (0,0)	1476,0 (0,0)	1476,0 (0,0)	1476,0 (0,0)	1476
B_2	1526,4 (17,6)	1519,2 (14,4)	1512,0 (0,0)	1476,0 (0,0)	1476,0 (0,0)	1476,0 (0,0)	1476
B_3	1540,8 (26,9)	1512,0 (0,0)	1512,0 (0,0)	1476,0 (0,0)	1461,6 (17,6)	1440,0 (0,0)	1440
B_4	1591,2 (21,6)	1555,2 (14,4)	1548,0 (0,0)	1544,4 (28,8)	1526,4 (7,2)	1515,6 (7,2)	1512
B_5	1562,4 (28,8)	1512,0 (0,0)	1512,0 (0,0)	1486,8 (31,4)	1440,0 (0,0)	1440,0 (0,0)	1440
B_6	1980,0 (0,0)	1969,2 (8,8)	1954,8 (14,4)	1904,4 (7,2)	1904,4 (21,0)	1890,0 (0,0)	1890
B_7	1587,6 (7,2)	1591,2 (8,8)	1584,0 (0,0)	1533,6 (26,5)	1526,4 (28,8)	1512,0 (0,0)	1512
B_8	1526,4 (28,8)	1512,0 (0,0)	1512,0 (0,0)	1501,2 (21,6)	1512,0 (0,0)	1497,6 (17,6)	1458
B_9	1976,4 (7,2)	1972,8 (8,8)	1972,8 (8,8)	1908,0 (0,0)	1893,6 (7,2)	1897,2 (8,8)	1890
B_{10}	2066,4 (13,5)	2012,4 (40,1)	2016,0 (32,2)	1971,0 (37,1)	1994,4 (17,6)	1972,8 (8,8)	1962
B_{11}	2300,4 (56,2)	2308,5 (7,8)	2196,0 (45,5)	2242,8 (26,9)	2217,6 (13,5)	2102,4 (28,8)	2088
B_{12}	1879,2 (14,4)	1850,4 (43,2)	1807,2 (52,9)	1814,4 (28,8)	1792,8 (14,4)	1692,0 (0,0)	1692
B_{13}	2077,2 (42,0)	2055,6 (28,8)	2023,2 (8,8)	1994,4 (53,9)	1965,6 (28,8)	1933,2 (14,4)	1908
B_{14}	2354,4 (125,4)	2293,2 (61,9)	2223,0 (46,8)	2318,4 (70,5)	2260,8 (83,3)	2133,0 (59,0)	2088
B_{15}	3285,0 (153,0)	3045,6 (62,8)	3002,4 (26,5)	3175,2 (125,5)	3072,0 (59,4)	2893,5 (56,0)	2808
B_{16}	2718,0 (60,2)	2613,6 (43,2)	2548,8 (35,3)	2638,8 (97,7)	2583,0 (69,1)	2448,0 (0,0)	2448
B_{17}	2948,4 (46,1)	2916,0 (111,0)	2781,0 (9,0)	2904,0 (17,0)	2760,0 (8,5)	2664,0 (50,9)	2592
B_{18}	2984,4 (75,0)	2826,0 (41,0)	2732,4 (58,5)	3015,0 (37,1)	2836,8 (70,7)	2692,8 (40,4)	2628
B_{19}	4248,0 (148,0)	3888,0 (123,4)	3780,0 (101,8)	3870,0 (101,3)	3852,0 (88,2)	3712,5 (98,9)	3582
B_{20}	3060,0 (36,0)	3042,0 (102,9)	2980,8 (61,9)	3229,2 (95,0)	3031,2 (69,8)	2966,4 (48,8)	2880
B_{21}	3558,0 (104,3)	3276,0 (77,4)	3159,0 (46,8)	3294,0 (73,5)	3276,0 (64,1)	3141,0 (57,6)	3096
B_{22}	3636,0 (64,4)	3450,0 (47,2)	3270,0 (51,6)	3604,5 (132,5)	3402,0 (99,2)	3177,0 (53,2)	3096
B_{23}	8820,0 (112,3)	7560,0 (95,5)	7574,4 (81,6)	8892,0 (147,8)	7911,0 (135,0)	7573,5 (140,8)	7470

(2) - I_2	Strategie - S1 (=S2)			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2550,0 (72,5)	2469,6 (92,8)	2376,0 (50,9)	2304
B_1	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1080
B_2	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1080
B_3	1123,2 (35,3)	1116,0 (0,0)	1116,0 (0,0)	1080
B_4	1134,0 (22,8)	1123,2 (14,4)	1116,0 (0,0)	1116
B_5	1281,6 (28,8)	1260,0 (32,2)	1238,4 (28,8)	1224
B_6	1548,0 (0,0)	1540,8 (8,8)	1530,0 (0,0)	1530
B_7	1170,0 (11,4)	1144,8 (24,4)	1126,8 (21,6)	1116
B_8	1144,8 (14,4)	1137,6 (17,6)	1144,8 (14,4)	1116
B_9	1544,4 (13,5)	1540,8 (8,8)	1530,0 (0,0)	1530
B_{10}	1605,6 (48,8)	1615,5 (7,8)	1616,4 (13,5)	1548
B_{11}	1872,0 (68,3)	1857,6 (52,7)	1814,4 (48,8)	1728
B_{12}	1432,8 (26,9)	1411,2 (14,4)	1411,2 (14,4)	1404
B_{13}	1651,5 (41,0)	1557,0 (29,8)	1558,8 (52,9)	1512
B_{14}	2008,8 (126,1)	1867,5 (41,0)	1740,0 (33,9)	1692
B_{15}	2727,0 (89,5)	2673,0 (59,0)	2475,0 (29,8)	2448
B_{16}	2307,6 (48,8)	2235,6 (80,8)	2160,0 (0,0)	2160
B_{17}	2592,0 (112,9)	2412,0 (121,0)	2326,5 (34,6)	2232
B_{18}	2613,6 (89,9)	2389,5 (85,7)	2340,0 (32,2)	2304
B_{19}	3546,0 (93,2)	3456,0 (36,0)	3312,0 (76,4)	3204
B_{20}	2760,0 (44,9)	2682,0 (18,0)	2602,8 (14,4)	2592
B_{21}	2934,0 (54,0)	2772,0 (66,1)	2656,8 (68,9)	2520
B_{22}	3186,0 (54,0)	2850,0 (61,2)	2750,4 (94,8)	2628
B_{23}	8244,0 (210,1)	7074,0 (156,0)	7189,2 (125,0)	7038

(2) - I_3	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2782,8 (144,3)	2548,8 (52,9)	2383,2 (57,6)	2736,0 (84,4)	2548,8 (61,9)	2419,2 (57,6)	2304
B_1	1195,2 (14,4)	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188
B_2	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1188,0 (0,0)	1209,6 (17,6)	1224,0 (0,0)	1188
B_3	1231,2 (14,4)	1224,0 (0,0)	1224,0 (0,0)	1202,4 (17,6)	1188,0 (0,0)	1188,0 (0,0)	1188
B_4	1260,0 (22,8)	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1188,0 (30,1)	1152,0 (0,0)	1152
B_5	1252,8 (42,0)	1224,0 (0,0)	1224,0 (0,0)	1227,6 (40,1)	1180,8 (35,3)	1152,0 (0,0)	1152
B_6	1584,0 (0,0)	1584,0 (0,0)	1580,4 (7,2)	1548,0 (0,0)	1548,0 (0,0)	1548,0 (0,0)	1548
B_7	1224,0 (0,0)	1245,6 (28,8)	1224,0 (0,0)	1238,4 (17,6)	1224,0 (0,0)	1224,0 (0,0)	1224
B_8	1281,6 (48,8)	1238,4 (17,6)	1224,0 (0,0)	1238,4 (17,6)	1231,2 (14,4)	1224,0 (0,0)	1224
B_9	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1548,0 (0,0)	1548,0 (0,0)	1548,0 (0,0)	1548
B_{10}	1648,8 (8,8)	1638,0 (0,0)	1638,0 (0,0)	1645,2 (37,1)	1598,4 (41,7)	1548,0 (0,0)	1548
B_{11}	1926,0 (75,5)	1792,8 (35,3)	1756,8 (35,3)	1922,4 (50,1)	1886,4 (34,9)	1742,4 (28,8)	1728
B_{12}	1605,6 (58,5)	1569,6 (36,7)	1512,0 (0,0)	1504,8 (35,3)	1418,4 (17,6)	1418,4 (17,6)	1404
B_{13}	1720,8 (42,0)	1641,6 (28,8)	1627,2 (35,3)	1728,0 (25,5)	1692,0 (39,4)	1641,6 (28,8)	1584
B_{14}	2052,0 (36,0)	2056,5 (60,2)	2002,5 (151,3)	2016,0 (0,0)	1987,2 (104,7)	1900,8 (73,4)	1764
B_{15}	2952,0 (0,0)	2754,0 (114,8)	2592,0 (31,2)	2943,0 (27,0)	2718,0 (62,4)	2520,0 (50,9)	2484
B_{16}	2430,0 (48,3)	2304,0 (39,4)	2289,6 (80,8)	2430,0 (116,7)	2268,0 (0,0)	2181,6 (28,8)	2160
B_{17}	2778,0 (69,5)	2505,6 (55,1)	2403,0 (63,0)	2622,0 (97,9)	2436,0 (44,9)	2299,5 (70,1)	2232
B_{18}	2664,0 (142,3)	2551,5 (54,6)	2426,4 (50,1)	2523,6 (112,9)	2491,2 (70,7)	2343,6 (36,7)	2304
B_{19}	3564,0 (95,3)	3528,0 (89,4)	3398,4 (36,7)	3528,0 (102,6)	3519,0 (99,0)	3402,0 (92,7)	3276
B_{20}	2808,0 (44,1)	2674,8 (31,4)	2592,0 (0,0)	2772,0 (58,8)	2721,6 (58,5)	2628,0 (44,1)	2592
B_{21}	2944,8 (82,6)	2812,5 (120,3)	2736,0 (64,4)	2988,0 (50,9)	2871,0 (89,5)	2710,8 (50,4)	2610
B_{22}	3282,0 (44,9)	3030,0 (8,5)	2838,0 (30,6)	3312,0 (29,4)	3054,0 (123,3)	2934,0 (96,4)	2808
B_{23}	8370,0 (83,1)	7416,0 (67,0)	7081,2 (58,7)	8028,0 (203,9)	7776,0 (185,3)	6786,0 (0,0)	6786

(3) - I_1	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2678,4 (87,0)	2620,8 (52,9)	2469,6 (62,8)	2656,8 (69,8)	2628,0 (32,2)	2448,0 (22,8)	2376
B_1	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_2	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368
B_3	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1396,8 (35,3)	1368,0 (0,0)	1368,0 (0,0)	1368
B_4	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1339,2 (35,3)	1296,0 (0,0)	1296,0 (0,0)	1296
B_5	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1339,2 (35,3)	1310,4 (28,8)	1296,0 (0,0)	1296
B_6	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656
B_7	1411,2 (35,3)	1382,4 (28,8)	1368,0 (0,0)	1382,4 (28,8)	1368,0 (0,0)	1368,0 (0,0)	1368
B_8	1411,2 (35,3)	1411,2 (35,3)	1368,0 (0,0)	1425,6 (53,9)	1396,8 (35,3)	1368,0 (0,0)	1368
B_9	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656,0 (0,0)	1656
B_{10}	1756,8 (14,4)	1735,2 (14,4)	1713,6 (28,8)	1728,0 (0,0)	1713,6 (28,8)	1706,4 (28,8)	1656
B_{11}	2001,6 (28,8)	1972,8 (14,4)	1886,4 (28,8)	1987,2 (35,3)	1944,0 (0,0)	1936,8 (14,4)	1872
B_{12}	1656,0 (0,0)	1598,4 (28,8)	1584,0 (0,0)	1670,4 (53,9)	1641,6 (17,6)	1584,0 (0,0)	1584
B_{13}	1843,2 (42,0)	1764,0 (32,2)	1728,0 (0,0)	1800,0 (45,5)	1749,6 (28,8)	1728,0 (0,0)	1728
B_{14}	2224,8 (57,6)	2073,6 (48,8)	2023,2 (61,9)	2224,8 (102,8)	2142,0 (93,5)	2023,2 (89,3)	1908
B_{15}	2836,8 (57,6)	2718,0 (54,0)	2628,0 (32,2)	2865,6 (74,1)	2790,0 (54,0)	2671,2 (57,6)	2592
B_{16}	2390,4 (28,8)	2318,4 (53,9)	2304,0 (0,0)	2448,0 (50,9)	2376,0 (0,0)	2311,2 (26,9)	2232
B_{17}	2520,0 (60,2)	2448,0 (45,5)	2332,8 (42,0)	2476,8 (26,9)	2412,0 (78,9)	2354,4 (36,7)	2268
B_{18}	2642,4 (36,7)	2511,0 (46,8)	2433,6 (36,7)	2613,6 (43,2)	2534,4 (36,7)	2404,8 (35,3)	2376
B_{19}	3546,0 (90,0)	3398,4 (66,8)	3268,8 (69,8)	3528,0 (72,0)	3408,0 (61,2)	3297,6 (89,9)	3204
B_{20}	2800,8 (80,2)	2721,6 (53,9)	2664,0 (0,0)	2844,0 (60,2)	2808,0 (0,0)	2678,4 (53,9)	2592
B_{21}	2894,4 (36,7)	2772,0 (50,9)	2649,6 (36,7)	2907,0 (64,3)	2793,6 (36,7)	2671,2 (14,4)	2592
B_{22}	3067,2 (42,0)	2980,8 (73,4)	2808,0 (50,9)	3096,0 (55,8)	2966,4 (36,7)	2822,4 (28,8)	2736
B_{23}	7344,0 (315,3)	7092,0 (233,3)	6508,8 (172,5)	7236,0 (213,7)	7416,0 (485,7)	6566,4 (28,8)	6300

(3) - I_2	Strategie - S1 (=S2)			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2275,2 (69,8)	2116,8 (26,9)	2066,4 (28,8)	2016
B_1	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_2	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_3	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_4	972,0 (0,0)	972,0 (0,0)	972,0 (0,0)	972
B_5	1080,0 (64,4)	1036,8 (35,3)	1022,4 (28,8)	1008
B_6	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1224
B_7	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_8	1029,6 (28,8)	1008,0 (0,0)	1008,0 (0,0)	1008
B_9	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1224
B_{10}	1404,0 (39,4)	1360,8 (14,4)	1310,4 (28,8)	1296
B_{11}	1605,6 (28,8)	1569,6 (48,8)	1497,6 (17,6)	1476
B_{12}	1238,4 (28,8)	1231,2 (14,4)	1224,0 (0,0)	1224
B_{13}	1418,4 (43,2)	1332,0 (32,2)	1296,0 (0,0)	1296
B_{14}	1692,0 (39,4)	1591,2 (80,2)	1569,6 (66,8)	1440
B_{15}	2352,0 (17,0)	2289,6 (77,5)	2181,6 (36,7)	2124
B_{16}	2008,8 (47,8)	1944,0 (0,0)	1872,0 (0,0)	1872
B_{17}	2138,4 (53,9)	1965,6 (28,8)	1886,4 (17,6)	1872
B_{18}	2167,2 (52,9)	2037,6 (17,6)	1972,8 (26,9)	1944
B_{19}	3060,0 (0,0)	3031,2 (69,8)	2829,6 (53,9)	2772
B_{20}	2376,0 (78,9)	2332,8 (57,6)	2304,0 (0,0)	2304
B_{21}	2376,0 (72,0)	2268,0 (32,2)	2181,6 (28,8)	2160
B_{22}	2556,0 (44,1)	2548,8 (105,3)	2318,4 (48,8)	2268
B_{23}	6960,0 (220,6)	6876,0 (358,6)	6098,4 (252,1)	5724

(3) - I_3	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2289,6 (53,9)	2174,4 (53,9)	2131,2 (35,3)	2498,4 (84,0)	2311,2 (73,4)	2160,0 (39,4)	2088
B_1	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_2	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1152,0 (0,0)	1152,0 (0,0)	1152,0 (0,0)	1080
B_3	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1152,0 (0,0)	1152,0 (0,0)	1152,0 (0,0)	1080
B_4	1036,8 (35,3)	1008,0 (0,0)	1008,0 (0,0)	1022,4 (28,8)	1008,0 (0,0)	1008,0 (0,0)	1008
B_5	1065,6 (28,8)	1022,4 (28,8)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008,0 (0,0)	1008
B_6	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368
B_7	1108,8 (35,3)	1080,0 (0,0)	1080,0 (0,0)	1101,6 (28,8)	1087,2 (14,4)	1080,0 (0,0)	1080
B_8	1116,0 (32,2)	1087,2 (14,4)	1080,0 (0,0)	1094,4 (28,8)	1094,4 (28,8)	1080,0 (0,0)	1080
B_9	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368
B_{10}	1411,2 (14,4)	1404,0 (0,0)	1368,0 (0,0)	1404,0 (0,0)	1396,8 (14,4)	1389,6 (17,6)	1368
B_{11}	1634,4 (17,6)	1620,0 (0,0)	1584,0 (0,0)	1684,8 (26,9)	1620,0 (0,0)	1591,2 (14,4)	1584
B_{12}	1368,0 (0,0)	1324,8 (35,3)	1296,0 (0,0)	1382,4 (28,8)	1339,2 (35,3)	1296,0 (0,0)	1296
B_{13}	1483,2 (26,9)	1418,4 (28,8)	1368,0 (0,0)	1519,2 (47,8)	1447,2 (14,4)	1404,0 (32,2)	1368
B_{14}	1809,0 (85,9)	1791,0 (39,2)	1706,4 (115,2)	1929,6 (53,9)	1828,8 (26,9)	1821,6 (66,8)	1584
B_{15}	2527,2 (89,3)	2394,0 (54,0)	2289,6 (17,6)	2565,0 (85,9)	2404,8 (26,9)	2260,8 (26,9)	2232
B_{16}	2160,0 (75,5)	2073,6 (53,9)	2016,0 (0,0)	2217,6 (70,5)	2095,2 (35,3)	2001,6 (28,8)	1944
B_{17}	2196,0 (101,8)	2102,4 (28,8)	2016,0 (0,0)	2268,0 (96,6)	2181,6 (36,7)	2030,4 (17,6)	2016
B_{18}	2253,6 (36,7)	2196,0 (32,2)	2080,8 (14,4)	2296,8 (47,8)	2203,2 (57,6)	2152,8 (66,0)	2052
B_{19}	3204,0 (95,5)	3067,2 (83,3)	2916,0 (75,5)	3288,0 (220,6)	3060,0 (60,2)	2966,4 (53,9)	2772
B_{20}	2512,8 (92,2)	2419,2 (35,3)	2376,0 (0,0)	2700,0 (106,0)	2548,8 (57,6)	2368,8 (35,3)	2304
B_{21}	2613,6 (62,8)	2448,0 (75,5)	2318,4 (17,6)	2584,8 (95,0)	2484,0 (91,1)	2390,4 (17,6)	2304
B_{22}	2750,4 (36,7)	2664,0 (45,5)	2527,2 (14,4)	2815,2 (95,0)	2764,8 (26,9)	2512,8 (35,3)	2448
B_{23}	6912,0 (298,9)	6750,0 (302,8)	6134,4 (87,0)	7488,0 (350,0)	7056,0 (211,1)	6192,0 (165,8)	6012

(4) - I_1	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	3145,5 (137,9)	3078,0 (82,5)	2880,0 (68,3)	3145,5 (34,6)	2988,0 (25,5)	2887,2 (73,4)	2772
B_1	1548,0 (0,0)	1548,0 (0,0)	1548,0 (0,0)	1476,0 (0,0)	1476,0 (0,0)	1476,0 (0,0)	1476
B_2	1548,0 (0,0)	1548,0 (0,0)	1548,0 (0,0)	1526,4 (28,8)	1512,0 (0,0)	1512,0 (0,0)	1512
B_3	1548,0 (0,0)	1548,0 (0,0)	1548,0 (0,0)	1519,2 (69,8)	1483,2 (14,4)	1476,0 (0,0)	1476
B_4	1594,8 (14,4)	1584,0 (0,0)	1584,0 (0,0)	1566,0 (16,1)	1555,2 (14,4)	1548,0 (0,0)	1548
B_5	1526,4 (28,8)	1512,0 (0,0)	1512,0 (0,0)	1540,8 (26,9)	1497,6 (48,8)	1440,0 (0,0)	1440
B_6	1980,0 (0,0)	1976,4 (7,2)	1954,8 (24,4)	1972,8 (14,4)	1976,4 (7,2)	1972,8 (8,8)	1926
B_7	1612,8 (21,6)	1638,0 (44,1)	1602,0 (0,0)	1616,4 (41,7)	1602,0 (11,4)	1620,0 (36,0)	1566
B_8	1587,6 (40,1)	1584,0 (0,0)	1584,0 (0,0)	1576,8 (14,4)	1584,0 (0,0)	1584,0 (0,0)	1530
B_9	1980,0 (0,0)	1976,4 (7,2)	1962,0 (11,4)	1980,0 (0,0)	1980,0 (0,0)	1983,6 (17,6)	1944
B_{10}	2073,6 (48,8)	2048,4 (28,8)	2034,0 (27,9)	2025,0 (53,2)	2011,5 (51,5)	1951,2 (14,4)	1944
B_{11}	2340,0 (32,2)	2282,4 (63,8)	2160,0 (0,0)	2358,0 (36,0)	2282,4 (36,7)	2160,0 (0,0)	2160
B_{12}	1850,4 (43,2)	1836,0 (45,5)	1850,4 (43,2)	1908,0 (32,2)	1836,0 (22,8)	1821,6 (28,8)	1764
B_{13}	2095,2 (69,8)	2095,2 (26,9)	2016,0 (11,4)	2102,4 (47,5)	2070,0 (27,9)	2030,4 (28,8)	1998
B_{14}	2547,0 (83,0)	2361,6 (55,1)	2286,0 (31,2)	2476,8 (100,3)	2371,5 (14,9)	2268,0 (78,9)	2160
B_{15}	3342,0 (123,3)	3157,2 (78,5)	3024,0 (70,9)	3270,0 (111,3)	3199,5 (72,4)	3051,0 (20,1)	2916
B_{16}	2749,5 (56,0)	2623,5 (26,6)	2570,4 (43,2)	2768,4 (52,7)	2682,0 (72,0)	2491,2 (52,9)	2448
B_{17}	3078,0 (18,0)	2911,5 (46,5)	2740,5 (60,2)	3114,0 (29,4)	2840,4 (40,1)	2763,0 (92,2)	2664
B_{18}	2992,5 (74,6)	2898,0 (36,0)	2808,0 (50,9)	2980,8 (14,4)	2854,8 (38,8)	2833,2 (64,0)	2736
B_{19}	4392,0 (116,7)	4059,0 (99,0)	3864,0 (80,9)	4098,0 (97,9)	3987,0 (45,0)	3820,5 (19,6)	3798
B_{20}	3172,5 (26,6)	3085,2 (55,3)	2944,8 (57,6)	3360,0 (33,9)	3168,0 (50,9)	2973,6 (66,8)	2844
B_{21}	3498,0 (174,1)	3264,0 (33,9)	3168,0 (19,7)	3427,2 (140,6)	3208,5 (41,0)	3146,4 (44,7)	3096
B_{22}	3757,5 (66,6)	3438,0 (54,0)	3252,0 (17,0)	3702,0 (170,3)	3546,0 (53,0)	3240,0 (0,0)	3240
B_{23}	8838,0 (450,4)	8406,0 (389,0)	8068,5 (359,2)	8424,0 (158,0)	7740,0 (85,0)	8160,0 (225,9)	7650

(4) - I_2	Strategie - S1 (=S2)			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2736,0 (87,3)	2592,0 (64,9)	2419,2 (26,9)	2376
B_1	1080,0 (0,0)	1080,0 (0,0)	1080,0 (0,0)	1080
B_2	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188
B_3	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188
B_4	1134,0 (22,8)	1130,4 (17,6)	1116,0 (0,0)	1116
B_5	1267,2 (26,9)	1267,2 (26,9)	1238,4 (28,8)	1224
B_6	1544,4 (7,2)	1544,4 (7,2)	1537,2 (8,8)	1530
B_7	1206,0 (30,1)	1166,4 (7,2)	1166,4 (7,2)	1152
B_8	1180,8 (35,3)	1152,0 (0,0)	1152,0 (0,0)	1152
B_9	1548,0 (0,0)	1548,0 (0,0)	1537,2 (8,8)	1530
B_{10}	1663,2 (24,4)	1634,4 (13,5)	1598,4 (26,5)	1548
B_{11}	1930,5 (84,8)	1872,0 (50,9)	1782,0 (66,4)	1728
B_{12}	1454,4 (28,8)	1411,2 (14,4)	1411,2 (14,4)	1404
B_{13}	1706,4 (28,8)	1612,8 (37,1)	1562,4 (41,7)	1512
B_{14}	1989,0 (63,0)	1920,0 (67,9)	1810,8 (55,3)	1728
B_{15}	2970,0 (71,4)	2673,0 (63,0)	2559,6 (64,8)	2484
B_{16}	2224,8 (86,4)	2232,0 (44,1)	2174,4 (28,8)	2160
B_{17}	2632,5 (92,1)	2466,0 (38,2)	2354,4 (41,7)	2304
B_{18}	2624,4 (69,6)	2484,0 (54,6)	2376,0 (32,2)	2340
B_{19}	3528,0 (103,9)	3744,0 (85,4)	3406,5 (91,2)	3294
B_{20}	2664,0 (50,9)	2674,8 (69,8)	2512,8 (57,6)	2484
B_{21}	3132,0 (144,0)	2902,5 (156,6)	2682,0 (48,3)	2592
B_{22}	3312,0 (111,0)	3060,0 (81,8)	2930,4 (50,1)	2844
B_{23}	9108,0 (487,9)	7452,0 (122,5)	7718,4 (366,7)	7182

(4) - I_3	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2671,2 (58,7)	2624,4 (58,5)	2534,4 (84,0)	2893,5 (32,1)	2718,0 (82,5)	2592,0 (84,4)	2448
B_1	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188,0 (0,0)	1188
B_2	1310,4 (17,6)	1310,4 (17,6)	1296,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1224
B_3	1296,0 (0,0)	1267,2 (14,4)	1260,0 (0,0)	1238,4 (28,8)	1224,0 (0,0)	1224,0 (0,0)	1224
B_4	1224,0 (0,0)	1224,0 (0,0)	1224,0 (0,0)	1180,8 (35,3)	1152,0 (0,0)	1152,0 (0,0)	1152
B_5	1231,2 (14,4)	1224,0 (0,0)	1224,0 (0,0)	1195,2 (35,3)	1152,0 (0,0)	1152,0 (0,0)	1152
B_6	1584,0 (0,0)	1584,0 (0,0)	1580,4 (7,2)	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1566
B_7	1288,8 (14,4)	1281,6 (17,6)	1296,0 (0,0)	1274,4 (17,6)	1281,6 (17,6)	1281,6 (17,6)	1260
B_8	1299,6 (28,8)	1288,8 (42,0)	1245,6 (28,8)	1288,8 (47,8)	1267,2 (42,0)	1274,4 (48,8)	1224
B_9	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584,0 (0,0)	1584
B_{10}	1659,6 (26,5)	1659,6 (26,5)	1638,0 (0,0)	1681,2 (31,4)	1666,8 (37,1)	1548,0 (0,0)	1548
B_{11}	2026,8 (50,4)	1832,4 (43,2)	1821,6 (58,5)	1944,0 (71,1)	1908,0 (62,4)	1828,8 (42,0)	1728
B_{12}	1591,2 (61,9)	1533,6 (17,6)	1526,4 (17,6)	1598,4 (48,8)	1540,8 (14,4)	1504,8 (35,3)	1440
B_{13}	1749,6 (43,2)	1699,2 (35,3)	1677,6 (17,6)	1741,5 (39,0)	1677,6 (28,8)	1656,0 (39,4)	1584
B_{14}	2016,0 (106,0)	2074,5 (71,3)	1956,0 (44,9)	2166,0 (22,4)	2037,6 (110,6)	1972,8 (111,9)	1836
B_{15}	2970,0 (74,2)	2646,0 (40,2)	2704,5 (51,5)	3105,0 (63,0)	2821,5 (71,3)	2568,0 (17,0)	2556
B_{16}	2376,0 (82,1)	2318,4 (48,8)	2314,8 (35,3)	2541,6 (111,2)	2383,2 (61,9)	2340,0 (106,8)	2196
B_{17}	2649,6 (94,8)	2538,0 (93,9)	2400,0 (86,1)	2646,0 (88,2)	2498,4 (87,7)	2430,0 (38,9)	2322
B_{18}	2718,0 (101,0)	2624,4 (36,7)	2480,4 (64,8)	2749,5 (60,2)	2548,8 (60,9)	2462,4 (36,7)	2394
B_{19}	3843,0 (99,0)	3645,0 (131,4)	3379,5 (76,8)	3834,0 (58,3)	3666,0 (37,0)	3444,0 (74,0)	3276
B_{20}	2764,8 (73,4)	2745,0 (46,8)	2620,8 (14,4)	3132,0 (106,9)	2948,4 (98,2)	2692,8 (52,9)	2592
B_{21}	3105,0 (130,7)	2862,0 (74,2)	2781,0 (45,0)	3078,0 (96,3)	2973,6 (80,8)	2776,5 (44,8)	2718
B_{22}	3444,0 (110,3)	3204,0 (132,3)	2934,0 (40,2)	3366,0 (132,3)	3150,0 (126,0)	2995,2 (97,7)	2844
B_{23}	8046,0 (498,0)	8082,0 (621,7)	7855,2 (549,0)	7812,0 (380,2)	8064,0 (214,6)	7878,0 (169,7)	7056

(4b) - I_3	Strategie - S1			Strategie - S2			
Bestell menge	20 av (dev)	100 av (dev)	1000 av (dev)	20 av (dev)	100 av (dev)	1000 av (dev)	best
B_0	2940,0 (61,2)	2851,2 (42,0)	2664,0 (60,2)	3372,0 (74,0)	2980,8 (83,3)	2721,6 (92,8)	2556
B_1	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_2	1324,8 (57,6)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_3	1310,4 (28,8)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296,0 (0,0)	1296
B_4	1260,0 (0,0)	1260,0 (0,0)	1260,0 (0,0)	1252,8 (26,9)	1224,0 (0,0)	1224,0 (0,0)	1224
B_5	1296,0 (22,8)	1288,8 (14,4)	1296,0 (0,0)	1252,8 (35,3)	1224,0 (0,0)	1224,0 (0,0)	1224
B_6	1720,8 (35,3)	1692,0 (0,0)	1663,2 (35,3)	1706,4 (28,8)	1548,0 (0,0)	1548,0 (0,0)	1548
B_7	1324,8 (35,3)	1296,0 (0,0)	1296,0 (0,0)	1339,2 (35,3)	1310,4 (28,8)	1310,4 (28,8)	1296
B_8	1360,8 (14,4)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1368,0 (0,0)	1332
B_9	1764,0 (0,0)	1735,2 (35,3)	1670,4 (43,2)	1735,2 (35,3)	1706,4 (28,8)	1706,4 (28,8)	1584
B_{10}	1828,8 (80,2)	1713,6 (43,2)	1692,0 (0,0)	1821,6 (92,8)	1728,0 (72,0)	1692,0 (0,0)	1692
B_{11}	2138,4 (132,0)	2044,8 (69,8)	1951,2 (35,3)	2138,4 (146,9)	1987,2 (14,4)	1922,4 (48,8)	1872
B_{12}	1706,4 (28,8)	1670,4 (28,8)	1656,0 (0,0)	1605,6 (84,0)	1569,6 (70,5)	1512,0 (0,0)	1512
B_{13}	1893,6 (62,8)	1792,8 (26,9)	1807,2 (26,9)	1908,0 (25,5)	1800,0 (39,4)	1807,2 (42,0)	1728
B_{14}	2448,0 (44,1)	2112,0 (74,0)	2142,0 (59,7)	2364,0 (84,9)	2352,0 (61,2)	2142,0 (59,7)	2016
B_{15}	3312,0 (128,1)	3096,0 (145,3)	2961,0 (133,2)	3480,0 (177,2)	2995,2 (131,6)	2865,6 (87,0)	2700
B_{16}	2656,8 (80,2)	2512,8 (61,9)	2484,0 (45,5)	2642,4 (66,8)	2548,8 (110,1)	2476,8 (89,3)	2304
B_{17}	2901,6 (48,8)	2772,0 (62,4)	2682,0 (120,7)	2916,0 (108,0)	2793,6 (123,9)	2707,2 (110,1)	2520
B_{18}	2934,0 (90,0)	2800,8 (127,6)	2714,4 (108,2)	2997,0 (99,8)	2750,4 (98,2)	2691,0 (89,5)	2592
B_{19}	4428,0 (0,0)	3996,0 (117,6)	3924,0 (29,4)	4320,0 (211,9)	4320,0 (157,3)	3981,6 (141,5)	3744
B_{20}	3196,8 (86,4)	2865,6 (70,5)	2786,4 (43,2)	3528,0 (125,6)	3072,0 (111,3)	2865,6 (70,5)	2700
B_{21}	3366,0 (194,7)	3211,2 (83,3)	3117,6 (103,3)	3333,6 (177,2)	3168,0 (50,9)	3074,4 (48,8)	2952
B_{22}	3660,0 (122,4)	3564,0 (147,0)	3249,0 (69,1)	3600,0 (229,6)	3468,0 (290,0)	3396,0 (44,9)	3060
B_{23}	9756,0 (324,0)	8820,0 (147,9)	8832,0 (138,9)	9072,0 (327,9)	9324,0 (290,0)	8952,0 (250,0)	8604

Danksagung

Hiermit möchte ich mich bei allen Personen bedanken, die direkt oder indirekt einen positiven Einfluss auf die Entstehung dieser Arbeit hatten. Insbesondere richtet sich mein Dank an

- **Prof. Dr. Ingo Morgenstern**, der mein Interesse für das Gebiet der Optimierung weckte und durch das Angebot, die vorliegende Arbeit durchzuführen und auch außerhalb der Sprechzeiten zu betreuen, dafür sorgte, dass ich mein Wissen auf diesem Gebiet intensivieren konnte,

- **Dr. Heinrich Braun**, Development Manager der Arbeitsgruppe **SAP-APO-OPT** der SAP AG Walldorf, der mich mit der Durchführung des Projekts *Sequenzplanung in der Automobilindustrie* (vgl. dazu v.a. Kapitel 10) betraute und stets gewissenhaft betreute,

- die **Arbeitsgruppe** von **Prof. Dr. Ingo Morgenstern** an der Universität Regensburg, besonders **Wolfgang Feil**, **Johannes Schneider** und **Anja Ebersbach** für die Unterstützung in wissenschaftlichen Fragen und die nötige Zerstreuung zwischendurch, sei es bei einer Tasse Kaffee oder am Ende eines Sicherungsseils beim Klettern,

- die **Arbeitsgruppe SAP-APO-OPT** der SAP AG Walldorf, für die gute Zusammenarbeit und die abwechslungsreiche Feierabendgestaltung, besonders **Jens Gottlieb** für die Zusammenarbeit in Bezug auf Veröffentlichungen im Bereich der Sequenzplanung, **Andreas Sandner** für die lehrreichen Diskussionen und **Thomas Dehoust** für den abendlichen Stressabbau mittels virtueller Konfliktlösung,

- **Herrn Prof. Dr. Kallrath** und **Frau Dr. Anna Schreieck** der BASF AG Ludwigshafen, die mir die Daten für die Problemstellungen der Kapitel 6, 7 und 9 zur Verfügung stellten,

- **Herrn Fischer**, **Herrn Mayer** und **Herrn Weiss** der Infineon AG Regensburg für Zusammenarbeit im Projekt *Erprobung einer Optimierungsmethode zur Verbesserung der Feinsteuerung in der Fertigung*,

- **Herrn Helmut Mayer**, **Herrn Rudolph** und **Herrn Dr. Brinkmann** der BMW AG München für die Möglichkeit, an einem Wettbewerb zur Sequenzplanung (*Erstellung von Montage-Reihenfolgen*) in der Automobilindustrie zum Vergleich kommerzieller Software teilzunehmen,

- **meinen Eltern** und meiner Freundin **Cordula**, die mich jederzeit unterstützten, ernährten und für den nötigen Ausgleich zwischen Arbeit und Freizeit sorgten.

Abbildungsverzeichnis

Abbildung 2.1: Die Bedingung der *detailed balance* ist nicht erfüllt! ..... **(18)**

Abbildung 2.2: Die Bedingung der *Ergodizität* ist nicht erfüllt! k_i bezeichnet man auch als ‚Zuckerhut'. ..... **(19)**

Abbildung 2.3: *Golf hole* k_i. ..... **(19)**

Abbildung 2.4: Ist das System im *golf hole* bei k_i gefangen, so werden andere, evtl. bessere, Zustände nicht mehr erreicht. ..... **(19)**

Abbildung 2.5: Näherung des exponentiellen Verhaltens der Übergangswahrscheinlichkeit von **SA** durch eine Stufenfunktion bei **TA**. ..... **(20)**

Abbildung 2.6: Temperaturverlauf des einfachen **TA** und zweier bouncing Varianten, bei konstantem $T_B(i)$ und bei logarithmisch gesenktem: $T_B(i) = \alpha * T_B(i-1)$ mit α=0,8. ..... **(25)**

Abbildung 3.2.1: Beispiel für die Bestimmung der Brutto-Dauer **b)** eines Produktionsvorgangs P mit gegebener Netto-Dauer **a)**. ..... **(40)**

Abbildung 3.2.2: Die Produktionsvorgänge werden auf der Primärressource semiaktiv geplant. Produktionsvorgang P_3 wird erst nach P_2 geplant, obwohl zu einem früheren Zeitpunkt (zwischen P_1 und P_2) noch ausreichend Kapazität zur Verfügung steht. ..... **(41)**

Abbildung 3.2.3: Die Produktionsvorgänge werden auf der Sekundärressource aktiv geplant. Produktionsvorgang P_1 wird zu seinem frühesten Zeitpunkt geplant. ..... **(41)**

Abbildung 3.2.4: Zu sehen ist die Entnahme (**a**,**c**) und Lagerung (**b**,**d**) eines Materials durch einen Produktionsvorgang P_1 im *batch* (**a**,**b**) bzw. im *continuous* (**c**,**d**) Betrieb. ..... **(42)**

Abbildung 3.2.5: Hier ist ein Beispiel einer Befüllung eines Silos im *batch* zu sehen. **a)** zeigt den Füllstand des Silos vor der Planung des Produktionsvorgangs, **c)** nach der Planung. Ab dem Startzeitpunkt des Produktionsvorgangs wird eine bestimmte Menge des Materials verbraucht (**a)**, **b)**). .....**(43)**

Abbildung 3.2.6: Zyklus bei den 3 Aktivitäten A_1, A_2 und A_3, die durch die Aktivitätslinks $A_1(1) \xrightarrow{ES,0} A_2(1)$, $A_2(1) \xrightarrow{ES,0} A_3(1)$ und $A_3(1) \xrightarrow{ES,0} A_1(1)$ miteinander verbunden sind. A_2 ist z.B. Vorgänger und Nachfolger bzgl. A_3. ....**(49)**

Abbildung 3.2.7: Oberfläche eines Programms zur Anzeige eines Gantt Diagramms einer Lösung. Vier Bereiche sind vorhanden: Liste der Primärressourcen (**a)**), Liste der Lager/Silo und Multiressourcen (**b)**), zeitliche Lage der einzelnen Modi (**c)**) und der zeitliche Verlauf des Lagerfüllstandes (**d)**). .....**(55)**

Abbildung 3.2.8: 2 der 6 *semiaktiven* Pläne sind zu sehen. Während Plan **a)** eine minimale Durchlaufzeit hat, ist bei Plan **b)** die Rüstzeit (schattiert) minimiert. Plan **a)** ist bei diesem Beispiel der einzige *aktive* Plan.....**(59)**

Abbildung 3.2.9: Suchraum: Die Konfigurationen sind hier als Eckpunkte in einem quadratischen Gitter dargestellt. Die Kreise zeigen Konfigurationen, die die Nebenbedingung verletzen. Die Linien entsprechen den Nachbarschaften (durchgezogene: N1, gestrichelt: N2). Es ist nicht möglich mit N1 von k_1 zu k_2 zu gelangen, falls die Konfigurationen, die die Nebenbedingung verletzen verboten sind. Diese Inselbildung durch die Nebenbedingung lässt sich durch N2 oder eine weiche Definition der Nebenbedingung mithilfe von Straftermen verhindern. .....**(60)**

Abbildung 4.1: Zu sehen ist ein zweidimensionales kubisches Gitter mit 3*3 Positionen. Aufgrund der periodischen Randbedingung ist Position 1 nicht nur der nächste Nachbar der Positionen 2 und 4, sondern auch der Positionen 3 und 7. .....**(64)**

Abbildung 4.2: Einer der möglichen Grundzustände eines dreidimensionalen $\pm J$ EA-Modells mit periodischen Randbedingungen, ohne äußeres Magnetfeld, mit 2*2*2 Ising-Spins (der gefüllte Kreis entspricht +1, der leere -1) und Wechselwirkungen zwischen nächsten Nachbarn (die durchgezogene Linie entspricht +J, die gestrichelte -J). Der Spin an Position **X** ist frustriert, er kann nicht alle Wechselwirkungen zu seinen nächsten Nachbarn gleichzeitig erfüllen. Beide Einstellungen des Spins **X** führen zur gleichen Energie. .....**(64)**

Abbildung 4.3: Es sind die Messgrößen Energie (**E**), Magnetisierungsbetrag (|**M**|), spezifische Wärme (**C**) und Suszeptibilität (**X**) in Abhängigkeit der Temperatur (**T**) für ein ferromagnetisches System (0% negative Wechselwirkungen) mit 32*32*32 Spins abgebildet. ..... **(65)**

Abbildung 4.4: Es ist die Abhängigkeit der beiden Messgrößen Energie (**E**) und Magnetisierungsbetrag (|**M**|) der jeweils besten gefundenen Lösung für verschiedene Anteile an negativen Wechselwirkungen (**%-J**) für ein System aus 32*32*32 Spins zu sehen. ..... **(66)**

Abbildung 4.5: Zu sehen ist ein Vergleich verschiedener Nachbarschaften zur Lösung des dreidimensionalen ± J EA-Modells mit verschiedener Anzahl an negativen Wechselwirkungen (0%, 50%, 100%) und 16*16*16 Spins..... **(67)**

Abbildung 4.6: Es sind die Messgrößen Energie (**E**), Magnetisierungsbetrag (|**M**|), spezifische Wärme (**C**) und Suszeptibilität (**X**) in Abhängigkeit der Temperatur (**T**) für ein ferromagnetisches System (0% negative Wechselwirkungen) mit 50 Spins eines ± J SK-Modells abgebildet. ..... **(68)**

Abbildung 4.7: Es sind die Messgrößen Energie (**E**), Magnetisierungsbetrag (|**M**|), spezifische Wärme (**C**) und Suszeptibilität (**X**) in Abhängigkeit der Temperatur (**T**) für ein antiferromagnetisches System (100% negative Wechselwirkungen) mit 50 Spins eines ± J SK-Modells abgebildet. ..... **(69)**

Abbildung 4.8: Zu sehen ist ein Vergleich verschiedener Nachbarschaften zur Lösung des ± J SK-Modells mit 50% negativen Wechselwirkungen mit 200 Spins. ..... **(70)**

Abbildung 4.9: In dieser Abbildung sind die Energie (**E**), Ziel- (**Z1**) und Straffunktionswerte (**S2**, **S4**) und der Ordnungsparameter (**O**) in Abhängigkeit von der Temperatur (**T**) in der linken Spalte zu sehen und die spezifische Wärme (**C**) und die einzelnen Suszeptibilitäten (**X**) rechts. Die Simulationen zu **MT6** wurden dabei mit Nachbarschaft (2) durchgeführt. ..... **(75)**

Abbildung 4.10: In dieser Abbildung sind die Energie (**E**), Ziel- (**Z1**) und Straffunktionswerte (**S2**, **S4**) und der Ordnungsparameter (**O**) in Abhängigkeit von der Temperatur (**T**) in der linken Spalte zu sehen und die spezifische Wärme (**C**) und die einzelnen Suszeptibilitäten (**X**) rechts. Die Simulationen zu **MT6** wurden dabei mit Nachbarschaft (3) durchgeführt. ..... **(76)**

Abbildung 4.11: Die beiden Lösungen **a)** und **b)** weisen die gleiche Durchlaufzeit auf. Es ist nicht möglich, beide Produktionsvorgänge zur gleichen Zeit auf der Maschine zu starten (Frustration); entartete Zustände sind die Folge. .....**(77)**

Abbildung 4.12: Neben der Energie (**E**) und der spezifischen Wärme (**C**) sind hier noch die Durchlaufzeit (**M**) und der Ordnungsparameter (**O**) zusammen mit deren Suszeptibilitäten (**X**) gegeben. .....**(78)**

Abbildung 4.13: 3 Produktionsvorgänge sind auf 2 Maschinen zu planen. Zwischen P_2 und P_3 ist eine Rüstzeit nötig. Die beiden Lösungen **a)** und **b)** unterscheiden sich zwar in der Durchlaufzeit und der Rüstzeit, haben jedoch die gleichen Kosten, falls die Summe aus Durchlaufzeit und Rüstzeit minimiert werden soll. .....**(79)**

Abbildung 4.14: Die Energie (**E**) der Simulation der Problemstellung Bench01 setzt sich zusammen aus den Verspätungen (**D**), der Durchlaufzeit (**M**) und der Rüstzeit (**S**). Neben diesen Größen ist noch der Ordnungsparameter (**O**), die spezifische Wärme (**C**) und die Suszeptibilitäten (**X**) gegen die Temperatur abgebildet. .....**(81)**

Abbildung 4.15: Zu sehen sind die Observablen der Simulation einer vereinfachten Instanz der Problemstellung Bench01. Die Energie (**E**) setzt sich zusammen aus den Verspätungen (**D**), der Durchlaufzeit (**M**) und der Rüstzeit (**S**). Neben diesen Größen ist noch der Ordnungsparameter (**O**), die spezifische Wärme (**C**) und die Suszeptibilitäten (**X**) gegen die Temperatur abgebildet.....**(82)**

Abbildung 5.1: Es wurden jeweils 100 Simulationen mit den Verfahren **SA** und **TA** für verschiedene Anzahl Sweeps pro Temperaturschritt (100, 500, 1000, 5000) durchgeführt. Zu sehen ist jeweils der Prozentsatz (**%**) der Lösungen, die eine Durchlaufzeit $\leq$ **M** haben.....**(86)**

Abbildung 5.2: Simulation mit **TA**. Es wurden Simulationen mit verschiedener Anzahl Sweeps pro Temperaturschritt (500, 1000, 5000) durchgeführt. Bei den Simulationen für 500 (bzw. 1000) Sweeps wurden dabei von jeweils 10 (bzw. 5) aufeinanderfolgenden Lösungen die besten genommen, so dass jede übernommene Lösung etwa 100 Sekunden Rechenzeit zur Verfügung hatte. Zu sehen ist jeweils der Prozentsatz (**%**) aus jeweils 100 übernommenen Lösungen, die eine Durchlaufzeit $\leq$ **M** haben. .....**(87)**

Abbildung 5.3: Ein optimaler Plan des MT10 Problems mit einer Durchlaufzeit von 930. ...**(87)**

Abbildung 5.4: Ein Plan für 50 Aufträge mit einer Durchlaufzeit von 3238. Die *bottleneck* Ressource M4 ist ab einem bestimmten Zeitpunkt (83) zu 100% ausgelastet. ..... **(89)**

Abbildung 6.1: Profile der benötigten Arbeitskräfte der Aktivitäten A_{1i}, A_{2i}, A_{3i} und A_{4i} .. **(95)**

Abbildung 6.2: Vorgänger-Nachfolgerbeziehung der Aktivitäten A_{1i}, A_{2i}, A_{3i} und A_{4i} ..... **(95)**

Abbildung 6.3: Benötigte Rechenzeit in Sekunden auf einem Pentium mit 650MHz für die Durchführung von 100000 Sweeps für die verschiedenen Instanzen. ..... **(99)**

Abbildung 7.1: Struktur des Fertigungsprozesses. ..... **(105)**

Abbildung 7.2: Beispiel eines Fertigungsprozesses mit 4 Stufen. ..... **(106)**

Abbildung 7.3: Verschiedene Möglichkeiten bei der Kombination von Ressourcen aufeinanderfolgender Stufen: **(a)** ein Vorgänger – ein Nachfolger, **(b)** ein Vorgänger – mehrere Nachfolger, **(c)** mehrere Vorgänger – ein Nachfolger, **(d)** mehrere Vorgänger – mehrere Nachfolger. ..... **(106)**

Abbildung 8.1: Struktur des Fertigungsprozesses. ..... **(115)**

Abbildung 8.2: Rezept zur Produktion eines Endproduktes ($1 \le a \le 4, 6 \le b \le 9, 1 \le x \le 10$). **(116)**

Abbildung 8.3: Zu sehen ist eine Lösung der Instanz 0 mit minimaler Rüstzeit. Rüstzeit 360 Minuten, Durchlaufzeit 6423 Minuten, Lieferzeitpunkte sind verletzt! ..... **(119)**

Abbildung 8.4: Zu sehen ist eine Lösung der Instanz 0 mit minimaler Durchlaufzeit (2051). Rüstzeiten und Lieferzeitpunkte wurden ignoriert. ..... **(120)**

Abbildung 8.5: Es werden die durchschnittlichen Ergebnisse von Simulationen mit den 5 verschiedenen Nachbarschaften gezeigt (linke Spalte). In der rechten Spalte sind die durchschnittlichen Ergebnisse der Simulationen mit allen Nachbarschaften (**avA**), den besseren Nachbarschaften (**avB**) und der adaptiven Nachbarschaft (**AD**) zu sehen. ..... **(124)**

Abbildung 8.6: Zu sehen sind die durchschnittlichen Lösungen für die Optimierung mit verschiedenen Lieferzeitpunkten. ....**(125)**

Abbildung 8.7: Pareto-Fronten verschiedener Rechenzeiten (100-10000 Sweeps pro Temperaturschritt) für Instanz 0. ....**(125)**

Abbildung 8.8: Abgebildet ist eine Lösung einer Optimierung der Kriterien **M+S** mit einer Durchlaufzeit von 2122 Minuten und einer Rüstzeit von 420 Minuten. ....**(126)**

Abbildung 8.9: Während bisher die Maschinen C_1, C_2, C_3 im *batch mode* (**a)**) arbeiteten, wird jetzt im *continuous mode* das Material abgegeben (**b)** ; Maschine C_1 und C_2) oder aufgenommen (**c)** ; Maschine C_3)....**(132)**

Abbildung 8.10: Die besten Ergebnisse der verschiedenen Verfahren bei kurzen Rechenzeiten (<=10 Minuten auf einem Pentium mit 200 MHz). ....**(140)**

Abbildung 8.11: Die besten Ergebnisse der verschiedenen Verfahren bei langen Rechenzeiten (>10 Minuten auf einem Pentium mit 200 MHz). ....**(140)**

Abbildung 9.1: Graphische Darstellung der Problemstellung. ....**(145)**

Abbildung 9.2: Mögliche Materialflüsse durch einen Modus: **a)** single input - single output, **b)** multi input - single output, **c)** single input - multi output, **d)** multi input - multi output, **e)** multi variable input - multi variable output. Bei multi input/output können statt den gezeigten 2 Materialien auch mehrere verbraucht/produziert werden. ....**(149)**

Abbildung 9.3: Zu sehen sind die jeweils besten Durchlaufzeiten aus den Literaturergebnissen und den beiden Strategien bei der Planung mit **SA**. ....**(155)**

Abbildung 9.4: Zu sehen sind die jeweils besten Durchlaufzeiten aus der Literatur für die Verfahren **TGB-VV** und **B+BS** (bei begrenzter Rechenzeit) und die durchschnittlichen Durchlaufzeiten des Verfahrens **SA** mit Strategie **S1** mit 20 und 1000 Sweeps pro Temperaturschritt. ....**(156)**

Abbildung 9.5: Zu sehen ist die beste Lösung für Aufgabe (4) mit Initialzuständen der Silos I_2 mit einer Durchlaufzeit von 2232 Minuten. ....**(157)**

Abbildung 9.6: Zu sehen sind die jeweils besten Durchlaufzeiten aus den Literaturergebnissen und den beiden Strategien bei der Planung mit **SA**. ....**(157)**

Abbildung 9.7: Zu sehen sind die jeweils besten Durchlaufzeiten aus der Literatur für die Verfahren **TGB-VV** und **B+BS** (bei begrenzter Rechenzeit) und die durchschnittlichen Durchlaufzeiten des Verfahrens **SA** mit Strategie **S2** mit 20 und 1000 Sweeps pro Temperaturschritt. ..... **(158)**

Abbildung 10.1: Energie und Wärmekapazität der Problemstellung mit linearer Definition der Kostenfunktion bei Simulationen mit **SA** und **TA**..... **(170)**

Abbildung 10.2: Es ist der temperaturabhängige Verlauf der summierten Kostenfunktionen der Nebenbedingungen zu sehen. ..... **(170)**

Abbildung 10.3: Ordnungsparameter (**Overlap**) und Suszeptibilität der Problemstellung mit linearer Definition der Kostenfunktion bei Simulationen mit **SA** und **TA**. ..... **(171)**

Abbildung 10.4: Ergebnisse der Verfahren **LSDED**, **IterDHU** und **AntDHU** für vier Instanzen der zweiten Gruppe. ..... **(177)**

Abbildung 10.5: Durchschnittliche Ergebnisse für die verschiedenen Verfahren für verschiedene Rechenzeitgrenzen. Die Rechenzeit bezieht sich dabei auf einen Pentium mit 600MHz. ..... **(179)**

Abbildung A.1: Flussdiagramm eines *Sweeps*. ..... **(187)**

Abbildung A.2: Flussdiagramm eines *Monte Carlo* Verfahrens..... **(188)**

Abbildung A.3: Flussdiagramm des *T-Bouncing*. ..... **(189)**

Abbildung A.4: Flussdiagramm des *Acceptance SA* (SA-Acc). ..... **(190)**

Tabellenverzeichnis

Tabelle 2.1: Benötigte Rechenzeit (***t***) für die vollständige Enumeration bei einer Problemstellung mit ***n*** binären Entscheidungsvariablen (z.B. Ising-Spins), falls die Berechnung einer Lösung 10^{-10} Sekunden benötigt. ..... **(9)**

Tabelle 5.1: Die Tabelle zeigt die erzeugten Lösungen für die Problemstellung mit n*10 Aufträgen. Die mit (*) gekennzeichneten Lösungen sind optimal., da sie entweder der unteren Grenze entsprechen, oder der optimalen Lösung des MT10X10 (930). ..... **(89)**

Tabelle 5.2: Für 100000 Sweeps benötigte Rechenzeit **t** in Sekunden für verschiedene Job-Shop **Instanzen** auf einem Pentium mit 650MHz ..... **(90)**

Tabelle 5.3: Es sind die durchschnittlichen Ergebnisse (**av**) mit ihrer Standardabweichung (**dev**) aus jeweils 10 Simulationen mit **1000** und **5000** Sweeps pro Temperaturschritt für die 40 Instanzen von Lawrence zusammen mit deren **Optima** gegeben. Zusätzlich sind noch Lösungen gegeben, die mit gespiegelten Instanzen (**5000 i**) mit 5000 Sweeps pro Temperaturschritt erhalten wurden. ..... **(91)**

Tabelle 6.1: Es werden 4 verschiedene Aktivitätstypen (**Akt**) benötigt, von denen jeweils **Num** Aktivitäten erzeugt werden. Jeder Aktivitätstyp ist gekennzeichnet durch seine Dauer (**d**) und das benötigte Kapazitätsprofil (**Profil**). ..... **(94)**

Tabelle 6.2: Kritischer Pfad (**cp**), optimales Ergebnis (**opt**) und untere Schranke (**lb**) für die einzelnen Instanzen. Zusätzlich sind noch die besten Ergebnisse verschiedener Verfahren aus der Literatur (**TS**, **PTS**, **AT**, **CP**, **LH**) zu sehen. Wurden die Ergebnisse mit gespiegelten Instanzen erzielt sind diese mit (*) markiert. ..... **(97)**

Tabelle 6.3: Gesamte Rechenzeiten der einzelnen Verfahren für verschiedene Größen der Instanzen. ..... **(97)**

Tabelle 6.4: Zu sehen sind die durchschnittlichen Durchlaufzeiten (Standardabweichung) von jeweils 20 Simulationen mit **GR** mit **5000** bzw. **25000** Sweeps für die originalen (**o**) und die gespiegelten (**i**) Instanzen mit 20 bis 30 Aufträgen. ..... **(99)**

Tabelle 6.5: Zu sehen sind die durchschnittlichen Durchlaufzeiten (Standardabweichung) von jeweils 20 Simulationen mit **SA** (**100**, **1000** und **5000** Sweeps pro Temperaturschritt) und **GR** (**25000** und **250000** Sweeps) für die originalen oder die gespiegelten Instanzen mit mehr als 40 Aufträgen. Ergebnisse von gespiegelten Instanzen sind mit (*) markiert. .....**(100)**

Tabelle 6.6: Gezeigt werden die besten Durchlaufzeiten (**bester**), die mit **SA** bei **sweeps** Sweeps pro Temperaturschritt, erzielt wurden. Zusätzlich sind noch die besten Resultate aus der Literatur (**bester(Lit)**) (siehe **Tabelle 7.2**) zu sehen. Ergebnisse von gespiegelten Instanzen sind mit (*) markiert. .....**(101)**

Tabelle 6.7: Es sind die durchschnittlichen Durchlaufzeiten (**av**) und deren Standardabweichung (**dev**) gemittelt über die 600 Instanzen zu sehen. Zusätzlich ist noch der prozentuale Abstand zur unteren Grenze (**%lb**), der oberen Grenze (**%ub**) und dem kritischen Pfad (**%cp**) gegeben. Die beiden letzten Spalten geben die Anzahl der Instanzen, bei denen bei jeder Simulation (**numAll**) und bei der besten aus 5 Simulationen (**numBest**) die obere Grenze erreicht wurden. .....**(103)**

Tabelle 6.8: Benötigte Rechenzeit für die einzelnen Verfahren für das Erstellen von 100000 Lösungen in Sekunden. Zu sehen ist der Mittelwert (**av**) aller 600 Instanzen, die minimal (**min**) und maximal (**max**) benötigte Rechenzeit.....**(104)**

Tabelle 7.1: Die **beste**n Lösungen für **CP** bei maximalen Rechenzeiten von 300 und 3600 Sekunden und für einen kombinierten Ansatz aus **MP** und **CP** bei einer maximalen Rechenzeit von 3600 Sekunden. Zusätzlich dazu sind die Zeiten (**t**) in Sekunden angegeben, in denen diese Lösungen gefunden wurden. .....**(108)**

Tabelle 7.2: Die **beste**n und **durch**schnittlichen Ergebnisse von **SA** für verschiedene Anzahl **Sweeps** pro Temperaturschritt bei minutengenauer (**SA1**) und sekundengenauer (**SA2**) Planung. .....**(109)**

Tabelle 7.3: Die verschiedenen Strategien, die bei den Simulationen verwendet werden, unterscheiden sich im frühesten Startzeitpunkt der Aufträge und deren vorgegebener Reihenfolge. .....**(110)**

Tabelle 7.4: Durchschnittliche Rechenzeiten in Sekunden für die Durchführung von 100000 Sweeps für verschiedene Anzahl an Tagespaketen auf einem Pentium mit 650 MHz. .....**(110)**

Tabelle 7.5: Durchlaufzeiten für die Simulation von 2-,3- und 4-Tages-Paketen. Zu sehen sind die besten Ergebnisse aus jeweils 10 Simulationen mit 50 (**_a**) bzw. 2000 (**_b**) Sweeps pro Temperaturschritt für die 4 Strategien **S1**–**S4**. Zum Vergleich sind noch die Ergebnisse der Strategien **Strat1**, **Strat2** und **Strat3** aus **[8.1]** zu sehen. Die Strategien **Strat1**-**Strat3** entsprechen dabei den Strategien **S2**-**S4**. .... **(111)**

Tabelle 7.6: Durchlaufzeiten für die Simulationen der 5-9-Tagespakete mit den Strategien **S1** und **S2** mit 50 (**_a**) bzw. 2000 (**_b**) Sweeps pro Temperaturschritt. Zusätzlich sind die Ergebnisse der Strategie **Strat1** aus **[8.1]** gegeben. ..... **(113)**

Tabelle 7.7: Durchlaufzeiten bei den Simulationen mit Strategien **S5** und **S6** mit 50 (**_a**) bzw. 2000 (**_b**) Sweeps pro Temperaturschritt. ..... **(114)**

Tabelle 8.1: Optimale Verteilung der 4 Rüstfamilien auf die Primärressourcen. ..... **(118)**

Tabelle 8.2: Ergebnisse für die Instanz 0 bei Modellierung mit *fixiertem Pegging* (**M2**) im Vergleich zu den Ergebnissen des Modells mit *variablem Pegging* (**M1**) aus **Kapitel 8.2**..... **(128)**

Tabelle 8.3: Vergleich der Ergebnisse der beiden Modellierungen **M1** und **M2** für Instanz 4. **(128)**

Tabelle 8.4: Vergleich der Ergebnisse der beiden Modellierungen **M1** und **M2** für Instanz 5. **(128)**

Tabelle 8.5: Ergebnisse für die Instanz 4 bei Modellierung mit *fixiertem Pegging* (**M2**) im Vergleich zu den Ergebnissen bei Erweiterung um zusätzliche Links zwischen Aktivitäten verschiedener Aufträge (**M2b**). ..... **(130)**

Tabelle 8.6: Vergleich der Ergebnisse der beiden Modellierungen **M2** und **M2b** für Instanz 5.**(130)**

Tabelle 8.7: Benötigte Rechenzeit in Sekunden für 100000 Sweeps auf einem Pentium mit 650MHz für die verschiedenen Instanzen und Modellierungen. ..... **(131)**

Tabelle 8.8: Durchschnittliche Ergebnisse zu den verschiedenen Kosteneinsparversuchen (**M3**, **M4**, **M5**) gegenüber der ursprünglichen Problemdefinition (**M1**). ..... **(134)**

Tabelle 8.9: Untere Schranken für die Durchlaufzeit für die verschiedenen Erweiterungen der Produktionskapazität. ..... **(134)**

Tabelle 8.10: Durchschnittliche Ergebnisse zu den verschiedenen erweiterten Produktionskapazitäten (**M6**, **M7**, **M8**) gegenüber der ursprünglichen Aufgabenstellung (**M1**). ..... **(135)**

Tabelle 8.11: Für die einzelnen Maschinen werden Pausen zur Durchführung von Wartungsarbeiten definiert. **(135)**

Tabelle 8.12: Je nach Produktionsstufe können Aktivitäten pausiert werden oder nicht. **(136)**

Tabelle 8.13: Durchschnittliche Ergebnisse bei Einführung von Wartungspausen (**M9**) gegenüber der ursprünglichen Problemdefinition (**M1**). **(136)**

Tabelle 8.14: Die besten Ergebnisse der Verfahren **TFP_SA**, **CP**, **B&B** und **GA**. Zusätzlich sind für **GA** noch durchschnittliche Ergebnisse angegeben **GA(av)**. Es wird dabei ein Rechenzeitlimit von 10 Minuten auf einem Pentium mit 200 MHz eingehalten. **(137)**

Tabelle 8.15: Die besten Ergebnisse der Verfahren **Vogl_SA** und **Vogl_TA** nach 350 Temperaturschritten mit je 50000 Moves. Dies entspricht 4-5 Stunden Rechenzeit auf einer UltraSparc mit 296 MHz. **(138)**

Tabelle 8.16: Die besten Ergebnisse der Verfahren **SA_N1**, **TA_N1** und **Bounce_N1** bei einer Rechenzeitgrenze von 524 Sekunden auf einem Pentium mit 200MHz. **(139)**

Tabelle 8.17: Die besten Ergebnisse der Verfahren **SA_N1**, **TA_N1** und **Bounce_N1**, erreicht innerhalb der Rechenzeitgrenze von 22 Stunden (**SA_N1**), 7 Stunden (**TA_N1**) und 16 Stunden (**Bounce_N1**) auf einem Pentium mit 200MHz. **(139)**

Tabelle 8.18: Die besten Ergebnisse von **SA_N2** mit einer Rechenzeitschranke von 1000 und 10000 Sweeps (entspricht 168 und 1680 Sekunden auf 650MHz). **(139)**

Tabelle 6.19: Es werden die jeweils besten Ergebnisse des **OmeGA** und **SA_N2** für eine veränderte Problemstellung gezeigt. Die Rechenzeit war bei **SA_N2** auf 1000 Sweeps pro Temperaturschritt beschränkt. **(141)**

Tabelle 8.20: Gezeigt werden die Ergebnisse der Simulationen mit einem *annealing* (**SA-Cool**) und mehreren konstanten Temperaturen (**T=...**). Zusätzlich sind noch die Ergebnisse eines **Greedy** Verfahren gegeben. **(142)**

Tabelle 8.21: Vergleich der Ergebnisse des bisher verwendeten **SA** (**SA-Cool**) mit denen der Simulation mit der neuen Parameterdefinition (**SA-Acc**). **(143)**

Tabelle 9.1: Initialzustände der Lager. **(147)**

Tabelle 9.2: Bestellmengen der Endprodukte. **(147)**

Tabelle 9.3: Ergebnisse der Verfahren **TGB** und **B+BS** für Aufgabe (3), I_3 und B_1-B_{22}. Bei **TGB** sind zusätzlich zu den Ergebnissen der ‚reinen' Heuristik (**H**) noch jene angegeben, die durch Hinzufügen eines Verbesserungsverfahrens (**VV**) erhalten wurden. Die Spalte (**beste / # opt**) zeigt die besten bekannten Ergebnisse aus **[9.3]**. Sind optimale Werte bewiesen worden, so wird dies durch (#) gekennzeichnet. Die Ergebnisse sind immer in der Zeiteinheit Minuten zu sehen. **(152)**

Tabelle 9.4: Benötigte Rechenzeit (PC Pentium II 266MHz) in Sekunden für **TGB** (**H**) und das Verbesserungsverfahren (**VV**) für die Lösung der Aufgabe (3), I_3 mit B_1–B_{22}. Die Rechenzeiten sind dabei Mittelwerte der in Paketen zusammengefassten Bestellmengen (aus **[9.6]**). **(152)**

Tabelle 9.5: Ergebnisse der Verfahren **TGB** und **B+BS** für Aufgabe (4b), I_3 und B_1-B_{22}. Bei **TGB** sind zusätzlich zu den Ergebnissen der ‚reinen' Heuristik (**H**) noch jene angegeben, die durch Hinzufügen eines Verbesserungsverfahrens (**VV**) erhalten wurden. Die Spalte (**beste**) zeigt die besten bekannten Ergebnisse aus **[9.3]**. Die Ergebnisse sind immer in der Zeiteinheit Minuten zu sehen. **(153)**

Tabelle 9.6: Benötigte Rechenzeit in Sekunden für **TGB** (PC Pentium II 266MHz) und **B+BS** (PC Pentium 800MHz) für die Lösung der Aufgabe (4b), I_3 mit B_1–B_{22}. Die Rechenzeiten sind dabei Mittelwerte der in Paketen zusammengefassten Bestellmengen. **(153)**

Tabelle 9.7: Ergebnis des Verfahrens **B+BS** für Aufgabe (4), I_2 und B_0. Benötigte Rechenzeit (**Zeit**) in Sekunden für **B+BS** (PC Pentium 800MHz). **(153)**

Tabelle 9.8: Zu sehen sind die besten Durchlaufzeiten für die verschiedenen Aufgabenstellungen (Aufgabe, Bestellmengen, Initialzustand der Silos), die mit dem Verfahren **SA** und den beiden Strategien innerhalb der Rechenzeitgrenze erhalten wurden. **(154)**

Tabelle 9.9: Benötigte Rechenzeit (PC Pentium mit 650MHz) in Sekunden für 100000 Sweeps mit dem Verfahren **SA** für die verschiedenen Aufgaben und Bestellmengen. Eine typische Simulation mit 200 Temperaturschritten und jeweils 1000 Sweeps benötigt also z.B. für Aufgabe (1) und Bestellmenge B_0 480 Sekunden. **(155)**

Tabelle 9.10: Zu sehen ist ein Vergleich der Durchlaufzeiten der Verfahren **SA** und **B+BS**. Bei **SA** ist die durchschnittliche Durchlaufzeit bei 20 und 1000 Sweeps pro Temperaturschritt und die beste gefundene Durchlaufzeit gegeben. **(156)**

Tabelle 9.11: Bei einer vereinfachten Planung der Problemstellung, die auf die Planung der Produktionsstufe 1 verzichtet, allerdings eine ‚Reparatur' des ungültigen optimierten Produktionsplans erforderte, wurden einige Ergebnisse noch verbessert. Diese sind hervorgehoben. .....**(159)**

Tabelle 9.12: Die besten Ergebnisse aus jeweils 10 Simulationen für die verschiedenen Aufgabenstellungen. Hier wurde mit einer fixierten Losgröße geplant. Die markierten Ergebnisse sind solche, die die gleiche Qualität aufweisen wie die besten aus **Tabelle 9.11**.....**(160)**

Tabelle 9.13: Verkaufspreise der einzelnen Produkte.....**(161)**

Tabelle 9.14: Erlös und Lösung bei den einzelnen Rahmenbedingungen für 1 Tag. .....**(161)**

Tabelle 9.15: Erlös und Lösung bei den einzelnen Rahmenbedingungen für 2 Tage.....**(161)**

Tabelle 9.16: Erlös und Lösung bei den einzelnen Rahmenbedingungen für 5 Tage.....**(161)**

Tabelle 10.1: Zu sehen sind die Ergebnisse der verschiedenen Heuristiken zu den verschiedenen Instanzen. Dabei sind die Mittelwerte aus jeweils 500 Simulationen und deren Standardabweichung gegeben. .....**(175)**

Tabelle 10.2: Erfolgsraten der Heuristiken für die Instanzen der ersten Gruppe. Darunter wird der Prozentsatz der Lösungen (von 500) verstanden, die alle Restriktionen erfüllt. .....**(175)**

Tabelle 10.3: Ergebnisse der Verfahren **Iter**, **Ant** und **LS** für die Instanzen der ersten Gruppe. Zu sehen sind jeweils die durchschnittlichen Ergebnisse von jeweils 100 Simulationen und deren Standardabweichung. .....**(176)**

Tabelle 10.4: Ergebnisse der Verfahren **Iter**, **Ant** und **LS** für die Instanzen der zweiten Gruppe. Zu sehen sind jeweils die durchschnittlichen Ergebnisse von jeweils 100 Simulationen und deren Standardabweichung. .....**(177)**

Tabelle 10.5: Vergleich der Ergebnisse mit denen aus **[10.3]**, **[10.7]**, **[10.11]**. **Best solution** gibt die besten Ergebnisse der in diesem Kapitel vorgestellten Verfahren an. Die Erfolgsrate (**SR**) gibt für jedes Verfahren den Prozentsatz der Lösungen an, die den besten Zielfunktionswert erreicht haben.....**(178)**

Tabelle 10.6: Anzahl der Moves die für die verschiedenen Nachbarschaftsoperatoren in 100 Sekunden auf einem Pentium mit 600MHz ausgeführt werden können. ..... **(179)**

Tabelle C.1: Auftragsliste (**Aufträge**) der einzelnen Tagespakete (**Tag**). ..... **(195)**

Tabelle C.2: Ressourcenabhängige Produktions- und Rüstzeiten in Minuten. ..... **(195)**

Tabelle C.3: Produktabhängige Produktionszeiten für die 1. und 2. Stufe und die Information, ob ein Produkt die 4. Stufe bei der Produktion überspringt. ..... **(196)**

Tabelle D.1: Liste der Produkte (**product ID**) mit ihren alternativen Maschinen (**machines**), ihren Produktionszeiten (**T_0 [h]**), den benötigten Kapazitäten (**number pool objects**) der Sekundärressourcen (**pool**), den benötigten Rohstoffen (**required preproducts**) und der zugehörigen Produktfamilie für die Rüstzeiten (**Family**)..... **(198)**

Tabelle D.2: Kapazität der Pools. ..... **(199)**

Tabelle D.3: Kapazität der speziellen Lager für bestimmte Produkte. ..... **(199)**

Tabelle D.4: Rüstzeiten der Maschinen (**Reactor**, **Feeder**, **Filter**) zwischen dem Vorgänger (**P**) und dem Nachfolger (**S**) aus den Produktfamilien (**F1**, **F2**, **F3**, **F4**). ..... **(199)**

Tabelle D.5: Verschiedene Instanzen der Problemstellung Bench01. Es ist jeweils eine Liste an zu fertigenden Produkten mit den zugehörigen Lieferzeitpunkten (in Minuten nach dem Produktionsstart) gegeben. **Instanz_0** ist die in der ursprünglichen Aufgabenstellung gegebene Instanz. ..... **(200)**

Tabelle D.6: Zu sehen sind die besten Lösungen für die verschiedenen Modelle und Instanzen, die während der Simulationen bei begrenzter Rechenzeit (siehe **Kapitel 8**) erhalten wurden. ..... **(201)**

Literaturverzeichnis

[1.1] S.Kiener, N.Maier-Scheubeck, M.Weiß, *Produktionsmanagement – Grundlagen der Produktionsplanung und –steuerung*, Oldenbourg, 1999.

[2.1] G. Reinelt, *The Traveling Salesman*, Lecture Notes in Computer Science 840, Springer, Berlin, 1994.

[2.2] J. Gottlieb, M. Puchta, C. Solnon, *A Study of Greedy, Local Search and Ant Colony Optimization Approaches for Car Sequencing Problems*, in G.R. Raidl et al. (eds.), *Applications of Evolutionary Computing*, 246-257, Lecture Notes in Computer Science, Volume 2611, Springer, 2003.

[2.3] F.F. Boctor, *Heuristics for scheduling projects with resource restrictions and several resource-duration modes*, International Journal of Production Research 31, 2547-2558.

[2.4] F.F. Boctor, *A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes*, European Journal of Operations Research 90, 349-361, 1996.

[2.5] E. Aarts, J.K. Lenstra, *Local Search in Combinatorial Optimization*, John Wiley & Sons, 1997.

[2.6] C.R. Reeves, *Modern Heuristic Techniques for Combinatorial Problems*, John Wiley & Sons, 1993.

[2.7] M. Puchta, *Physikalische Optimierungsverfahren angewendet auf Produktionsplanungsprobleme*, Diplomarbeit, Universität Regensburg, 1998.

[2.8] G. Dueck, *New Optimization Heuristics*: *The Great Deluge Algorithm and the Record-toRecord Travel*, Journal of Computational Physics, Vol. 104, pp. 86-92, 1993.

[2.9] G. Dueck, T. Scheuer, H.-M. Wallmeier, *Toleranzschwelle und Sintflut: Neue Ideen zur Optimierung*, Spektrum der Wissenschaft 3, 42, 1993.

[2.10] G. Dueck, *Das Sintflutprinzip – Ein Mathematik-Roman*, Springer, 2004.

[2.11] F. Glover, M. Laguna, *Tabu Search*, Kluwer Academic Publishers, 1997.

[2.12] J.H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[2.13] I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Herzog Verlag, Stuttgart, 1973.

[2.14] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Third Edition, Springer, 1996.

[2.15] D.E. Goldberg, *Genetic Algorithms in Search Optimization & Machine Learning*, Addison-Wesley, 1989.

[2.16] T. Stützle, *Applying Iterated Local Search to the Permutation Flow Shop Problem*, Technical Report AIDA-98-04, Darmstadt University of Technology, Computer Science Department, Intellectics Group.

[2.17] H.R. Lourenço, O. Martin, T. Stützle, *A Beginner's Introduction to Iterated Local Search,* in Proceedings of MIC 2001, pages 1-6, Porto, Portugal, 2001.

[2.18] H.R. Lourenço, O. Martin, T. Stützle, *Iterated Local Search*, in F. Glover and G. Kochenberger, editors, *Handbok of Metaheuristics*, pages 321-353, Kluwer Academic Publishers, Norwell, MA, 2002.

[2.19] S. Lin, B.W. Kernighan, *An effective heuristic algorithm for the traveling-salesman problem*, Operations Research, 21:498-516, 1971.

[2.20] J. Schneider, *Effiziente parallelisierbare physikalische Optimierungsverfahren*, Dissertation, Universität Regensburg, 1999.

[2.21] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, G. Dueck, *Record Breaking Optimization Results --- Using the Ruin & Recreate Principle*, Journal of Computational Physics 159, 139-171, 2000.

[2.22] S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, *Optimization by Simulated Annealing*, Science, Vol. 220, pp. 671-680, May 1983.

[2.23] K. Binder, D.W. Heermann, *Monte Carlo Simulation in Statistical Physics*, Springer, 1992.

[2.24] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller: *Equation of state calculation by fast computing machines*, J. Chem. Phys., Vol. 21, pp. 1087-1092, 1953.

[2.25] S. Kirkpatrick, *Optimization by Simulated Annealing: Quantitative Studies*, Journal of Statistical Physics 34, pp. 975-984, 1984.

[2.26] D. Mitra, F. Romeo, A. Sangiovanni-Vincentelli, *Convergence and Finite-Time Behavior of Simulated Annealing*, Proc. 24th Conf. On Decision and Control, pp. 761-767, Dec. 1985.

[2.27] G. Dueck, T. Scheuer, *Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing*, Journal of Computational Physics, Vol. 90, pp. 161-175, 1990.

[2.28] D.T. Connolly, *An improved annealing scheme for the QAP*, EJOR, 46, 93-100, 1990.

[2.29] R. Tafelmayer, K. H. Hoffmann, *Adaptive Schedules for Ensemble Based Threshold Accepting*, TU Chemnitz-Zwickau, 1994.

[2.30] M. Kolonko, M.T. Tran, *Convergence of Simulated Annealing with Feedback Temperature Schedules*, Problems in the Engineering and Informational Sciences, 11, 279-304, 1997.

[2.31] J. Schneider, I. Morgenstern, J. M. Singer, *‚Bouncing' towards the optimum – Improving the results of Monte Carlo optimization algorithms*, Phys Rev E 58, 5058, 1998.

[5.1] H. Fisher, L. Thompson, *Probabilistic learning Combinations of local job-shop scheduling rules*, Industrial Scheduling, Prentice Hall, Englewood Cliffs, New Jersey, 1963.

[5.2] OR-Library: http://mscmga.ms.ic.ac.uk - zuletzt zugegriffen am 1.4.2004.

[5.3] R.J.M. Vaessens, E.H.L. Aarts, J.K. Lenstra, *Job Shop Scheduling by Local Search*, Memorandum COSOR 94-05, Eindhoven University of Technology, 1994.

[6.1] S. Heipcke, *A new constraint programming approach to large scale resource constrained scheduling*, Diplomarbeit, Katholische Universität Eichstätt, 1995.

[6.2] Cavalcante et al.: http://www.dcc.unicamp.br/~cris/SPLC.html - zuletzt zugegriffen am 1.2.2001.

[6.3] C.C.B. Cavalcante, V.C. Cavalcante, C.C. Ribeiro, C.C. de Souza, *Parallel cooperative approaches for the labour constrained scheduling problem*, Preprint, Instituto de Computação, UNICAMP, Campinas, Brazil, 2000.

[6.4] C.C.B. Cavalcante, C.C. de Souza, M.W.P. Savelsbergh, Y. Wang, L.A. Wolsey, *Scheduling Projects with Labour Constraints*, Technical Report LEC-98-08, Georgia Institute of Technology, Atlanta, 1998.

[6.5] S. Heipcke , *Scheduling under Labour Resource Constraint*, Kapitel 12 in *Combined Modelling and Problem Solving in Mathematical Programming and Constrained Programming*, Dissertation, University of Buckingham, U.K., 1999.

[6.6] M. Uetz, *Algorithms for Deterministic and Stochastic Scheduling*, Kapitel 5 in *Ressource Constraint Project Scheduling Problems*, Dissertation, Technische Universität Berlin, 2001.

[6.7] PSPLib: http://www.bwl.uni-kiel.de/Prod/psplib/index.html - zuletzt zugegriffen am 25.4.2004.

[6.8] R. Kolisch, A. Sprecher, *PSPLib – a project scheduling problem library*, European Journal of Operational Research, Vol. 96, pp. 205-216, 1996.

[6.9] R. Kolisch, C. Schwindt, A. Sprecher, *Benchmark instances for project scheduling problems*, in J. Weglarz, *Handbook on Recent Advances in Project Scheduling*, 147-178, Kluwer, Amsterdam, 1999.

[6.10] R. Kolisch, S. Hartmann, *Experimental Investigation of Heuristics for Resource-Constrained Project Scheduling: An Update*, Working Paper, Technische Universität München, 2004.

[6.11] D. Merkle, M. Middendorf, H. Schmeck, *Ant Colony Optimization for Resource-Constrained Project Scheduling*, IEEE Transactions on Evolutionary Computation, 6:333-346, 2002.

[7.1] Heipcke S., *Production Flow Planning with Machine Assignment*, Kapitel 10 in *Combined Modelling and Problem Solving in Mathematical Programming and Constrained Programming*, Dissertation, University of Buckingham, U.K., 1999.

[8.1] *Benchmark Bench01*, unpublished working paper, SAP AG, Walldorf, 1997.

[8.2] T. Engelmann, *Prozeßflußoptimierung mit evolutionären Algorithmen*, Diplomarbeit, Universität Karlsruhe, 1998.

[8.3] C. Schwindt, *Verfahren zur ressourcenbeschränkten Projektdauerminimierungsproblems mit planungsabhängigen Zeitfenstern*, Dissertation, Universität Karlsruhe, 1997.

[8.4] M. Vogl, *Produktionsplanung mit Hilfe Physikalischer Verfahren*, Diplomarbeit, Universität Regensburg, 2000.

[8.5] D. Knjazew, *OmeGA – A Competent Genetic Algorithm for Solving Permutation and Scheduling Problems*, aus der Reihe *Genetic Algorithms and Evolutionary Computation*, Kluwer Academic Publishers, 2002.

[8.6] C. Schumm, *Parallele genetische Algorithmen zur multikriteriellen Optimierung von Schedulingproblemen*, Diplomarbeit, Universität Karlsruhe, 1999.

[9.1] H. Westenberger, J. Kallrath, *Formulation of a Job Shop Problem in Process Industry*, unpublished working paper, Bayer AG, Leverkusen and BASF AG, Ludwigshafen, Germany, 1994.

[9.2] J. Kallrath, *Planning and Scheduling in the process industry*, OR Spectrum, Volume 24, Issue 3, 2002.

[9.3] F. Blömer, *Produktionsplanung und –steuerung in der chemischen Industrie – Ressourceneinsatzplanung von Batchprozessen auf Mehrzweckanlagen*, Gabler, Wiesbaden, 1999.

[9.4] F. Blömer, H.O. Günther, *Scheduling of a Multi-Product Batch Process in the Chemical Industry*, Computers in Industry, 36, 245-259, 1998.

[9.5] F. Blömer, H.O. Günther, *LP-based heuristics for scheduling chemical batch processes*, International Journal of Production Research, 38, No. 5, 1029-1051, 2000.

[9.6] F. Blömer, H.O. Günther, *Numerical evaluation of MILP models for scheduling chemical batch processe*", Diskussionspapier 1999/05, TU Berlin, 1999.

[9.7] N. Trautmann, *Anlagenbelegungsplanung in der Prozessindustrie*, Dissertation, TU Karlsruhe, 2000.

[9.8] K. Neumann, C. Schwindt, N. Trautmann, *Advanced production scheduling for batch plants in process industry*, OR Spectrum, Volume 24, Issue 3, 2002.

[10.1] J. Schneider, J. Britze, A. Ebersbach, I. Morgenstern, M. Puchta, *Optimization of Production Planning Problems – A Case Study for Assembly Lines*, International Journal of Modern Physics C, Volume 11, No. 5, 949-972, 2000.

[10.2] M. Puchta, J. Gottlieb, *Solving Car Sequencing Problems by Local Optimization*, in S. Cagnoni et al. (eds.), *Application of Evolutionary Computing*, 132-142, LNCS 2279, Springer, 2002.

[10.3] I.P. Gent und T. Walsh, *CSPLib: a benchmark library for constraints*, technical report APES-09-1999. Adresse: http://4c.ucc.ie/~tw/csplib.

[10.4] A. Davenport, E.P.K. Tsang, *Solving constraint satisfaction sequencing problems by iterative repair*, in *Proceedings of the First International Conference on the Practical Applications of Constraint Technologies and Logic Programming*, 345-357, 1999.

[10.5] J.H.M. Lee, H.F. Leung, H.W. Won, *Performance of a Comprehensive and Efficient Constraint Library using Local Search*, in *Proceedings of 11th Australian Joint Conference on Artificial Intelligence*, 191-202, LNAI 1502, Springer, 1998.

[10.6] C. Solnon, *Solving Constraint Satisfaction Problems with Artificial Ants*, in *Proceedings of ECAI-2000*, 118-122, IOS Press, 2000.

[10.7] J.C. Regin, J.F. Puget, *A Filtering Algorithm for Global Sequencing Constraints*, in *Principles and Practice of Constraint Programming*, 32-46, LNCS 1330, Springer, 1997.

[10.8] B.Smith, *Succeed-first or fail-first: A case study in variable and value ordering heuristics*, in *Third Conference on the Practical Applications of Constraint Technology*, 321-330, 1996.

[10.9] M. Dorigo, G. Di Caro, *The Ant Colony Optimization Meta-Heuristic*, in D. Corne, M. Dorigo, F. Glover (eds.), *new Ideas in Optimization*, 11-32, McGraw Hill, 1999.

[10.10] T. Stützle, H.H. Hoos, *MAX-MIN Ant Systems*, Journal of Future Generation Computer Systems, Volume 16, 889-914, 2000.

[10.11] I.P. Gent, *Two Results on Car-sequencing Problems*, Technical report APES, 1998.

[10.12] L. Perron, P. Shaw, Combining Forces to Solve the Car Sequencing Problem, in Integration of AQI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 225-239, LNCS 3011, Springer, 2004.