REDUCING DATACENTER ENERGY USAGE THROUGH EFFICIENT JOB ALLOCATION

Christian Bodenstein, Chair of Information Systems Research, Albert-Ludwigs-Universität Freiburg, Platz der Alten Synagoge, 79085, Freiburg, Germany
christian.bodenstein@is.uni-freiburg.de

Guido Schryen, Department of Management Information Systems, University of Regensburg, Universitätsstraße 31, 93055 Regensburg, Germany
Schryen@gmx.net

Dirk Neumann, Chair of Information Systems Research, Albert-Ludwigs-Universität Freiburg, Platz der Alten Synagoge, 79085, Freiburg, Germany
dirk.neumann@is.uni-freiburg.de

Abstract

A large portion of power consumed by datacenters in small and medium sized businesses is wasted consumption, as machines mostly run idle. Commonly, this is accepted by operators as many countermeasures, such as layout redesign or hardware upgrades are too. In this work we analyze how much power can be saved if an exact optimization problem is set up to determine the cost minimizing allocation of resources. Since the exact solution cannot be found in a timely manner due to the non-linear nature of energy consumed by a system in operation, we show the goodness of various approximations in terms of power savings using numerical evaluations based on real workload traces compared to a commonly used resource allocation policy used in datacenters today.

Keywords: Resource Management, Green IS, Decision Support, Datacenters.

1 Introduction

Reducing power consumption is often one of the major goals in today's commercial organizations in their quest for profit maximization among other corporate aims. In nearly every modern business, to achieve higher profits, departments are forced to cut down on expenses. The same is true for modern datacenters. With fierce competition from Amazon EC2 or Sun's per hour computing campaigns, datacenters are forced to cut down on costs, or else risk dropping out of the markets. These "costs" can be divided into components and while no single cost component dominates, the cost of electricity for datacenters is a substantial operating cost factor – Google reports that over 10% of their operating costs are energy costs and these costs are doubled when the additional energy to cool the servers is included (Burge et al. 2006).

While most large datacenters have launched massive campaigns to reduce their energy consumption and thus increasing their efficiency, some even moving to cooler areas to save on cooling costs, smaller enterprise datacenters have missed the opportunities, or were unable to bring up the required investments to reduce their own costs; increase their own efficiency. It is well known that for most small and medium sized business (SMB) datacenters the systems are run very inefficiently with only 15% utilization on average – the rest is wasted. Various low-cost methods to reduce energy consumption are readily available in terms of architectural design, be it through virtualization, more efficient cooling through hot-cold aisle layout or more energy-efficient hardware. These methods however involve costly remanagement, often relocations to newer and larger buildings enabling only large enterprises to reap its benefits.

It is up to IT researchers to find methods for Enterprise and SMB data centers to work more efficiently, posing the question: *"How effective are energy minimizing scheduling algorithms compared to current scheduling mechanisms?"* In this work, we intend to promote energy-efficient optimizations in the management of datacenters for operators of spatially limited datacenters which are unable to re-design their hardware, but are willing to alter their assignment strategies as a short-term measure. The contribution of this work is threefold:

- The derivation of formal requirements to job allocations in modern datacenters
- The formulation and numerical evaluation of various energy consumption models
- Analysis of effects each requirement has on the potential reduction of energy usage

In conducting our research, we follow the guidelines for good design science research as suggested by (Hevner et al. 2004). This work is structured as follows. Section two presents related work on cost/energy-aware scheduling and energy efficiency and derives the requirements set before our models. Since power consumption reduction and cost reduction go hand-in-hand, because reducing power consumption in most cases results in reducing costs, we use these terms *power usage* or *power consumption* reduction and *cost* reduction synonymously. In section three we elaborate our models and show small examples, reflecting the unique properties inherent to each version as stipulated by the model requirements. Section four contains our numeric evaluations based on which section five indicates the value of the mechanisms to IT service management, before section six concludes this work and points to promising future research.

2 Related Work and Requirements Analysis

This section attempts a categorization of some related work in allocation and scheduling mechanisms (it is by no means exhaustive) in the field of chip design architecture, server architecture, clusters and datacenters. To date, a lot of research has also gone into hardware inefficiencies and thermal hotspots in computing. (See 2008) found that most energy flowing into systems is given off as heat, resulting that only about 10% (percentage varies in hardware complexity) of energy is actually available for computing. This insight has fueled numerous research projects ranging from single chip workload positioning (Coskun et al. 2007) to provisioning on servers and server clusters (Bansal et al. 2007).

(Kurp 2008) focused mainly on thermal efficiency in datacenters, suggesting viable alternatives to reduce heat buildup in server clusters to save cooling costs. Work on thermal hotspots in datacenters is equally dominant with numerous authors combining and summarizing state-of-the-art thermal measures to reducing heat build-up in next-generation datacenters. (Moore et al. 2005; Nathuji et al. 2008; Rajamani et al. 2003; Sharma et al. 2008)

With ample means of reducing heat in IT-infrastructures, authors looked towards reducing energy consumption by the equipment itself. Work by (Albers et al. 2007) address energy consumption on a chip-level on battery-operated systems, concluding that less energy is consumed over the duration of a process if the speed of the processors is reduced. (Yao et al. 1995) addressed efficiency on a server level, while (Chase et al. 2001) proposed solutions for server clusters, ranging from energy-saving power supplies, to the powering down of unused hard drives to using different power states to conserve energy.

In consolidation, some authors suggest power management for single server environments, combining energy-efficiency and management of servers. These attempts include the goal minimizing the average percentage of energy consumed by the system, while meeting an execution time constraint. This is similar to (Shivle et al. 2006), where the goal of the allocation is to minimize the average percentage of energy consumed by the application to execute across the machines in the ad hoc Grid, while meeting an application execution time constraint. This design is also present in recent work by (Pinheiro et al. 2003) for server cluster environments. Other work has focused mainly on turning idle machines off, to save power as shown in (Burge et al. 2007; Chase et al. 2001) for datacenters. Consolidated attempts are present in the work of (Mastroleon et al. 2005; Meisner et al. 2009) ranging from automated power management, to forcing servers to 'power-nap' in underutilized phases.

What all of the above have in common is the need for cost minimization in one form or another. They merely differ in their approaches. Looking specifically at the needs and requirements of modern SMB datacenters, and waging what can and cannot be done with a limited budget, the following requirements can be derived and will be used to form our goals: (The requirements have been listed in table 1 selectively matching authors' work with each requirement)

Author	R1) Cost Min.	R2) PowerScaling	R3) On/Off	R4) Compare
(Sharma et al. 2008)	X	X		
(Moore et al. 2005)	X	X		
(Rajamani et al. 2003)	X		X	
(Raghavendra 2008)	X			
(Hamann 2008)	X			
(Rivoire et al. 2007)	X	X		
(Nathuji et al. 2008)	X	X		
(Burge et al. 2007)	X	X	X	
(Chase et al. 2001)	X	X	X	
(Mastroleon et al. 2005)	X	X		
(Meisner et al. 2009)	X		X	
Our contribution	**X**	**X**	**X**	**X**

Table 1. Selected Author-to-Requirements Categorization

Requirement R1: Energy consumption minimizing allocation: The profit maximizing datacenter provider essentially performs optimizations for two cases: *undersaturation* versus *oversaturation*. When the datacenter is under-saturated, the scheduler may accept all, or near to all requests, leaving profits as a constant. According to (Burge et al. 2007) most datacenters provide for enough capacity to handle peak utilization, meaning they have excess capacity on average utilization. From this we derive our requirement, namely *all jobs must be allocated to a resource node in a cost minimizing manner.*

Requirement R2: Scaling power usage featuring increasing returns to scale: Data centers nearly always have a heterogeneous pool of machines: different generation servers, different kinds of blade servers, or even a mix of blades and servers. Different machines provide different performance and consume different amounts of power and they are seldom linear. The second requirement is thus *allocations must consider non-linear cost functions*. We model this non-linearity by using logarithmic cost functions.

Requirement R3: Dynamic power management featuring on-off decisions: Costs in management of datacenters are not generated through utilization alone. This is in accordance to (Moore et al. 2005) who found that idle machines consume roughly 50% of the power compared to those at full utilization. Compared to the SPEC data, this is largely due to the addition of cooling costs and "base" operations (such as fans, hard-disks, etc.) during idle time. Leaving a system running idle thus generates costs that are unnecessary. In our third requirement *the cost minimizing allocation must include the decisions which nodes to leave running and which nodes to turn off*.

Requirement R4: Linear and Non-Linear Model comparison: The last requirement for this work outlines the key contribution of this work, namely the comparison of costs, and to a certain extent (though not extensively) the computational costs of both linear and non-linear varieties of modelling costs in computing. We intend to, using realistic modelling parameters, show under what circumstances it would make more sense to approximate, and when to calculate exactly.

3 Model Presentation

Following the requirements suggested above we proceed to the model definitions proposed by this work. To begin with, a base model (referred to as LIN) will be presented which aims to minimize costs through allocating jobs to nodes efficiently based on a linear mixed integer problem (MIP). To show the effect of considering each requirement, three model extensions will be presented, each addressing a further requirement. The first extension will cover the addition of fixed costs in LINFIX, assuming linear cost functions with fixed costs. The second and third extension will cover the effect of considering the non-linear versions in non-linear mixed integer problems (MINLP). The second extension (referred to as SCALE) will show the effect of a non-linear cost function and SCALEFIX, the third extension, will show these non-linear cost functions with fixed costs. For ease of comparison, all models will be subject to the same parameters and variable sets.

3.1 Model Parameters

Computing power and memory are the central scarce resource to be traded in a centrally controlled datacenter environment. There are two players: (I) requesters, who wish to obtain computing resources and (II) the datacenter which supplies resources, attempting to minimize costs. N describes the set of resource offers ("node") available to the datacenter. Each node can be seen as a perfectly divisible virtual machine capable of executing multiple jobs in parallel. This technology is already present in most state-of-the-art virtualization technologies, since unused resources by current virtual machines (VM), are made available to other VM's. The datacenter has the following information available to him:

Computing units supplied by node n	$\overline{cs_n}$
Memory units supplied by node n	$\overline{ms_n}$
Logarithmic cost function parameters of node n	$f[log(\dots)]$
Variable cost component of node n	$varco_n$
Fixed cost component of node n	$kfix_n$

Resource requests are posted in the form of orders. Each order contains the information of the resources required:

Computing units required by j	c_j
Memory units required by j	m_j

First timeslot where job j is to be executed	$first_j$
Last timeslot where job j is to be executed	last_j

For demonstration purposes, an example case has been generated to show the functionality and uniqueness of each model. This test scenario will be used in each model to generate a solution. The data sample is shown in table 2.

Job	c_j	m_j	$first_j$	$last_j$	Nod	$\overline{cs_n}$	$\overline{ms_n}$	$varco_n$	$lfunb_n$	$lfunc_n$	$kfix_n$
J1	30	45	1	4	N1	150	249	4,3	2	30	3
J2	25	31	2	3	N2	145	259	8,8	9	5	2
J3	55	78	3	4	N3	155	253	8,6	3	80	1
J4	60	14	2	4							
J5	35	51	1	2							
J6	20	29	3	3							

Table 2. Sample Job Requests and Node Offers

J1 is a request for a job to be run from timeslot 1 to 4 and requires a minimum of 30 CPU's and 45 units of memory in each timeslot. Node N1 offers 150 CPU's and 249 units of memory and costs approximately 4.3 monetary units (MU) per 10% increase in utilization and timeslot. Note, since the costs of operation are non-linear, as stipulated by requirement R2, $varco_n$ is only an approximation of the logarithmic function and is required for the linear models ($varco_n$ are calculated using ordinary least squares estimation on each nodes logarithmic cost function for a linear function of the form $y = 0 + bx$. This ensures the logarithmic and linear functions both have the same *y-intercept*, namely $kfix_n$.). The exact costs are determined by the log function and the parameters $lfunb_n$ and $lfunc_n$. The idle costs $kfix_n$ for N1 are 3 MU per timeslot if the node is in operation. If J1 were allocated to N1, the computing costs would amount to 6.76 MU $((last_j - first_j + 1) * f[log(\dots)] = (4 - 1 + 1) * 2 * log(1 + 30 * 30/150) = 6.76)$ plus the 12 MU for powering the node over the required 4 periods. In the following sections, the proposed models will be formulated. Using the test scenario above shown in table 2, each model will be demonstrated and the resulting sample allocation will be discussed and compared with its predecessors.

3.2 Model Setting: LIN – Cost minimization with linear cost functions

LIN Model: : In this setting we begin by relaxing the formal model requirements R2, R3 and R4 by proposing a base model featuring *cost minimization (R1)* showing *constant returns to scale.* For simplicity all nodes are assumed to be *available and powered during all timeslots*, and that all *reported resource capacity parameters are finite and accurate*. Allocation is determined by the binary decision variable x_{jnt}, where $x_{jnt} = 1$ if job j is allocated to node n in time slot t, and $x_{jnt} = 0$ if not. The time slot horizon $T = \{first_j; last_j\}$ defines the set of all time slots for the underlying allocation problem. This ensures that t is only defined for those time periods where allocation for the given combination of jobs and nodes is actually feasible. Hence, if J is defined as the index of job requests (resource requests), N the index of defined resource nodes (node suppliers) and T the time horizon index, the cost minimizing assignment problem can be mathematically formalized as the following:

$$min_x K = \sum_n^N \sum_t^T \left(\frac{\sum_j^J x_{jnt} c_j}{\overline{cs_n}} \right) * varco_n \qquad (1)$$

Subject to:

$$x_{jnt} \in \{0,1\}, \qquad \forall j \in J, \forall n \in N, \forall t \in T \qquad (2)$$

$$\sum_n^N x_{jnt} = \begin{cases} 1, & \forall j \in J, \forall t \in [first_j; last_j] \\ 0, & \forall j \in J, \forall t \in \text{T}\backslash[first_j; last_j] \end{cases} \qquad (3)$$

$$\sum_j^J x_{jnt} c_j \leq \overline{cs_n}, \qquad \forall n \in N, \forall t \in T \qquad (4)$$

$$\sum_j^J x_{jnt} m_j \leq \overline{ms_n}, \qquad \forall n \in N, \forall t \in T \qquad (5)$$

Eq. 1 shows the objective function of the base model. Its function is to minimize costs based on the utilization of CPU's of each job. We have opted to use CPU's only in our target function, since most costs are generated through the operation of cores. For simplicity, these costs indirectly include all operation costs incurred in operation (i.e. cooling, memory requirements, network costs etc.). Eq. 2 introduces the binary decision variable *x*. Eq. 3 ensures the enforcement of agreements allocating all jobs in the schedule as prescribed by requirement a) and ensures each job is only executed on one node at a time. Eq. 4 and Eq. 5 are the resource constraints and ensure that allocations do not exceed the capacity of the systems.

Node	**t1**	**t2**	**t3**	**t4**
N1	J1, J5	J1, J2, J4, J5	J1, J3, J4	J1, J3, J4
N2				
N3			J2, J6	

Table 3. Sample allocation for LIN

Example: For the request and offer sample presented in table 2, the costs for the allocation amount to 59.313 MU's (allocation shown in table 3). In attempt to minimize costs, the mechanism preferred N1 due to its low variable costs ($varco_n$= 4.3 MU). Once the node no longer had any capacities left, it proceeded to the next lowest cost, and assigned J2 and J6 to N3 which only had $varco_n$ of 8.6 MU.

3.3 Model Setting: LINFIX – Cost minimization with linear cost functions

LINFIX Model: In this setting we augment the initial setting (**LIN**) by keeping the relaxation of the formal model requirements R2 and R4 intact, proposing a first extension to the *cost minimization (R1)* approach showing *constant returns to scale*. In this extension we model the dynamic nature of availabilities in datacenters we now assume that *unused nodes may operate in a sleep mode (R3)* incurring no further costs. This state is different to the one where a node is simply unused, as a nodes still generates costs (largely idle operation costs to keep all drives spinning, cooling costs, management and upkeep, etc.). As a simplification measure, switching between sleep mode and operational status generates no additional costs; they are included in the cost functions. As before, the resource capacity parameters reported by datacenter providers remain *finite and accurate*. Eq. 6 shows the objective function which aims to minimize costs based on the utilization of CPU's of each job plus the sum of all active nodes $kfix_n$. Equation 7 is added to include the node activation constraint y_{nt}. M is an arbitrarily big number (the so called Big M).

$$min_x K = \sum_n^N \sum_t^T \left(\frac{\sum_j^J x_{jnt} c_j}{\overline{cs_n}} \right) * varco_n + \sum_n^N \sum_t^T y_{nt} kfix_n \qquad (6)$$

Subject to:
... as before (Equations (2) – (5) in Section 3.2)

$$y_{nt} \in \{0,1\}, \ \sum_j^J x_{jnt} \leq M y_{nt} \qquad \forall n \in N, \forall t \in T \qquad (7)$$

Example: For the request and offer sample the costs for this allocation amount to 51.662 MU's based on the logarithmic cost parameters shown in table 4. In attempt to minimize costs, LINFIX is faced with a trade-off between operation costs $varco_n$ and node fix costs $kfix_n$. Again, N1 is preferred due to its low variable costs ($varco_n$= 4.3 MU) which outweigh the fixed costs of 3 MU when the node is fully utilized, which is the case for timeslots two to four. In timeslot one, however, the worth of including idle operation costs in optimization becomes evident. In the LIN example, N1 was only utilized by 43%. Even though the computation costs seem higher at first it becomes evident why N3 is chosen for lower utilization levels. The costs in timeslot one of 4.8 MU's (1.8 MU + 3 MU) for N1 are higher than the 4.7 MU's (3.7 MU + 1 MU) generated if the job were executed on N3.

Node	t1	t2	t3	t4
N1	OFF	J1, J2, J4, J5	J1, J3, J4	J1, J3, J4
N2	OFF	OFF	OFF	OFF
N3	J1, J5	OFF	J2, J6	OFF

Table 4. Sample allocation for LINFIX

3.4 Model Setting: SCALE – Logarithmic cost functions

SCALE Model: In this setting we augment the initial settings by keeping the relaxation of the formal model requirements R3 and R4 intact, proposing a first extension to the *cost minimization (R1)* now addressing the formal requirement of *increasing returns to scale (R2)* often observed in cost management of datacenters. The relative gain from powering a node at high utilization levels is higher than operating a node a lower levels, promoting a consolidation of operations to a single system. To observe this effect nodes are again assumed to be *available and powered during all timeslots*, and that all *reported resource capacity parameters are finite and accurate.* Again allocation is determined by the binary decision variable x_{jnt}, results in the mathematical formulation for SCALE:

$$min_x\, K = \sum_n^N \sum_t^T f\left[log\left(1 + \left(\frac{\sum_j^J x_{jnt} c_j}{\overline{cs_n}}\right)\right)\right] \quad (8)$$

Subject to:
... as before (Equations (2) – (5) in Section 3.2)

Eq. 12 shows the objective function of SCALE which minimizes costs based on the utilization of CPU's. Different to the model before is that the cost function f_n is logarithmic in nature.

Node	t1	t2	t3	t4
N1	J1, J5	J1, J2, J4, J5	J1, J3, J4	J1, J3, J4
N2			J2, J6	
N3				

Table 5. Sample allocation for SCALE

Again, the request and offer sample shown in table 4 is taken, resulting in the allocation shown in table 7. For SCALE, the costs amount to 58.182 MU's. In attempt to minimize costs, the mechanism again preferred N1 due to its shallow logarithmic cost curve. Once the node no longer had any capacities left, it proceeded to the next lowest cost, and assigned J2 and J6 to N2. This is different to the allocation in FIN, as N3, although showing a lower OLS gradient than N2, has a very steep climb in the first portion of the cost curve, making it less attractive for lower utilization levels. As a result this solution is generates 98% of the costs given in LIN, but is worse than LINFIX, since unused nodes are still left in operation.

3.5 Model Setting: SCALEFIX – Cost minimization with logarithmic cost functions

Featured Model: In this setting we further realize the given requirements by addressing all but requirement R4. The third extension, SCALEFIX, features, *cost minimization (R1),* addresses the formal requirement of working with cost functions showing *increasing returns to scale (R2),* as well as providing for a solution to the question *which node should be used, and which sent to sleep mode (R3)*. Eq. 9 shows the objective function of SCALEFIX which minimizes costs based on the utilization of CPU's and fixed costs incurred if nodes are active. As before the cost function f_n is dependent on the current utilization. Additionally the costs incurred during operation of nodes $kfix_n$ are added for each node y_{nt} used by the resulting allocation.

$$min_x K = \sum_n^N \sum_t^T f\left[log\left(1 + \left(\frac{\sum_j^J x_{jnt} c_j}{\overline{cs_n}}\right)\right)\right] + \sum_n^N \sum_t^T y_{nt} kfix_n \quad (9)$$

Subject to:
... as before (Equations (2) – (5) in Section 3.2 and (7) in Section 3.3)

Example: For SCALEFIX the effects observed in the example cases above come together. Again, the request and offer sample shown in table 4 are taken. For SCALEFIX, the costs amount to 48.755 MU's. Similar to LINFIX, SCALEFIX is faced with a trade-off between the slope of the logarithmic cost curves and node fix costs $kfix_n$. As in SCALE the mechanism again preferred N1 due to its shallow logarithmic cost curve, which outweighed the effect of this node having the highest fixed costs. However two changes are apparent.

Node	**t1**	**t2**	**t3**	**t4**
N1	J1, J5	J1, J2, J4, J5	J1, J2, J6, J4	J1, J3, J4
N2	OFF	OFF	OFF	OFF
N3	OFF	OFF	J3	OFF

Table 6. Sample allocation for SCALEFIX

Here the error of linearizing cost curves becomes evident. Recall that LINFIX allocated J1 and J5 to N3, based on its calculation that costs for N1 (4.8 MU's) were higher than for N3 (4.7 MU's) using $varco_n$ as a cost estimate. However, when using the logarithmic functions, SCALEFIX finds that N1 (5.3 MU's) is more expensive than N3 (5.7 MU's). Since all three log functions intersect each other at utilization level of approximately 30%, below this point, N2 is cheaper, but for utilizations above 30%, N1 and N3 operate more efficiently. As a result the best possible solution generates only 82% of costs in LIN, 84% against SCALE and 94% of LINFIX. As such, these numbers are quite small, but for larger numbers of jobs and nodes, and for idle costs ranging above 50% of total utilization costs, the difference only increases, and will be shown in the evaluation section.

4 Evaluation

With numerical analysis there are two possibilities to come up with data on which to test the models: Practical use cases derived from real workload traces of datacenters are of high practical relevance. However, it is hard to obtain traces where all the data required by the underlying model is readily available. More importantly, to the best of our knowledge there is currently no trace available containing all the costs involved in computing broken down onto the workload traces that generated them. Consequently, the only option left is the creation of artificial instances. Further, artificial data benefits in that they can be constructed so as to specifically investigate certain properties which we investigate in our models.

4.1 Data Generation

To remain as close as possible to practical cases, we chose to test our models using numeric simulations based on randomly generated workloads derived from real workload traces based on a constant set of parameters. Each combination generated using these parameters are referred to as an "instance". Each order pair is referred to as a "case". In order to simulate this scenario, the resource requirements were all drawn randomly using a positively skewed lognormal distribution $\log \mathbb{N}_0^+ (\mu, \sigma^2)$, as recommended by (Feitelson 2002). The distributions are shown in table 7.

Variable Description	Variable	Distribution
Computing units supplied by node n	$\overline{cs_n}$	Lognormal LogN (4.1; 1.1)
Memory units supplied by node n	$\overline{ms_n}$	Lognormal LogN (6.4; 1.7)

Logarithmic cost parameter :	$lfunb_n$	Uniform [1; 10]
	$lfunc_n$	Uniform [1; 100]
Fixed cost component of node n	$kfix_n$	Uniform [0.1; 8]
Computing units required by j	c_j	Lognormal LogN (3; 1.1)
Memory units required by j	m_j	Lognormal LogN (5; 1.7)
First timeslot where job j is to be executed	$first_j$	Uniform [1; 5]
Last timeslot where job j is to be executed	$last_j$	Uniform [1; 5] > $first_j$

Table 7. Data Generation Parameter Distribution

For every case and instance all models were subject to the same dataset, allowing the solutions to be directly comparable. For each case, ten instances were generated and the average taken. The simulations were run using GAMS where the MIP problems were solved using CPLEX and the MINLP problems were solved using SBB. The solvers were not restrained in their iterations and time limits, but for the MINLP problems solutions within 5% of the theoretic optimum (option optcr=0.05) were acceptable. These were run with separate instances executed in parallel on multiple systems. In total over 3200 instances were generated of which most were solved within reasonable bounds.

To serve as a benchmark, we enquired about allocation methods commonly used in datacenters today and found an algorithm as presented by (Vengerov 2009). In this work we will use a simpler version of the model and refer to it as BESTFIT (described as such by the authors). Following Vengerov's intuition "jobs are allocated where they fit best" (SUN Microsystems). To allow comparison to our models, BESTFIT differs from our model only in the target function, and is described in equation 11:

$$min_x\, K = \sum_n^N \sum_t^T \left[\overline{cs_n} - \left(\sum_j^J x_{jnt} c_j\right)\right] \quad (11)$$

Subject to:

... as before (Equations (2) – (5) in Section 3.2)

For single period allocations the allocation timeframe was limited to one, where for multi-period allocations groups of allocations over two, three, four and five periods were simulated. Increasing the allocation timeframe decreased the computing time of feasible solutions considerably. As a result, most simulations were run using five timeslots and results of these traces are summarized below. Additionally, scenarios where the magnitude of resources available in relation to those required was too great (such as 20 jobs on 100 nodes) were not observed as these results would be biased towards shutting most systems down, showing more than 50% reduction of costs, which although a very high savings, is a rather intuitive solution, and thus omitted. Also, since a requirement to all models is that all jobs are accepted, cases where not all jobs could be accepted were deemed infeasible and not subject to analysis.

4.2 Data Analysis

Before proceeding to review the results of our simulations, a few statistical tests need to be mentioned to found our results. Following the simulations, we tested whether the samples collected for each case were subject to a normally distributed population using the Shapiro-Wilk Normality Test (at a chosen alpha level of 0.05). We chose the Shapiro-Wilk Normality Test as the test is quite sensitive against a wide range of alternatives even for small samples (n < 20). The statistic is responsive to the nature of the configuration of the sample as compared with the configuration of expected values of normal order statistics used by other tests (Shapiro et al. 1965).

After successfully accepting the normality hypothesis for our samples, we tested the variances with an F-test to see whether the variances were equal. We found that for 94% of the cases, these variances differed significantly (alpha level of 0.05). In the case where the samples are from a normally distributed population and have equal variance, the two-sample t-test can be used. For unequal variances, we determined the approximate solution (for unequal variances no exact solution exists),

adapted from the two-sample t-test, and called the Welch test. We performed the Welch test with the H_0 hypothesis that the means of each model was significantly lower than the mean for BESTFIT.

j/n	5		10		15		20	
5	0,21** (23.9)	0,28*** (28.3)	0,30*** (12.6)	0,37*** (9.0)	0,15** (9.1)	0,22*** (15.1)	0,13*** (9.0)	0,20*** (12.3)
	0,36*** (20.0)	0,39*** (25.3)	0,47*** (9.8)	0,53*** (12.2)	0,42*** (7.2)	0,46*** (15.8)	0,45*** (10.8)	0,48*** (11.3)
10	0,11* (15.5)	0,13** (19.0)	0,25*** (6.4)	0,34*** (12.4)	0,27*** (14.7)	0,31*** (16.0)	0,24*** (7.4)	0,30*** (9.4)
	0,17** (17.6)	0,24*** (20.5)	0,40*** (9.4)	0,49*** (16.4)	0,46*** (13.3)	0,49*** (17.0)	0,45*** (7.2)	0,49*** (11.1)
15	0,02 (24.1)	0,07 (18.4)	0,16*** (13.6)	0,22*** (11.9)	0,27*** (7.4)	0,31*** (10.8)	0,28*** (9.4)	0,32*** (12.4)
	0,10 (22.5)	0,15 (16.0)	0,35*** (10.8)	0,41*** (11.8)	0,41*** (8.8)	0,44*** (11.6)	0,46*** (11.7)	0,49*** (14.5)
20	0,15** (14.8)	0,18***(17.7)	0,13** (11.0)	0,22*** (9.4)	0,25*** (8.1)	0,28*** (10.2)	0,28*** (9.1)	0,32*** (9.9)
	0,25*** (15.5)	0,27***(17.4)	0,31*** (11.1)	0,36*** (9.8)	0,35*** (8.8)	0,40*** (11.0)	0,42*** (8.9)	0,46*** (14.4)
	40		60		80		100	
40	0,29*** (5.3)	0,31*** (7.2)	0,27*** (5.8)	0,34*** (7.2)	0,23*** (4.4)	0,29*** (3.7)	0,21*** (2.6)	0,24*** (5.1)
	0,43*** (6.1)	0,46*** (9.6)	0,45*** (9.3)	0,50*** (8.5)	0,47*** (8.9)	0,51*** (7.7)	0,46*** (3.3)	0,49*** (4.5)
60	0,30*** (5.4)	0,34*** (5.4)	0,29*** (4.5)	0,35** (4.9)	0,28*** (4.5)	0,35*** (5.5)	0,25*** (4.8)	0,30*** (3.4)
	0,40*** (6.2)	0,44*** (8.1)	0,44*** (5.9)	0,49*** (8.6)	0,45*** (5.4)	0,49*** (6.3)	0,45*** (5.6)	0,48*** (4.1)
80	0,20*** (11.0)	0,22** (12.0)	0,28*** (6.9)	0,31*** (7.8)	0,29*** (3.9)	0,32*** (2.5)	0,27** (4.8)	0,30*** (5.5)
	0,25*** (12.2)	0,29*** (10.6)	0,39*** (8.4)	0,44*** (11.2)	0,43*** (5.2)	0,48*** (5.0)	0,45*** (5.6)	0,48*** (5.1)
100	0,13** (8.8)	0,16** (9.6)	0,26** (4.4)	0,30*** (5.1)	0,30*** (6.4)	0,35*** (6.3)	0,30*** (3.8)	0,34*** (3.7)
	0,17*** (9.3)	0,20** (11.4)	0,35*** (4.8)	0,38*** (9.3)	0,40*** (5.9)	0,43*** (6.6)	0,44*** (3.9)	0,47** (7.1)

*** mark a significance level of more than 99%. ** show a level between 95% and 99% and * indicates a significance of more than 90%. The top left value of the matrix belongs to the linear results LIN, the top right result belongs to the logarithmic model SCALE. The lower left result is from LINFIX and the lower right result belongs to SCALEFIX. The succeeding value in brackets shows the coefficient of variation (CV).

Table 8. Simulation Solutions

The evaluations were categorically grouped for more convenient presentation in table 8. The first value in each cell shows the improvement of each model (in the order LIN, SCALE, LINFIX, SCALEFIX) compared to the benchmark model BESTFIT. Following this improvement, the significance level determined by the Welch test is noted with the use of stars, as is common for expressing the significance of the mean improvement. The value in brackets indicates the coefficient of variation. Since standard deviation has little interpretable meaning on its own unless the mean value is known it is easier to get an idea of variability in a distribution by dividing the standard deviation with the mean. Represented as a % of mean, this quotient is known as the coefficient of variation (CV). The coefficient of variation is particularly useful when comparing dispersion in datasets with different means, which is the case for our results, since the means differ greatly.

5 Discussion and Interpretation for IT Service Management

In this work we set out to explore to what extent operation costs of SMB datacenters be reduced through energy-aware allocations of computational tasks. Understanding how costs are generated, and which factor generates what potential is essential for successful cost reduction strategies. This work contributed to this understanding of costs in datacenters by formulating as well as numerically evaluating various cost models in cost minimization approaches. Through analysis of related work and methods used by datacenters today, we found that for small and medium sized datacenters are architected (from a power perspective) to maximize utilization for maximum energy efficiency. However the state of maximum utilization is seldom reached. Where large datacenters spend a LOT of

time and work to achieve greater efficiencies in operation, enterprise and SMB centers are stuck with their inefficiencies; and therein lies the opportunity.

We elaborated the models and reflected the unique properties inherent to each version as stipulated by the model requirements. We found that the four models performed better than the BESTFIT mechanisms currently used for cases of underutilization significantly providing for meaningful alternatives to reduce costs of operation by allocating jobs to resources efficiently.

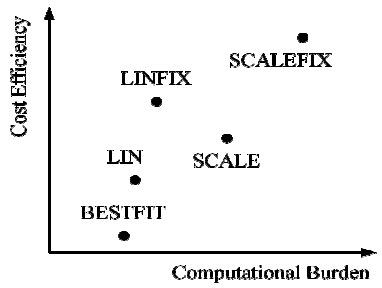

Figure 9. Model Comparison Overview

While the experimental results above must not be generalized too hastily, they do suggest practical use for the cost efficient management of datacenters. While not a single model setting can be coined as a "one-size-fits-all" solution, each does have its specific uses given certain scenarios. These scenarios could include the following:

BESTFIT, though fast and computationally inexpensive shows weaknesses when faced with heterogeneous computing environments, where the costs generated for systems are no longer equal for all systems. This is where both versions of LIN and SCALE performed well, as they consider these changing costs in their optimization. In comparison LIN and SCALE differ in their need for accuracy. For cases simulated where the SCALE function was quasi-linear (low variance in OLS estimations for the $varco_n$ parameter) the difference between the LIN and SCALE models was so low, that the computational burden to compute the solution for SCALE models makes its slight improvement in energy usage questionable.

The same principle is true for LINFIX and SCALEFIX. For all combinations, job stacking was present for all SCALE models, where all active nodes were picked and stacked to a 100% utilization with only a single node not fully stacked to execute leftovers. For linear models utilization was high, but ranged between 0.5 and 1. Both models show great potential in cost reduction and are a must for underutilized datacenter. If technologies are available to reduce the idle costs through PowerNap or comparable features, not including these methods in optimization is costly. As shown in related work, ample methods are readily available. Once datacenter managers know the costs for each power state of each system, costs for powering and cooling can be cut by up to 50%. These models serve to aid IT Service providers to implement insights gained through our research to allocate their tasks to their resources in a cost-efficient manner, provided their cost situation resembles those presented in this work.

References

Albers, S., and Fujiwara, H. "Energy-efficient algorithms for flow time minimization," ACM Trans. Algorithms (3:4) 2007, p 49.

Bansal, N., Kimbrel, T., and Pruhs, K. "Speed scaling to manage energy and temperature," J. ACM (54:1) 2007, pp 1-39.

Burge, J., Ranganathan, P., and Wiener, J.L. "Cost-aware scheduling for heterogeneous enterprise machines (CASH'EM)" in: CLUSTER IEEE, 2007, pp. 481-487.

Chase, J.S., Anderson, D.C., Thakar, P.N., Vahdat, A.M., and Doyle, R.P. "Managing energy and server resources in hosting centers," in: Proceedings of the eighteenth ACM symposium on Operating systems principles, ACM, Banff, Alberta, Canada, 2001.

Coskun, A.K., Rosing, T.S., and Whisnant, K. "Temperature aware task scheduling in MPSoCs," in: Proceedings of the conference on Design, automation and test in Europe, EDA Consortium, Nice, France, 2007.

Feitelson, D.G. "Workload Modeling for Performance Evaluation," in: Performance Evaluation of Complex Systems: Techniques and Tools, Performance 2002, Tutorial Lectures, Springer-Verlag, 2002.

Hamann, H.F. "A Measurement-Based Method for Improving Data Center Energy Efficiency," in: Proceedings of the 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (sutc 2008) - Volume 00, IEEE Computer Society, 2008.

Hevner, A., March, S., Park, J., and Ram, S. "Design Science in Information Systems Research," MIS Quarterly (28:1) 2004, pp 75-105.

Hillier, F., and Lieberman, G. Introduction to Operations Research, (5th Edition ed.) McGraw-Hill Publishing Company, 1990.

Kurp, P. "Green computing," Commun. ACM (51:10) 2008, pp 11-13.

Lefurgy, C., Rajamani, K., Rawson, F., Felter, W., Kistler, M., and Keller, T.W. "Energy Management for Commercial Servers," Computer (36:12) 2003, pp 39-48.

Mastroleon, L., Bambos, N., Kozyrakis, C., and Economou, D. "Automatic power management schemes for Internet servers and data centers," Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE, 2005, p. 5 pp.

Meisner, D., Gold, B.T., and Wenisch, T.F. "PowerNap: eliminating server idle power," in: Proceeding of the 14th international conference on Architectural support for programming languages and operating systems, ACM, Washington, DC, USA, 2009.

Moore, J., Chase, J., Ranganathan, P., and Sharma, R. "Making scheduling "cool": temperature-aware workload placement in data centers," in: Proceedings of the annual conference on USENIX Annual Technical Conference, USENIX Association, Anaheim, CA, 2005.

Nathuji, R., Isci, C., Gorbatov, E., and Schwan, K. "Providing platform heterogeneity-awareness for data center power management," Cluster Computing (11:3) 2008, pp 259-271.

Pinheiro, E., Bianchini, R., Carrera, E.V., and Heath, T. "Dynamic cluster reconfiguration for power and performance," in: Compilers and operating systems for low power, Kluwer Academic Publishers, 2003, pp. 75-93.

Raghavendra, P. "Optimal algorithms and inapproximability results for every CSP?," in: Proceedings of the 40th annual ACM symposium on Theory of computing, ACM, Victoria, British Columbia, Canada, 2008.

Rajamani, K., and Lefurgy, C. "On evaluating request-distribution schemes for saving energy in server clusters," in: Proceedings of the 2003 IEEE International Symposium on Performance Analysis of Systems and Software, IEEE Computer Society, 2003.

Rivoire, S., Shah, M.A., Ranganathan, P., and Kozyrakis, C. "JouleSort: a balanced energy-efficiency benchmark," in: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, ACM, Beijing, China, 2007.

Shapiro, S., and Wilk, M. "An analysis of variance test for normality (complete samples)," Biometrika (52:3/4 December) 1965, pp 591-611.

Sharma, R.K., Shih, R., Bash, C., Patel, C., Varghese, P., Mekanapurath, M., Velayudhan, S., and Manu Kumar, V. "On building next generation data centers: energy flow in the information technology stack," in: Proceedings of the 1st Bangalore annual Compute conference, ACM, Bangalore, India, 2008.

Shivle, S., Siegel, H.J., Maciejewski, A.A., Sugavanam, P., Banka, T., Castain, R., Chindam, K., Dussinger, S., Pichumani, P., Satyasekaran, P., Saylor, W., Sendek, D., Sousa, J., Sridharan, J., and Velazco, J. "Static allocation of resources to communicating subtasks in a heterogeneous ad hoc grid environment," Journal of Parallel Distributed Computing (66:4) 2006, pp 600-611.

SPEC "SPEC 2008 Power Results: Standard Performance Evaluation Corporation," 2008.

Vengerov, D. "A reinforcement learning framework for utility-based scheduling in resource-constrained systems," The International Journal of Grid Computing and eScience: Future Generation Computer Systems (25:7) 2009, pp 728-736.

Yao, F., Demers, A., and Shenker, S. "A scheduling model for reduced CPU energy," in: Proceedings of the 36th Annual Symposium on Foundations of Computer Science, IEEE Computer Society, 1995.