A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers

Benjamin Speitkamp and Martin Bichler

Abstract—Today's data centers offer IT services mostly hosted on dedicated physical servers. Server virtualization provides a technical means for server consolidation. Thus, multiple virtual servers can be hosted on a single server. Server consolidation describes the process of combining the workloads of several different servers on a set of target servers. We focus on server consolidation with dozens or hundreds of servers, which can be regularly found in enterprise data centers. Cost saving is among the key drivers for such projects. This paper presents decision models to optimally allocate source servers to physical target servers while considering real-world constraints. Our central model is proven to be an NP-hard problem. Therefore, besides an exact solution method, a heuristic is presented to address large-scale server consolidation projects. In addition, a preprocessing method for server load data is introduced allowing for the consideration of quality-of-service levels. Extensive experiments were conducted based on a large set of server load data from a data center provider focusing on managerial concerns over what types of problems can be solved. Results show that, on average, server savings of 31 percent can be achieved only by taking cycles in the server workload into account.

Index Terms—Management of services delivery, modeling of resources, data center management services, optimization of services systems.

1 Introduction

Enterprises nowadays host central servers either in internal data centers or in those of commercial IT service providers. These data centers host most of the essential IT services (e.g., ERP modules, databases, or Web servers) on dedicated physical servers. The complex resource requirements of enterprise services and the desire to provision for peak demand are important reasons for overprovisioning. As a consequence, the average server utilization is typically very low, which incurs high investment and operational costs. The Gartner Group estimates that the utilization of servers in data centers is less than 20 percent [1]. However, efficiency in the production of IT services is key in an industry where services are becoming more and more standardized and where revenues are therefore decreasing. In particular, the bare savings of servers is of primary importance for most data centers regarding the cost of energy and cooling, which sometimes account for 40-50 percent of the total data center operation costs [2] (see also Section 7).

Server consolidation is an approach to the efficient usage of (physical) servers in order to reduce the total number of servers that an organization requires. The practice developed in response to the above-described server sprawl, a situation in which multiple, underutilized servers take up more space and consume more energy than can be justified by their workload. *Server virtualization* provides technical means to consolidate multiple servers leading to increased utilization of physical servers. The term refers to the abstraction of computing resources across many aspects of computing and has been used to describe different techniques. Server virtualization describes a virtual machine, which appears to a "guest" operating system as hardware, but is simulated in a contained software environment by the host system. This way, a single physical server can be partitioned into multiple virtual servers. Parallels, Xen, and VMware are only a few of the products on the market enabling server virtualization. Apart from higher server utilization levels, benefits are: reduced time for deployment, easier system management, and thereby overall lower hardware and operating costs. Virtualization has been a growing trend in the past few years, and it can now be considered an established tool that is nowadays used regularly in large-scale server consolidation projects with IT service providers [3].

Capacity planning, quality-of-service, and performance modeling have long been central research issues in Computer Science and Information Systems [4]. Increased outsourcing of IT services [5], as well as on-demand software provisioning for thousands or millions of users, has led to automation and further industrialization of data center operations. Production efficiency has become crucial for IT service management and capacity management is therefore a key component of related standards such as ITIL [6] or ISO/IEC 20,000 [7]. Target areas of capacity planning and performance modeling include file and database systems, computer networks, operating systems, fault-tolerant systems, and real-time systems (see, for example, [8], [9], and [4]). A traditional analytical approach to support capacity planning is queuing theory, allowing for the determination of response time, service time, server utilization, and many other metrics essential for capacity planning with dedicated servers [10].

• *The authors are with the Department of Informatics (I18), TU München, Boltzmannstr. 3, 85748 Garching/Munich, Germany. E-mail: speitkamp@mytum.de, bichler@in.tum.de.*

Manuscript received 1 May 2009; revised 18 Sept. 2009; accepted 12 Apr. 2010; published online 29 Apr. 2010.
For information on obtaining reprints of this article, please send e-mail to: tsc@computer.org, and reference IEEECS Log Number TSCSI-2009-05-0100. Digital Object Identifier no. 10.1109/TSC.2010.25.

Server consolidation is orthogonal to this type of capacity planning. Mostly, the workload and system requirements for different services are known or can be estimated with a high degree of confidence from historical workload data. The challenge is finding an allocation of source servers (*services* in the following) to target servers that minimizes costs, considering quality of service (QoS) requirements. This poses new and widespread capacity planning problems that will be the focus of this work. *Storage virtualization* and *storage-consolidation-related* problems are not considered in this paper. Server consolidation problems can be treated separately from the storage issue.

In this paper, we consider the problem of IT service providers hosting the IT services of multiple (also internal) customers. Based on the users' demands or historical workloads, the IT service provider needs to determine the allocation of these services to target servers using some predefined objective function (e.g., minimizing the amount or costs of servers used). We propose a server consolidation method that combines data analysis to characterize variations in workload traces and algorithms to optimally assign source servers to physical target servers. These types of problems are computationally hard, and can be found in almost all server consolidation projects. We will focus on problem sizes of up to 600 servers in this paper. We found that even in large enterprise-wide consolidation projects with thousands of servers, the overall problem was divided in individual consolidation projects with a few dozens or hundreds of servers for technical or administrative reasons. There can, however, be larger problems beyond 600 servers that will require different heuristic solutions than the ones described in this paper.

Cloud Computing and Infrastructure-as-a-Service (IaaS) have become very important trends recently. Note also that IaaS providers face similar resource allocation problems. The relation to Cloud Computing and IaaS will be discussed in Section 8.

One contribution of this paper is decision support for IT service providers following a design science approach [11]: We provide a mathematical formulation of widespread server consolidation problems, a complexity analysis showing that the fundamental problem is an NP-hard optimization problem, an LP-relaxation-based heuristic to solve this problem, and a data preprocessing method characterizing workload traces and deriving parameters for the decision model. The data preprocessing method allows the exploitation of seasonalities in the decision problem on the one hand and offers the possibility to adjust the quality of each service individually on the other hand.

The main contribution of this paper is an extensive experimental evaluation based on real-world workload traces that focus on the managerial questions for IT service providers. Given the specifics of the server consolidation problem, it is important for managers to understand which problem sizes can be solved exactly, which can be solved heuristically, and what the impact of various model parameters is on the solution quality and time. Both our IP-based decision models and our LP-based heuristic allow for the consideration of business constraints, but at the same time, solve even large-scale server consolidation problems with hundreds of servers as they can be found in practice in less than 20 minutes. We provide extensive sensitivity analyses describing how the time resolution of load data considered, or the size of the source servers relative to the target servers, impacts the size of the problems that can be solved. We also show how the risk parameters and the time resolution of load data considered impact the quality of the allocation. A key result is that only leveraging the daily seasonalities in the optimization accounts for 31 percent savings on average in the number of target servers needed as compared to the optimal solution ignoring daily variations in the workload. We also show that there are significant differences between the two types of applications with respect to the size of instances that could be solved.

While our work follows a design science approach, it does have managerial implications that go beyond the fact that IT service managers can now plan server consolidation projects in an automated way. Server consolidation has been embraced as a possibility to cut costs in the provisioning of IT. How much savings one can expect also depends on the characteristics of the workloads at hand. Our models allow the calculation of the potential savings in terms of hardware cost for a specific server consolidation problem at hand. Clearly, virtualization does provide a number of other benefits as well, such as easy management and migration of servers to other hardware. In this paper, however, we focus on savings due to the optimal allocation of virtual to physical servers.

The paper is structured as follows: In Section 2, we will provide a brief overview of virtualization techniques. In Section 3, typical server consolidation scenarios found in practice are presented. Then, in Section 4, two fundamental capacity planning problems are introduced and discussed regarding their computational complexity. These models are supplemented by important constraints in Section 4.2. Section 5 describes the experimental setup, while Section 6 discusses the results of computational experiments based on real-world data sets. In Section 7, further managerial implications are presented. Section 8 provides an overview of related work, and in Section 9, we draw conclusions.

2 Virtualization

Virtualization is performed on a given hardware platform by a control program, which creates a simulated virtual machine for its "guest" software. The "guest" software, which is often itself a complete operating system plus applications, runs as if it was installed on a dedicated server. Typically, many such virtual machines are simulated on a given physical machine. Virtualization is an umbrella term for many techniques. For example, symmetric multiprocessor (SMP) servers can be subdivided into fractions, each of which is a complete server and able to run an operating system. This is often described as physical or logical *hardware partitioning*. Example products are HP nPAR, or IBM DLPAR. *Software virtualization* includes approaches on or below the operating system level, or on the application level. So-called hypervisor software creates virtual servers whose physical resource use can be adjusted dynamically, enabling multiple isolated and secure virtualized servers on a single physical server. The market for software virtualization products is growing and a description of products and techniques is beyond the scope of this paper.

A typical hypervisor allows the creation and execution of multiple virtual servers simultaneously. Each virtual server

instance can execute a guest operating system, such as Windows or Linux. As such, a hypervisor is a software that allows one physical server to run numerous operating systems plus applications simultaneously. These products bridge calls to network adapters, CD-ROM readers, hard disk drives, and USB devices. Some products even support migration of virtual servers across multiple host machines at runtime. Administrators can monitor the runtime data of logical and physical software landscapes, start, stop, or relocate servers, and assign hardware resources to application services automatically or manually. There are a number of use cases demonstrating how virtualization can be used in a data center [12]. These new technical possibilities require new methods for capacity management.

Virtualization causes overhead in terms of additional CPU cycles necessary for the hypervisor, every virtual machine, and also a workload-dependent component. In the subsequent decision models, we will consider overhead in a data preprocessing step with simple additive or multiplicative factors. Virtualization overhead will be further reduced with current hardware developments which support certain virtualization features.

Virtualization has benefits that are beyond savings in investment and energy costs through consolidation. For example, managing and migrating software from one server to another becomes much easier. However, virtualization software also comes at a cost. Whether the benefits outweigh the (actual) cost is a topic frequently discussed in reports by market research companies. It heavily depends on the cost of introducing virtualization software and the technical progress in the field.

3 Server Consolidation

Server consolidation is an approach for the efficient usage of computer server resources in order to reduce operating costs. These costs mainly originate from space and energy consumption (for servers and data center cooling), data center maintenance, server administration, and purchasing costs for servers. Consolidation is used in situations in which multiple, underutilized servers take up more space and consume more resources than can be justified by their workload. We refer to server consolidation as the process of combining the workloads of several different (source) servers or services and assigning them to a set of target servers.

3.1 Decision Problems

In the following, we present three widespread scenarios in server consolidation tasks which our decision models apply to: The objective of the decision models is to minimize the sum of server costs, which might be purchasing, maintenance, administration, or the sum of them. Depending on the scenario, the models provide

- the information on *how many* and *which servers* are required for the given set of services (i.e., virtual servers), given a set of potential (hardware) servers (all scenarios),
- *an optimal allocation* of services to servers with respect to the objective function (all scenarios), and
- possibly additional support for decisions that aim to minimize investment and/or operational costs (scenarios 1 and 2).

The *first decision scenario* refers to an investment decision, i.e., the data center operator wants to switch to a new technology of hardware, such as a new generation of multicore processors, and therefore, has to decide on how many machines to buy. There may be alternative server types he wants to evaluate against each other. In this first scenario, coefficient c_i represents the purchasing costs of single server i from a set of potential servers I (with $i \in I$). Alternatively, c_i may represent the total cost of ownership for each single server i, including energy and administrative costs, etc. (for a list of symbols, see Appendix D, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25). In the case of alternative server types to choose from, respective models may be configured, with each model offering servers of one specific type. The objective values of the optimization models (i.e., the total cost of the servers) can help to decide on a configuration. Of course, the different server types can also be included in a single model. Sometimes, additional strategic considerations, such as reusability of hardware with future services, customers, technologies, and planning periods do play a role. In these situations, the decision model still serves as a decision support tool complementing the above considerations.

The *second decision scenario* refers to the case where servers have already been purchased and purchasing costs are considered as "sunk costs." Nevertheless, the use of existing servers is required to be as efficient as possible, i.e., the operational costs should be kept to a minimum. Note that the costs of electrical power supply for servers and cooling accounts for a large part of operational costs in data centers (see Section 7). Therefore, in this second scenario, the costs c_i represent the operational costs per server, e.g., comprising energy, cooling, and administrative costs.

As a special case, a *third decision scenario* originates from the a priori assumption of identical servers in terms of both costs and capacities, as is the case with a rack of identical blade servers. Here, the costs c_i may just represent the information concerning which subset of servers should be used if not all servers are required. This may be of particular importance when optional technical allocation constraints have to be considered. Here, the decision models return no cost value but, for instance, the minimum number of servers required and an allocation that satisfies additional technical constraints.

3.2 Available Data in Data Centers

Data centers reserve certain amounts of IT resources for each single service or customer. For this purpose, CPU capacity may be measured in SAPS or HP Computons, memory in Gigabyte, and bandwidth in Megabits per second. SAP Application Performance Standard (SAPS) is a widespread hardware-independent unit that describes the performance of a system configuration in the SAP environment. It is derived from the Sales and Distribution Benchmark, where 100 SAPS is defined as 2,000 fully business processed order-line items per hour [13].

Data centers typically log the workloads of their servers over time. These include data about CPU, memory, and bandwidth utilization based on measurements every 5 minutes or every hour, for example. Usually, resource demand of this sort has seasonal patterns on a daily, weekly, or monthly basis. For example, payroll accounting is

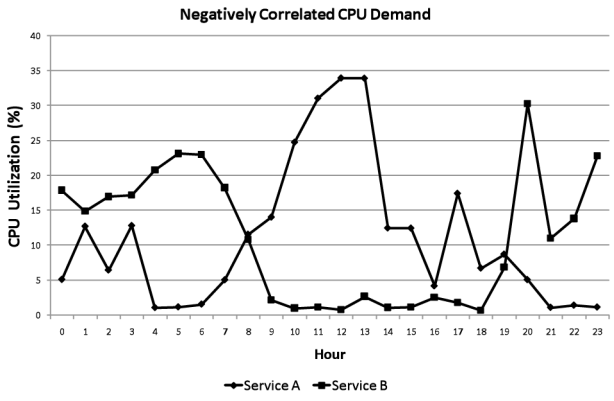


Fig. 1. Workload of two services with complementary workload during a day.

performed at the end of the week, while demand for an OLAP application has a daily peak in the morning hours, when managers access their daily reports.

Consolidation can leverage these cycles and attempts to assign those services on a physical server whose demand is negatively correlated, i.e., workload peaks at different times of the day or week (see Fig. 1 for a real-world example of two services). An analysis of the large set of workload traces that we have received from our industry partner can be found in Appendix B, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25. Although we did find different daily and also weekly cycles, there was hardly any trend component in any of the different types of services. One reason might be that IT managers tend to preselect services for consolidation that exhibit no or little trend. Nevertheless, workload traces can change in extended time periods and it is important for IT service managers to monitor workload developments in the data center regularly and also reallocate servers if necessary. The models in the following section provide decision support for the initial and for subsequent allocation decisions.

4 Problem Formulation

First, we introduce two optimization models that formalize the consolidation problem based on the cost and workload data as they are typically available in data centers. Then, we analyze the computational complexity of this problem and algorithmic approaches to solve the problem for practical problem sizes.

4.1 Static Server Allocation Problems

A basic server consolidation problem is the **Static Server Allocation Problem (SSAP)** (for a list of abbreviations, see Appendix E, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25). Here, the IT service manager needs to consolidate servers by assigning *services* (i.e., virtual servers) to (physical) target *servers* so as to minimize the number of servers used or minimize overall server costs. We start out with this basic model to introduce the mathematical structure of the fundamental decision problem that will be extended later on.

Suppose that we are given n services $j \in J$ that are to be served by m servers $i \in I$. Different types of IT resources $k \in K$ may be considered (e.g., CPU and memory, bandwidth). In this model, for service j, the customer orders u_{jk} units of resource k (e.g., SAPS, HP Computons, Megabyte, etc.) and each server has a certain capacity s_{ik} of resource k. y_i are binary decision variables indicating which servers are used, c_i describes the potential cost of a server, and x_{ij} describes which service is allocated to which server. Considering multiple types of resources, such as memory and bandwidth, the problem can be formulated as follows (see (1)):

$$\begin{aligned} \min \quad & \sum_{i=1}^{m} c_i y_i \\ s.t. \quad & \\ & \sum_{i=1}^{m} x_{ij} = 1, && \forall j \in J, \\ & \sum_{j=1}^{n} u_{jk} x_{ij} \leq s_{ik} y_i, && \forall i \in I, \forall k \in K, \\ & y_i, x_{ij} \in \{0,1\}, && \forall i \in I, \forall j \in J. \end{aligned} \tag{1}$$

The objective function minimizes server costs, while the first set of constraints makes sure that each service is allocated exactly once, and the second set of constraints ensures that the aggregated workload of multiple services does not exceed the capacity of a single server.

The SSAP represents a single service's resource demand as constant over time. In the following we want to consider variations in the workload, and time is divided into a set of intervals T indexed by $t = \{1, \ldots, \tau\}$: Cyclic workloads over time are now represented in the matrix u_{jkt} describing how much capacity service j requires from resource type k in time interval t. Based on this matrix, we can reformulate the second set of side constraints to

$$\sum_{j=1}^{n} u_{jkt} x_{ij} \leq s_{ik} y_i, \qquad \forall i \in I, \forall j \in J, \forall t \in T. \tag{2}$$

We will call this model variation the Static Server Allocation Problem with variable workload (SSAPv).

The number of servers an IT service provider can save using virtualization has been one of the main sales arguments for software vendors in this field. This obviously depends very much on the level of interference of the respective workload traces (i.e., time series). SSAPv helps to quantify how much can actually be saved compared to dedicated server hosting based on historic demand data. The coefficients u_{jkt} depend on the load characteristics of the servers to be consolidated. Section 5.3 will describe how these coefficients can be derived. In general, we assume $u_{jkt} \leq s_{ik}$.

The target servers in SSAPv may be the same type of physical machine as the source servers or different machines with different resource capacities (e.g., CPU architectures). If the computer architecture of source and target servers is different, utilization parameters need to be converted. Such conversion rates can easily be estimated from server tests and benchmark studies [14].

4.2 Extensions of the Static Server Allocation Problems

In the following, we present constraint types, which have been elicited from practical applications: All the model extensions have been incorporated in a software tool (vPlan), which is now in use with our industry partner:

1. Max No. of Services Constraint: For any target server i, the maximum number of services n_i may be defined as that which must not be exceeded by the resulting allocation. The purpose of this constraint may be to limit administrative time and effort in the event of a server failure, etc.,

$$\sum_{j \in J} x_{ij} \leq n_i, \qquad \forall i \in I. \tag{3}$$

2. Separation Constraints: A subset of services S may have to be allocated on different servers for security or other technical reasons

$$\sum_{j \in S} x_{ij} \leq 1, \qquad \forall i \in I. \tag{4}$$

3. Combination Constraints: A subset of services S may have to be allocated on the same server due to enhanced interapplication communication, or the same operating system requirements. With e being an element of S:

$$-(|S|-1) \cdot x_{ie} + \sum_{j \in S - \{e\}} x_{ij} = 0, \quad e \in S, \forall i \in I. \tag{5}$$

4. Technical Constraints and Preassignment Constraints: A subset $R \subseteq I$ of servers may exhibit a technical feature, e.g., a particular (storage area) network access. A service j may require this particular server attribute. Thus, the service must only be allocated to one of the servers in R providing this attribute. If $|R| = 1$, this constraint will be called *preassignment* constraint

$$\sum_{i \in R} x_{ij} = 1, \qquad j \in J. \tag{6}$$

5. Limits on the number of reallocations. In the event that SSAP or SSAPv has to be solved repeatedly, for subsequent planning periods, for example, it may be required that the current allocation does not change too much in order to limit administrative costs. The number of migrations of services already allocated in the present solution may therefore be restricted. Let X be the set of x_{ij} with $x_{ij} = 1$ in the current allocation, and let n be the number of already allocated services (i.e., $n = |X|$) and let r be the number of reallocations allowed

$$\sum_{x_{ij} \in X} x_{ij} \geq n - r. \tag{7}$$

4.3 Algorithms and Computational Complexity

Complexity analysis can help us to understand whether we can hope to find exact solutions even for large instances. Unfortunately, as we can show, the problem is strongly NP-hard, even for the simplest case with only one resource and one time interval.

Theorem 1. *SSAP is strongly NP-hard.*

Corollary 1. *SSAP is strongly NP-hard even when only one resource is considered, and all servers have the same cost and the same capacity.*

We provide a straightforward proof in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25, by reducing SSAP to the multidimensional bin packing problem (MDBP). Bin packing was described by Garey & Graham [15], who have shown that it is strongly NP-hard. Many variations of bin packing have attracted a considerable amount of scientific interest over the last few decades, partly due to their relevance in diverse scheduling applications.

For exact solutions, one can use traditional integer programming techniques such as branch-and-bound algorithms. Knowing that the problem is NP-hard is important, but it does not necessarily mean that it is intractable for practical problem sizes. So, for IT managers, it is important to understand which problem sizes can be solved exactly, and how far one can get with heuristic solutions, both in terms of problem size and solution quality.

For the simplest version of SSAP with only one resource and no side constraints, one can apply heuristics as they have been developed for the bin packing problem. Two well-known heuristics are the *best-fit decreasing* (BFD) and the *first-fit decreasing* (FFD) approach [16]. It was shown that the solutions they produce require not more than $11/9\ \mathrm{OPT} + 1$ bins (with OPT being the number of bins required by an optimal solution) [17], [18]. For SSAP with only a single resource, this means that in the worst-case, one would need around 22 percent more target servers than necessary using the FFD.

MDBP is harder to solve and polynomial-time approximation schemes (PTAS) with worst-case guarantees on the solution quality have been published only recently. The first nontrivial result was produced by Chekuri and Khanna [19] who gave a polynomial-time algorithm that, for any fixed $\varepsilon > 0$, delivers a $(1 + \varepsilon \mathrm{d} + \mathrm{O}(\log \varepsilon^{-1}))$-approximate solution for constant d. The approach is based on solving a linear programming relaxation for the problem. The basic feasible solution would make fractional assignments for at most dm vectors in d dimensions and m bins or servers. In a second step, the set of fractionally assigned vectors is assigned greedily. Recently, Bansal et al. [20] showed a polynomial-time randomized algorithm with approximation guarantee arbitrarily close to $\ln d + 1$ for fixed d. For small values of d, this is a notable improvement.

5 Experimental Setup

Complexity results provide general worst-case results, but provide little managerial insight into when the approach is applicable for server consolidation and what problem sizes can be solved for specific inputs of this problem in due time. As we will see, the resource demands of different types of applications or the number of time intervals heavily impact the problem sizes that can be solved. Also, it is important to understand the impact of parameters, such as the required service level, on the solution quality. In the following, we

will provide a detailed experimental analysis focusing on these application-specific questions.

5.1 Experimental Data

From our industry partner, we obtained two extensive sets of workload data. The first set contains 160 traces for the resource usage of Web/application/database servers (W/A/D). The second set contains 259 traces describing the load of servers exclusively hosting ERP applications. Both sets contain data of three consecutive months measured in intervals of 5 minutes.

We provide a statistical analysis of the workload data in Appendix B, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25. The main characteristics of the data relevant for the consolidation of enterprise applications are the resource demands in terms of CPU and memory. We found strong diurnal seasonalities with nearly all servers and some weekly seasonalities, but almost no long-term seasonal patterns or trend components in the workload traces. One reason is that system administrators typically select those services for consolidation that do not exhibit a significant trend and are more predictable in their overall resource demand over time. Many applications that are hosted in data centers exhibit a low but, as we could see, quite regular utilization of server resources [1]. Of course, one will also find services exhibiting strong trends or long-term seasonal patterns. With or without such services in a consolidation project, it is recommended to monitor developments in the service workloads and, if necessary, reoptimize and reallocate the servers on a regular basis.

CPU is well known to be the bottleneck resource for these types of applications under consideration [21], [22], [23]. This obviously depends on the capacity ratio of CPU power to memory size of the target servers. This ratio causes CPU to be the bottleneck resource with our applications. For the type of applications in our data set, this was also the only resource that was considered in consolidation planning of our industry partner. Without loss of generality, we restrict our attention to CPU workload in our experiments. The method and toolset have been applied to multiple resources (CPU and memory), in the past as well, without any changes.

It is interesting to note that CPU demand of the ERP services is typically significantly higher than that of the W/A/D services, which included a number of very different applications (see Table 2, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25). For the broad class of W/A/D services, it was not straightforward to define recognizably different classes based on CPU demand and functionality (e.g., portal server, domain server, CRM application, etc.). We decided to treat ERP and W/A/D services as different classes throughout our analysis. While this does not require different models or algorithms, the ratio of resource demands to the capacity of the physical server has an impact on the problem sizes one can solve (see Section 6.1).

For our experiments, we generated sets of time series containing different numbers of W/A/D or ERP services by sampling with replacement from the original W/A/D or ERP set. The same problem instances were used when models (SSAP versus SSAPv) and algorithms (see Section 5.2) were compared with respect to runtime or solution quality.

5.2 Allocation Algorithms

For SSAP with only one resource, we have used branch-and-bound (B&B), as well as first fit (FF) and first-fit decreasing (FFD) heuristics. FFD, for example, operates by first sorting services to be assigned in decreasing order by volume, and then assigns each to the first server in the list with sufficient remaining capacity. FF does not include sorting.

SSAPv is more complicated as it involves several time intervals or dimensions. Therefore, we propose an LP-relaxation-based heuristic. As compared to the PTAS described in Chekuri and Khanna [19], we also use the results of an LP-relaxation in the first phase, but use an integer program in the second step to find an integral assignment of those services that were fractionally assigned in the LP relaxation. This has two reasons: First, in particular, for W/A/D services, the number of servers used was very low compared to the number of services, leading to a low number of integer variables. As we will see, the integer program in the second phase was responsible for only a small proportion of the overall computation time. Second, an advantage of the LP-based approach and the IP in the second phase (as compared to simple generalizations of FF or FFD) is the fact that constraints (see Section 4.2) can easily be integrated. We will call this LP-relaxation-based procedure the *SSAPv Heuristic*.

For SSAP B&B, SSAPv B&B, and the SSAPv Heuristic, the number of servers used does have a significant impact on the computation time. Each additional server increases the number of binary decision variables by $1 + n$. In order to keep the number of decision variables as low as possible, we have used a specific iterative approach for all three algorithms. In the first iteration, we solved the problem with m being the lower bound number of servers (*LB*) first. *LB* can be calculated as in (8), with s being the server capacity assumed to be equal for all servers

$$LB = \left\lceil 1/s \cdot \max_{t \in T} \left(\sum_{j \in J} u_{jt} \right) \right\rceil. \tag{8}$$

The lower bound is based on the assumption that services could be allocated in fractional quantities of the demand of the target servers. Therefore, no integral solution can be lower. If this problem turns out to be infeasible, m is incremented by 1. Infeasibilities are typically detected very quickly, while feasible solutions with a high number of servers m can take a very long time. This is repeated until a feasible solution is found. The first feasible solution found in the B&B search tree is obviously an optimal solution, minimizing the number of target servers. The computation times reported in our experiments will summarize all iterations.

Note that with identical servers (i.e., equal costs and equal capacities), there are at least *m!* equivalent solutions for SSAP and SSAPv, which will cause huge search trees in traditional B&B algorithms. This problem is sometimes referred to as "symmetry" in integer programming [24]. A straightforward approach that reduces the computation time considerably is to differentiate the cost of servers c_i by a small amount ε. In our experiments, c_i was set to $i (i = 1, \ldots, m)$.

5.3 Data Preprocessing

An integral part of the server consolidation method is the data preprocessing. Data preprocessing results in a discrete characterization of daily patterns in the workload traces and allows to solve the allocation problem as a discrete optimization problem (see Section 4). The number of time intervals τ considered impacts the number of constraints in our models. More time intervals might better exploit complementary resource demands and thus lead to denser packing. Assuming that there is seasonality in the workload traces, one can derive estimators for the resource demands in different discrete time slots that reflect the variations of workload over time.

Our experimental data represent the average CPU utilization of consecutive 5-minute intervals measured over 3 months for 160 W/A/D and 259 ERP services. The original workload traces are denoted by u_{jkt}^{raw}, whereas u_{jkt} will be an estimator for the demand of resource k of a service j in interval t that is derived from the original workload traces.

In the following, we will describe a two-step process to derive the parameters u_{jkt} for our optimization models from the original workload traces. In the first step, we will derive an estimator for individual 5-minute time intervals, while in the second step, we will aggregate these intervals to reduce the number of parameters for the optimization.

This procedure is based on the analysis of the workload traces (see Appendix B, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25) that suggested leveraging daily seasonalities in the time series. We will describe a day as a period of observation. Let p denote the number of *periods* contained in the load data (in our case, $p = 92$ days in load traces of 3 months). A single period is then described by τ' *intervals*, which are individual measurements in the raw data (in our case, $\tau' = 288$ five-minute intervals per day). We can now derive the empirical distribution of a random variable U_{jkt} for each interval $t = 1, \ldots, 288$, each of which has 92 observations. For example, U_{jkt} might contain all measurements for the time interval from 12:00 p.m. to 12:05 p.m. across 3 months

$$U_{jkt} = \bigcup_{q=0}^{p-1} \{u_{jk(q\cdot\tau'+t)}^{raw}\}, \quad \forall j \in J, \forall k \in K, \quad \forall t \in \{1, \ldots, \tau'\}. \tag{9}$$

In Fig. 2, the CPU requirements of a service (in custom unit $SPU = 50$ *SAPS*) are plotted for all 24 hours of a day. Hence, each parallel to the y-axis captures a sample of about 92 values according to the 92 days of measurements for each 5-minute interval of the day.

These empirical distributions of U_{jkt} for the various intervals enable us to derive an estimator u_{jkt}. This estimator takes into account the risk attitude of the IT service manager. For instance, the 0.95-quantile of U_{jkt} is an estimator for the resource requirement of service j where 95 percent of all requests can be satisfied.

However, even if at one point in time, the workload of a particular service exceeds the capacity of a server, this might only mean that the response time increases, not that the request is canceled. Also, depending on the correlation, adding multiple such services or random variables might lead to a reduction of variance. The experimental results (Section 6) will provide more information on how this setting impacts the quality of service.

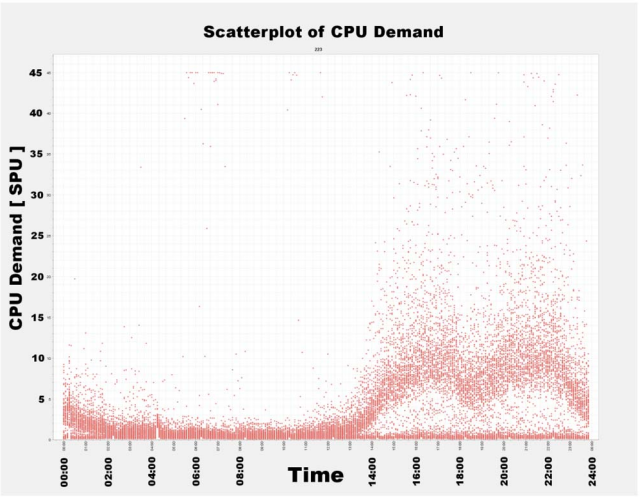


Fig. 2. Workload profile of an ERP service.

In order to reduce the number of model parameters, it is useful to consider coarser-grained intervals (e.g., hours). This can be achieved by aggregating the estimators of 5-minute intervals in a second step. For example, 12 adjacent five-minute intervals may be aggregated to a 1-hour interval by choosing the maximum of these 12 five-minute interval values. This ensures that the service level is maintained in all 5-minute time intervals. For SSAP, where we can only consider a single utilization parameter u_{jk}, the maximum of all 5-minute intervals was considered.

5.4 Experimental Design

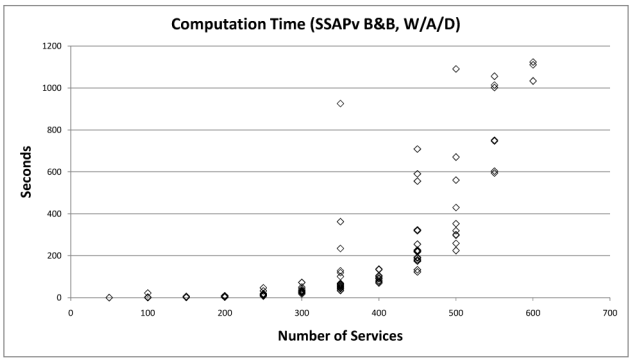
The experiments were designed to answer managerial questions as to which problem sizes can be solved with which quality, and how these variables depend on the models, algorithms, and parameter settings above. It should provide IT managers with guidance in what methods and model parameters are appropriate for a consolidation task. In our experiments, we use the following treatment variables:

- model (SSAP or SSAPv),
- algorithm,
- service type (W/A/D or ERP services),
- number of services,
- server capacity (CPU capacity in SAPS),
- risk attitude (as quantiles of the workload distributions),
- number of time intervals considered in SSAPv (e.g., 1-hour versus 6-hour intervals), and
- sensitivity with respect to additional allocation constraints (see Section 4.2).

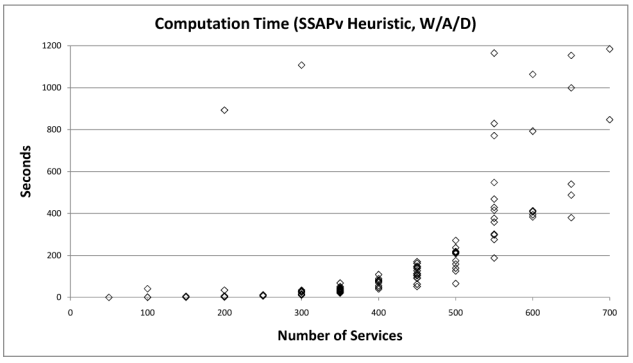
Depending on the model, we analyzed different solution algorithms. SSAP (with only one resource) was solved using Branch & Bound, First Fit, and First Fit Decreasing, while SSAPv was solved using a Branch & Bound and the SSAPv Heuristic described in Section 5.2.

5.5 Dependent Variables

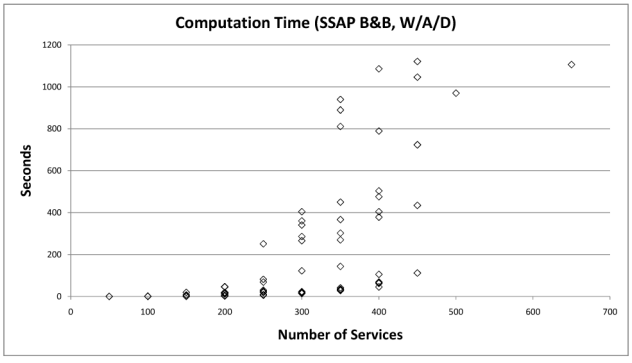
For the analysis of the solution quality, the lower bound number of servers *LB* and the actual number of servers



(a)



(b)



(c)

Fig. 3. Computation times for SSAP B&B, SSAPv B&B, SSAPv Heuristic; server cap. $= 5{,}000$, W/A/D Services.

required were measured. In addition, we measure computation times of the algorithms employed.

The open source solver lp_solve 5.5.0.9 was used for the revised simplex and the B&B algorithm. To double-check the results, we also used the COIN-OR CBC branch-and-cut IP solver with the CLP LP solver. All other software, including the implementation of the FF and FFD heuristics, was implemented in Java 1.5.0. Experiments were executed on workstations running Windows XP Professional (Service Pack 2) on an AMD Athlon XP $2{,}100+$ Model 8 with 2.1 GHz. The timeout for all single lp_solve runs (both Simplex and B&B) and for the single FF and FFD runs was set to 20 minutes. First, this has already allowed us to solve very large consolidation problems of up to 700 servers, i.e., we could solve those sizes that were considered practically relevant by our industry partner. Second, respective tools are used in an

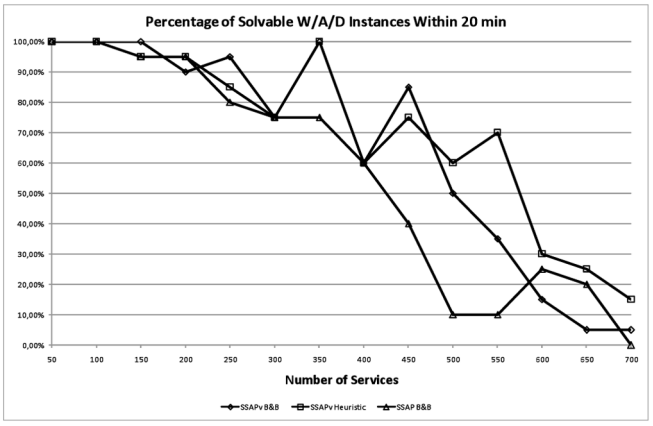


Fig. 4. Percentage of solvable W/A/D instances within 20 minutes regarding different algorithms.

iterative manner and field consultants want to explore different settings in a matter of minutes rather than waiting for hours. It also allowed us to conduct the large number of experiments that are summarized in the following. Nevertheless, it is interesting to understand what happens to larger problem sizes that have not been time-constrained. Therefore, a number of instances were analyzed without setting a timeout, or one of 24 hours only.

6 Results

In the following, we will describe selected experimental results. Unless otherwise stated, we will report the results of W/A/D instances with 24 time intervals based on the 95th percentile. We will first discuss computation time and then solution quality with respect to different treatments.

6.1 Computation Time Depending on Problem Size

Fig. 3 shows the computation time of the different algorithms. For each of the different numbers of services (x-axis), 20 instances have been sampled (with replacement) from the set of *W/A/D services*. These 20 instances were then solved using the different algorithms. Here, we examine a target server capacity of 5,000 SAPS which was identical to the environment of our industry partner. Furthermore, we examine 24 time intervals and determine service demand based on the 95th percentile of 5-minute intervals.

Fig. 3 illustrates the computation time of different instances, while Fig. 4 shows the proportion of all 20 instances that could be solved in time. Beyond 500 services, the SSAPv Heuristic is capable of solving more instances within 20 minutes than the SSAPv B&B. In total, the SSAPv Heuristic could solve 70 percent and SSAPv B&B could solve 65 percent of all 280 instances within 20 minutes.

It is interesting to note that most of the time with the SSAPv Heuristic was spent on the initial LP relaxation, while the subsequent B&B could be solved in a few seconds due to a low number of fractional assignments, which is restricted by the number of servers multiplied by the number of time intervals considered. The runtimes of the SSAP B&B exhibit a larger variance beyond 300 services than the SSAPv solutions. This is due to the fact that SSAP needs more target servers, and consequently, has more decision variables but fewer constraints than SSAPv. Accordingly, Fig. 4

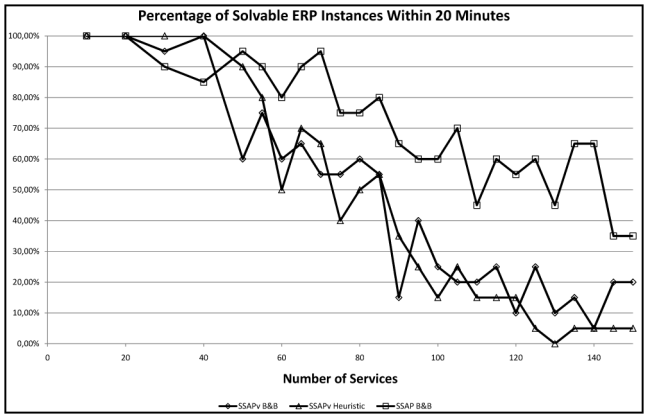


Fig. 5. Proportion of solvable ERP instances within 20 minutes (5,000 SAPS).

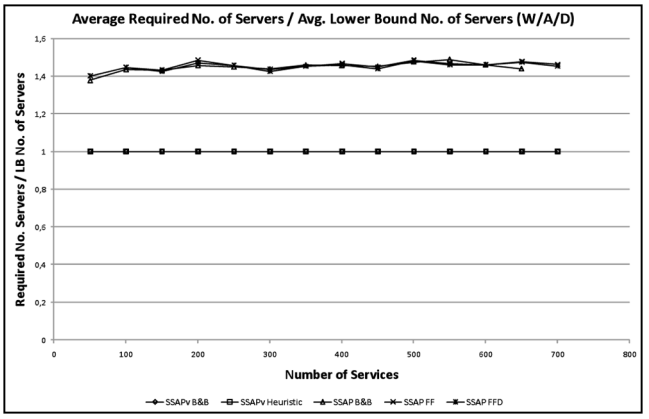


Fig. 6. Solution quality of Algorithms for W/A/D services and varying number of services (5,000 SAPS).

shows that SSAP B&B often solves fewer instances than the SSAPv algorithms when used with W/A/D services (in total, 56 percent of 280 instances).

We have also analyzed runtimes of large instances without a timeout. A few instances with 700 or even 800 services could be solved in 100 minutes, while most took much longer. Instances with 900 services or above could not even be solved within 24 hours.

For *ERP services*, we have also used the same server capacities of 5,000 SAPS. A notable difference is that the CPU demands of ERP services are significantly higher than those of W/A/D services. As we will see, this does have a strong effect on the empirical hardness of the problem. Fig. 5 shows that in this case, SSAP B&B solves more instances within 20 minutes than the SSAPv algorithms (71 percent of all instances). SSAPv B&B solved 45 percent, and the SSAPv Heuristic 43 percent of all instances. The SSAPv Heuristic, however, was faster in 72 percent of those cases where at least the SSAPv B&B or the SSAPv Heuristic produced a solution within 20 minutes. Compared to the W/A/D services, we could only solve much smaller instances before the solver hit the 20-minute timeout.

One explanation is the number of servers, and consequently, decision variables needed. Due to the resource demands of ERP services, on average, only 8.47 services could be assigned to a target server, as compared to 24.06 W/A/D services that would fit on such a server on average. Consequently, the number of decision variables increased more than the number of constraints, compared to a similar instance with W/A/D workload traces. For example, for an instance with 150 ERP workload traces, we had 2,567 decision variables and 558 constraints; 150 W/A/D traces resulted in only 985 decision variables and 306 constraints. Consequently, we could only solve problems with much smaller numbers of services. For ERP services, we have tested larger instances. Several instances with up to 250 services could be solved within 5 hours, instances with 275 and more services could not be solved within 24 hours.

For both W/A/D and ERP services, the SSAP First Fit and SSAP First Fit Decreasing show very low computation times of less than 1 minute for instances of up to 1,200 services. Also here, ERP instances take longer to solve than W/A/D instances, since more target servers are required. Note that SSAP First Fit and First-Fit Decreasing can both only consider a single resource and a single time dimension.

6.2 Solution Quality Depending on Problem Size

For the experiments, we assume servers with equal capacity. In (8), we have already introduced the lower bound number of servers required (*LB*). The lower bound is calculated as if services could be fractionally assigned to servers. Note that *LB* depends on the number of time slots considered. Considering more time slots tends to result in a smaller lower bound of servers. In our experimental setting, the *LB* for W/A/D led to an upper bound pack ratio (n/LB) of 24.06 services per server on average. Fig. 6 shows the factor Q by which the computed number of required servers exceeds the lower bound number of servers, i.e., $Q = 1/LB \cdot \sum_{i=1}^{m} y_i$. We will refer to this excess ratio as *solution quality*—that is, the closer Q is to 1, the better the solution is, i.e., the closer it is to the lower bound.

SSAPv B&B always achieved an allocation with the lower bound number of servers (i.e., $\bar{Q}_{SSAPvB\&B} = 1.0$). The solution quality of SSAP B&B and SSAP solved with the first-fit-decreasing (FFD) heuristic is on average 1.45 times the lower bound ($\bar{Q}_{SSAPB\&B} = \bar{Q}_{SSAPFF} = 1.45$). FF and FFD achieved the same solution quality most of the time ($\bar{Q}_{SSAPFF} = 1.46$). In other words, SSAPv with 24 time intervals per day required, on average, 31 percent fewer servers than SSAP (with only one time interval per day). For example, a consolidation problem with 250 W/A/D services resulted in an allocation of 10 servers with SSAPv, but it required 15 target servers with SSAP. The average quality of SSAPv Heuristic was equal to the optimal solution ($\bar{Q}_{SSAPvHeuristic} = 1.0$).

These results are in line with our analysis of ERP services. As mentioned earlier, problem instances with ERP load traces are harder to solve than with W/A/D workload traces. However, we found a similar solution quality for SSAPv and SSAP for ERP services compared with W/A/D services: The SSAP B&B solution was, on average, 1.48 times worse than *LB* ($\bar{Q}_{SSAPB\&B} = 1.48$), SSAP FFD was, on average, 1.49 times worse than *LB* ($\bar{Q}_{SSAPFFD} = 1.49$, $\bar{Q}_{SSAPFF} = 1.50$), and the average quality of SSAPv B&B solutions $\bar{Q}_{SSAPvB\&B}$ was equal to 1.0, and $\bar{Q}_{SSAPv\ Heuristic} = 1.01$ (with $n/LB = 8.47$ on average).

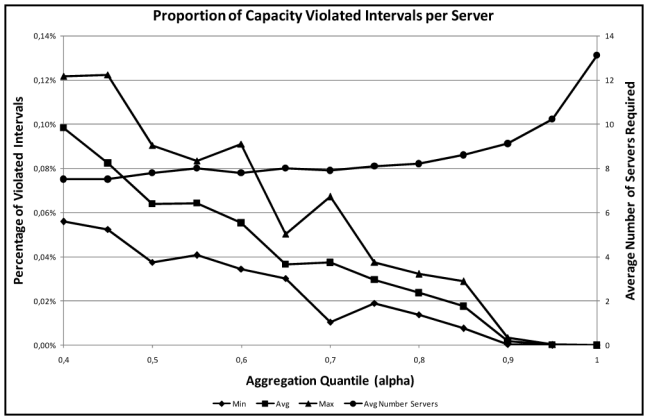


Fig. 7. Min, Avg, Max percentage of capacity violations per server for 10 problem instances with 250 W/A/D services depending on aggregation quantile. Server capacity $= 5{,}000$ SAPS.

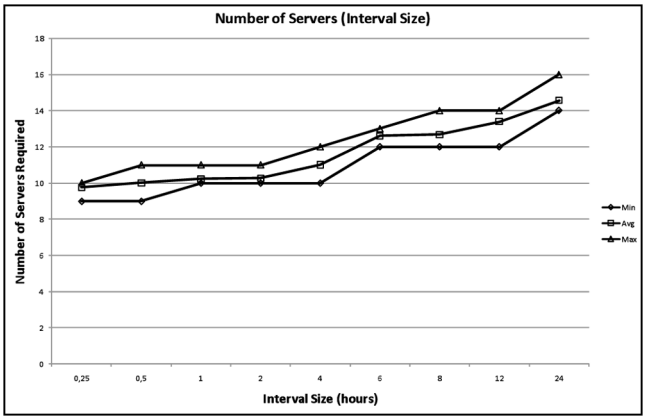


Fig. 8. 250 W/A/D services, with 10 sampled instances per interval size, cap $= 5{,}000$, SSAPv B&B.

6.3 Impact of Risk Attitude on Solution Quality

So far, we have always assumed the decision maker to select the 95th percentile in data preprocessing. We call this parameter the decision maker's risk attitude or the aggregation quantile. In other words, if each service was considered in isolation, 95 percent of the historical service demand would have been satisfied without delay at this capacity. It is not clear whether such a parameter setting leads to actual overbooking of server resources or if it is a more conservative estimate due to a reduction in variance when multiple services are assigned to the same server. The level of overbooking depends on the aggregate demand of services on a server and their correlation.

One way to analyze the effect of different parameter settings is the analysis of capacity violations based on the historic workload traces that are assigned to a single server.

We applied different quantiles $(0.4, 0.45, \ldots, 1)$ of the sets U_{jkt} (see Section 5.3), solved 10 different consolidation problems of 250 W/A/D services each using SSAPv B&B, and measured the capacity violations on all servers. The number of capacity violations is depicted as average, minimum, and maximum percentage of the 10 instances of the overall number of intervals (see Fig. 7).

Even using a 0.4 quantile, only in around 0.12 percent of the 5-minute intervals in 3 months did the resource demands exceed the server capacity. It depends heavily on the type of application whether such capacity violations matter. The 0.8 quantile resulted in 0.025 percent capacity violations, which translates on average to 6.2 five-minute intervals per server, where the CPU demand exceeded the capacity in 3 months. In addition to a reduction of variance through consolidating multiple services on a server, the low number of capacity violations that result from these parameter settings is mainly to be explained by data preprocessing. When aggregating from 5-minute intervals to a 60-minute interval, the maximum of 12 five-minute interval quantiles is used in order to ensure that enough capacity is available in all 5-minute intervals.

Fig. 7 also shows the number of servers needed for a particular aggregation quantile. Interestingly, the average number of servers does not change significantly between the 0.4 and 0.8 quantile, whereas the number of capacity violations decreases. In contrast, the number of required servers increases for quantiles between 0.8 and 1, which is based on the fact that CPU utilization has a long tail with observations where the service needed a lot of CPU. To account even for these bursts requires an extra number of servers.

6.4 Influence of the Interval Size Considered

The second model parameter is the granularity of time intervals, which impacts the number of constraints in SSAPv, and the solution quality. Fig. 8 shows the average, minimum, and maximum number of required servers for 10 different sets of 250 W/A/D services. Note that an interval size of 24 hours (i.e., a single interval) reduces SSAPv to SSAP. On average, there is almost no improvement in the number of servers needed, whether one considers 15-minute or 2-hour intervals. However, while we only required 10 servers on average for the 2-hour interval, the single 24-hour interval (SSAP) required between 14 and 16 servers.

6.5 Influence of Additional Allocation Constraints

In Section 4.2, we introduced a number of additional side constraints that can become relevant in server consolidation projects. It is easy to incorporate these constraints in LP-based heuristics and the IP formulation, as compared to SSAP FF. These side constraints, however, will impact the solution time and solution quality (see Appendix C, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSC.2010.25, for details).

If managers set an upper bound on the number of services per server (1), the number of servers typically increases. As an increasing number of servers leads to more decision variables, it also has a negative impact on computation time. The combination and separation constraints (2/3) had little effect on the solution quality for W/A/D services, even for large numbers of such constraints. They had a negative impact on computation time. Similarly, the technical constraints (4) had little effect on the number of servers needed, and the computation time decreased, as long as the number of required servers did not increase.

6.6 Summary

The following provides a summary of the main results and of the experimental analysis relevant to IT managers:

1. **The server consolidation problem (SSAP and SSAPv) is an NP-hard optimization problem.**

   The problem can be reduced to the vector bin packing problem, which is strongly NP-hard.

2. **The solution time depends on the resource demand of services relative to server capacity.**

   There was a significant difference between W/A/D and ERP services in our experiments with respect to their resource demands. As we have shown, the relation between the resource demands of services and server capacity heavily impacts the size of the problems that can be solved. While, on average, only 8.5 ERP services could be allocated to a server in our experiments, it was possible to install 22-25 W/A/D services on such a server.

   Out of 20 SSAPv problem instances, most could be solved within 20 minutes with both the Heuristic and the B&B for up to 350 W/A/D services. In contrast, for ERP services, we could reliably solve these instances in time in only up to around 50 services. For W/A/D services, some larger instances of 600-800 services could be solved within 100 minutes, but 900 instances were beyond what could be solved in a day.

   For W/A/D services, the SSAPv Heuristic solved a higher number of instances than SSAPv B&B. This is due to the fact that there are a smaller number of fractional assignments compared to the number of variables in W/A/D problems. SSAP not only resulted in a lower solution quality, but the computation time was also higher since more servers were needed, and consequently, more decision variables. Although it would appear simpler, there is no reason to use SSAP for W/A/D services.

   When applied to ERP services, the performance of SSAPv decreases as compared to SSAP. ERP services required more resources, so a larger number of servers were required, leading to around three times as many variables. This had more impact on the SSAPv Heuristic, since the number of fractionally assigned variables in the initial LP relaxation increased. For ERP services, the percentage of instances solved was not significantly higher for the SSAPv Heuristic than for the B&B. The instances that could be solved, however, were solved faster with the SSAPv Heuristic. As a consequence, the SSAPv Heuristic turns out to be excellent for W/A/D services, but it loses some of its advantages when consolidating ERP services.

   In the future, new multicore CPU architectures will expand the capacity of servers significantly. Since we expect hardware capacity to grow faster than resource demands of software, we can expect to solve even larger ERP consolidation problems in the future.

3. **Considering daily variations in the workload in SSAPv reduces the number of servers used by 31 percent, on average, compared to SSAP.**

   Regarding W/A/D services, the average number of servers needed by SSAPv with 24 time slots was 69 percent of the number of servers required by SSAP. Almost equally, the corresponding savings with ERP services is 32 percent on average. Interestingly, the SSAPv Heuristic almost always resulted in the same number of servers needed, but was faster. The SSAPv Heuristic and the B&B found the lower bound number of servers in almost all cases for both W/A/D and ERP services. For SSAP with only a single time interval and a single resource, the FF and FFD heuristics found the optimal solution most of the time. However, it is difficult to consider side constraints in these heuristics. In summary, while SSAPv seems more complex at first, the solution quality, i.e., the utilization increases significantly, and the problems can even be solved faster.

4. **A larger number of time intervals has a positive impact on solution quality.**

   Applying the SSAPv, 2-hour intervals yield already 29.6 percent server savings compared to SSAP, while smaller intervals added little value.

5. **High aggregation quantiles have a negative impact on solution quality.**

   The risk attitude can have a significant influence on the number of servers needed. Its impact depends on the number of time slots. We have seen, however, that there is little difference between a 0.8 quantile and a 0.6 quantile in the number of servers needed. A 0.95 quantile avoids almost all capacity violations.

6. **Additional side constraints are easy to incorporate, but sometimes at the expense of solution time.**

   An important advantage of LP-based heuristics is the fact that it is easy to consider technical side constraints. The impact of side constraints on solution time varies. Some can make the problem harder to solve, while others even decrease the solution time. If the number of target servers increases through additional constraints, then computation time will increase (as is naturally the case with upper bounds on the number of services per server). If the number of servers is not increased by the constraints, then the *preassignment* constraints and *technical* allocation constraints will shorten the computation time, while the *combination* and *separation* constraints will increase the computation time.

7 Discussion of Additional Business Concerns

Cost reduction and energy considerations in data center operations are some of the main motivations for this paper. The hardware cost component of most data centers is less than 15 percent (for servers, storage, etc.) and is falling—although the volume of new equipment being installed is increasing [25]. The volume of installed servers has been growing at an approximately 11 to 15 percent compound annual growth rate for the x86 market during the past 4 years. More importantly, energy consumption of hardware components and for cooling is increasing [26]. Over the past decade, the cost of power and cooling has increased 400 percent and these costs are expected to rise. In some cases, energy costs account for 40-50 percent of the total data center operations costs [2].

Another motivation for energy conservation in data centers is the current debate on climate change. The IT industry is estimated to contribute 2 percent of the global CO_2 emissions; 23 percent of these emissions are produced by running servers, including cooling [27]; this means

about 0.5 percent of global CO_2 emissions are caused by running servers.

By means of a more efficient resource usage, both energy and hardware costs can be decreased. There are different approaches to slowing down the increasing energy demand in data centers. While energy consumption for cooling can be decreased by more efficient data center design in terms of infrastructure topology, floor space, placement of server racks etc., the power consumption of hardware—and thereby for cooling, too—can be decreased by server consolidation, increasing hardware utilization rates [28]. According to Gartner's "Server Consolidation Poll" [29], the top three reasons for the interest in server consolidation are "Control server sprawl" (33 percent), "Reduce power and cooling needs" (25 percent), and "Provide TCO savings" (20 percent).

In summary, beyond the minimization of investment costs as achieved through scenario 1 or 2 (see Section 3.1), the consolidation of servers is of significant importance for data centers regarding the reduction of energy and cooling costs.

Although energy costs as well as server capacities are constantly changing, we will provide a small example, illustrating expected savings through optimal server consolidation. We assume a server consolidation project with 250 source servers. A state-of-the-art midrange server with average investment costs of €2,500 results in energy costs of approximately €735.80 per year. This estimate is based on 700 W energy consumption and €0.12/kWh. Note that these costs do not include costs for cooling in a data center. Experts believe that for cooling, one could assume the same amount of energy cost per year as for operating the server.

In the optimal assignment with SSAPv, we require 10 new midrange servers. The optimal assignment in SSAP (ignoring daily variations) led to a demand for 15 new midrange servers. The energy costs for the operations of these five additional servers in 4 years (typical writeoff time frame) amount to €14,716, with cooling around €28,000. Investment costs for the five servers are $5 * €2{,}500 = €12{,}500$, totaling €40,500 as a lower bound on the cost savings. Note that we did not consider costs for space and administration of additional hardware. Moreover, a manual assignment will likely be suboptimal and not be able to consider multiple resources, multiple time intervals, and side constraints. If IT managers do not do a thorough analysis of the workload traces, it could easily happen that they consolidate servers with positively correlated workloads, leading to further losses in production efficiency. Apart from basic savings, decision support for server consolidation reduces the error-prone and time-consuming process of manual planning.

Obviously, virtualization has benefits that are beyond savings in investment and energy costs. For example, managing and migrating servers from one hardware server to another becomes much easier. However, virtualization software also comes at a cost. Whether these benefits outweigh the (actual) cost is a topic frequently discussed in reports by market research analysts. It heavily depends on the cost of introducing virtualization software. For these reasons, a total cost analysis of server consolidation projects is outside the scope of this paper.

8 Related Work

Our paper follows a design science approach as outlined in [30]. While there has been a lot of work on capacity planning and resource allocation in general, little work has focused on server consolidation and respective planning problems.

Closest in spirit to SSAP is a workshop paper by Rolia et al. [31], who suggest an integer program for allocation problems in a data center. Another heuristic for a similar allocation problem was presented in a talk by HP Labs [32]. Further details have not yet been published. In contrast to this work, we discuss a set of models focused on server consolidation, where we explicitly consider variations in workload over time and the server costs, suggest specific algorithms for data aggregation and optimization, and provide an extensive empirical evaluation focusing on managerial questions.

Seltzsam et al. [33] describe a system called AutoGlobe for service-oriented database applications. In contrast to SSAP(v), the static allocation heuristic in [33] determines an initial schedule trying to *balance* the workload across servers, while a fuzzy controller is used to handle overload situations based on fuzzy rules provided by system administrators.

There is a growing literature on Cloud computing and Infrastructure-as-a-Service (IaaS) [34]. IaaS providers running virtualized data centers need to solve a similar resource allocation problem. A difference is that with IaaS providers new customers come and go. Also, nowadays they offer predefined server sizes in terms of CPU and memory capacity. Typically they use rules to assign the virtual server of a new customer to one of many physical servers. The problem that such IaaS providers face can be interpreted as the online version of SSAP. If only a single resource such as CPU is considered per day, the online version of SSAP can be solved using online bin packing algorithms [35]. In contrast, static server consolidation problems of the sort described in this paper occur regularly in enterprise data centers, where old hardware technology gets replaced, or where virtualization technology is introduced.

9 Conclusion

Efficiency in the production of IT services is crucial in today's competitive markets. Virtualization and server consolidation can lead to increased utilization of hardware, and therefore, to increased production efficiency. Server consolidation poses a number of new and fundamental decision and planning problems in IT service management.

In this paper, we have proposed a capacity planning method for virtualized IT infrastructures that combines a specific *data preprocessing* and an *optimization model*. We characterized the computational complexity of these models, proposed algorithms, and provided extensive experimental evaluations based on workload traces from an industry partner. The consideration of variations in the workload in SSAPv yielded hardware savings of around 31 percent as compared to optimal allocations in SSAP. This result holds for two widespread classes of applications, namely Web/application/database servers on the one hand, and ERP services on the other. The SSAPv Heuristic allowed us to solve large-scale server consolidation problems within minutes, while easily integrating additional technical side constraints for the allocation. The different resource demands of the two types of services, however, had a significant impact on the problem sizes that could be solved. The approach is now in use in the field.

Workloads can change over time. It is important to analyze the workload traces regularly and reoptimize the

allocation if necessary. If workloads are less predictable or exhibit significant trend, an automated controller can perform these tasks and move applications automatically from one physical server to another, allowing for adaptive, self-organizing data center management. The models discussed in this paper can serve as the basis for respective controllers.

ACKNOWLEDGMENTS

This work was accomplished in collaboration with Siemens IT Solutions and Services (SIS). We thank SIS for their technical and financial support.

REFERENCES

[1] K. Parent, "Server Consolidation Improves IT's Capacity Utilization," Court Square Data Group, 2005.
[2] D. Filani, J. He, S. Gao, M. Rajappa, A. Kumar, P. Shah, and R. Nagappan, "Technology with the Environment in Mind—Dynamic Data Center Power Management: Trends, Issues, and Solutions," *Intel Technology J.,* vol. 12, pp. 59-68, Feb. 2008.
[3] IDC, "Increasing the Load: Virtualization Moves Beyond Proof of Concept in the Volume Server Market," press release, Oct. 2005.
[4] D. Menasce, *Performance by Design: Computer Capacity Planning.* Prentice Hall, 2004.
[5] J.-N. Lee, S.M. Miranda, and Y.-M. Kim, "IT Outsourcing Strategies: Universalistic, Contingency, and Configurational Explanations of Success," *Information Systems Research,* vol. 15, pp. 110-131, June 2004.
[6] OGC, *ITIL Best Practice for Service Delivery,* fourth ed. The Stationary Office, 2002.
[7] *ISO 20000 IT Service Management Standards, Int'l Organization for Standardization ISO/IEC,* ISO, 2005.
[8] E. Noel and K.W. Tang, "Performance Modeling of Multihop Network Subject to Uniform and Nonuniform Geometric Traffic," *IEEE/ACM Trans. Networking,* vol. 8, no. 6, pp. 763-774, Dec. 2000.
[9] S.S. Thakkar and M. Schweiger, "Performance of an OLTP Application on Symmetry Multiprocessor System," *Proc. 17th Ann. Int'l Symp. Computer Architecture,* pp. 228-238, 1990.
[10] D. Menasce, V. Almeida, and L. Dowdy, *Performance by Design: Computer Capacity Planning by Example.* Prentice Hall, 2004.
[11] A.R. Hevner, S.T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly,* vol. 28, pp. 75-105, Mar. 2004.
[12] M. Missbach and J. Stelzel, *Adaptive Hardware-Infrastrukturen für SAP.* SAP Press, 2005.
[13] SAP, "SAP Standard Application Benchmarks: Measuring in SAPS," vol. 2009, 2008.
[14] Q. Zhang, L. Cherkasova, G. Mathews, W. Greene, and E. Smirni, "R-Capriccio: A Capacity Planning and Anomaly Detection Tool for Enterprise Services with Live Workloads," *Proc. Int'l Middleware Conf.,* pp. 244-265, 2007.
[15] M. Garey and R. Graham, "Resource Constrained Scheduling as Generalized Bin Packing," *J. Combinatorial Theory, Series A,* vol. 21, pp. 257-298, Nov. 1976.
[16] M.Y. Kao, *Encyclopedia of Algorithms.* Springer, 2008.
[17] M. Yue, "A Simple Proof of the Inequality FFD (L) $\leq$ 11/9 OPT (L) + 1, for All L, for the FFD Bin-Packing Algorithm," *Acta Mathematicae Applicatae Sinica (English Series),* vol. 7, pp. 321-331, 1991.
[18] G. Dósa, "The Tight Bound of First Fit Decreasing Bin-Packing Algorithm Is FFD (I)$\leq$ 11/9 OPT (I) + 6/9," *Proc. Int'l Symp. Combinatorics, Algorithms, Probabilistic and Experimental Methodologies,* B. Chen, M. Paterson, and G. Zhang, eds., pp. 1-11, 2007.
[19] C. Chekuri and S. Khanna, "On Multi-Dimensional Packing Problems," *Proc. ACM-SIAM Symp. Discrete Algorithms,* pp. 185-194, 1999.
[20] N. Bansal, A. Caprara, and M. Sviridenko, "Improved Approximation Algorithms for Multidimensional Bin Packing Problems," *Proc. IEEE Symp. Foundations of Computer Science,* pp. 697-708, 2006.
[21] M. Andreolini, S. Casolari, and M. Colajanni, "Models and Framework for Supporting Runtime Decisions in Web-Based Systems," *ACM Trans. Web,* vol. 2, pp. 1-43, 2008.
[22] S. Elnikety, E. Nahum, J. Tracey, and W. Zwaenepoel, "A Method for Transparent Admission Control and Request Scheduling in E-Commerce Web Sites," *Proc. Int'l Conf. World Wide Web,* 2004.
[23] C. Pu, A. Sahai, J. Parekh, J. Gueyoung, B. Ji, C. You-Kyung, T. Garcia, D. Irani, L. Jae, and L. Qifeng, "An Observation-Based Approach to Performance Characterization of Distributed n-Tier Applications," *Proc. IEEE 10th Int'l Symp. Workload Characterization (IISWC '07),* pp. 161-170, 2007.
[24] F. Margot, "Symmetric ILP: Coloring and Small Integers," *Discrete Optimization,* vol. 4, pp. 40-62, 2007.
[25] R. Kumar, "Important Questions from Gartner's EMEA Data Center Conference 2006: Data Center Trends," Gartner, Inc., Dec. 2006.
[26] R. Kumar, "Data Center Power and Cooling Scenario through 2015," Gartner, Inc., Mar. 2007.
[27] R. Kumar and L. Mieritz, "Conceptualizing 'Green' IT and Data Center Power and Cooling Issues," Gartner, Inc., Sept. 2007.
[28] R. Kumar and S. Mingay, "How IT Management Can 'Green' the Data Center," Gartner, Inc., Jan. 2008.
[29] J.R. Phelps, "Data Center Conference 2007 Server Consolidation Poll Finds Projects Increasing, Reasons Changing and Outside Assistance Growing," Gartner, Inc., Jan. 2008.
[30] K. Peffers, T. Tuunanen, M.A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *J. Management Information Systems,* vol. 24, pp. 45-77, 2008.
[31] J. Rolia, A. Andrzejak, and M. Arlitt, "Automating Enterprise Application Placement in Resource Utilities," *Proc. 14th IFIP/IEEE Int'l Workshop Distributed Systems: Operations and Management (DSOM '03),* pp. 118-129, 2003.
[32] A. Zhang, F. Safai, and D. Beyer, "Server Consolidation: High-Dimensional Probabilistic Bin-Packing," *Proc. INFORMS Conf.,* 2005.
[33] S. Seltzsam, D. Gmach, K. Krompass, and A. Kemper, "AutoGlobe: An Automatic Administration Concept for Service-Oriented Database Applications," *Proc. 22nd Int'l Conf. Data Eng. (ICDE '06),* 2006.
[34] B. Hayes, "Cloud Computing," *Comm. ACM,* vol. 51, pp. 9-11, 2008.
[35] S. Seiden, "On the Online Bin Packing Problem," *J. ACM,* vol. 49, pp. 640-671, 2002.
[36] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, 1979.
[37] R.H. Shumway and D.S. Stoffer, *Time Series Analysis and Its Applications,* second ed. Springer, 2006.
[38] H.P. Williams, *Model Building in Mathematical Programming.* John Wiley & Sons Ltd, 1999.

Benjamin Speitkamp received the diploma degree from the Universität Karlsruhe (TH) and the PhD degree from the Technische Universität (TU) München, Germany, in 2009. He joined the Department of Informatics at TU München in 2005. His research is at the intersection of information systems and operations management and focuses on IT service management in particular. In 2008, he received the INFORMS ISS Design Science Award.

Martin Bichler received the MSc degree in information systems from the Technical University of Vienna, and the PhD and as habilitation degrees from the Vienna University of Economics and Business Administration. He is a full professor in the Department of Informatics at the Technische Universität (TU) München. He has worked as a research fellow at the University of California, Berkeley, and as a research staff member at the IBM T.J. Watson Research Center, New York. He has been involved in research and development in the areas of auction design, operations research, and IT service operations management.