Capacity Planning for Virtualized Servers[1]

Martin Bichler, Thomas Setzer, Benjamin Speitkamp
Department of Informatics, TU München
85748 Garching/Munich, Germany
(bichler | setzer | benjamin.speitkamp)@in.tum.de

Abstract

Today's data centres offer many different IT services mostly hosted on dedicated physical servers. Virtualization provides a technical means for server consolidation leading to increased server utilization. The term refers to the abstraction of computing resources across many aspects of computing and has been used to describe different techniques. Virtualization engines allow hosting multiple virtual servers (including operating system plus applications) on a single physical server, and sometimes to migrate and dynamically allocate these virtual servers to physical servers on demand. This allows for much flexibility in capacity management. This article presents a number of capacity planning problems in the presence of virtualized IT infrastructures and decision models to allocate these virtual servers optimally. Based on a data set of CPU traces from a data centre provider, we present the results of experiments using different problem formulations.

1 Introduction

Nowadays data centers host most of their 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 reasons for using high-capacity physical servers. As a consequence, server utilization is typically very low, which incurs high investments and high operational cost. The Gartner Group estimates that the utilization of servers in a typical company is around 20 percent [1].

Virtualization can be used as technical means for server consolidation. Benefits are higher server utilization levels, reduced time for deployment, easier system management, and overall lower hardware and operating costs. While traditional capacity planning literature heavily relies on queuing networks and discrete event simulation, new types of capacity planning problems arise in the presence of virtualization. Essentially, an IT service manager needs to decide which services should be hosted on which servers. In case of volatile demand, these decisions need to be made dynamically by a controller, in order to make sure enough capacity is available for each service at each point in time.

In this paper we present a set of capacity planning problems for virtualized IT infrastructures. We consider the problem of an IT service provider hosting the IT services of multiple customers. Based on the users demands the IT service provider needs to determine the partitioning of servers using pre-defined objective functions (e.g., minimizing the amount of servers used). We have developed a set of decision models and a software implementation to determine optimal allocations of application services or virtual servers resp. to physical servers.

The paper is structured as follows: In section 2 we continue with a brief overview of virtualization techniques. In section 3 we introduce three capacity planning problems and adequate optimization models. In section 4 and 5 we evaluate two of the proposed models using industry data. In section 6 related work is discussed. In section 7 conclusions are drawn and future work is discussed.

[1] Published at the 16th WITS 2006, Milwaukee, Wisconsin, USA.

2 Virtualization Technology

Multiple approaches allow for server virtualization: SMP (symmetric multiprocessor) servers can be subdivided into fractions, each of which is acting as a single 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 such as Linux VServer, Microsoft Virtual Server, Parallels, Quemu, VMware, and XEN create virtual servers whose physical resource use is dynamically adjustable, enabling multiple isolated and secure virtualized servers on a single physical server. According to IDC server spending around virtualization is a rapidly growing market that will amount to nearly US$ 15 billion by 2009 [2].

3 Problem Formulations

3.1 Static Server Allocation Problem

A basic server consolidation problem is the **Static Server Allocation Problem (SSAP)**. Here, the IT service manager needs to consolidate servers and assign *services* (i.e., virtual servers) to (physical) *servers* so as to minimize the number of servers used or minimize overall server costs, respectively. Suppose that we are given n services $j \in J$ that are to be served by m servers $i \in I$. For simplicity, we assume that no one service requires more capacity than can be provided by a single server. Customers order u_j units of capacity (e.g., SAPS) and each server has a certain capacity s_i. For safety reasons, s_i might be set below the physical capacity limit. y_i is a binary decision variable indicating which servers are used, c_i describes a potential cost of a server, and x_{ij} describes which service is allocated to which server. The problem can be formulated as an instance of the well-known bin packing problem (see equations (1)).

$$\begin{aligned}
&\min \sum\nolimits_{i=1}^{m} c_i y_i \\
&s.t. \\
&\qquad \sum\nolimits_{i=1}^{m} x_{ij} = 1 && \forall j \in J \\
&\qquad \sum\nolimits_{j=1}^{n} u_j x_{ij} \le s_i y_i && \forall i \in I \\
&\qquad y_i, x_{ij} \in \{0,1\} && \forall j \in J, \forall i \in I
\end{aligned} \tag{1}$$

Bin packing is known to be NP-hard. However, polynomial and linear time approximation algorithms exist. For example, the first fit decreasing algorithm orders the items from largest to smallest, then places them sequentially in the first bin in which they fit, and can be implemented in O(n log n). Johnson et al. showed that this strategy is never suboptimal by more than 22%, and furthermore that no efficient bin-packing algorithm can be guaranteed to do better than 22% [3].

In addition to the basic formulation in (1), often additional side constraints have to be considered. For example, system administrators want to make sure that a particular service j is only allocated to one out of a restricted set K of k servers, e.g., running a certain operating system (2) or that two services j and l are not allocated to the same server for some reason (3).

$$\sum\nolimits_{i=1}^{k} x_{ij} = 1 \tag{2}$$

$$x_{ij} + x_{il} \le 1 \qquad \forall i \in I \tag{3}$$

Fixed allocations of services to servers can be done in a pre-processing step where the services are assigned manually and the capacity of the respective servers is decreased accordingly. Virtualization engines such as VMware allow to set lower and upper bounds for the capacity allocated to a virtual server (i.e., service). The minimum capacity is guaranteed, and the virtual server’s workload can fluctuate up to a maximum capacity. This interval between minimum and maximum capacity is shared with other services. Therefore, the optimization problem should assign those services on a physical server that have their demand, i.e. workload peaks at different times of the day or week. Typically, service demand of this sort has seasonal patterns on a daily or weekly basis. For example, the payroll accounting is done at the end of every week and requires specific services, while demand for an OLAP application is constant during a week, but has an everyday peak in the morning hours, when managers access their daily reports.

Suppose, time is divided into a set of discrete time steps T indexed by $t=\{1, \ldots, \tau\}$. The workloads vary over time, so that we get a separate matrix u_{jt} describing how much capacity service j requires in time step t. This matrix can for example be generated by taking the maximum demand value of each time step in a workload trace. Based on this matrix we formulate the **Static Server Allocation Problem with variable workload** (**SSAPv**) in (4).

$$
\begin{aligned}
&\min \sum\nolimits_{i=1}^{m} c_i y_i \\
&s.t. \\
&\qquad \sum\nolimits_{i=1}^{m} x_{ij} = 1 && \forall j \in J \\
&\qquad \sum\nolimits_{j=1}^{n} u_{jt} x_{ij} \le s_i y_i && \forall i \in I, \forall t \in \tau \\
&\qquad y_i, x_{ij} \in \{0,1\} && \forall j \in J, \forall i \in I
\end{aligned}
\tag{4}
$$

3.2 Dynamic Server Allocation Problem

Data centre virtualization also allows services to be migrated over time to different servers. For example, the “dynamic workload balancing” use case in [4] describes how servers can be dynamically installed and de-installed depending on changing service workload. The **Dynamic Server Allocation Problem (DSAP)** describes the decision of how many servers are required overall in a planning period (say a week) and how services are allocated to servers in the individual time steps (e.g., 3 hour time slots of the day).

$$
\begin{aligned}
&\min \sum\nolimits_{i=1}^{m} c_i y_i \\
&s.t. \\
&\qquad \sum\nolimits_{i=1}^{m} x_{ijt} = 1 && \forall j \in J, \forall t \in T \\
&\qquad \sum\nolimits_{j=1}^{n} u_{jt} x_{ijt} \le s_i y_i && \forall i \in I, \forall t \in T \\
&\qquad x_{ijt}, y_i \in \{0,1\} && \forall i \in I, \forall j \in J, \forall t \in T
\end{aligned}
\tag{5}
$$

In addition, however, the administrator might want to minimize the number of re-allocations over time. The binary decision variable z_{ijt} determines double the amount of re-allocations (each re-allocation is counted twice). This way, a system administrator can limit the number of re-allocations from one period to another to p (6).

$$
\begin{aligned}
&x_{ijt} - x_{ijt-1} \le z_{ijt} && \forall i \in I, \forall j \in J, \forall t \in \{2,\ldots,\tau\} \\
&x_{ijt-1} - x_{ijt} \le z_{ijt} && \forall i \in I, \forall j \in J, \forall t \in \{2,\ldots,\tau\} \\
&\sum\nolimits_{i=1}^{m} \sum\nolimits_{j=1}^{n} \sum\nolimits_{t=2}^{\tau} z_{ijt} \le 2p \\
&z_{ijt} \in \{0,1\} && \forall i \in I, \forall j \in J, \forall t \in T
\end{aligned}
\tag{6}
$$

The number of servers an IT service provider can save using data centre virtualization has been one of the main sales arguments for software vendors in this field. Quite obviously, this depends very much on the level of interference of the different workload traces (i.e., time series). The above DSAP helps to quantify how much can actually be saved compared to dedicated server hosting based on historic workload data.

4 Description of Data

From a professional data centre we obtained workload traces containing the CPU utilization (measured in so called performance units, PUs) of 30 dedicated servers hosting different types of application services (e.g., web servers, application servers and databases). The data describes five minute average values during a month. Most of these workload traces exhibit cyclic patterns on a daily basis. Thus, we consider daily workload fluctuations.

As the number of time steps considered directly impacts the number of variables in SSAPv and DSAP, we aggregated the 5 minute average values and calculated an estimator for the hourly CPU workload requirements to lower the number of variables. For each service we determined the maximum workload of the 5 minute intervals in an hour based on a sample of multiple days and determined this as our estimator for $\{u_{jt}\}$. This is a rather conservative estimator, which is based on the type of service and the risk attitude of the data centre provider. If providers are willing to overbook resources to a certain degree they can lower u_{jt} based on additional statistical considerations.

All decision models in Section 3 are binary programs. In particular, SSAPv and DSAP might become intractable for practical problem sizes with a large number of services and time steps considered. There are several strategies to solve the above mentioned decision problems in practically acceptable times.

1. Decrease the number of time steps considered per day.
2. Pre-select services with destructive interference by identifying negatively correlated workload traces using for example covariance matrices or Principle Component Analysis.
3. Use heuristics or meta-heuristics such as Genetic Algorithms to solve the problem.

5 Experiments and Results

Based on the data from our industry partner we conducted experiments with SSAP and SSAPv with respect to runtime and number of servers assigned. SSAPv takes into account the hourly workload variations during a day u_{jt}, while SSAP only considers the maximum workload u_j of a day in order to satisfy the workload demands during a day. We considered four distinct scenarios of different sizes A (8 services), B (15 services), C (26 services), and D (30 services). Each scenario represents an arbitrarily chosen subset of services from our data set. We assumed that all servers have identical capacity and cost (e.g., a rack of homogeneous blade servers).

Using the *First Fit* approach for the SSAP for each scenario an upper bound for the number of servers was obtained to host each service. Based on this upper bound for the number of servers, the number of decision variables and the number of LP-constraints are derived for the SSAP and the SSAPv. Table 1 illustrates characteristics of the respective binary programs.
SSAP and SSAPv were solved optimally for each of the four scenarios. SSAP solves the problem using fewer servers than the *First Fit* approach in three of four cases. SSAPv further reduces the number of required servers compared to SSAP in three of four cases (see Fig. 1). The larger the scenarios, the more can be gained by optimization formulations. Further improvement may be achieved by pre-selecting services that exhibit strong complementarities.

Scenario (# Services)	*No. of decision variables*	*No. of constraints SSAP*	*No. of constraints SSAPv*
A (08)	27	11	80
B (15)	64	19	111
C (26)	216	34	218
D (30)	279	39	246

Table 1: Problem sizes of different scenarios

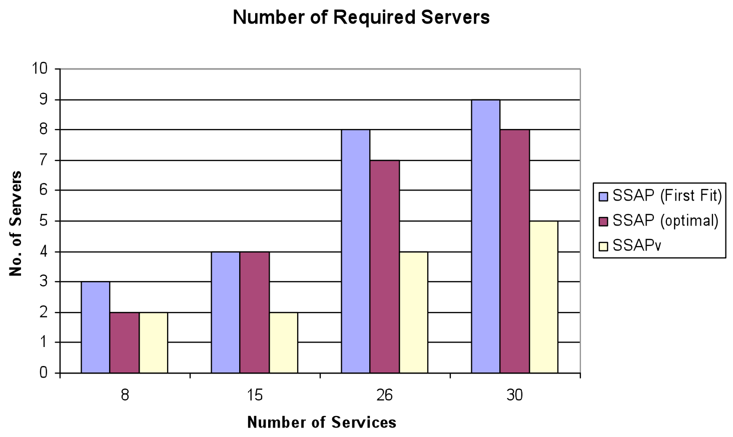

Fig. 1: Server requirements depending on server allocation method

All calculations were performed on a laptop with an AMD Athlon XP 2600+ processor, 1.9 GHz, with 512 MB RAM using a COTS branch-and-bound IP solver (Frontline®). The branch-and-bound algorithm included "strong branching (determination of the next variable to branch upon based on pseudo costs calculated from the dual simplex) and "probing" (setting certain binary integer variables to 0 or 1 and deriving values for other binary integer variables, or tightening bounds on the constraints).

The experimental results in Fig. 1 illustrate the potential cost savings of taking workload patterns into account. Of course, the more complex formulation SSAPv incurs higher computational costs. Optimal solutions for the SSAP were calculated within less than 30 seconds for all four scenarios. Plus, there exist polynomial time schemes for this problem (see Section 3.1).. SSAPv took much longer. Depending on the parameter settings, the determination of the optimal solution took 5 minutes to several hours. More extensive sensitivity analyses need to be done here.

6 Related Work

A number of approaches found in literature address resource allocation problems as described in this article. Urgaonkar et al. [5] analysed the overbooking of shared resource pools by exploiting knowledge about demand and workload profiles (statistical multiplexing). The authors analysed best-fit, worst-fit and random placement heuristics to bundle complementary services on common servers. Seltzsam et al. describe a system called AutoGlobe for service-oriented database applications [6]. Among others the system includes components for load monitoring, and algorithms for static planning and dynamic capacity management. The static allocation heuristic determines an initial schedule balancing the workload across the servers, while a fuzzy controller is used to handle overload situations based on fuzzy rules provided by system administrators. Rolia et al. [7] also propose an initial approach to assign applications to servers using an integer program. The authors statistically characterize the demand profiles of business applications based on historical traces and projections, providing overall bounds on application resource requirements. Rolia et al. take the

traces of application demand as input and assign them to servers by building demand profiles based on historical observations and present two approaches to assign them to a small set of servers using a Genetic Algorithm [7]. In the R-Opus framework, Cherkasova et al. present a *Workload Placement Service* based on statistical multiplexing [8]. The total amount of servers is decreased by using genetic algorithms that penalize low utilizations of resource pools as well as resource pools with insufficient capacity.
Our contribution is the proposal and comparison of different allocation approaches for virtualized servers. Sensitivity analyses are performed regarding computational costs, the quality of different solution algorithms and different problem sizes.

7 Conclusions and Future Work

Virtualization enables much flexibility for capacity management in a data centre. In this paper, we have proposed three decision models for capacity planning problems that can be found in virtualized IT infrastructures. We have implemented the SSAP and the SSAPv and provided first experimental evaluations based on workload traces from an industry partner.

In our future research we plan to evaluate a larger set of workload traces from our industry partner and cluster them based on the type of application. We plan to work on the development of heuristics and the application of meta-heuristics such as Genetic Algorithms to solve SSAPv and DSAP for much larger problem sizes. In addition, we want to do sensitivity analyses with respect to the size of the time steps used (1 hour vs. 3 hour time steps) and develop further approaches to aggregate the workload traces based on the decision makers risk attitude and information in service level agreements. Pre-selection of promising candidate services based on covariance of workload traces is another approach, with which we expect to improve the results further.

References

[1] K. Parent, "Server Consolidation Improves IT's Capacity Utilization" Vol. 2006: Court Square Data Group, 2005.

[2] IDC-Press, "Increasing the Load: Virtualization Moves Beyond Proof of Concept in the Volume Server Market.", 2005.

[3] D. Johnson, A. Demers, J. Ullman, M. Garey, and R. Graham, "Worst-Case Performance Bounds for Simple One-Dimensional Packing Algorithms" *SICOMP*, Vol. 3, 1974.

[4] M. Missbach and J. Stelzel, *Adaptive Hardware-Infrastrukturen für SAP*: SAP Press, 2005.

[5] B. Urgaonkar, P. Shenoy, and T. Rescoe, "Resource Overbooking and Application Profiling in Shared Hosting Platforms" presented at Usenix OSDI, 2002.

[6] S. Seltzsam, D. Gmach, K. Krompass, and A. Kemper, "AutoGlobe: An Automatic Administration Concept for Service-Oriented Database Applications" presented at 22nd International Conference on Data Engineering (ICDE 2006), Atlanta, 2006.

[7] J. Rolia, A. Andrzejak, and M. Arlitt, "Automating Enterprise Application Placement in Resource Utilities" presented at Workshop on Distributed Systems: Operations and Management, Heidelberg, 2003.

[8] L. Cherkasova and J. Rolia, "R-Opus: A Composite Framework for Application Performability and QoS in Shared Resource Pools" Hewlett-Packard Labs, Palo Alto, 2006.