Improving enterprise VM consolidation with high-dimensional load profiles

Andreas Wolke and Carl Pfeiffer
Technische Universität München
Boltzmannstraße 3, 85748 Garching, Germany
Email: andreas.wolke@in.tum.de, carl.pfeiffer@mytum.de

***Abstract*—Modern enterprise data centers take advantage of virtual machine consolidation to allocate virtual machines to virtualized servers to increase energy efficiency. One key problem is to minimize the number of virtualized servers required while maintaining service quality. A promising approach is to exploit recurring load patterns exhibited by enterprise VMs for increased allocation efficiency. This paper shows that bin packing heuristics can deliver the same allocation quality as integer linear programs if calculation time is constrained. There were no significant differences between vector bin packing heuristics in simulations based on CPU load profiles obtained from enterprise data centers. We further show that consolidating in clusters of a few hundred virtual machines is sufficient as solution quality does not improve with larger clusters.**

I. Introduction

Modern state of the art enterprise data centers use virtualization technology to improve resource utilization of their physical servers. Applications once run on physical servers are now run in VMs, of which many are assigned to a single physical server.

In this work we analyze how the number of servers required to run such VMs can be minimized. The motivation is primarily economic: according to InformationWeek [1], energy consumption accounts for twenty-five percent of a data center's operational costs - and energy costs keep climbing. Public interest in energy efficiency and green data centers [2] is another motivation to save energy.

The main technical reason for minimizing the number of servers is their high baseline energy consumption [3]. Servers maximize their energy efficiency when they are either in hibernate mode, or when the CPU is fully utilized [4]. The optimization objective is thus to find an assignment of VMs to servers that minimizes the total number of servers needed while maintaining service quality.

Fortunately, many VMs exhibit load profiles similar to Figure 1 with patterns recurring daily or weekly. These patterns help refine the VM's demand characteristics compared to time-invariant demand specifications like static CPU cores and memory demand. Speitkamp et al. [5] showed with their SSAPv[1] integer linear program that leveraging such profiles strongly increases consolidation density while maintaining service quality.

In short, our findings show that heuristics are on par with SSAPv solutions if calculation time is constrained. Further, there are no significant differences between heuristics - simpler heuristics perform just as well as more complex ones. Also, solution quality does not increase with input sizes in excess of 200 VMs, allowing calculations to be clustered on a per rack basis. In scenarios with unlimited calculation time SSAPv still outperforms all heuristics.

This work contributes threefold. First, we evaluate the effectiveness of well-known vector bin packing heuristics on daily sampled load profiles of enterprise data centers. To our knowledge, this is the first study to combine high-dimensional load profiles with vector bin packing heuristics. In related work, vectors are used to represent single demand values for multiple resources. Second, we provide comprehensive guidelines for parametrizing vector bin packing heuristics for the purpose of VM allocation. Third, we show that vector bin packing heuristics can substitute linear program solvers if solution time is constrained and load profiles of similar dimensionality are used.

II. Problem formulation

We ask for an allocation which assigns load profiles to servers without oversubscribing any of these. We consolidate servers based on a single restrictive resource, the CPU. To maximize a server's energy efficiency it is necessary to fully utilize each CPU as under-utilization leads to lower efficiency [3].

A load profile describes a VM load of a twenty-four hour day for a single resource in s time slots of equal lengths. A vector $\vec{d_i}$ with $s = dim(\vec{d_i})$ values thus describes the CPU demand of a VM i over all s time slots, with one value per time slot.

A server j is represented by a vector of residual capacities $\vec{r_j}$ with dimensionality s. It explains the amount of free server resources for each time slot. Whenever a VM i is assigned to a server j, its load vector is subtracted from the server's residual capacity vector $r_j - d_i$. If one vector component turns out to be negative, this particular server is oversubscribed. An allocation of VM i to server j is valid if $d_i^k \leq r_j^k$ for $1 \leq k \leq s$.

We search a valid allocation for a given set of VMs and their respective load profiles with the objective of minimizing the number of servers, no oversubscriptions allowed. This problem is known as the multidimensional vector bin packing problem [6].

[1]Static Server Allocation Problem with variable load

III. Existing approaches

Various works on integer linear programs to solve the multidimensional vector bin packing problem exist [5][7][8]. Speitkamp et al. recommend load profiles with less than 24 dimensions, arguing that downsampling load profiles hurts solution quality only marginally and reduces major complexity hurdles.

Setzer et al. propose a novel approach to cut complexity [9]. Singular value decomposition is leveraged to extract workload characteristics used by an integer linear program that originated from the one proposed by Speitkamp. This approach combines the advantages of using load profiles with a reduction in computational complexity.

Panigrahy et al. [10] propose a vector bin packing heuristic for server consolidation based on the dot product of two vectors. Li et al. [11] subsequently showed that the cosine of two vectors yields slightly better results. Both compared their approaches to existing heuristics such as First Fit or the Euclidean distance metric proposed by Srikantaiah et al. [12]. In these and other studies [13], vector components represent singular demand values for server resources, for example CPU or memory.

The vectors in this study instead represent the load profile of a single constrained resource (CPU) over time, where time is partitioned into a fixed number of equal-length time slots. Compared to previous studies, vectors used by this approach are of higher dimensionality. To our knowledge, this is the first study combining vector bin packing heuristics with load profiles to calculate dense allocations for a given set of VMs.

The approach closest to ours is by Maruyama [14], who propose a generic algorithm for vector bin packing. They evaluate the impact of different parameters by conducting simulations founded on synthetic data, where vector elements are randomly chosen from a bell shaped or skewed distribution. However, histograms drawn from representative load profiles (Figure 1) suggest different distributions, leaving room to exploit complementarities and questioning the applicability of [14]'s approach.

IV. Simulation setup

Our simulations make use of precalculated load profiles to evaluate the effectiveness of vector bin packing heuristics for allocating VMs to servers. These load profiles stem from large European data centers and are not generated in a synthetically manner as done by other studies. An extensive descriptive statistical analysis of the raw data is provided by Speitkamp et al. [5].

For each of the 450 time series we calculate a profile closely following the approach described by Speitkamp. We select the CPU as the single limiting resource, and ignore other resources, since the scalability of CPU power is inferior to the scalability of all other server resources. The majority of the time series exhibit a daily load pattern if weekends are excluded.

A load profile is calculated in two steps. In step one, the CPU time series is split into days with a sampling rate of five minutes, totalling 288 values for twenty-four hours. We superimpose splits such that multiple values for each index exist. In step two, we take the 95th percentile from these values to form a new time series of 288 values. To further downsample a load profile, we merge time slots with their maximum value as the merged time slot's value.

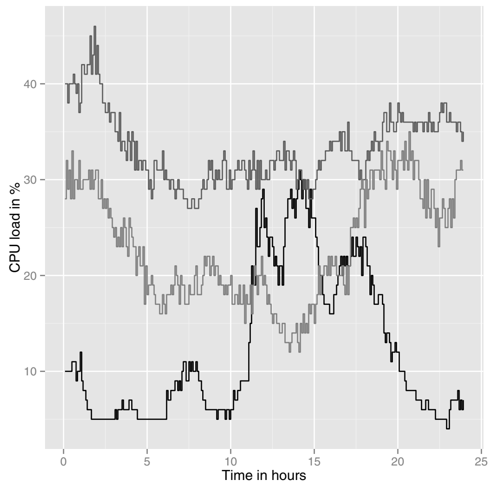

Fig. 1. Sample workload profiles of VMs

We created twenty sets of load profiles. For each set, 180 load profiles were randomly picked from the 450 load profiles. Each simulation treatment was replicated twenty times, once for each set.

Simulations were conducted as full factorial experiments with three factors. The factor scale describes the number of VMs to allocate with levels 30, 90, 120, 150 and 180. The factor profile length describes the load profile downsampling with levels 288 (no downsampling), 24 (one per hour), 12 (one per two hours) and 6 (one per three hours). We consider four bin packing heuristics and one integer linear program as the third factor, algorithms, with five levels: 1) FF (First Fit), 2) FFv (First Fit with variable workload), 3) DotProduct [10], 4) Cosine [11], and 5) SSAPv [5]. We shortly summarize the working of each algorithm.

First Fit is a one-dimensional bin packing heuristic and therefore needs to extract a single scalar value from the VM's load profile vector, in this case the vector component with the maximum value. It then places the VM on the first server that has enough residual capacity to accomodate the VM. First Fit with variable workload (FFv) on the other hand does a component-wise comparison of the load profile and the server's vector of residual capacities and then similarly places the VM on the first server that can host the VM.

SSAPv [5] is an integer linear program to calculate an allocation with the objective of minimizing the number of servers, which we solved using the Gurobi solver. We limited its computation time to 25 minutes for small scenarios with up to 60 VMs and to 45 minutes for larger scenarios with up to 180 VMs. Gurobi was able to produce valid, yet potentially suboptimal solutions for almost all treatments. Since our study is about the practical applicability of the described approaches, we argue that data center operators in operational environments limit the calculation time of solvers like Gurobi. We therefore compare Gurobis (sub)optimal solutions with vector bin packing heuristics.

Panigrahy et al. [10] propose the dot product of the the server's residual capacity and the VM's demand vectors as the metric for determining the allocation target. For each new VM the dot product of a weight $\vec{w}$, residual $\vec{r}$, and demand $\vec{d}$ is calculated as shown in Equation 1. The VM is placed on the server i with the greatest dot product.

$$\sum_{k=1}^{s} w_i^k r_i^k d^k \tag{1}$$

Xi [11] et al. instead propose the minimum cosine (see Equation 2).

$$\cos(\vec{r}, \vec{d}) = \frac{\vec{r} \cdot \vec{d}}{|\vec{r}||\vec{d}|} \tag{2}$$

V. Results

We conducted simulations in a full factorial experiment design. Each treatment was replicated twenty times, once for each profile set. Simulations were conducted in fully randomized order. Simulations results and R-scripts for data analysis are available on-line[2].

The performance of the different vector bin packing heuristics was measured by their allocation density $d = \frac{n}{m}$, where n denotes the number of VMs to allocate and m the number of required servers. Since solutions obtained with the Gurobi solver were time-constrained and thus potentially suboptimal, we were unable to use the competitive value (i.e. heuristic vs. optimum solution) as metric. We now describe the simulation results in more detail.

A. Effect of profile length

Profile lengths with levels 6, 12, 24 and 288 were used for each heuristic. The factor scale, i.e. the number of VMs, was fixed at level 120. Figure 2 and 3 depict algorithm performance with respect to density d and the number of required servers. First Fit is represented by a single point owing to its independence of profile length.

As expected, SSAPv delivers the highest allocation density and lowest server count regardless of profile length. First Fit on the other hand yields the worst results regardless of profile length. Evidently, vector bin packing heuristics successfully exploit profile complementarities without much difference between them. Dot Product and FFv slightly outperform Cosine, which does not contradict the results of Xi et al. [11] as our vectors are of higher dimensionality.

Allocation density improves with increasing profile length: going from 6 to 24 yields the greatest improvement. This is consistent with the results of Speitkamp et al. [5] with regard

[2]https://github.com/jacksonicson/paper_IC2E_2014

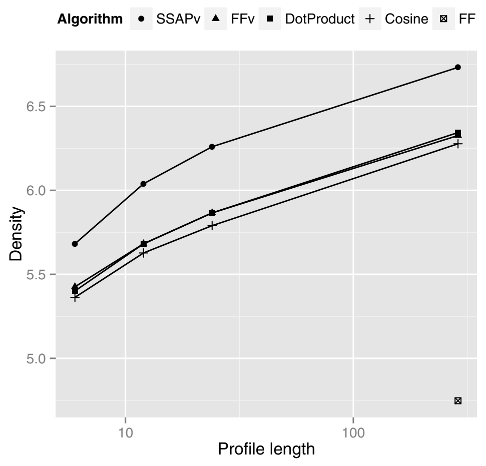

Fig. 2. Allocation density in relation to profile length, scale at 120 VMs.

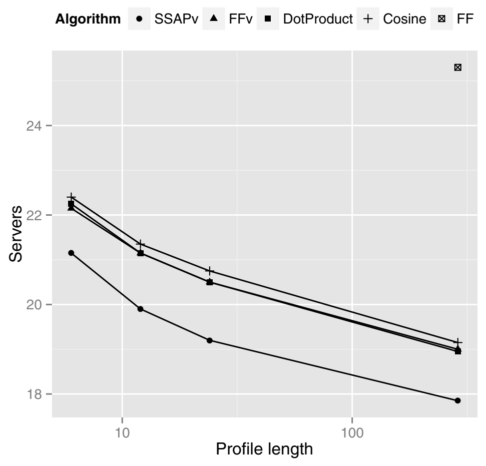

Fig. 3. Required servers in relation to profile length, scale at 120 VMs.

to SSAPv. The findings disagree with those of Talwar et al. [10], as increasing profile length and thus vector dimensionality have a positive effect on density.

SSAPv delivers the best allocation density but takes much longer to calculate solutions. For treatments with 288 slots and 120 VMs we limited calculation time to 45 minutes. In contrast, no heuristic took longer than 5 seconds to calculate the allocation without optimizations to the Python code base.

B. Effect of scale

In this set of simulations we wanted to analyze the effect of scenario scale on density and server count. We fixed the profile length at 288 values. The scale was varied between 30, 90, 120, 150, and 180 VMs. There is missing data for SSAPv as Gurobi was unable to provide valid solutions within 45 minutes for treatments in excess of 120 VMs.

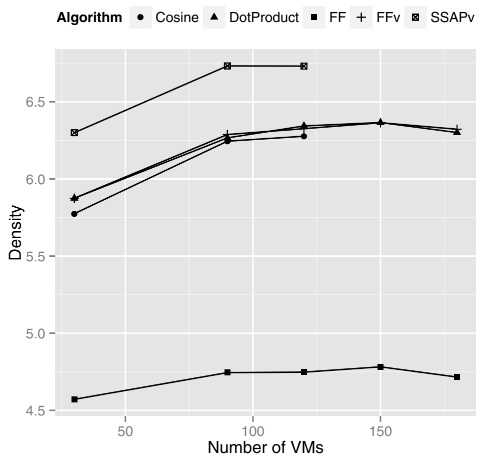

Fig. 4. Allocation density in dependency to scenario scale at a profile length of 288 values.

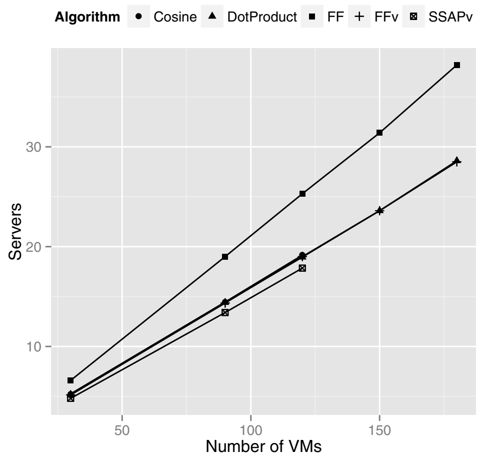

Fig. 5. Required servers in relation to scale, profile length at 288.

Results are depicted in Figure 4 and Figure 5. SSAPv again outperforms all heuristics with regard to server count and allocation density, while the one-dimensional First Fit heuristic produces the worst overall results.

The required number of servers increases almost linearly with the number of VMs for all heuristics and SSAPv. This is different from an increase in profile length, where server count decreases in a non-linear way. The slope of vector bin packing heuristics is about 0.15 and slightly greater than the one of SSAPv with 0.145. First Fit has the greatest slope with 0.21 which is especially bad for very large scenarios.

Increasing scale from 30 to 120 VMs again positively affects allocation density for all approaches as shown in Figure 4. This makes sense as larger scenarios should provide more complementarities between profiles. Increasing scale beyond 120 VMs does not increase allocation density further. We

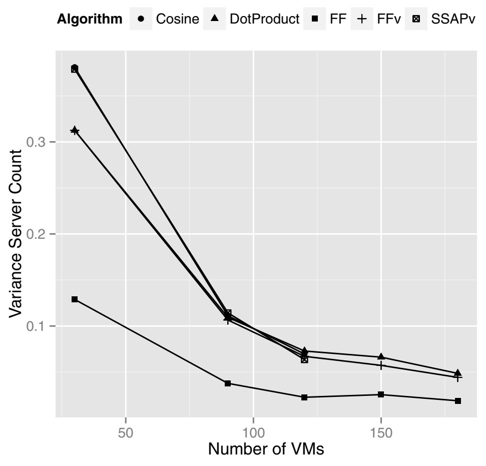

Fig. 6. Variance of the number of required servers in relation to scale, profile length at 288.

conclude that there is no benefit in simulations for larger scenarios.

We examined the variance of the number of required servers and scale between different treatments. We found density to be low if only a few VMs are used. As shown in Figure 6 the variance strongly decreases with scale going from 30 to 120 VMs and remains approximately constant for larger scenarios. This is likely because small scenarios do not provide enough complementarities that can be leveraged by vector bin packing algorithms.

This result suggests scenarios with a scale between 90 and 200 VMs. Smaller scenarios produce less dense allocations with a high variance of the number of required servers. Depending on the (random) choice of VM types and thus the ability to exploit complementarities, allocations can benefit or hurt accordingly.

Increasing scale above 200 VMs does not improve allocation density much, but takes longer to solve, especially for SSAPv. Although SSAPv provides better results, its practical applicability is questionable as solvers consume large amounts of time and might not even return valid results within bounded time.

C. Interaction of scale and profile length

Our findings show that increases in scale or profile length lead to denser allocations and thus decrease the number of required servers. We now pose the question of interaction effects between these two factors. Current results suggest that a simultaneous increase in both factors yields the densest allocations. However, adverse interaction effects might lead to less dense allocations.

We thus conducted an ANOVA analysis based on our simulation results. We set density as the target variable and included the two factors profile length (levels: 6, 12, and 24) and scale (levels: 30, 90, and 120), resulting in a full factorial

TABLE I
ANOVA RESULTS FOR FACTORS SIZE AND PROFILE LENGTH IN RELATION TO ALLOCATION DENSITY

	SumSq	MeanSq	Fvalue	Pr(>F)
VM Count	15.36	15.361	105.157	<2e-16***
Profile Length	27.56	13.780	94.329	<2e-16***
VM Count:Profile Length	0.08	0.038	0.257	0.773
Residuals	104.30	0.146		

TABLE II
TUKEYHSD COMPARISON OF ALLOCATION ALGORITHMS REGARDING ALLOCATION DENSITY

	diff	lwr	upr	padj
DotProduct-Cosine	0.06666667	-0.1472012	0.2805345	0.9082794
FF-Cosine	-1.52933552	-1.7432034	-1.3154676	0.0000000
FFv-Cosine	0.04912281	-0.1647451	0.2629907	0.9683139
SSAPv-Cosine	0.45473979	0.2408719	0.6686077	0.0000005
FF-DotProduct	-1.59600218	-1.8098701	-1.3821343	0.0000000
FFv-DotProduct	-0.01754386	-0.2314117	0.1963240	0.9993892
SSAPv-DotProduct	0.38807312	0.1742053	0.6019410	0.0000211
FFv-FF	1.57845833	1.3645905	1.7923262	0.0000000
SSAPv-FF	1.98407531	1.7702074	2.1979432	0.0000000
SSAPv-FFv	0.40561698	0.1917491	0.6194849	0.0000082

3^2 design. Refer to Table I for the results of the two-way ANOVA analysis.

As expected from our chart analysis, both factors are significant at p-values below 0.001. Interaction between profile length and scale is not significant with a p-value of 0.967. In other words, increasing both factors yields the best allocation results.

The plots of residuals versus fitted values and residuals versus run order of treatments did not exhibit any heteroscedasticity, clear pattern or trend. We found a single outlier in the residuals. Excluding this outlier had no effect on the ANOVA results. The normal probability plot reveals a deviation from the normality assumption, indicating a slightly skewed distribution that should not have any impact on the results.

D. Comparing algorithms

As the previous charts show, vector bin packing heuristics produce very similar results, with First Fit consistently performing worse and SSAPv better. To further verify this finding we conducted a one-way ANOVA analysis on the allocation density. The factor algorithm had five levels: FF, FFv, DotProduct, Cosine and SSAPv. The data for a scale of 120 VMs and a profile length of 288 values was used for analysis. The results of the TukeyHSD test are shown in Table II.

The residual plots of residuals versus fitted values and residuals versus run order of treatments did not show any heteroscedasticity, clear pattern or trend. Normal probability plot was unobtrusive again.

SSAPv is significant compared to any of the heuristics with p-values < 0.001. FF is significant compared to any of the algorithms. The results are consistent with our previous findings gained by chart analysis.

There is no significant differences between vector bin packing heuristics. The more advanced heuristics DotProduct and Cosine have no advantage over the straightforward FFv heuristic. Again, this result is in line with our expectations from chart analysis. For low dimensional vectors [10] and [11] showed that DotProduct and Cosine outperform FFv. At least for high dimensional vectors, we could not confirm this finding.

TABLE III
TUKEYHSD COMPARING SSAPV (PROFILE LENGTH 24) WITH HEURISTICS (PROFILE LENGTH 288) WITH REGARD TO ALLOCATION DENSITY

	diff	lwr	upr	padj
DotProduct-Cosine	0.06666667	-0.1464126	0.2797460	0.9071496
FF-Cosine	-1.52933552	-1.7424148	-1.3162562	0.0000000
FFv-Cosine	0.04912281	-0.1639565	0.2622021	0.9678878
SSAPv-Cosine	-0.01754386	-0.2306232	0.1955355	0.9993802
FF-DotProduct	-1.59600218	-1.8090815	-1.3829229	0.0000000
FFv-DotProduct	-0.01754386	-0.2306232	0.1955355	0.9993802
SSAPv-DotProduct	-0.08421053	-0.2972898	0.1288688	0.845066298
FFv-FF	1.57845833	1.3653790	1.7915376	0.0000000
SSAPv-FF	1.51179166	1.2987123	1.7248710	0.0000000
SSAPv-FFv	-0.06666667	-0.2797460	0.1464126	0.9071496

Despite significant differences between SSAPv and vector bin packing heuristics, these might be irrelevant in practical applications. The results shown in Figure 5 show that SSAPv and vector bin packing heuristics produce almost identical results if values are rounded to the next integer. We should stress once more that a time limit was imposed on the Gurobi solver. Given unbounded time, SSAPv solutions may still improve.

To contain solvers with respect to calculation time for large scenarios, the SSAPv is often parametrized with shorter profile lengths. We pose the question of how well heuristics with profile lengths of 288 perform compared to SSAPv solutions based on a downsampled profile length of 24.

We conducted an ANOVA analysis to compare the allocation density of SSAPv with heuristic solutions, with profile lengths of 288 for the heuristics and 24 for the SSAPv. Table III shows the results of the TukeyHSD test.

In this scenario, SSAPv does not perform significantly better than vector bin packing heuristics. Similar to our previous ANOVA analysis, FF performs significantly worse than any other approach including SSAPv.

The finding is practically relevant, since it indicates that heuristics perform just as well as the time-constrained SSAPv solver, especially if the time limit is in the order of seconds. While heuristics can be parametrized with high profile lengths, SSAPv has to be solved for small profile lengths to curb calculation time. The trade-off obviously decreases SSAPv solution quality to a point where it performs just like heuristic solutions.

We found that the 99th percentile of execution time for a scale of 120 VMs and a profile length of 288 values amounts to 3.6 seconds for heuristics and 30 minutes for the SSAPv. Most of the SSAPv solutions were not proven optimal by Gurobi.

VI. SERVICE QUALITY

Penalties usually take effect if service quality goals are not fulfilled. Therefore, the quality of service provisioning is an important aspect for data center operators. Often, VMs are oversized to handle unexpected load spikes. Profile based consolidation as proposed by this work pursues the goal of eliminating entirely such security buffers.

Still, security buffers can go side by side with profiles. Instead of overprovisioning resources, security buffers can be incorporated into profile calculation. A profile can be calculated such that 100% of past loads are coverd. An additional security buffer can be added to profile values that exhibit a high demand so that buffer size is varied with expected demand. Alternatively, a profile could include only 90% of the past load demands. This would be a more aggressive profile which would result in more aggressive allocations.

One might expect that slight deviations of the load behavior in an aggressively calculated profile result in service level violations. However, this is not the case in realistic environments. In a previous study that is still under review we analyzed the performance of consolidations calculated by the SSAPv in a testbed infrastructure [15]. It showed that even significant differences between the actual load and an aggressively calculated consolidation profile only cause slight increases in service level violations. Taking into account that vector bin packing heuristics on average tend to calculate conservative allocations mitigates the concern even more.

VII. Conclusions

We analyzed one bin packing and three vector bin packing heuristics alongside an SSAPv solver to allocate VMs to a minimum number of servers based on VM load profiles. Our analysis leads to a number of guidelines that may benefit data center operators as well as software engineers facing the problem of (VM) allocation.

First, allocations for small scale infrastructures with less than 50 VMs should be calculated using SSAPv or other linear programming techniques. They deliver significantly better results than all heuristics we have tested. Modern linear program solvers frequently find a viable solution within a couple of minutes.

Second, vector bin packing outperforms one-dimensional packing but there are no significant differences between vector bin packing heuristics. We compared four packing heuristics ranging from the very simple, one-dimensional FF to ones that leverage vector algebra. FF reduces the load profile of a VM to a single scalar value, performing significantly worse than FFv or any other vector bin packing heuristic leveraging load profiles with profile lengths from 6 to 288.

Third, including more than one server rack consisting of approximately 40 servers and 200 VMs into a single allocation calculation does not improve allocation density. While SSAPv and heuristic solutions improved with scale increasing up to 180 VMs, increasing scale even further did not yield significantly better results (or worse, for that matter). Instead, calculation time increases significantly for the SSAPv. We therefore recommend per-rack calculations.

Fourth, vector bin packing heuristics should be parametrized with detailed load profiles. If the profile length is increased from 6 to 288 values, allocation density increases as well. The biggest improvement is noticeable between 6 and 24 values. Beyond that, allocation density increases only slowly.

Fifth, medium and large scale scenarios should be calculated using a heuristic if calculation time is constrained. While such scenarios can still be solved by linear program solvers, the problem complexity and in turn the profile length have to be reduced to obtain results within minutes. However, going below profile lengths of 24 negatively affects allocation density. We found that heuristics running with highly detailed profiles on large scenarios produce the same allocation quality as SSAPv running on downsampled, low-dimensional profiles.

References

[1] C. Babcock, “Data Centers May Not Gobble Earth, After All,” 2013. [Online]. Available: http://www.informationweek.com/hardware/data-centers/data-centers-may-not-gobble-earth-after/231300144

[2] J. Glanz, “Power, Pollution and the Internet,” 2012. [Online]. Available: http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html

[3] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Y. Zomaya, “A taxonomy and survey of energy-efficient data centers and cloud computing systems,” *CoRR*, vol. abs/1007.0066, 2010.

[4] L. A. Barroso and U. Hölzle, “The Case for Energy-Proportional Computing,” *Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007. [Online]. Available: http://portal.acm.org/citation.cfm?id=1339817.1339894 http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4404806

[5] B. Speitkamp and M. Bichler, “A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers,” *IEEE Transactions on Services Computing*, vol. 3, no. 4, pp. 266–278, Oct. 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5467027

[6] J. Csirik, J. B. G. Frenk, M. Labb, and S. Zhang, “On the multidimensional vector bin packing.” *Acta Cybern.*, vol. 9, no. 4, pp. 361–369, 1990. [Online]. Available: http://dblp.uni-trier.de/db/journals/actaC/actaC9.htmlCsirikFLZ90

[7] A. Andrzejak, M. Arlitt, and J. Rolia, “Bounding the Resource Savings of Utility Computing Models,” *HP Labs*, 2002. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.5.1094

[8] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, “Resource allocation algorithms for virtualized service hosting platforms,” *J. Parallel Distrib. Comput.*, vol. 70, no. 9, pp. 962–974, Sep. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.jpdc.2010.05.006

[9] T. Setzer and M. Bichler, “Using matrix approximation for high-dimensional discrete optimization problems: Server consolidation based on cyclic time-series data,” *European Journal of Operational Research*, vol. 227, no. 1, pp. 62–75, 2013. [Online]. Available: http://ideas.repec.org/a/eee/ejores/v227y2013i1p62-75.html

[10] K. Talwar, R. Ramasubramanian, R. Panigrahy, and U. Wieder, “No Title,” Microsoft Research, Tech. Rep., 2011. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=144571

[11] X. Li, A. Ventresque, N. Stokes, J. Thorburn, and J. Murphy, “iVMp: an Interactive VM Placement Algorithm for Agile Capital Allocation,” in *IEEE Conference on Cloud Computing*, 2013.

[12] S. Srikantaiah, A. Kansal, and F. Zhao, “Energy aware consolidation for cloud computing,” in *Proceedings of the 2008 conference on Power aware computing and systems*, ser. HotPower’08. Berkeley, CA, USA: USENIX Association, 2008, pp. 10–10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855610.1855620

[13] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, “Resource allocation algorithms for virtualized service hosting platforms,” *Journal of Parallel and Distributed Computing*, vol. 70, no. 9, pp. 962 – 974, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731510000997

[14] K. Maruyama, S. Chang, and D. Tang, “A general packing algorithm for multidimensional resource requirements,” *International Journal of Computer and Information Sciences*, vol. 6, no. 2, pp. 131–149, 1977. [Online]. Available: http://dx.doi.org/10.1007/BF00999302

[15] A. Wolke, M. Bichler, and T. Setzer, “Energy efficient capacity management in virtualized data centers: optimization-based planning vs. real-time control,” under review.