Energy aware Colocation of Workload in Data centers

Madhurima Pore, Zahra Abbasi, Sandeep K. S. Gupta and Georgios Varsamopoulos
Impact Lab, School of Computing, Informatics and Decision Systems Engineering, ASU, Tempe, Arizona
{madhurima.pore,zahra.abbasi,sandeep.gupta and georgios.varsamopoulos}@asu.edu

***Abstract*—[1] There exists an interference due to colocating applications which depends on the applications' workload types, degrades the performance and affects the energy consumption of applications. We hypothesize the interference energy consumption and model it as "interference coefficient" and use it to develop an application-aware colocation management policy to colocate the applications in data centers.**

Including the interference effect in simulations using synthetic workload in the colocation management policy results energy savings of up to 8%.

***Index Terms*—Energy efficient colocation, energy proportionality and Data center energy efficiency**

I. Introduction

Increase of energy proportionality (i.e. power consumption increases with workload and idle power is very low) and energy efficiency (i.e. performing more workload in specific power budget) contribute in making data centers greener. Ideal energy-proportional systems have zero idle power and the workload-to-power ratio (i.e., energy efficiency) is constant (refer Figure 1). Hence, data center workload management policies can be highly simplified. Overall energy efficiency of the data center directly impacts the cost of operations. However, current data center servers are not ideally energy proportional (see example power plots in Figure 1). Though the idle power of modern servers has reduced, the power utilization curve is not linear. Thus, increasing energy proportional behavior of data centers involves server consolidation to operate servers at their peak energy efficiency. This is because the peak energy efficiency of servers is when they run near 100% utilization. However, recent works suggest that consolidation increases the contention on the shared resources such as on-chip caches, buses, main memory, CPUs and network [9], [10]. This contention results in performance degradation of applications. The performance overhead due to contention depends on the workload type of applications [9], [10] and can be minimized through colocating workloads that yield the least resource sharing. However, the colocation of applications can also induce energy consumption overhead due to increase in runtime. In this work, we propose an energy composition model that considers the overhead energy to design an application-aware colocation management policy.

[1]This work has been partly funded by NSF, CRI grant #0855527, CNS grant #0834797, CNS grant #1218505 and Intel Corp.

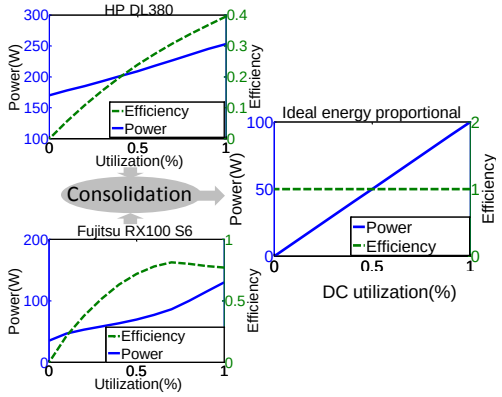


Fig. 1: Illustration of energy efficiency and power-utilization curve of two modern systems and an ideal consolidation which results in a energy proportional data center with constant energy efficiency.

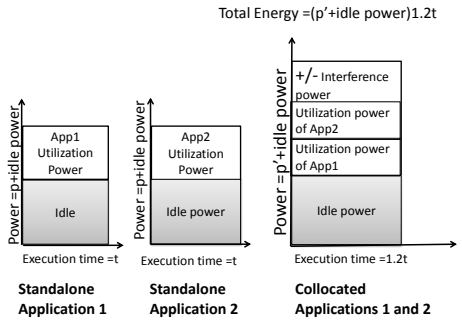


Fig. 2: Energy Composition: Energy of Application A and B when run in standalone is composed of idle and utilization energy. Application colocation results into a composition energy that depends on application energy of A and B as well as the interference energy due to contention.

The major research question of this paper is that if the standalone energy consumption of applications in a server is known, is it possible to estimate the energy consumption and performance of applications when colocating them in the server? This problem can be described in Figure 2. When applications are run separately, they consume idle energy as well as energy that depends on the utilization of the server. We estimate

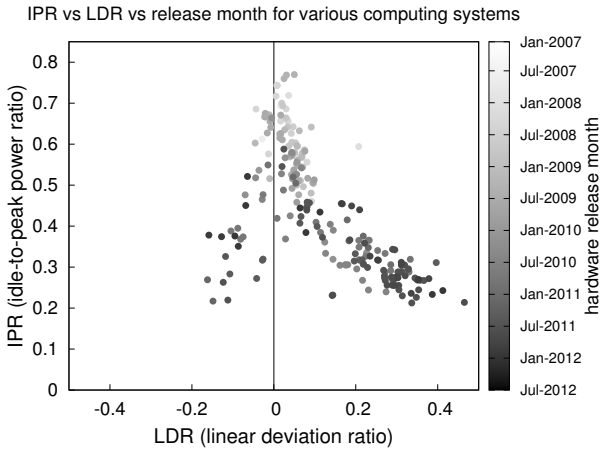


Fig. 3: Scatterplot of historical trends of energy proportionality (IPR vs LDR) for server systems for the past five years (data source: SPECPower_ssj2008 public results).

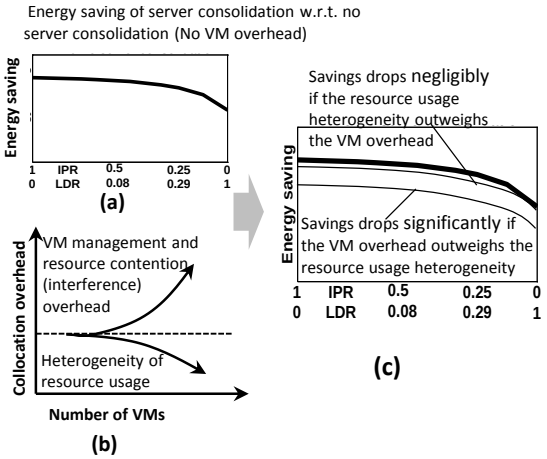


Fig. 4: (a) Savings of server consolidation over a spectrum of servers' IPR and LDR values with no colocation overhead assumption. **(b)** colocation of VMs induces overhead but the overhead is minimized if VMs are not competing for the same resources. **(c)** Depending on the overall overhead, the server consolidation savings in (a) significantly or negligibly reduces.

that the colocation incurs energy that is the sum of utilization energy of individual applications, the idle energy and extra energy consumption due to interference.

E.g., consider standalone run of a batch application that requires average utilization power of p (i.e. total power minus idle power) and execution time t. If two such applications are colocated, the average utilization power of the host server becomes p', where p' may become greater or less than $2p$ depending on the contention type. Ideally, p' should equal $2p$ and performance degradation should be zero. However, due to contention, the performance of the applications is reduced e.g., by 20% (i.e., execution time $=1.2t$). Hence, the total energy can be estimated as $(p' + \text{idle power}) \times 1.2t$. By being aware of type of applications, we make p' close to $2p$ and performance degradation to zero.

The main contributions of the paper are as follows:

- **Energy composition model**: The paper introduces an energy composition model to estimate the energy consumption of colocated applications. The model is based on "interference coefficient " which depends on the workload type of applications and the energy consumption of applications when running standalone. We verified the model using a empirical study with synthetic workload.
- **Developing an energy efficient colocation management aware of the application types**: The energy composition model is used to design an *application aware colocation management* policy to minimize total energy without compromising applications' performance requirements. It is modeled as combinatorial optimization problem and is further decomposed to set-cover problem and *Apriori* algorithm to be solved in a time efficient way [11].

II. Motivation and Related Work

This section provides a motivation for this work, and also reviews the related work.

A. Energy proportionality and application colocation in data centers

Modern data center servers demonstrate energy proportional behavior that can be captured using the metrics Idle to Peak power Ratio (IPR) and Linear Deviation Ratio (LDR) [12]. IPR measures how close to zero the idle power is, and LDR measures the linearity of the power curve (i.e., how close the power curve is to the hypothetical linear curve connecting idle power to peak power).

As the historical trends of IPR and LDR values in Figure 3 show the servers' IPR value is reducing, however their power-utilization curve is becoming nonlinear as well. According to a previous study the energy saving of the server consolidation over energy proportionality metrics spectrum is as shown in Figure 4(a) which is based on assuming zero colocation overhead. However, in this work, we show how colocation overhead decreases the energy efficiency that can be achieved by server consolidation. The total colocation overhead in a host server is governed by two trends: contention overhead and heterogeneity of resource usage by colocated VMs as illustrated in Figure 4(b). If the heterogeneity of resource usage outweighs the contention overhead of shared resources, increasing the number of colocated VMs does not significantly increase the colocation energy overhead.

Putting the aforementioned trends together, application aware colocation management can play a role in increasing the energy saving of consolidation as illustrated in Figure 4(c). It is therefore prudent to profile the colocation interference and introduce it as a parameter to the decision making of workload consolidation.

B. Performance implication in colocating applications

The interference effect of colocations on applications' performance has been studied using empirical methods in some recent works [8], [9]. The "Bubble-up" methodology predicts performance degradation due to colocation of applications [9]. It is used to design a colocation policy for delay-sensitive Google workloads. Authors in [8] consider the applications' workload type as well as the effect of underlying hardware to avoid performance degradation caused by colocation. The above works are motivation to study energy-efficient colocation of applications, since the performance degradation (e.g., increasing the execution time of applications) is tightly correlated to the energy consumption. Much of the research has proposed VM consolidation as an energy-efficient technique in data center environment [4], [6], [7]. However, very few of them consider the performance degradation caused by colocation interference and workload dynamics [4]–[7]. The most related works to our work are [9], [5] as they consider the workload type in colocation policy to reduce contention.

C. Complicating Factors

The colocation of applications has its own challenges.

- Workload intensity is dynamic.
- The data center hardware has inherent heterogeneity, in terms of power profiles. This raises the question of how to distribute the data center workload amongst servers to yield the best performance.
- The power performance profiles of servers are not linear. The nonlinearity increases the complexity of models and solutions of the energy efficient colocation management.
- There is no benchmark to characterize the interferences among colocated VMs.

III. System architecture

Data centers host different applications ranging from batch jobs [9] to web services (e.g., e-commerce) [3]. The workload types of such applications are different e.g., CPU-intensive workloads (e.g., to analyze or organize data) and memory-intensive workloads (e.g., data retrieval tasks). We consider such a data center with heterogeneous workloads. We assume applications in a data center consist of one or more tasks where each task can be assigned to a VM. Let the set $A_t = \{a_{1,t} \dots a_{i,t} \dots a_{|A|,t}\}$ be the set of tasks to be assigned to n total identical servers at a time t where, $|A|$ is the number of tasks in set A. Each $a_{i,t}$ represents a VM and is associated with a workload type $w_{i,t}$. Each pair of tasks $a_{i,t}$ and $a_{k,t}$ may or may not have the same workload type.

We represent a colocation set by $C_{j,t} \subseteq A_t$; each $C_{j,t}$ runs on a single server. The colocation policy partitions A_t into $\hat{n} \leq n$ colocation sets such that all the tasks belong to one set: $\exists C_{j,t} | a_{i,t} \in C_{j,t}, \forall i = 1 \dots k_t$.

The energy consumption, e^C_j, of each colocation set $C_{j,t}$, captures the energy consumption of its applications and their interference effect. The next section explains the model of interference effect for a colocation policy through an empirical study; This model is further used in §V to design an energy efficient colocation management scheme.

IV. Modeling the colocation Interference

The interference effect of colocation specifies how the performance and energy consumption of applications are affected, i.e. how much application "A" co-running with application "B" deviates from its standalone performance and energy. Studies in [8], [9] show promising results that there is contention of resources due to colocation. They characterize the performance degradation of colocated applications due to the contention.

Our proposed **hypothesis** is that, if such interference exists amongst the colocated application, it can be modeled as a "separate task". The utilization energy of an application is defined as its total energy consumption minus its idle energy consumption. Consider a case with two colocated applications, indexed by i and k, in a host server with an equal running time, the total energy consumption of the server can be written as follows:

$$e^c = e^{idle} + e_i + e_k + e^{ief}_{ik}, \tag{1}$$

where e_i and e_k denote the utilization energy consumption of tasks i and k respectively when running solely on the same system, e^{idle} denotes the idle energy to run either of the tasks i and k when running separately, and e^{ief}_{ik} denotes additional energy consumption due to the colocation interference between tasks i and k which depends on the workload type of the tasks.

To calculate e^{ief}_{ik}, we introduce *"interference coefficient"*, denoted by α, which estimates the interference energy:

$$e^{ief}_{ik} = \alpha_{ik}(e_i + e_k + e^{idle}) \tag{2}$$

The following analysis provides insight into the above equation. Consider two applications indexed by i and k. Both take t time unit to complete, and consume p_i and p_k utilization power at a server with idle power of p^{idle}. The ideal colocation energy of applications is as follows:

$$e^{c,ideal} = (p^{idle} + p')t = (p^{idle} + p_i + p_k)t = e^{idle} + e_i + e_k.$$

where p' is the ideal power consumption of the host server. However, in practice, due to colocation interference, p' may be greater or less than the sum of utilization power of applications and can be written as $p' = p_i + p_k + p^o_{i,k}$, where $p^o_{i,k}$ is a real number and , $p^o_{i,k} > -(p_i + p_k)$. Further, the execution time of applications, denoted by t' increase, $t' \geq t$. Let $\tau > 0$

where $t' = t + \tau$. The energy consumption of colocation can be written as follows:

$$e^c = (p^{idle} + p_i + p_k + p^o_{i,k})(t + \tau)$$
$$= e^{idle} + e_i + e_k + (p^{idle} + p_i + p_k)\tau + p^o_{i,k}(t + \tau). \quad (3)$$

We assume that the term $p^o_{i,k}(t+\tau)$ can be approximated as a percentage of the term $(p^{idle} + p_i + p_k)\tau$ such that $(p^{idle} + p_i + p_k)\tau + p^o_{i,k}(t+\tau) = (p^{idle} + p_i + p_k)\tau'$, where $\tau' \geq 0$. Let $\alpha_{i,k} = \frac{\tau'}{t}$. Therefore the colocation energy is:

$$e^c = e^{idle} + e_i + e_k + (p^{idle} + p_i + p_k)t\alpha_{i,j}$$
$$= e^{idle} + e_i + e_k + \alpha_{i,k}(p^{idle} + p_i + p_k). \quad (4)$$

The following experiments show how e^{ief}_{ik} depends on the workload type of applications.

A. Characterizing the interference effect of colocation

To understand the interference effect, we performed some experiments using CPU-intensive and memory-intensive applications.

CPU-intensive benchmark (CPU): We design a multi-thread benchmark that generates CPU workload using a prime number generation program. Benchmark input is the percentage of maximum workload sustained by the system (i.e. in our experiment, 22000 prime number generations). For e.g., CPU[10%] is generated by adjusting the number of prime number computations such that each core sleeps 90% of time and works during rest 10% with time granularity of 100ms.

Memory-intensive benchmark (MEM): We design a memory benchmark that scales the memory workload by generating random accesses to the memory and adjusting the total number of memory operations (e.g. reading), sleep time between operations and working set size. Maximum working set size sustained in our experiments is 1MiB. For e.g. MEM[10%] means we generate a workload for working set size 1KiB and adjust the number of memory operations such that the memory workload is offered 10% of time compared to the peak workload.

We run the following experiments to examine the interference effect under different workload types and intensity. When a benchmark is run, the power is observed during the complete run of the benchmark code. The energy is computed as the product of average power during the run into the total execution time.

1) Colocation energy overhead vs. workload type: We run three experiments on one physical server: (i) running two CPU benchmarks both at 30% workload, (ii) running two memory benchmarks both at 25% workload, and (iii) running one CPU benchmark at 30% workload and one memory benchmark at 25% workload. Figure 5(a) indicates that colocated CPU MEM application incur less energy consumption than two CPU and two MEM colocations. This can be due to less contention in shared resources in the colocated CPU-MEM applications.

2) Colocation overhead vs. workload intensity: We run the following experiments. Firstly, on one physical server, we colocate two CPU applications with increasing workload intensity, CPU [10%-50%]. Similarly, in the second experiment, we run two memory intensive applications with increasing workload intensity MEM[10%-50%]. Figures 5(b) and (c) show that colocation energy overhead and performance degradation increase as workload intensity increases for both CPU and MEM workload. Colocation of CPU MEM workload show that the average interference coefficient of CPU CPU colocation is observed to be four times that of CPU MEM colocations.

3) Colocation with realistic workloads: To obtain a realistic value of the interference coefficient, we setup SPECpower_ssj2008 workload on VMs and observed the energy consumption when two such VMs are colocated. The colocation incurs energy overhead, and the interference coefficient is observed to be 0.09.

V. Application aware colocation management

Using the results from the previous section, we model the total energy of the colocation set C_j as the sum of idle energy, utilization energy of each tasks and the overhead of interference energy as follows.

$$e^C_j = e^{idle} + \sum_{a_i \in C_j} e_i + \frac{1}{2} \sum_{a_i \in C_j} \sum_{a_k \in C_j, k \neq i} \alpha_{i,k}(e_i + e_k + e^{idle}), \quad (5)$$

where $e^{idle} = \max_{a_i \in C_j}(e^{idle}_i)$ denotes the idle energy to run the longest task of C_j when running standalone. The ratio 1/2 ensures that the interference effect for pair of tasks is considered only once. To consider the applications' performance, we model each server as a M/M/1 queue. In M/M/1, the response time can be modeled as $d = \frac{1}{\mu - \lambda}$ where μ and λ denote the server service rate and the workload arrival rate, respectively. Let μ_i, λ_i be the the service rate and workload arrival rate of task i respectively. The service rate of the colocation set j, denoted by μ^C_j, can be estimated using weighted average:

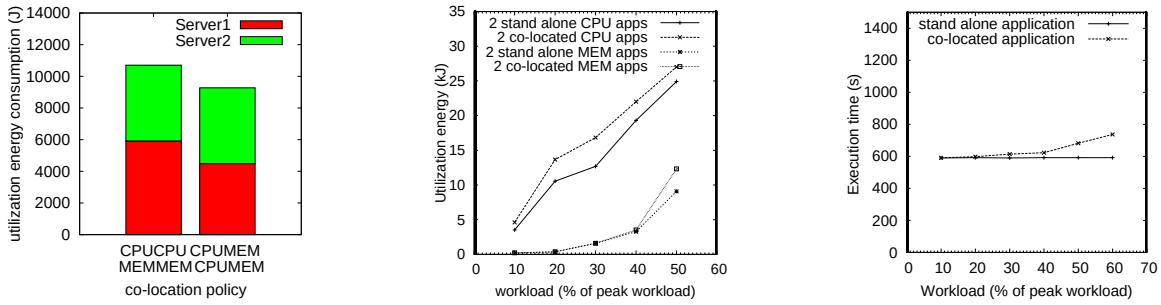
$$\mu^C_j = \sum_{a_i \in C_j} \frac{\lambda_i \mu_i}{\sum_{a_i \in C_j} \lambda_i}$$

The applications performance constraints bound the maximum possible workload arrival to a server as follows:

$$d^C_j = \frac{1}{\mu_j - \left(\sum_{a_i \in C_j} \lambda_i + \lambda^{ief}_j\right)} \leq d^{ref}_i, \ \forall a_i \in C_j \quad (6)$$

where λ^{ief}_j denotes the extra workload due to the interference effect of applications which can be estimated similar to the interference energy (see Eq. 5), as follows:

$$\lambda^{ief}_j = \frac{1}{2} \sum_{a_i \in C_j} \sum_{a_k \in C_j, i \neq j} \alpha_{i,k}(\lambda_i + \lambda_k)$$



(a) (b) (c)

Fig. 5: (a) Energy consumption versus colocation policy. **(b)** Utilization energy overhead w.r.t. workload type and intensity of the colocated applications, **(c)**Performance degradation due to colocation.

A. Problem formulation of application aware colocation management

Given the above factors, the energy-aware colocation problem can be formulated as to minimize the total energy consumption of colocated sets such that all the tasks are serviced and their performance (delay) requirement is met. Let x denote the selection of a colocation set, then the optimization problem can be stated as:

Minimize

$$\sum_j e_j^C x_j \tag{7}$$

such that $\sum_{j|a_i \in C_j} x_j = 1 \ , \forall a_i \in A$ [service constraint] (8)

$$d_j^C \leq d_i^{ref} \ , \forall a_i \in C_j \text{ [performance constraint]} \tag{9}$$

.

B. Solution of application aware colocation management

The above problem can be decomposed into two subproblems: (i) to find all *performance-feasible* colocation sets C_j that respect the applications performance requirements (Eq. 9), and (ii) to find energy efficient colocation sets among performance-feasible ones that cover all applications (i.e., solving the minimization problem as depicted by Eq. 7 subject to the service constraint, Eq. 8 over performance-feasible colocation sets) which is a set-covering problem [13].

The performance-feasible colocation sets i.e., a colocation set that meets its applications' performance requirement can be denoted by $\hat{C} = \{C_j | d_j^C \leq d_i^{ref} \ , \forall a_i \in C_j\}$ which represents all colocation sets that satisfy the performance requirement possibly searching through all subsets (i.e., $2^{|A|}$). Although the size of the power set grows exponentially in the number of applications, efficient search is possible using "anti-monotonicity" property, "Apriori" or "downward-closure" property [11]). This property guarantees that for a performance-feasible colocation set, all its subsets are also feasible and thus for a performance-infeasible colocation set, all its supersets must also be *performance-infeasible* . After obtaining performance-feasible solutions the energy efficient colocation sets are found using existing polynomial-time algorithms that solve the set covering problem with a bounded approximation ratio [13].

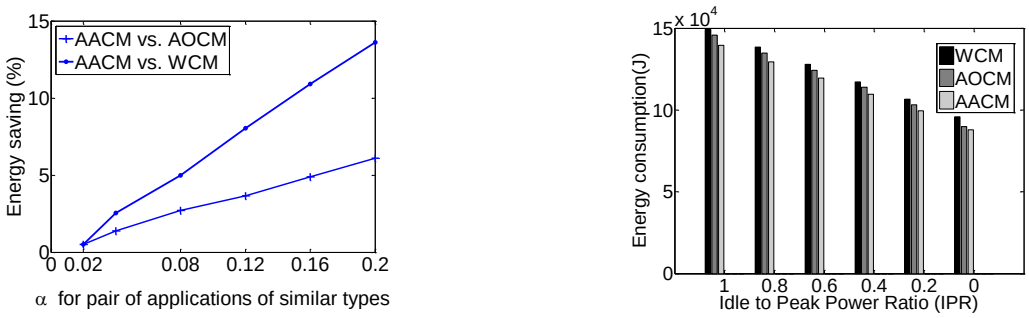
VI. Simulation study

We performed a simulation study to evaluate our Application Aware Colocation Management (AACM) (§V) with respect to magnitude of interference effect and servers power consumption model.

AACM is compared with two baseline algorithms: (i) Application Oblivious Colocation Manager (AOCM), which performs consolidation to minimize total applications' standalone energy, disregarding the interference effect of applications' colocation; this is a standard consolidation management scheme which is commonly used in previous works, and (ii) Worst case Colocation Manager (WCM), which is similar to AACM but prefers colocations of applications that incur higher interference effect; this scheme shows the maximum energy wastage of a consolidation manager that is not aware of colocation overhead. All schemes run on the performance-feasible colocations sets as described in §V-B. We use MATLAB 2010 and *GNU Linear Programming Kit (GLPK)* to optimally solve AACM, AOCM and WCM using branch-and-bound technique. Due to the NP-hardness of the problem, GLPK solver does not scale to large problem instances, performs well in our simulation study. For larger problems, our approxifmation algorithm (§V-B) can be used. Server model is IBM Systems x3650 M2 www.spec.org/power_ssj2008/results/) with idle power 100 W and peak power 300 W.

We assume applications of two workload types in the simulation study. We vary the workload intensity such number of total VMs (i.e. $|A|$) varies between 3 and 20 during a simulated time of 24 hours. Further, the average service rate is set to 5000 requests per second, and the workload arrival rate of tasks is such that every server has a maximum of 3 VMs.

The consolidation manager (i.e., both AACM, and AOCM) runs every hour. The workload of VMs during



(a) (b)

Fig. 6: Energy savings of AACM where **(a)** $\alpha = 0.02$ for applications of different types, and **(b)** $\alpha = 0.0.02$ for applications of similar types, and $\alpha = 0.1$ for applications of different types over IPR values.

every two consecutive consolidation is considered to be fixed. Both of the applications have equal VMs and reference delay of $3ms$. According to §IV-A for pair of similar applications $\alpha = 0.09$, which is almost four times greater than α for applications of different types. Therefore, in the experiments we fixed α for applications of different types to 0.02 and vary it for pair of similar applications between $0.02 - 0.2$ to investigate the energy saving of AACM in the proximity of the observed value. Note that very high value for α (e.g., in our experiment $\alpha > 0.2$) forbids colocation due to exceeding the performance threshold.

A. AACM energy efficiency w.r.t. interference effect

Figure 6(a) indicates that the energy saving of AACM linearly increase with increasing interference coefficient.

B. AACM energy efficiency w.r.t. IPR

Figure 6(b) shows that AACM saves energy over IPR spectrum (i.e., 2-4.2% energy saving w.r.t. AOCM and 6-8% energy saving w.r.t. WCM). since both idle power and utilization power contribute in interference energy and consequently in energy saving of AACM.

VII. Conclusion

This paper tackles the problem of profiling and modeling the application colocation interference overhead on the energy and performance, which affects the energy savings and achieved QoS of data center management policies. The colocation interference changes with the workload type of the applications. To model it, we perform an empirical study and introduce an interference coefficient which rates the colocation of every application of specific workload types. Since characterizing the interference effect is order of different workload types in the data center and not on the number of applications, this model is scalable for data centers with large number of applications. Using the interference coefficient, we proposed application-aware colocation management and showed its energy efficiency is shown through a simulation study. Currently we consider a constant interference coefficient with respect to workload intensity, which will be investigated in future. Also more investigations are required to incorporate this model into server provisioning, workload management and cross-data-center management algorithms using realistic workload benchmarks and non-linear power models [1], [2].

References

[1] Z. Abbasi, T. Mukherjee, G. Varsamopoulos, and S. K. S. Gupta. DAHM: A green and dynamic web application hosting manager across geographically distributed data centers.

[2] Z. Abbasi, G. Varsamopoulos, and S. K. S. Gupta. Tacoma: Server and workload management in internet data centers considering cooling-computing power trade-off and energy proportionality. *ACM Trans. Archit. Code Optim.*, 9(2):11:1–11:37, June 2012.

[3] D. Ardagna, B. Panicucci, M. Trubian, and L. Zhang. Energy-aware autonomic resource allocation in multi-tier virtualized environments. *IEEE Transactions on Services Computing*, 2010.

[4] N. Bobroff, A. Kochut, and K. Beaty. Dynamic placement of virtual machines for managing sla violations. In *Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium on*, pages 119–128. IEEE, 2007.

[5] G. Dhiman, G. Marchetti, and T. Rosing. vgreen: a system for energy efficient computing in virtualized environments. In *Proceedings of the 14th ACM/IEEE international symposium on Low power electronics and design*, ISLPED '09, pages 243–248, New York, NY, USA, 2009. ACM.

[6] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper. Resource pool management: Reactive versus proactive or let's be friends. *Comput. Netw.*, 53:2905–2922, December 2009.

[7] F. Hermenier, X. Lorca, JM. Menaud, G. Muller, and J. Lawall. Entropy: a consolidation manager for clusters. In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, VEE '09, pages 41–50, New York, NY, USA, 2009. ACM.

[8] J. Mars, L. Tang, and R. Hundt. Heterogeneity in homogeneous warehouse-scale computers: A performance opportunity. *Computer Architecture Letters*, 10(2):29 –32, july-dec. 2011.

[9] J. Mars, L. Tang, R. Hundt, K. Skadron, and M.L. Soffa. Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations. Porto Alegre, Brazil, Dec. 2011.

[10] A. Merkel, J. Stoess, and F. Bellosa. Resource-conscious scheduling for energy efficiency on multicore processors. pages 153–166, 2010.

[11] P.N. Tan, M. Steinbach, V. Kumar, et al. *Introduction to data mining*. Pearson Addison Wesley Boston, 2006.

[12] G. Varsamopoulos, Z. Abbasi, and S. K. S. Gupta. Trends and effects of energy proportionality on server provisioning in data centers. In *International Conference on High performance Computing Conference (HiPC2010)*, December 2010.

[13] V. Vazirani. *Approximation algorithms*. Springer USA, 2003.