Load Balancing for Sustainable ICT *

Alexandru-Adrian Tantar
University of Luxembourg
Coudenhove-Kalergi
Luxembourg
alexandru.tantar@uni.lu

Emilia Tantar
University of Luxembourg
Coudenhove-Kalergi
Luxembourg
emilia.tantar@uni.lu

Pascal Bouvry
University of Luxembourg
Coudenhove-Kalergi
Luxembourg
pascal.bouvry@uni.lu

ABSTRACT

The herein paper addresses the issue of providing a model and guidelines for constructing a sustainable ICT environment at the University of Luxembourg. A particular context is thus considered, based on a real-life project that has as aim to provide a sustainable environment for the ICT infrastructure of the university. According to the different environment constraints and requirements, the objectives are to minimize electricity consumption by employing virtualization techniques and also to reduce carbon emissions by creating a load balanced charge of the computers that build the infrastructure. The quality of service is also addressed by provisioning factors. A multi-objective dynamic approach is considered in order to cope with the simultaneous optimization of the mentioned objectives and the dynamic nature of the system.

Categories and Subject Descriptors

I.2 [**Artificial intelligence**]: Miscellaneous; J.0 [**Computer applications**]: General

General Terms

Design, Experimentation

Keywords

load balancing, sustainable ICT

1. INTRODUCTION

The main purpose of the project that encloses the results of this paper is to reduce the carbon emissions that result out of the University of Luxembourg's ICT operations. In this sense one of the goals is to implement measures that reduce the electricity consumption needed for the well-functioning of the university's ICT facilities. To this end, a viable solution for reducing energy consumption can be implemented by using virtual machines. This solution allows grouping multiple virtual machines on a reduced number of resources thus leading to a more efficient exploitation of the available computational power. In addition, low energy consumption terminals can be used to access the virtual machines, dispatching the computational load over a cloud of more powerful physical resources. At the same time, sharing access across services or end-users via virtual machines can reduce the number of active physical machines hence providing a straightforward approach for reducing energy consumption and carbon emissions.

The experience of more than 900 universities using virtualization solutions [1] proved that important cuts in electricity expenses can be attained. Statistics for regular usage with alternating peaks and no-activity periods place computational requirements at an average of less than 40% of the total available computational power. Administrative and user groups can be defined for most ICT environments delimiting the requirements of a research department, for example, from the ones of students or personnel. End-users like the students at the university or the administrative staff can thus share via multiple virtual machines the same resource directly reducing the number of running machines.

The aim of this work is to provide a model and guidelines for the assignment of virtual machines over a cloud of real machines. An important characteristic of cloud computing is related to the three architectural layers involved in the management of cloud computing environments [8], each of them serving a different purpose and handling different products: (1) Infrastructure as a service **(IaaS)**, (2) Platform as a Service **(PaaS)**, and (3) Software as a Service **(SaaS)**. The perspective we address in this work regards the Platform as a Service layer, the final goal being to manipulate the virtual machines (seen as platforms that run an operating system).

At an optimization level, the main challenges raised by virtualization come from the need of ensuring a load balancing over the cloud of resources in order to have an efficient energy consumption model. In order to fulfill the sustainable ICT desiderata, the alocation of virtual machines to mainframe computers aims at ensuring a dynamic provisioning as to minimize energy consumption. Also, with the administration and energy costs of the server rooms surpassing the hardware costs and given the rising cost of electricity, green computing and load balancing become even more important.

The problem we are facing can be modeled as a **dynamic preemptive load balancing with stochastic execution times in heterogeneous environments**. The difficulty of

*

GECCO'11, July 12–16, 2011, Dublin, Ireland.

the problem is not only given by the dynamic character fo the problem, but also by the fact that several objectives need to be optimized simultaneously in the presence of several stochastic factors.

We therefore propose a solution for the efficient distribution of virtual machines across a cloud of computational resources as to minimize the energy consumption. In the effort of creating a sustainable environment, besides the integration of renewable energy resources, the University of Luxembourg is also planning the reduction of electricity consumption for the future Belval campus which will gather all current facilities and research laboratories.

1.1 Background Notions and Notation

Let F be an objective function, defined over X (decision space) and taking values in Y (objective space), $F : X \to Y$, one may consider the $\min_{x \in X} F(x)$ minimization problem.

For the multi-objective case, F is defined as a vector of objective functions $F : X \to \mathbb{R}^k, \quad F(x) = [f_1(x), \ldots, f_k(x)]$. The set $X \subset \mathbb{R}^n$ of all the *feasible* solutions defines the *decision space*, while the vector F maps *feasible solutions* into the *objective space*. In addition, the Pareto optimality concept is used, based on partial order relations defined as follows:

DEFINITION 1. *Let $v, w \in \mathbb{R}^k$. We say that the vector v is* less than *w ($v <_p w$), if $v_i < w_i$ for all $i \in \{1, \ldots, k\}$. The $\leq_p$ relation is defined analogously.*

DEFINITION 2 (DOMINANCE). *A point $y \in X$ is* dominated *by $x \in X$ ($x \prec y$) if $F(x) \leq_p F(y)$ and if $\exists i \in \{1, ..., k\}$ such that $f_i(x) < f_i(y)$. Otherwise y is called non-dominated by x.*

DEFINITION 3. *A point $x \in X$ is called a* Pareto *point if there is no $y \in X$ which dominates x. The set of all Pareto solutions forms the* Pareto set.

Furthermore, in order to model the dynamic behavior of the optimization problem, given t a monotonically increasing value on a time period $[t_0, t_{end}] \in \mathbb{R}^+$ we have:

$$\int_{t_0}^{t_{end}} F dt = \left(\int_{t_0}^{t_{end}} f_1 dt, \ \ldots \ , \int_{t_0}^{t_{end}} f_k dt \right) \qquad (1)$$

The following notation will be alternatively used to describe dynamic formulations, for simplicity reasons:

$$\int_{t_0}^{t_{end}} F dt = \left(\int_{t_0}^{t_{end}} f_i dt \right)_{1 \leq i \leq k} \qquad (2)$$

2. THE MULTI-OBJECTIVE LOAD BALANCING CONTEXT

2.1 Load balancing

The load balancing problem consists in mapping jobs, e.g. virtual machines, to computational resources (real machines). The simplest way of handling this problem is by using approaches like the sender-initiater, where the machine which is heavily loaded sends some of its charge to another random machine. In turn the machine that receives the request can accept or not the demand given that the extra charge may imply overshooting a certain critical charge level.

Depending on the nature of the environment two types of load balancing approaches can be identified: static and dynamic. For the static case, once an assignment solution is applied, no job can be changed, i.e. moved to a different machine that the already assigned one. For dynamic cases, we distinguish two sub-classes: *preemptive* and *non-preemptive* [5]. In the first case the jobs can change their assignment at execution time, while in the non-preemptive case, once a job is started on a machine that job must be completed on the same machine. In this latter case, provisioning occurs, meaning that new jobs can appear during the run and they need to be added to the existing resources. Dynamic load balancing can be treated using diffusion models [2] or flow models [3].

Another important aspect for determining the type of problem dealt with is related to the nature of the jobs that need to be assigned (deterministic or stochastic). For virtual machines, as the execution time depends on the end-user, one can not know in advance the total required execution time, where form the stochastic nature of the problem.

Load balancing formulations for cloud environments suppose different perspectives according to the followed goal and the available data. Classical formulations include the following:

Scheduling problems , seen as job-shop scheduling, multiprocessor scheduling, grid task scheduling [9]. Associated objectives: makespan minimization (when the jobs are deterministic) generally using evolutionary algorithms [6].

Flow problems (Multicast flows) Associated goals: flow-time (minimization of the total completion time of all tasks), minimization of the communication costs needed in transfering the jobs between resources; minimization of the communication time, minimization of the response time (the quantity of information to be exchanged, communication costs and delays).

Also other goals co-exist as maximum or average link utilization, flow assignation or packet loss.

In the current situation we adopt a load balancing perspective taking into account energy consumption derived by considering base operations. The execution time of each virtual machine is not known in advance so scheduling strategies can not be employed. Also as we are not interested in the communication related factors and therefore we did not adhere to a flow formulation. The approach we are employing consists in dynamically balancing the load of each machine. We therefore use a dynamic formulation given the dynamic characteristics of each job (execution time, amount of processor needed).

2.2 Dynamic multi-objective optimization formulation

The dynamic multi-objective optimization field is a growing area acknowledging four main types of dynamism sources. According to [7] the formulations can be classified in four classes. The load balancing problem is a 4^{th} order dynamic problem, being defined as an online dynamic multi-objective

problem of the form:

$$H(F_\sigma, D, x, t) = F_{D(\sigma,t)}(x, t)$$

For the general case and assuming a minimization context, the goal is to identify a sequence of solutions $\mathbf{x}(t)$, with $t \in [0, t^{end}]$ leading to the following:

$$\min_{\mathbf{x}(t)} \int_{t_0}^{t_{\text{end}}} F_{D(\sigma,t)}(\mathbf{x}(t), t)\ dt$$

$$\min_{\mathbf{x}(t)} \left\{ \left(\int_{t_0}^{t_{\text{end}}} f_{D(\sigma,t),i}(\mathbf{x}(t), t)\ dt \right)_{1 \le i \le k} \right\}$$

We used minimization in the formulation, as maximization is treated similarly. The objective functions we will be using are both minimization and maximization objectives.

3. PROBLEM FORMULATION

In the studied model a mixture of min and max functions is used. The σ time dependent environment parameters are given by the memory capacity of each virtual machine, the memory capacity of the used machines and the processing power needed by each virtual machine to be processed in any of the real machines. The dynamicity is given by the fact that the virtual machines being represented by users change in time. Each virtual machine has a life time given by the login moment (t_0) and an ending moment given by the moment when the user closes the session (t_{end}). As the real machines can be turned on and off the current environment machines give the memory capacity, implying that the vector of capacities changes also in time acording to the available machines. Finally, the processing power needed by a user at a given time moment strongly depends on the applications running on the virtual machine, that dynamically change with time.

- A **set of physical computational resources or machines described by power states**
  $M_t = \{m_1, m_2, \ldots, m_{\eta(t)}\}$, with $|M_t| = \alpha(t)$ the total number of power states for all the machines. For each machine we have $\{m_i\} = \{m_i^1, m_i^2, \ldots, m^{\gamma(i)}\}$, $|m_i| = \gamma(i)$, number of power states for the machine m_i, $1 \le i \le \eta(t)$. The number of available resources is assumed to dynamically change over time due to sharing policies acting across administrative domains, e.g. shared computers at institution level, stochastic factors or scheduling constraints.

  As an extension, let $\hat{M}_t$ be a set of resources defined as $\hat{M}_t = M_t \cup \{q\}$, where q represents a queue. No processing is done for the virtual machines assigned to the queue.

- A **time dependent set of virtual machines** defined as $VM_t = \{vm_1, vm_2, \ldots vm_{\beta(t)}\}$, $|VM_t| = \beta(t)$. As for the physical machines, different scenarios can be considered where the number of virtual machines varies due to incoming or finishing services, service level agreement terms or end of contractual duration.

Environment Parameters

The $D(\sigma, t)$ in our case is defined over a set σ of parameters described by $c_t(i)$, $cvm_t(j)$ and $p_t(i, j)$:

- A **maximal capacity per machine**, equal for all the states of a machine, $c_t(i), 1 \le i \le \alpha(t)$, subject to stochastic changes over time in response to external factors. A finite capacity is assumed per physical machine, e.g. memory or a different resource to balance across machines. An infinite capacity is assumed for the queue, used only to put virtual machines on hold.

- **Energy consumption per power state**, $e_t(i), 1 \le i \le \alpha(t)$, knowing that a higher computational power leads to a higher energy consumption.

- **Capacity required by a virtual machine** vm_j at a given time moment t, denoted by $cvm_t(j)$, $1 \le j \le \beta(t)$. For the herein study, the capacity required by a virtual machines, e.g. memory amount, is considered to vary over time with respect to internal functioning constraints.

- **Computational load** of the virtual machine vm_j when assigned to a physical machine for a power state m_i, here given by
  $p_t(i, j)$ with $1 \le i \le \alpha(t)$, $1 \le j \le \beta(t)$, seen in the following as the amount of processor power (percentage) required by the virtual machine vm_i on the physical machine m_j.

Variables

The problem consists in assigning the virtual machines to real machines as to balance the memory and processor usage:

- $X(t) = (x_{ij}(t))_{1 \le i \le \alpha(t)+1}$ a binary variable that, when set to 1, designates the assignment of the j^{th} virtual machine to the i^{th} state or queue (i equal to $\alpha(t)+1$);

- $Y(t) = (y_i(t))_{1 \le i \le \alpha(t)}$, a binary variable denoting if the i^{th} state is active or not. If no states are active for a given machine, the machine is considered to be switched off. Please note that a machine can be in only one state at a time.

Constraints

The assignment constraint

$$\sum_{i=1}^{\alpha(t)} \sum_{j=1}^{\beta(t)} x_{ij} y_i + \sum_{j=1}^{\beta(t)} x_{\alpha(t)+1,j} = \beta(t)$$

Also, for all the machines m_k, $k \in \{1, \ldots, \eta(t)\}$ and the corresponding states $y_i \in m_k$ the following constraints apply:

The capacity constraint $y_i \sum_{j=1}^{\beta(t)} x_{ij} cvm_t(j) \ \le \ c_t(i)$,

The processing constraint $y_i \sum_{j=1}^{\beta(t)} x_{ij} p_t(i, j) \le 1$,

The active machines constraint $\sum_{i=1}^{\alpha(t)} y_i \le \eta(t)$,

$$\sum_{i \in m_k} y_i \le 1, 1 \le k \le \eta(t).$$

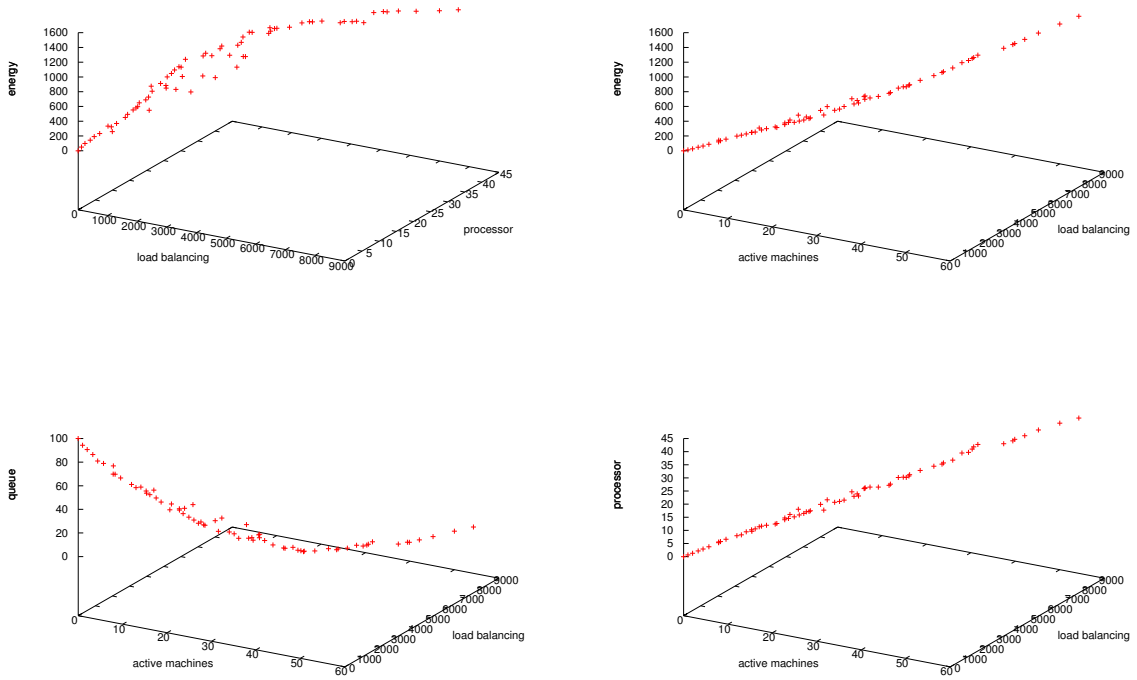

Figure 1: Preliminary results depicting different approximate Pareto fronts (different objectives) obtained with the Indicator Based Evolutionary Algorithm (IBEA) with 1000 generations per time step.

Objective functions

The chosen objective functions aim in the first place at minimizing the electricity consumption (through one explicit objective and also by minimizing the number of used machines) and at balancing the memory and computational power load. The quality of service is also addressed through provisioning by minimizing the number of virtual machines in the queue. The multi-objective formulation is fully motivated by the fact that the objectives are contradictory, e.g. by minimizing the number of active machines the workload per machine increases, or different by nature (memory vs computational load balancing).

Memory load balancing :

$$\min \sum_{k=1}^{\alpha(t)} y_k \xi_k$$

$$\xi_k = \left| \sum_{j=1}^{\beta(t)} x_{kj} cvm_t(j) - \left(1 + \sum_{i=1}^{\alpha(t)} y_i\right)^{-1} \sum_{i=1}^{\alpha(t)} y_i \sum_{j=1}^{\beta(t)} x_{ij} cvm_t(j) \right|$$

Electricity consumption minimization :

$$\min \sum_{i=1}^{\alpha(t)} e_t(i) y_i$$

Number of machines in the queue :

$$\min \sum_{j=1}^{\beta(t)} x_{\alpha(t)+1,j}$$

Load balancing of the processing power :

$$\max y_i \sum_{j=1}^{\beta(t)} x_{ij} p_t(i,j), \ \forall i \in \{1, \ldots, \alpha(t)\}$$

Number of used machines $\min \sum_{i=1}^{\alpha(t)} y_i$.

The problem is already NP-hard in its mono-objective form, just by considering the load balancing in the processing power that reduces in fact to a bin packing problem which is already NP-hard [4] and by extension its multi-objectie variants become NP-hard also.

4. EXPERIMENTATION

For the initial experimentation phases we have considered a static environment with a fixed number of physical machines and a fixed number of virtual machines having as objective functions the ones described in the above section. In order to provide a glance of the topology of the Pareto front we enclose in Figure 1 some preliminary results including approximate Pareto fronts obtained during the dynamic process be means of the Indicator Based Evolutionary Algorithm (IBEA).

A homogeneous environment is modeled in the following where all the machines are described by different power

steps, i.e. computational power vs amount of energy required. For each virtual machine the computational requirement per state is given as a percentage of the total processor's power. The objectives to optimize, as previously described, are (1) to minimize the number of machines used, e.g. all machines switched off when no virtual machines are present, (2) assure load balancing, here expressed with respect to the amount of memory used, (3) minimize the number of inactive virtual machines, i.e. placed in the queue, (4) maximize overall processor usage, and (5) reduce energy consumption. The following extreme points, used hereafter to assess the performance of the used algorithms, are of particular interest:

- *no machines used* – no energy consumption, only useful when no virtual machines are present, i.e. all machines switched off, here assuming no knowledge regarding the future state of the system in terms of arrival of v irtual machines. An algorithm exploring the different possible configurations the system can be in has to provide one solution for which no physical machines are active.

- *empty queue* – no inactive virtual machines. As all virtual machines are assigned and running, the minimal number of physical machines that can be used has to be determined while efficiently switching into the availab le power states. As a simplification, one can consider a case where no queue can be used and all the virtual machines have to be scheduled. The tradeoff to be assured in this case hence only considers the number of machines used, loa d balancing efficiency, provided computational power and energy consumption. Like for the previous point, if all machines can be scheduled, e.g. equal number of virtual and physical machines, and if all constraints can be fulfilled, a n efficient algorithm should provide a solution for which all the virtual machines are assigned to some physical machines.

- *load balancing* – given a fixed number of active physical machines, the exploration algorithm has to be able to determine corresponding power states such that (a) a maximum number of virtual machines can be executed w ith maximum performance, and (b) load balancing in terms of used/available memory is assured.

A graphical representation of different Pareto approximate fronts, for different objectives, is given in Figure 1. All the above described extremes are included in the approximated front, offering the possibility, for a real situation, to chose between maximizing performance, minimizing energy consumption or ensuring a strong load balancing.

5. CONCLUSIONS

In this paper we proposed a new load balancing model adapted to the needs of sustainable ICT in an university campus environment. The model follows the requirements of the future Belval campus, University of Luxembourg. New test scenarios were defined and preliminary results presented for the case of dynamic load balancing in heterogeneous environments. The complexity of the problem conducts us further to the development of learning strategies to be included in the dynamic framework in order to anticipate consumption needs, e.g. of particular importance when relying on renewable sources like photovoltaic cells. The results obtained so far allowed us to provide an initial overview of the compromise solutions that can be obtained in order to observe their evolution over time given environment changes.

6. REFERENCES

[1] Vmware helps worldś top universities cut costs and go green, June 2008.

[2] G. Cybenko. Dynamic load balancing for distributed memory multiprocessors. *J. Parallel Distrib. Comput.*, 7:279–301, October 1989.

[3] K. D. Devine, E. G. Boman, R. T. Heaphy, B. A. Hendrickson, J. D. Teresco, J. Faik, J. E. Flaherty, and L. G. Gervasio. New challenges in dynamic load balancing. *Applied Numerical Mathematics*, 52(2-3):133 – 152, 2005. ADAPT '03: Conference on Adaptive Methods for Partial Differential Equations and Large-Scale Computation.

[4] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman Co. New York, NY, USA, 1979.

[5] M. Harchol-balter and A. B. Downey. Exploiting process lifetime distributions for dynamic load balancing. *ACM Transactions on Computer Systems*, 15:13–24, 1996.

[6] S. N. Sivanandam and P. Visalakshi. Dynamic task scheduling with load balancing using parallel orthogonal particle swarm optimisation. *IJBIC*, 1(4):276–286, 2009.

[7] E. Tantar, A.-A. Tantar, and P. Bouvry. On dynamic multi-objective optimization - classification and performance measures. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2011)*, Juin 2011.

[8] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A break in the clouds: towards a cloud definition. *SIGCOMM Comput. Commun. Rev.*, 39:50–55, December 2008.

[9] T. Wilcox. Dynamic load balancing of virtual machines hosted on xen. Master's thesis, Brigham Young University, USA, 2009.