SESAMES: a Smart-Grid Based Framework for Consuming Less and Better in Extreme-Scale Infrastructures

Mohammed El Mehdi Diouri, Olivier Glück and Laurent Lefèvre
INRIA Avalon Team, Laboratoire de l'Informatique du Parallélisme
Ecole Normale Supérieure de Lyon, 46 Allée d'Italie, 69364 Lyon Cedex 07, France
{mehdi.diouri, olivier.gluck, laurent.lefevre}@ens-lyon.fr

***Abstract*—As they will gather hundreds of million cores, future exascale supercomputers will consume enormous amounts of energy. Besides being very important, their power consumption will be dynamic and irregular. Thus, in order to consume energy efficiently, powering such systems will require a permanent negotiation between the energy supplier and one of its major customers represented by exascale platforms. In this paper, we present SESAMES, a smart and energy-aware service-oriented architecture manager that proposes energy-efficient services for exascale applications and provides an optimized reservation scheduling. The paper focuses on the new features of this framework which are the design of a smart grid and a multi-criteria green job scheduler. Simulation results show that with the proposed multi-criteria job scheduler, we are able to save up to 2.32 % in terms of energy consumption, 24.22 % in terms of financial cost and reduce up to 7.12 % the emissions of CO_2.**

I. Introduction

A supercomputer is a system built from a collection of computers performing tasks in parallel, in order to achieve very high performance. According to the TOP 500 list [1] published in June 2013, the most powerful supercomputer is the Tianhe-2 platform, a machine with more than 3 million cores and able to perform 34 PFLOPS. Such systems support a wide range of scientific applications, including manufacturing with the design of cars and aircraft, and environment with the prediction of tsunami damage and seismic waves. However, new scientific challenges, such as modeling and simulating the complexity of life, demand more and more performant computing resources. For this reason, designing exascale systems is identified by the high performance computing (HPC) community as a real need. Indeed the IESP[2] and the EESI[3] have defined roadmaps in order to build exascale systems by the 2020 time frame. An exascale machine is a supercomputer capable of performing more than 10^{18} floating point operations per second (1 EFlop/s). The Tianhe-2 supercomputer consumes more than 17 MW for a maximum performance of 34 PFLOPS while the Defense Advanced Research Projects Agency (DARPA) has set to 20 MW, the maximum energy consumption of an exascale supercomputer [1]. Building exascale supercomputers requires to take into account the energy-efficiency of computing resources.

However, to ensure the transition to the exascale era, supercomputer designers must be able to address two main challenges that will become even more problematic for exascale systems:

- How to program applications running on these supercomputers? An exascale application will consume several megawatts and involves exabytes of data transferred. Moreover, exascale supercomputers will experience many failures per day [2]. Thus, future applications need to be run with energy-aware services[4] such as fault tolerance and collective data operations.
- How to provide them energy? Besides being very important, the power consumption of future extreme-scale systems will be irregular and dynamic. At this end, we need to be able to predict the energy consumption of the exascale supercomputer and to be able to constantly dialog with the energy provider in order to consume energy efficiently.

This paper presents SESAMES: a Smart and Energy-aware Service-oriented Architecture Manager at Extreme-Scale. In [3], we proposed some solutions to tackle the first challenge by presenting the framework components that optimize the energy consumption of the services running on an exascale system. In this paper, our goal is to extend this framework by taking into account the second challenge. To this end, SESAMES relies on a smart grid that establishes a permanent communication flow with the energy provider. Through a bidirectional negotiation with the energy provider, SESAMES schedules the reservations in a multi-criteria way by taking into consideration the energy consumption, the power capping, the financial cost and the pollution factor.

This paper is organized as follows. Section II describes previous related works. Section III presents an overview of the framework architecture. Section IV presents the design of the smart grid on which relies SESAMES. In Section V, we present how SESAMES schedules reservations of physical nodes in an energy-aware way. Section VI presents the validation results while Section VII presents the conclusion and future works.

[1] http://www.top500.org/

[2] IESP: International Exascale Software Project (http://www.exascale.org)

[3] EESI: European Exascale Software Initiative (http://www.eesi-project.eu)

[4] A service is an algorithm or a protocol that is performed to satisfy a need or to fulfill a demand (i.e. a checkpointing protocol or a broadcasting algorithm)

II. Related works

Many energy-aware scheduling algorithms have been proposed in previous works. By respecting several constraints, these algorithms aim at minimizing the energy consumption, the financial cost or the pollution impact. One of the early works about energy-aware scheduling was proposed in [4]. In this paper, Pinheiro et al. have proposed a technique for managing a cluster of physical machines with the objective of minimizing the power consumption, while providing the required QoS. The main technique to minimize power consumption is the load concentration, or unbalancing, while switching idle computing nodes off. In [5], the authors suggest to minimize the financial cost at the consumer side by scheduling applications at times when energy is less expensive. They show that we can provide significant economic savings to consumers. In another example [6], the authors proposed an algorithm that schedules jobs with the objective of maximizing the green energy consumption while respecting the jobs' deadlines. However, these previous works do not propose to optimize several objective functions together (energy consumption, financial cost and pollution impact).

III. Overview of SESAMES

In this paper, we propose several techniques and solutions in order to improve the energy efficiency in extreme-scale supercomputers. In order to enable a coordination of these techniques, we need to design an unified framework. To this end, we present a Smart and Energy-aware Service-oriented Architecture Manager at Extreme-Scale, called SESAMES.

A. Global Architecture

The goal of SESAMES is to manage the execution of extreme-scale applications on future supercomputers in an energy-efficient way. Since these large scale distributed systems are major energy consumers and their energy consumption is irregular over the time [7], SESAMES needs to establish a permanent communication with the energy provider. Such communication flows aim at optimizing the production, distribution and consumption of energy. Figure 1 presents the global architecture of the considered infrastructure. On the one hand, the energy providers adapt the supply according to the demand of supercomputers. On the other hand, the users of supercomputers will prefer to consume energy at times when it is the "cleanest" and the least expensive. By "cleanest", we mean that the energy production is green and generates a reduced quantity of CO_2 emissions (like wind and solar energies).

Besides, in order to reduce the global energy consumption, SESAMES directly acts on the supercomputer nodes. An energy sensor is plugged to each node and measures the current power consumption. SESAMES collects these energy logs.

In order to gather the execution context, SESAMES also establishes a dialog with the supercomputer users, either at the moment of reserving computing nodes or when they want to run applications and services. In Section III-B, we specify what we mean by a service and detail the different services considered in SESAMES.

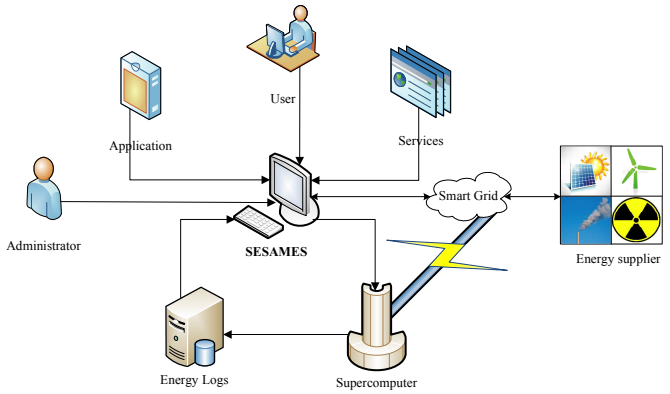

Fig. 1. SESAMES: Global Architecture

B. SESAMES Considered Services

Exascale applications need to meet with several challenges. An exascale supercomputer will typically gather from half a million to several millions of CPU cores running up to a billion of threads. From the current knowledge and observations of existing large systems, it is anticipated that exascale systems will experience various kind of faults many times per day [2]. Thus, in order to run exascale applications on extreme-scale systems, reliable fault tolerance mechanisms are mandatory. Exascale infrastructures will also face important challenges related to data processing and data exchanges. Exascale applications will involve large volumes of data: hundreds of exabytes of data are expected by 2018. To increase performance in exascale applications, significant improvements in the data collective algorithms are necessary.

In order to overcome these challenges, we should enable several services to run harmoniously together with extreme-scale scientific applications.

As concerns fault tolerance, SESAMES includes two services. On the one hand, one is dedicated to the checkpointing step that is performed during the normal functioning of the application [8]. It consists in storing a snapshot image of the current application state [9]. On the other hand, one is dedicated to the restart in case of failure. It consists in restarting the execution of the application from the last checkpoint.

As concerns data collective operations, SESAMES includes the following services: (i) scattering data over several processes, (ii) gathering data from several processes, (iii) broadcasting data to all processes [10], and (iv) all gathering (i.e. gathering data from all tasks and distributing the combined data to all tasks).

We may also need some additional services. A service for retrieving a specific data among all processes and a service for visualizing the application logs in real time. Monitoring the hardware resources that are involved in the extreme-scale system is also another important service that is required if we would like to visualize in real time the energy consumption, and to detect failures. In order to run harmoniously, the power/energy consumption issue must always be regarded as a main concern whatever the service considered.

Each service can be implemented using different protocols or algorithms. For instance, as concerns data broadcasting, two

main MPI implementations exist: MPICH2[5] and OpenMPI[6]. The algorithms used are different from one MPI implementation to another. Indeed, for a broadcast of a large volume of data between a large number of processes, OpenMPI uses a pipelining with a configurable chunk size while MPICH2 uses a *Scatter* followed by an *AllGather*. MPICH2's scatter algorithm is implemented as a binomial tree with swap by considering a chunk size equal to the total volume of data divided by the total number of processes. Another promising approach for broadcasting data is to use hybrid programming by combining MPI for inter-node communications, with OpenMP for intra-node communications [11].

One objective of SESAMES is to help users to select the most convenient version (i.e. protocol or algorithm) of the service. To this end, SESAMES some components presented in the following section.

C. Components

Designing an unified energy-aware framework is essential to enable a coordination between all the actors of the platform and all the components used to improve the energy efficiency. Figure 2 presents the main components of this framework.

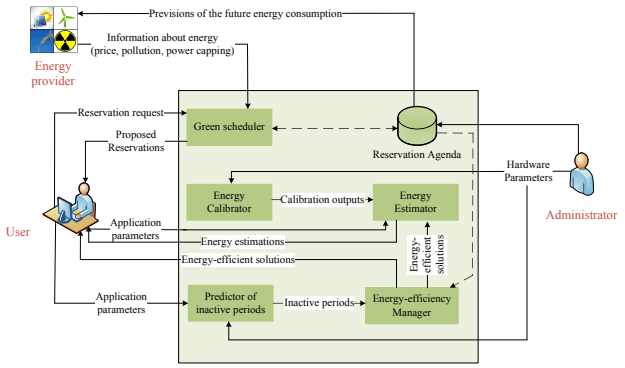

Fig. 2. SESAMES components

SESAMES includes a multi-criteria green job scheduler, which is in charge of scheduling efficiently the incoming reservation requests and allocating resources to them. The green job scheduler attempts to solve a multi-criteria optimization problem by taking into account several constraints. How can we allocate the supercomputing nodes at the moment desired by the user by consuming the least amount of energy, the cleanest energy, at the lowest financial cost and without exceeding the power capped by the energy provider? The reservation process is detailed in Section V. Once the reservation planned, the green job scheduler updates the reservation agenda in SESAMES.

Since extreme-scale supercomputers have an important and irregular energy consumption, SESAMES incorporates the ability to estimate accurately the energy consumption of the different services presented in Section III-B and mandatory in exascale applications. In order to take into account the hardware specifications, the energy estimator relies on a complete hardware calibration. At this moment, the energy estimation with SESAMES is already proposed for the checkpointing [8] and the data broadcasting [10] services.

To optimize the energy consumption of supercomputers, SESAMES includes an energy-efficiency manager that is in charge of proposing to apply some green levers at the component level: shutting down or slowing down idle resource components (CPU, RAM, HDD, etc.). The energy-efficiency manager proposes these green levers depending on the predicted inactive periods of nodes and the rights assigned by the supercomputer administrator to the user. Such green solutions are evaluated by the energy estimator in order to inform the user about the significance of the energy-efficient solutions suggested.

Since we have already presented in [3] the components that estimate and propose to reduce the energy consumption of the different services, we focus on the new features of this framework which are the design of the smart grid and the green job scheduling.

IV. Smart Grid Design

Extreme-scale supercomputers consume enormous amounts of power and are regarded as being important customers by energy suppliers. In order to establish a better communication between energy suppliers and these important customers, SESAMES relies on a smart grid.

A. From Energy Provider to SESAMES

The interaction between SESAMES and the energy provider is a double negotiation. Through permanent communication flows, SESAMES gathers from the energy provider a set of information:

- the energy financial cost agenda in order to know how expensive is the consumed energy;
- the energy pollution impact agenda in order to know how "clean" is the consumed energy;
- the fixed power capping agenda in order to know the maximum total power that the energy supplier can provide.

Indeed, energy providers usually offer price per kWh depending on the time and day when energy is consumed. For example, EDF[7], the French energy provider, shows prices in euros per kWh depending on whether it is a "blue day", a "white day" or a "red day", but also depending on whether it is an off-peak hour (from 10pm to 6am) or a full hour (from 6am to 10pm). The price per kWh on "red days and in full hours is more than seven times the price per kWh on "blue days" and in off-peak hours, which is far from being negligible!

Moreover, the energy provided may be produced from coal, natural gas, petroleum, sun, wind, etc. Then the pollution impact is a function of the resource from which the energy is provided. The maximum total power that can provide the energy supplier may vary over the time. For this reason, SESAMES should constantly get informed about the total power consumption not to exceed.

[5]http://www.mcs.anl.gov/research/projects/mpich2/
[6]http://www.open-mpi.org/
[7]EDF: http://bleuciel.edf.com

B. From SESAMES to Energy Provider

Thanks to its energy calibrator and estimator components, SESAMES can provide a set of information to the energy supplier about the power and the energy consumed by the supercomputer. In case the supercomputer is planned to experience a reduced energy consumption, SESAMES warns the energy provider, so that he can disable the production of energy produced from a polluting source. Furthermore, if the supercomputer is in need of high peaks in terms of power consumption, SESAMES asks the energy supplier to provide an extra power.

In order to predict the power consumption of the supercomputer, SESAMES needs to gather a set of power measurements on the supercomputer that calibrates our power estimations. Even if the supercomputing nodes are identical from the hardware point of view, their power consumption is different while they are idle or while they are performing the same operation [12]. We show in Figure 3 the idle P_{idle} and the maximum P_{max} power consumption of all the nodes from two different clusters of Grid5000 [13]. The idle power consumption is obtained by measuring the power consumption of each node while running nothing else than the operating system. The maximum power consumption is obtained by running a CPUBurn[8] on each core. The power consumption of a node executing an application ranges between the idle and the maximum power consumption. The first cluster is located in Lyon and incorporates 64 identical nodes. Each node gathers two CPUs with one core each, 2048 MBytes of memory capacity and 73 GBytes of storage capacity. The second cluster is located in Reims and includes 44 identical nodes. Each node gathers two CPUs with twelve cores each, 49152 MBytes of memory capacity and 250 GBytes of storage capacity. On the x axis, the nodes of each cluster are sorted in an ascending order in terms of idle power consumption.

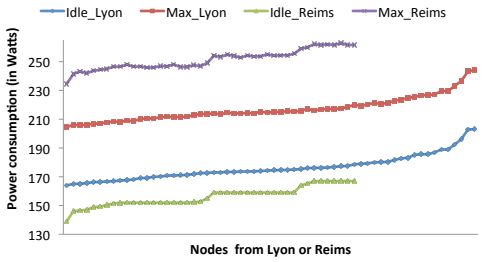

Fig. 3. Idle and Maximum Power consumptions of identical nodes from two different Grid5000 clusters (Lyon and Reims)

The difference in terms of power consumption can be explained by fluctuations caused by the external environment, such as external temperature and position of the node in the rack [14], [15]. We have also shown in [12] that this power heterogeneity mainly comes from the CPUs and/or the fans cooling the CPUs and to fluctuations in the manufacturing process. So even if the end user can have the feeling of benefiting from a completely homogeneous cluster, heterogeneity (at least in terms of energy consumption) is present.

SESAMES measures P_{idle} and P_{max} for each node. In the following section, we describe how SESAMES utilizes the calibrated idle and the maximum power consumption for all the nodes of the supercomputer.

[8]http://packages.debian.org/stable/cpuburn

V. Multi-Criteria Energy-Aware Scheduling

Thanks to the information coming from the energy provider, SESAMES schedules reservation requests of physical nodes in an energy-aware way. The green job scheduler of SESAMES maintains a reservation agenda for each node. This agenda stores all the reservations concerning the node.

In order to run their applications, users send to SESAMES a reservation request in order to book some of the supercomputer's nodes. A reservation request consists of a number n of nodes required, a reservation duration D, an earliest possible start time t_{se} and a latest possible start time t_{sl}. In order to make a reservation, the green job scheduler of SESAMES solves the following multi-objective optimization problem (P):

Determine the starting date t_0 of the reservation such as:

- C_1: t_0 is between $t_{d_{min}}$ and $t_{d_{max}}$;
- C_2: at least, N nodes are available during t_0 and $t_0 + D$;
- C_3: the power capping $P_{capping}$ fixed by the energy supplier is not reached;
- O_1: the energy $\xi(t_0, t_0 + D)$ consumed during the reservation is minimized;
- O_2: the financial cost $\zeta(t_0, t_0 + D)$ of the reservation is minimized;
- O_3: the pollution impact (CO_2 emissions) $\rho(t_0, t_0 + D)$ of the reservation is minimized.

C_1, C_2 et C_3 are the constraints of the problem P. O_1, O_2 et O_3 are the objective of the problem P.

The pollution impact $\rho(t)$ depends on the agenda of the energy resources provided by the energy supplier as it is specified in Section IV-A. It is evaluated by considering the quantity of CO_2 emitted per joule consumed. Analogously, the financial cost $\zeta(t)$ depends on the financial cost agenda of the energy provided by the energy supplier. Is is evaluated by considering the price per joule.

We denote by $P_{MAX}^{N_{res}}(t, t + D)$ the maximum power consumed by the N_{res} nodes reserved between t and $t + D$. To evaluate $P_{MAX}^{N_{res}}(t, t + D)$, SESAMES measures P_{max}^i, the maximum power consumption for each node i (as explained in Section IV-B). $P_{MAX}^{N_{res}}(t, t + D)$ is equal to $max_{betweentett+D}(\sum_{i=1}^{N_{res}} P_{max}^i)$. Therefore, the power consumption of a node executing a given application is between P_{idle}^i and P_{max}^i. P_{idle}^i is obtained by measuring the power consumption of each node i while it is executing nothing (except the OS). We consider that for a node i, the mean power consumption during D is $\alpha_i P_{max}^i$, where α_i is a coefficient between $\frac{P_{idle}^i}{P_{max}^i}$ and 1.

For the computation of the different objective functions $\xi(t, t + D)$, $\zeta(t, t + D)$ and $\rho(t, t + D)$, we also consider only the N least consuming nodes between t and $t + D$. $\xi(t, t + D)$, $\zeta(t, t + D)$ and $\rho(t, t + D)$ are such as:

$$\xi(t,t+D) = \sum_{i=1}^{N}(\alpha_i P^i_{max} D) = D \cdot \frac{\sum_{i=1}^{N}(P^i_{max})}{N} \cdot \sum_{i=1}^{N}(\alpha_i)$$

$$\zeta(t,t+D) = \xi(t,t+D) \times \frac{\int_t^{t+D} \zeta(t)\,\mathrm{d}t}{D}$$

$$\rho(t,t+D) = \xi(t,t+D) \times \frac{\int_t^{t+D} \rho(t)\,\mathrm{d}t}{D}$$

We obtain the first equation by using the fundamental property of the barycenter in the system $\{(P^i_{max}, \alpha_i), i \in [1,n]\}$. $\sum_{i=1}^{n}(\alpha_i)$ depends only on the application that will be running during the reservation. The problem (P) is NP complex. We solve it using Algorithm 1.

The dialog between the user and the green job scheduler of SESAMES is presented in Figure 4. Once the reservation request is expressed by the user, SESAMES informs the user either:

- that the requested reservation is not possible; in this case, the user can try another reservation.
- that there is a starting time t_0 which simultaneously minimizes $\xi(t_0, t_0+D)$, $\zeta(t_0, t_0+D)$ and $\rho(t_0, t_0+D)$. In this case, the user either confirms or not the optimal reservation proposed.
- that there are six different starting times ($t_{\xi\zeta\rho}$, $t_{\xi\rho\zeta}$, $t_{\rho\xi\zeta}$, $t_{\rho\zeta\xi}$, $t_{\zeta\rho\xi}$ ou $t_{\zeta\xi\rho}$) that minimize ξ, ζ et ρ by giving different weights to these three criteria. Then by considering the corresponding values of ξ, ζ and ρ, the user can select one of these six starting dates. This means that SESAMES asks either the user to choose a ranking of three criteria: energy consumption, financial cost, pollution of environment.

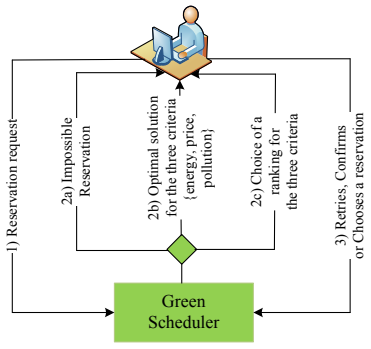

Fig. 4. Communications User/SESAMES during a user reservation

VI. Evaluation of the Multi-Criteria Energy-Aware Scheduler

This section presents the simulation environment and the evaluation results of the multi-criteria green job scheduler. The goal is to compare our scheduling approach (with a given ranking of the three criteria) to the scheduling approach that consists of starting the user reservation **as soon as possible**.

Algorithm 1 Resolution of the multi-criteria optimization problem (P)

$t_{tmp}[]$: array of integers; nb, t : integers ;
$nb \leftarrow 0$;
*/*** Determining the possible starting dates t, those that satisfy C_1, C_2 et C_3 ***/*
for $t \in [t_{d_{min}} .. t_{d_{max}}]$ **do**
  if [(at least N nodes are available during $[t, t+D]$) AND (the N nodes that have the least energy consumption are such as : $P^{N_{res}+N}_{MAX}(t, t+D) < P_{capping}$)] **then**
    nb_{++} ;
    $t_{tmp}[nb] \leftarrow t$;
  end if
end for
if (nb == 0) **then**
  Inform user that the reservation is not possible due to a lack of nodes or to the power capping.
  return -1
else
  $\xi[], \zeta[], \rho[]$: tableaux de dcimaux ;
  */*** Computation of the differents objective functions for all the possible starting dates t ***/*
  for $i \in [1..nb]$ **do**
    $\xi[i] \leftarrow \xi(t_{tmp}[i], t_{tmp}[i]+D)$;
    $\zeta[i] \leftarrow \zeta(t_{tmp}[i], t_{tmp}[i]+D)$;
    $\rho[i] \leftarrow \rho(t_{tmp}[i], t_{tmp}[i]+D)$;
  end for
end if
x, y, z : integers ;
$Min\xi, Min\zeta, Min\rho$: $floats$; */* Minimum values for each function */*
$TMin\xi[], TMin\zeta[], TMin\rho[]$: array of integers;
/ Get all the t_{tmp} minimizing each objective function */*
/ Definition of Get_TMin() in Algorithm 2 */*
$(Min\xi, TMin\xi, x) \leftarrow Get\_TMin(\xi, t_{tmp}, nb)$;
$(Min\zeta, TMin\zeta, y) \leftarrow Get\_TMin(\zeta, t_{tmp}, nb)$;
$(Min\rho, TMin\rho, z) \leftarrow Get\_TMin(\rho, t_{tmp}, nb)$;

/ If it exists, return the value of the smallest $t_{tmp}[i]$ minimizing O_1, O_2 et O_3 */*
for $i \in [1..nb]$ **do**
  if [$(t_{tmp}[i] \in TMin\xi)$ et $(t_{tmp}[i]) \in TMin\zeta)$ et $(t_{tmp}[i]) \in TMin\rho)$] **then return** $t_{tmp}[i]$, $Min\xi$, $Min\zeta$ et $Min\rho$;
  end if
end for
/ Else : provide all the combinations depending on the 6 possible rankings of the 3 criteria*/*
$t_{\xi\zeta\rho}, t_{\xi\rho\zeta}, t_{\zeta\xi\rho}, t_{\zeta\rho\xi}, t_{\rho\xi\zeta}, t_{\rho\zeta\xi}$: integers ;
/ Minimum values for the energy consumption */*
$Min\rho\xi, Min\zeta\xi, Min\rho\zeta\xi, Min\zeta\rho\xi$: floats
/ Minimum values for the financial cost */*
$Min\xi\zeta, Min\rho\zeta, Min\xi\rho\zeta, Min\rho\xi\zeta$: floats ;
/ Minimum values for the pollution of environnement */*
$Min\xi\rho, Min\zeta\rho, Min\xi\zeta\rho, Min\zeta\xi\rho$: floats ;
/ 1 energy, 2 price, 3 pollution */*
/ Find the smallest $TMin\xi$ minimizing ζ then ρ */*
/ Definition of Find_T_MinALL() in Algorithm 3 */*
$(t_{\xi\zeta\rho}, Min\xi, Min\xi\zeta, Min\xi\zeta\rho) \leftarrow Find\_T\_ALL(\xi, \zeta, \rho, TMin\xi, x)$;
/ 1 energy, 2 pollution, 3 price */*
/ Find the smallest $TMin\xi$ minimizing ρ then ζ */*
$(t_{\xi\rho\zeta}, Min\xi, Min\xi\rho, Min\xi\rho\zeta) \leftarrow Find\_T\_ALL(\xi, \rho, \zeta, TMin\xi, x)$;
/ 1 price, 2 energy, 3 pollution */*
/ Find the smallest $TMin\zeta$ minimizing ξ then ρ */*
$(t_{\zeta\xi\rho}, Min\zeta, Min\zeta\xi, Min\zeta\xi\rho) \leftarrow Find\_T\_ALL(\zeta, \xi, \rho, TMin\zeta, y)$;
/ 1 price, 2 pollution, 3 energy */*
/ Find the smallest $TMin\zeta$ minimizing ρ then ξ */*
$(t_{\zeta\rho\xi}, Min\zeta, Min\zeta\rho, Min\zeta\rho\xi) \leftarrow Find\_T\_ALL(\zeta, \rho, \xi, TMin\zeta, y)$;
/ 1 pollution, 2 energy, 3 price */*
/ Find the smallest $TMin\rho$ minimizing ξ then ζ */*
$(t_{\rho\xi\zeta}, Min\rho, Min\rho\xi, Min\rho\xi\zeta) \leftarrow Find\_T\_ALL(\rho, \xi, \zeta, TMin\rho, z)$;
/ 1 pollution, 2 price, 3 energy */*
/ Find the smallest $TMin\rho$ minimizing ζ then ξ */*
$(t_{\rho\zeta\xi}, Min\rho, Min\rho\zeta, Min\rho\zeta\xi) \leftarrow Find\_T\_ALL(\rho, \zeta, \xi, TMin\rho, z)$;
return $t_{\xi\zeta\rho}$, $Min\xi$, $Min\xi\zeta$ and $Min\xi\zeta\rho$; $t_{\xi\rho\zeta}$, $Min\xi$, $Min\xi\rho$ and $Min\xi\rho\zeta$; $t_{\zeta\xi\rho}$, $Min\zeta$, $Min\zeta\xi$ and $Min\zeta\xi\rho$; $t_{\zeta\rho\xi}$, $Min\zeta$, $Min\zeta\rho$ and $Min\zeta\rho\xi$; $t_{\rho\xi\zeta}$, $Min\rho$, $Min\rho\xi$ and $Min\rho\xi\zeta$; $t_{\rho\zeta\xi}$, $Min\rho$, $Min\rho\zeta$ and $Min\rho\zeta\xi$;

Algorithm 2 Definition of Get_TMin

Get all the values of t minimizing the function F
function GET_TMIN(F {array of floats}, t {array of integers}, size {size of the array t})
$MinF \leftarrow \infty$: float ; *minimum value of F*
$m : integer$; */* number of t minimizing F */*
$TMinF[]$: array of integers ; */* array of t minimizing F */*
for $i \in [1..size]$ **do**
if $F[t[i]] <= MinF$ **then**
if $F[t[i]] < MinF$ **then**
$m \leftarrow 0$; */* new minimum */*
$MinF \leftarrow F[t[i]]$; */* new minimum */*
end if
m_{++}
$TMinF[m] \leftarrow t[i]$;
end if
end for
return (MinF, TMinF[ ], m) ;
end function

Algorithm 3 Definition of $Find\_T\_ALL$

/ Find the smallest value of $TMinF$ minimizing G then H */*
/ To this end, we look for all theTMinF qui minimizing G. Among the remaining times, we look for those that minimize H */*
function FIND_T_ALL(F, G, H {array of floats}, TMinF {array of integers minimizing F}, $size_F$ {size of the array}))
$TMin_{FG}[]$: array of integers ; */* array of $TMinF$ minimizing F then G */*
$TMin_{FGH}[]$: array of integers ; */* array of $TMinF$ minimizing F, G then H */*
$size_{FG}, size_{FGH}$: integers ; */* size of the arrays $TMin_{FG}[]$ and $TMin_{FGH}[]$ */*
Min_{FG} : float ; */* minimum value of G after minimizing F */*
Min_{FGH} : float ; */* minimum of H after minimizing F then G */*
$t_{FGH} : integer$; */* minimum of t after minimizing F, G then H */*
/ Choose among $TMin_F$ all the values minimizing G */*
$(Min_{FG}, TMin_{FG}, size_{FG}) \leftarrow Get\_TMin(G, TMin_F, size_F)$;
/ Choose among $TMin_{FG}$ all the values minimizing H */*
$(Min_{FGH}, TMin_{FGH}, size_{FGH}) \leftarrow Get\_TMin(H, TMin_{FG}, size_{FG})$;
/ Choose among $TMin_{FGH}$ the smallest values (earliest times) */*
$t_{FGH} \leftarrow min(TMin_{FGH})$;
return t_{FGH}, Min_F, Min_{FG} et Min_{FGH} ;
end function

A. Simulation Environment

In order to evaluate the benefits of the multi-criteria job scheduler, we simulated it in Matlab in order to compare it to the approach that consists of scheduling the user reservation as soon as possible.

The simulation is considered over 30 days. The time is discretized in equal time slots. Each time slot lasts ΔT minutes. In our simulations, ΔT is equal to 15 minutes. We remind that the mean power consumption of a node i during the reservation is equal to $\alpha_i P^i_{max}$. For each node i, α_i is generated randomly between $\frac{P^i_{idle}}{P^i_{max}}$ et 1.

This simulator takes as inputs:

- a list of the idle P_{idle} and the peak P_{max} power consumptions of the N_{tot} supercomputing nodes. N_{Tot} is the total number of the supercomputing nodes. P_{idle} et P_{max} of each node are obtained from the power consumption measured on 44 nodes of the Reimscluster of Grid'5000 platform. To reach the exascale, our simulations need to use a number of nodes with an order of magnitude equal to 10^6. To obtain realistic values at the exascale, we considered that each node has an energy efficiency of 50 GFLOPS/W (1 EFLOPS for 20 MW as requested by the DARPA [1]). As a consequence, we have considered $1 \cdot 10^6$ nodes, that the power consumption of a supercomputing node is 1/10th the power consumption of a node from the Reims cluster, and that each simulated node is able to reach 5 TFLOPS. This enables to simulate a supercomputer able to reach 5 EFLOPS.
- an agenda of the energy financial cost. The agenda that we considered is the one of the real electricity prices from EDF (the energy provider in France) during January 2013[9]. During this month, the electricity price varied from 0.0212×10^{-6} to 0.1422×10^{-6} euros per joule.
- an agenda of the pollution impact. The agenda that we considered is the one of the real emissions of CO_2 in France during January 2013[10]. During this month, the emissions of CO_2 varied between 7.5 to 31.1 μg per joule.
- an agenda of the fixed power capping. For each time slot, the maximum of power that can be delivered to the supercomputing system is generated randomly between 0% to 200% of the peak instantaneous power consumption of the whole supercomputer ($\sum_{i=1}^{N_{Tot}} P^i_{max}$). These values are generated according to a normal probability distribution with a mean equal to 100 and a variance equal to 25.

Moreover, the simulator takes as input a flow of R reservation requests. Each reservation request r is generated randomly and is composed of:

- a number of nodes n^r generated randomly between 1 and N nodes.
- a duration D^r generated between 15 minutes and a maximal duration corresponding to 24 hours.
- an early starting time $t_{d_{min}}$ generated randomly between the first time slot over the whole simulation until the last possible one.
- a late starting time $t_{d_{max}}$ considered equal to $t_{d_{min}} + T^r$ where T^r is the maximum starting time delay for this reservation.

After each reservation request, the simulator maintains and updates three agendas:

- an agenda of the placed reservations using the scheduling approach that consists of starting the reservation **as soon as possible** in the interval $[t^r_{d_{min}}; t^r_{d_{max}}]$ and which consists of taking the first n^r available nodes.
- an agenda of the placed reservations using our scheduling algorithm with a user that ranks the criteria as follows: financial cost, pollution impact, energy consumption.

[9] http://particuliers.edf.com/gestion-de-mon-contrat/options-tempo-et-ejp/option-tempo/l-historique-52426.html
[10] http://www.rte-france.com/fr

- an agenda of the placed reservations using our scheduling algorithm with a user that ranks the criteria as follows: pollution impact, financial cost, energy consumption.

In order to compare the three scheduling policies, we collect from each reservation agenda:

- the mean energy consumed during a reservation;
- the mean financial cost of a reservation;
- the mean quantity of CO_2 emitted per reservation;
- the mean starting time lag by taking as reference the early starting time per reservation;
- the number of unsuccessful reservations (due to the unavailability of nodes or to the power capping fixed by the energy supplier).
- the mean occupancy rate of the platform.

For the energy, the financial cost, the quantity of CO_2, the starting time lag, the mean values are calculated over the scheduled reservations; The mean occupancy rate of the platform is computed over the whole reservation agenda. We run this simulation 100 times and we computed the mean values that we obtain on average over the 100 simulations.

B. Evaluation Results

Figure 5 presents the results of simulations. On the top left is represented the mean energy consumed over the scheduled reservations. On the top right is represented the mean financial cost over the scheduled reservations. On the bottom left is represented the mean CO_2 emissions over the scheduled reservations. On the bottom right is represented the mean starting delay over the scheduled reservations.

First, we notice through Figure 5 that the multi-criteria scheduling approach (with a given ranking of the three criteria) consumes less energy, costs less money and generates less CO_2 emissions. Indeed, if we compare the two versions of the multi-criteria scheduling approach to the "as soon as possible" approach, we obtain the following relative savings after 100 scheduled reservations:

	SESAMES (price, pollution, energy)	SESAMES (pollution, price, energy)
Saved energy	2.32 %	0.7 %
Money saved	24.22 %	18.23 %
CO_2 emitted	6.98 %	7.12 %

The mean values presented in Figure 5 seem to be high. On average, a user requests half of the platform (5×10^5 nodes) during a mean reservation duration (half of a day, i.e. 12×3600 = 43200 s). On average, a node consumes 17.5 W ($0.75 \times 25W$). Therefore, the mean energy consumed per reservation is $(5 \times 10^5 nodes) \times 17.5 \times 43200s$. Hence, its order of magnitude is 3×10^{11} Joules (same order of magnitude as the top left figure). On average, the electricity price is equal to 5×10^{-8} euros per Joules. Therefore, the order of magnitude of the mean financial cost per reservation is $(3 \times 10^{11}) \times (5 \times 10^{-8})$ which is equal to 1.5×10^4 (same order of magnitude as the top right figure). Similarly if we multiply the mean energy consumed per reservation by the quantity of CO_2 emitted per joule, we obtain the same order of magnitude as we obtain for the mean CO_2 emitted per reservation.

As the number of scheduled reservations grows, we notice that both the mean financial cost (top right) and the mean CO_2 emissions (bottom left) tend to increase for the two versions of the multi criteria scheduling approach. Indeed, as the number of scheduled reservations grows, the time slots with the lowest price and the lowest pollution impact become more and more unavailable. Moreover, for the the two versions of the multi-criteria scheduling approach, we notice that the mean starting time delay stabilizes at 30 minutes per reservation.

Furthermore, for the "as soon as possible" scheduling approach, we notice that mean starting delay per reservation increases with a growing number of scheduling reservations. Indeed, since the "as soon as possible" reservation agenda is getting filled up, it is becomes more and more difficult to schedule reservations with a starting time that is not delayed. We also notice that the energy consumed for the three scheduling schemes tend to decrease with a growing number of scheduling reservations. Since the reservation agendas are getting filled up, it becomes more and more difficult to schedule reservations with a high number of nodes. Hence, as the number of nodes per reservation decreases, the mean energy consumed per reservation decreases.

Over 100 simulations of 100 reservation requests each, the platform occupancy rate and the ratio of non scheduled reservations are as follows:

	Occupancy rate	Non scheduled reservations due to the number of nodes	Non scheduled reservations due to $P_{capping}$
"As soon as possible"	52.59 %	7.56 %	3.52 %
SESAMES (price, pollution, energy)	51.60 %	7.12 %	4.02 %
SESAMES (pollution, price, energy)	52.29 %	8.16 %	3.41 %

For the three scheduling schemes, we notice more than half of the platform is booked after the 100 reservation requests. The two versions of the scheduling approach proposed in SESAMES generate non scheduled reservation rates very close to those of the "as soon as possible" approach. We observe that a non negligible part of the scheduled reservations is due to the power capping. Therefore, it is important to take into consideration this power limitation.

VII. Conclusions

In this paper, we address the issue of power and energy management in exascale supercomputers. To this end, we propose a smart and energy-aware service-oriented architecture manager at extreme-scale called SESAMES. In order to "consume less energy", SESAMES allocates the supercomputing nodes at the moment desired by the user by consuming the least amount of energy without exceeding the power capped by the energy provider. In order to "consume a better energy", SESAMES optimizes the energy financial cost and its pollution impact on the environment. To this end, this framework involves external interactions with the user and with the energy supplier through a smart grid. SESAMES

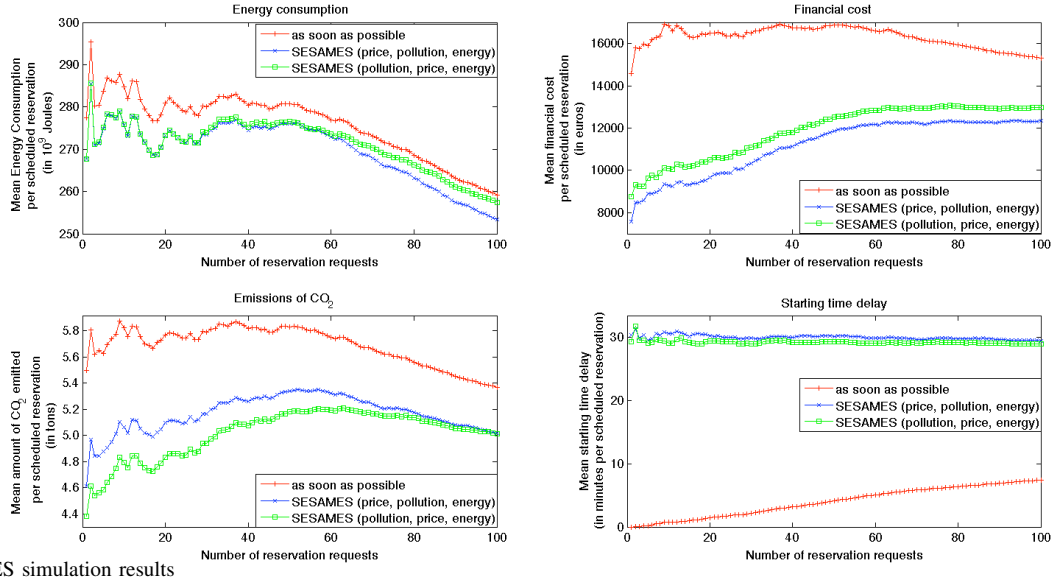

Fig. 5. SESAMES simulation results

proposes to "consume better" and to "consume less" energy by scheduling the user reservations by optimizing in a multi-criteria way the energy consumption, its financial cost and its pollution impact.

Compared to the "as soon as possible" approach, simulation results show that thanks to the multi-criteria job scheduler in SESAMES, we save up to 2.32 % in terms of energy consumption, up to 24.22 % in terms of financial cost and reduce up to 7.12 % the emissions of CO_2. Moreover, we take into consideration the power capping, i.e. the maximal power that the energy supplier can provide. The reservation starting time delay generated by our multi-criteria green job scheduler is on average 4.20 % of the mean reservation duration (30 minutes out of 12 hours). In our future work, we plan to enrich the knowledge base of SESAMES in order to incorporate additional services.

ACKNOWLEDGMENT

Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

REFERENCES

[1] P. M. Kogge and et al, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," in *DARPA Information Processing Techniques Office*, Washington, DC, September 28 2008, p. pp. 278.

[2] F. Cappello, A. Geist, B. Gropp, S. Kale, B. Kramer, and M. Snir, "Toward exascale resilience," *International Journal of High Performance Computing Applications*, vol. 23, pp. 374–388, November 2009.

[3] M.E.M. Diouri, O. Glück, L. Lefèvre, "Towards a novel smart and energy-aware service-oriented manager for extreme-scale applications," in 1^{st} *Workshop on Power-Friendly Grid Computing, co-located with the 2012 International Green Computing Conference, IGCC 2012, San Jose, CA, USA, June 4-8, 2012*, 2012, pp. 1–6.

[4] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, "Load balancing and unbalancing for power and performance in cluster-based systems," in *Workshop On Compilers and Operating Systems for Low Power*, 2001.

[5] T. T. Kim and H. V. Poor, "Scheduling power consumption with price uncertainty," *IEEE Trans. Smart Grid*, vol. 2, no. 3, pp. 519–527, 2011.

[6] I. Goiri, R. Beauchea, K. Le, T. D. Nguyen, M. E. Haque, J. Guitart, J. Torres, and R. Bianchini, "Greenslot: scheduling energy consumption in green datacenters," in *Conference on High Performance Computing Networking, Storage and Analysis, SC 2011, Seattle, WA, USA, November 12-18, 2011*. ACM, 2011, p. 20.

[7] A.-C. Orgerie, L. Lefèvre, and J.-P. Gelas, "Chasing gaps between bursts: Towards energy efficient large scale experimental grids," in *Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies, PDCAT 2008, Dunedin, Otago, New Zealand, 1-4 December 2008*, 2008, pp. 381–389.

[8] M. E. M. Diouri, O. Glück, L. Lefèvre, and F. Cappello, "ECOFIT: A Framework to Estimate Energy Consumption of Fault Tolerance protocols during HPC executions," in *13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, Delft, Netherlands, May 2013.

[9] K. M. Chandy and L. Lamport, "Distributed snapshots: Determining global states of distributed systems," *ACM Trans. Comput. Syst.*, vol. 3, no. 1, pp. 63–75, 1985.

[10] M.E.M. Diouri, O. Glück, L. Lefèvre, J.-C. Mignot, "Energy Estimation for MPI Broadcasting Algorithms in Large Scale HPC Systems," in 20^{th} *European MPI Users' Group Meeting on Recent Advances in Message Passing Interface (EuroMPI 2013), Madrid, Spain*, Sep. 2013.

[11] R. Rabenseifner, G. Hager, and G. Jost, "Hybrid mpi/openmp parallel programming on clusters of multi-core smp nodes," in *Parallel, Distributed and Network-based Processing, 2009 17th Euromicro International Conference on*, feb. 2009, pp. 427 –436.

[12] M.E.M. Diouri, O. Glück, L. Lefèvre, "Your Cluster is not Power Homogeneous: Take Care when Designing Green Schedulers!" in *IGCC2013 : International Green Computing Conference, Alington, VA USA*, Jun. 2013.

[13] F. Cappello et al, "Grid'5000: A large scale, reconfigurable, controlable and monitorable grid platform," in *6th IEEE/ACM International Workshop on Grid Computing*, Seattle, Washington, USA, Nov. 2005.

[14] G. Varsamopoulos, A. Banerjee, and S. K. S. Gupta, "Energy efficiency of thermal-aware job scheduling algorithms under various cooling models," in *Second International Conference on Contemporary Computing, IC3 2009, Noida, India, August 17-19, 2009*, ser. Communications in Computer and Information Science, vol. 40. Springer, pp. 568–580.

[15] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Thermal-aware task scheduling for data centers through minimizing heat recirculation," in *IEEE International Conference on Cluster Computing, 17-20 September 2007, Austin, Texas, USA (CLUSTER 2007)*, pp. 129–138.