Scaling Energy Adaptive Applications for Sustainable Profitability

Fabien Hermenier[1,2], Giuliani Giovanni[3], Andre Milani[3], and Sophie Demassey[4]

[1] Université Côte d'Azur, CNRS, I3S, France fabien.hermenier@unice.fr
[2] Nutanix inc. fabien.hermenier@nutanix.com
[3] Hewlett Packard Enterprise, {giuliani, andrea.milani}@hpe.com
[4] Centre for Applied Mathematics – MINES ParisTech, PSL
sophie.demassey@mines-paristech.fr

Abstract. Energy efficiency in data centres is addressed through workload management usually to reduce the operational costs and as a by-product, the environmental footprint. This includes to minimise total power consumption or to minimise the power issued from non-renewable energy sources. Hence, the performance requirements of the client's applications are either totally overlooked or strictly enforced.

To encourage profitable sustainability in data centres, we consider the total financial gain as a trade-off between energy efficiency and client satisfaction. We propose Carver to orchestrate energy-adaptive applications, according to performance and environmental preferences and given forecasts of the renewable energy production. We validated Carver by simulating a testbed powered by the grid and a photovoltaic array and running the Web service HP LIFE.

1 Introduction

Energy efficiency in data centre is a major topic over the last decade. The earliest approaches focused on spatial optimisation using workload consolidation. The use of renewable energies led to temporal optimisation where the workload is shifted to periods of green and cheap energy [1–5]. In parallel, applications moved from monolithic to elastic then to *energy-adaptive* [6–8] designs to align their performance with explicit energy concerns. Finally, smart city energy authorities pressure data centre managers to reduce their environmental footprint, conflicting with the data center client's performance expectations. While clients and energy authorities have competing objectives, they both interact with the data centre through financial agreements and their associated penalties. Energy-adaptive applications should then be scaled according to economic concerns when looking either for service or sustainability or both.

In this paper, we present Carver, a tool to reach profitable sustainability for data centre managers inside a smart city. Carver orchestrates energy-adaptive applications on a 24-hour horizon, based on performance level, the variable availability of renewable energies, and the variable price of multiple power sources. It adjusts the application's working modes to minimise a financial cost combining the power cost and penalties for not complying with the client's performance and the smart city authority's expectations in terms of renewable energy use.

We validated Carver benefits by simulating a real testbed powered by the grid and a photovoltaic array and running the application HP LIFE. Carver reduced running costs by 34.71% compared to the current scenario, which runs the applications at peak performance. In practice, it reduced incomes by 2.17% to increase the use of renewables by 3.48%. Carver also reduced running costs by 52.40% compared to a scenario that maximises the usage of renewable energies, by decreasing the use of renewables by 1.61% to increase incomes by 52.35%.

The rest of this paper is organised as follows: Section 2 presents Carver's architecture. Section 3 details the implementation. Section 4 evaluates our prototype. Section 5 describes related works and Section 6 presents our conclusions.

2 CARVER overview

Carver orchestrates energy-adaptive applications for profitable sustainability. At regular intervals, it analyses the forecasts of the energy providers powering the data centre, the application's characteristics and the smart city authority's expectations in terms of sustainable development, to compute an economically viable way to run the applications. In this section, we introduce Carver's architecture and detail the supported application's characteristics. We finally present Carver's general behaviour.

2.1 Architecture

Carver is a Web service written in Java. It interacts with energy providers, a smart city authority, and energy-adaptive applications using Rest APIs.

The energy providers characterise the power sources that are connected to the data centre. This includes commercial energy providers, through power distribution grids, but also a local production using a photovoltaic array for example. Each energy provider comes up with forecasts on a typical two-day horizon of the variable energy pricing and availability, as well as the proportion that comes from renewable sources. Pure renewable sources such as photovoltaic arrays display a constant 100% proportion while the power grid usually exhibits a variable ratio over time. Forecasting issues are outside the scope of this paper. We consider that the data supplier possesses its own algorithms or existing ones [9].

The smart city authority draws up agreements to regulate the energy used by consumers inside a smart city. An agreement declares a sustainable objective to reach and the financial penalties to pay in case of failure. By default, Carver supports agreements stating a minimal percentage of renewable energy to consume. It is however extensible enough to support other kinds of energy-related agreements.

The energy-adaptive applications form the auto-scalable workload of the data centre. Each application can run under different *working modes* that vary in performance and power consumption. Carver interacts with the applications through an *Energy Adaptive Software Controller* (EASC) connected to each of them. This controller enables developers to make their application adaptive to variable energy availability and sets out the application's characteristics in a textual description. [7]

2.2 The EASC description script

A script describes the scalability characteristics of an application and its service-level objectives.

The Service Level Objective (SLO) is a business objective to reach over a validity period, and a pricing model. Carver currently supports *cumulative* and *instant* performance models. Cumulative models mainly concern batch-oriented applications that must achieve a given amount of work within a given time frame. Instant models concern interactive applications, such as a Web service that fulfills a given amount of requests per second that vary over the day.

The pricing model defines the base price the client pays when the objective is achieved, as well as a list of *modifiers*, i.e. financial penalties refund to the client depending on the gap between the actual and target performance levels. Each price modifier specifies a threshold and a flat or linear penalty. When the objective is not reached, the penalty to apply is given by the modifier with the largest threshold that is below or equal to the performance achieved. A flat penalty value is subtracted once from the base price, while a linear value is expressed as a price per performance unit.

The Working modes correspond to the possible deployments of the application (*e.g.* by a variable number of running replicas). It is described by a performance level, a power consumption, and a shell command to use to start the reconfiguration. When the application switches to a new working mode, it may temporary undergo a performance loss that depends on the previous working mode. A matrix indicates these transition costs. We assume that establishing the working modes it is done manually during a training phase, or using automated methods. [10].

2.3 General workflow

Carver schedules the application execution on a time frame spanning from 24 to 48 hours. The time frame is discretised into time slots of 15 minutes and its duration is computed to include the complete validity period of each contract established between the data centre business service, the smart city authority and the application owners. Each contract starts at midnight and imposes a validity period of 24 hours. Accordingly, the time frame covers the ongoing day only when Carver is called at midnight; otherwise it covers the period from the present time to the end of the following day.

By default, Carver is called every 15 minutes. This means that future decisions can be revised with regards to the refinement of the forecasts made by the energy providers. Carver first computes the time frame duration, then requests the forecasts from the energy providers, the smart city authority objective, and the EASC descriptions. Using this data, Carver computes, for each time slot of the time frame, the most suitable working mode for each application in order to maximise the data centre's profits. Once the computation is done, it informs each EASC of the schedule to follow.

3 Implementation

We refer to the optimisation problem computed by Carver as the EASC Allocation Problem (EascAP). The basic EascAP with only one power source, two modes per application, no smart city objectives, no instant performance goals, and no transition costs is already NP-hard in the strong sense as it contains Bin Packing as a special case because of the limited amount of power available at each time slot. As a consequence, no simple algorithm exists to solve real instances of the EascAP. The situation naturally becomes even more complicated when the instances exhibit all of the problem facets and heterogeneous components.

Carver relies on constraint programming to solve EascAP, using the Choco solver [5]. Constraint programming (CP) is a declarative paradigm to solve combinatorial decision problems. [11] We choose CP over alternative declarative approaches, such as mathematical or logic programming, because it tends to be more efficient on allocation and scheduling problems, and even more so with composite aspects: its higher-level modelling language enables the direct encoding of a wide variety of constraints and prevents the aggregated model from becoming intractable because of its size.

Applications and SLO The execution of an application $a \in A$ is defined by its working mode $m \in M_a$ during each time slot $t \in T$. This is modelled as a sequence of decision variables $\texttt{mode}_a = (\texttt{mode}_{at})_{t \in T}$, each having an initial domain M_a. Each variable-value assignment $\texttt{mode}_{at} = m$ is associated with the amount of work done by application a in mode m, possibly lowered by the transition cost incurred by a mode switch at time t, i.e. when $\texttt{mode}_{a(t-1)} = m' \neq m$. Let $W_a(m', m)$ denote the resulting value. The total instant and cumulative performance penalty costs over the time frame T can be respectively computed as $\texttt{icost}_a = \sum_{t \in T} K_I(W_a(\texttt{mode}_{a(t-1)}, \texttt{mode}_{at}))$ and $\texttt{ccost}_a = K_C(\sum_{t \in T} W_a(\texttt{mode}_{a(t-1)}, \texttt{mode}_{at}))$ where K_I and K_C describe any penalty functions associated with the non-achievement of the instant and cumulative performance goals.

The two later relations could be encoded as such in a CP model. However, we obtain a deeper inference if they are grouped into one *multicost-regular* constraint [12]. This constraint captures the dependence of the mode transitions on the penalty values. This constraint is specified with a weighted finite automaton, over the variable sequence $\texttt{mode}_a$ and two numerical variables $\texttt{perf}_a$ and $\texttt{icost}_a$, and enforces that: (1) sequence $\texttt{mode}_a$ has no forbidden mode switches

[5] http://choco-solver.org

if any exist, (2) $\texttt{perf}_a$ is the cumulative performance of application a over time frame T, i.e. $\texttt{perf}_a = \sum_{t \in T} W_a(\texttt{mode}_{a(t-1)}, \texttt{mode}_{at})$, and (3) $\texttt{icost}_a$ is the total instant penalty cost of application a.

The instant penalty function K_I is directly modelled by this constraint, within the pre-computed automaton weights. Since the cumulative penalty cost depends on the entire sequence, function K_C must be modelled with an extra constraint relating the cumulative performance to the penalty cost variable:

$$\texttt{ccost}_a = K_C(\texttt{perf}_a) \tag{1}$$

Without any assumption on the nature of the function K_C, we model Equation (1) with the *element* constraint after calculating the penalty costs associated with every possible value of $\texttt{perf}_a$.

Power sources and consumption The data centre is powered with a dedicated set of power sources S. Each source $s \in S$ is available in a limited amount at each period $t \in T$. We introduce a variable $\texttt{psrc}_{st}$ stating the power usage and a constant k_{st} stating the unit energy price of source s at time t. Equation (2) models the total energy cost $\texttt{pcost}$ of the data centre over one day:

$$\texttt{pcost} = \sum_{t \in T} \sum_{s \in S} \texttt{psrc}_{st} \times k_{st} \tag{2}$$

The power consumption of application $a \in A$ during time slot t is modelled as a variable $\texttt{papp}_{at}$ and its value only depends on the current working mode $\texttt{mode}_{at}$. This relation is modelled using an *element* constraint. Finally, Equations (3) link the power consumed by the applications to the power provided by the sources at each time slot:

$$\forall t \in T, \sum_{a \in A} \texttt{papp}_{at} = \sum_{s \in S} \texttt{psrc}_{st} \tag{3}$$

Smart city objective The smart city authority controls the minimum rate of renewable energy consumed by the data centre. According to the energy forecasts, i.e. the percentage r_{st} of green power provided by any source s at time t, the total ratio is a variable $\texttt{rpart}$ computed as:

$$\texttt{rpart} = \frac{\sum_{t \in T} \sum_{s \in S} \texttt{psrc}_{st} \times r_{st}}{\sum_{t \in T} \sum_{s \in S} \texttt{psrc}_{st}} \tag{4}$$

The penalty cost is again modelled by an *element* constraint, whatever the applied penalty function K_R:

$$\texttt{rcost} = K_R(\texttt{rpart}) \tag{5}$$

Economic profit The daily net benefits for the data centre are given as the base revenue for running the applications minus the energy cost and the penalties incurred for violating the application performance goals and the smart city objective. The optimisation criterion is then to minimise the sum of the expenses:

$$\min \sum_{a} (\texttt{ccost} + \texttt{icost}) + \texttt{pcost} + \texttt{rcost} \tag{6}$$

4 Evaluation

The goal of Carver is to minimise the total running costs of a data centre inside a smart city so that its sustainability-oriented investments are viable. To evaluate its practical benefits in as realistic an environment as possible, we used a simulator to replay a 4-day trial inside a company data centre powered by the grid and a photovoltaic array set from 16 to 20 January 2015. We replayed the environment and the production workload with different solar profiles and compared the running cost obtained by Carver with the scaling used in production, and another one focusing on maximising the use of renewable energy.

4.1 Environment setup

The simulated environment reproduces the testbed of the HP Innovation Lab in Milan. The testbed is composed of 20 Moonshot cartridges [6] that have a power consumption in the range of 20 Watt-peaks each. Such a size in terms of node stay representative in the context of a private cloud. Indeed Cano *et al.* [13] studied more than 2,000 private cloud installations. They observed their size vary between 3 and 40 servers, with an average of 6.18 nodes per cluster.

The testbed is powered by the grid and a photovoltaic array. The grid provides energy at a price of 0.16€/kWh. The photovoltaic array produces 1 kWatt peak. Because the trial represents a negligible period compared to the array's lifespan, we ignore the investment cost for the photovoltaic array and assume an energy production cost of zero. The solar irradiation and the percentage of renewable energy in the grid vary on a daily basis. The supply of dual energy is simulated using 4 characteristic pre-defined profiles for each source, extracted from a 7-month history using the methodology in [14].

The testbed runs the HP LIFE project, a highly available Web-based e-learning platform spread over the globe. The platform is composed of 3 energy-adaptive applications whose characteristics are summarised in Table 1. The *Website* application has 6 working modes and uses from 10 to 17 cartridges. The unused ones are turned off to save power. The 10 cartridges that are always running host the database replicas and the load balancers to ensure high availability. This results in high consumption in the lowest working mode relative to its low performance. The *E-learning* and *G-learning* applications are responsible for indexing the content of the *Website* application. Each uses 1 cartridge,

[6] `http://www8.hp.com/us/en/products/servers/moonshot/`

has 3 working modes, and a number of concurrent crawlers that increases with the working mode performance. The applications are deployed statically on the cartridges without any virtualisation layer or co-location. The trial workload is based on the scenarios used by the operation team of the HP LIFE project. For the *Website*, we analysed the production traces to identify the day on which the highest request rate was observed. This load was multiplied by a correction factor to cope with the higher computing power of the Moonshots compared to the production cluster. The workload for *E-learning* and *G-learning* consists in crawling 200,000 then 300,000 pages.

Application	Performance	Power (W)	Working Modes	Cartridges
Website	1050 – 3250 Req/s	360 – 550	6	10-17
G-learning	0 – 565 kPages/h	6 – 33	3	1
E-learning	0 – 60 kPages/h	6 – 33	3	1

Table 1: HP LIFE project characteristics. The *Website* application is the 3-tier Web service that delivers static and dynamic contents. The *E-learning* and *G-learning* applications are responsible for indexing *Website* content.

The *Website* SLO requires the first working mode that can absorb the average hourly load, and uses a piecewise linear penalty function with a penalty of up to 2 € when the request rate is below the expectations by at least 800 requests per second. Inside the simulator, we consider that the request rate is constant per time slot, and equal to its SLO. Accordingly, the simulated behaviour slightly exaggerates the power consumption. This simplification does not impact the evaluation as the simulated conditions are constant for the whole experiment.

The SLOs for the *G-learning* and the *E-learning* require to crawl all of the *Website* pages, their penalty functions being linear with the number of missing indexations. For *G-learning* and *E-learning*, they equal 0.2 € and 0.01 € per 1000 missing indexations, respectively. For each time slot, these applications always run at peak performance within the limits of their working mode. Inside the simulator we then assume a constant power consumption and performance.

Finally, we stated that the smart city authority's objective requires that the data centre use at least 65% of renewable energy every day, with a penalty of 100 € per missing percent point. These values were chosen to evaluate Carver in situations where it is not always possible to attain the expected threshold, and with a penalty high enough to evaluate Carver trade-off decisions.

We evaluated the simulator accuracy in [14] and reported the amount of renewable energy used in reality and in the simulator deviated by less than a percent point. We ran the evaluation on a MacBook Pro with 3.1 GHz Intel Core i7 and 16 GB RAM and gave Carver 15 seconds to compute the best solution possible. As Carver is called every 15 minutes by default, it is possible to increase the time limit. However, we observed that usually Carver reaches a local optimum in less than 15 seconds.

We compared Carver with two representative scenarios named perf and green that were also executed using the simulator to avoid any comparison bias. perf mimics the scaling used in production. The working mode used for the Website is the one that ensures the SLO while using the least power. The *G-learning* and *E-learning* applications are launched at 00:00 and 03:00, respectively; a common time for background jobs. Each runs at peak performance until it reaches its SLO. In the green scenario, the applications are scaled to use as much renewable energy as possible. For this scenario, we ran Carver with a heuristic that selects the most efficient working mode when the photovoltaic array produces enough energy to power the entire testbed, and the least efficient working mode otherwise. Furthermore, each time a solution is computed, Carver must compute a new solution that provides a better renewable energy percentage.

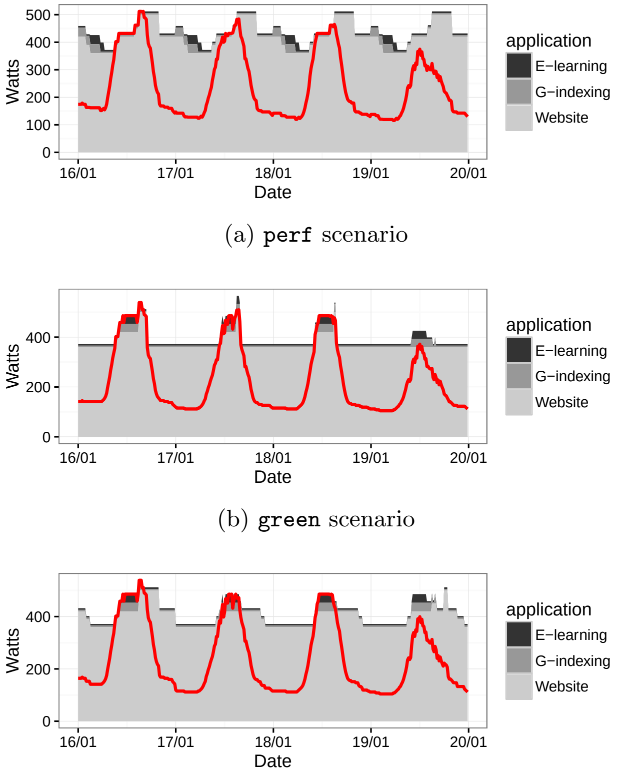
4.2 Results

The resulting power profiles for the three scenarios are depicted in Figures 1a to 1c. We observe that the profiles vary with the scenario despite identical workloads. This confirms that the applications were scaled differently depending the photovoltaic array production.

Figure 2 depicts the achieved daily renewable energy percentage per scenario. We first observe that the percentages vary depending on the day and the scenario with a maximum margin of 3.5 percentage points. This shows that perf provides a respectable value even when the renewable energy availability is ignored. First, this is because the servers have low power consumption and because the photovoltaic array can supply the entire testbed at peak period. Second, the Website, which is the biggest energy consumer, delivers pages upon request from daytime workers. Its workload, so its power consumption, is thus naturally aligned with the solar irradiation. Finally, its flexibility in terms of energy proportionality is limited, as its lowest working mode uses only 34.5% less energy than the most efficient one, and yet is 67.7% less efficient.

We also observe that Carver improves the daily percentage by 3.48% on average compared to perf, while green exhibits a 1.61% increase with regards to Carver. Figures 1a to 1c explain this increase. Indeed, the power profiles of Carver and green indicate that the two indexing applications were deferred to periods with maximum renewable energy, while the Website was scaled down during the evenings of 18 to 20 January to cope with the low share of renewables. Finally, the extra gain obtained in green is justified by its aggressive scale down of the Website to its minimal working mode each time that energy was not provided by the photovoltaic array. In terms of energy consumption, perf was the highest energy consumer, while green was the lowest one. Carver consumed 5.21% less than perf and 6.01% more than green.

Figure 3 shows the daily incomes. This income equals the cost of running the HP LIFE project (estimated at 1 euro per user per year by the HP LIFE managers), reduced by the penalties the data centre must pay to the application owner when the SLO is not reached. As expected, perf provides the maximum



(a) `perf` scenario

(b) `green` scenario

(c) Carver

Fig. 1: Power profiles depending on the running scenario. The line indicates the share of renewable energy used.

incomes as the applications are scaled manually and statically to ensure the SLOs. `Green` provides the lowest income. This is explained by its unique objective of maximising the renewable percentage, without any consideration of the SLOs. We finally observe that `Carver` establishes a trade-off with these two baselines. It provides an income 2.17% lower than `perf` but 52.48% higher than `green`. This exhibits the consequences of not having scaled down the Website during the evenings of 18 to 20 January.

Figure 4 summarises the daily running cost per scenario. This cost is the sum of the energy to be paid for and any financial penalties paid to the smart city authority and to the HP LIFE project owner. We first observe that energy does not play a significant role here. This is explained by the pricing model. For a data centre business manager, it is important to be sure that it is always economically viable to host a client's application even when the SLO is not achieved, to avoid bankruptcy. The incomes must then always cover energy expenses or the payment of human resources. For example, Amazon refunds its clients a maximum of 30%

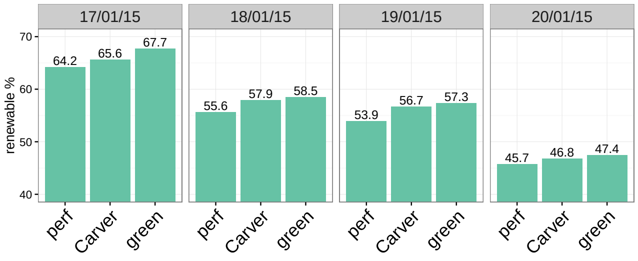


Fig. 2: Daily usage of renewable energy. The expected threshold is 65%.

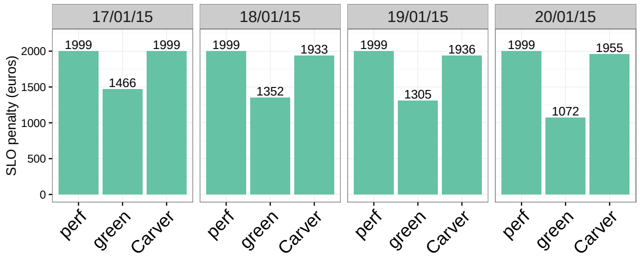


Fig. 3: Daily incomes

of their bill[7]. We observe that Carver provides the lowest daily running costs. This shows that Carver found a valuable compromise between prioritising the HP LIFE application or the smart city authority. Compared to `perf`, Carver reduced incomes by 2.17% to increase renewable energy use by 3.48% and the resulting running costs by 34.71%. Contrary to `green`, Carver did not over-commit on the smart city objective. It traded 1.61% of renewable energy to increase the incomes by 52.48% and the resulting running costs by 52.35%.

It is worth noting that the running costs of Carver and `perf` are close on the first and the last days. The difference on the first day is justified by the high solar irradiation as stated earlier. The difference on the last day is explained by the lack of flexibility in terms energy consumption for the Website application coupled with the low solar irradiation. Indeed, the objective was unrealistic and the high power consumption of the Website, even in its lowest working mode, precludes paying a small SLO penalty to avoid paying a high one to the smart city authority. In addition to validating the benefits of Carver, this evaluation also confirms the important of employing energy-adaptive applications with high variable working modes.

[7] `https://aws.amazon.com/fr/ec2/sla/`

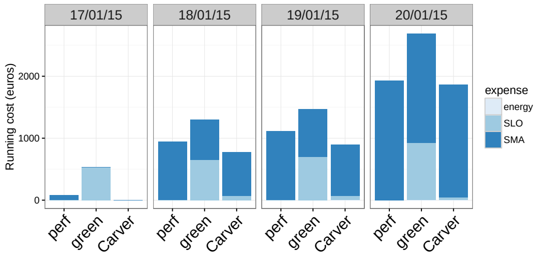


Fig. 4: Data centre running costs

5 Related works

Initial solutions to address energy efficiency focused on minimising the number of online servers, the power consumption, and gas emissions. With the democratisation of renewable energies, solutions have evolved to integrate their intermittent nature. Accordingly, the community integrated a temporal dimension going as far as to leave the spatial dimension to the underlying resource manager.

Li *et al.* [4] propose a scheduler for batch and service jobs that dynamically adjusts the number of online servers to maximise the usage of a purely renewable power source. Contrary to Carver, SLOs are not explicitly considered and the proposed scheduler presumes non-elastic jobs.

Goiri *et al.* propose with GreenSlot [15] and Greenswitch [2] a job scheduler that reduces the grid's electricity cost when the grid is up and reduces the performance degradation otherwise. The proposed scheduler is devoted to a modified version of Hadoop, called GreenHadoop [3] that supports deferrable and non-deferrable jobs, and the performance degradation denotes the amount of the workload to process. Lui *et al.* [5] consider the cooling costs along with the energy cost and the renewable share. The SLO represents a certain response time for interactive applications, or a completion time for the batch jobs. However, the proposed scheduler forces SLO satisfaction. Similarly to these works, Carver supports deferrable and non-deferrable applications through cumulative and instant SLOs while the penalty model is flexible enough to support their notion of performance degradation, but also instantaneous performance metrics. Carver does not integrate a cooling model, but does include the flexibility offered by constraint programming, and the economic objective provides the required entry points to integrate this concern. Most importantly, Carver offers the benefits of elastic applications, while its economic objective helps identify which of the clients or smart city authority is likely to minimise the total running costs.

Recently, Wang *et al.* [16] proposed a game-based, cloud-pricing framework to maximise the cloud profit regarding energy costs. Carver also addresses profit maximisation but at a lower level. While these authors centre on computing a pricing model that guides clients to profitable periods, Carver focuses on scaling applications with regards to their pricing model and trade-off possibilities.

In terms of energy-adaptive applications, Oliviera *et al.* [6] proposed applications that revise their design with regards to their current performance expectations and an energy regulation imposed by the underlying resource manager that focuses on maximising data centre revenue. Similarly, Carver supports variadic application architectures with the notion of working modes and trading possibilities to minimise running costs. However, Carver differs through its integration of the temporal dimension of the problem to integrate intermittent power sources. This makes it possible to study the energy dispatch over time, support cumulative performance models, and take care of transition costs. Hasan *et al.* [8] provides green awareness in interactive cloud applications. Their controllers increase revenues through better response times, and decrease renewable and non-renewable energy use. Carver also scales interactive applications. Thanks to an explicit manipulation of the temporal dimension, it also supports non-interactive applications and join the business and the energy-awareness dimensions using an economical model to reach a sustainable profitability.

6 Conclusion

Energy efficiency in data centres is addressed through workload management to reduce the environmental footprint or operational costs. Recent approaches take also into account the intermittent nature and lower prices of renewable energy sources and aim to limit the *brown* energy input by shaving or shifting the load. In doing so, they often overlook the performance requirements of the client's applications, even though the economic return of a data centre depends on these two factors. Our approach is driven by a broader view: (1) data centre managers are encouraged to adopt environmental measures, if these measures translate in terms of financial gain; and (2) financial gain results from a trade-off between energy consumption and client satisfaction.

We proposed Carver, a tool to orchestrate on a 24-hour horizon, energy-adaptive applications according to their performance level, the variable availability of renewable energies, and the variable price of multiple power sources. Carver adjusts the application's working modes in order to minimise a global financial cost combining the power cost and penalties for not complying with the client service level objectives and the smart city authority's requirements in terms of renewable energy use. Carver currently supports performance models for batch- and service-oriented applications, and flexible penalty functions that go beyond standard piecewise linear models.

We validated Carver by simulating a 4-day execution of a testbed powered by the grid and a photovoltaic array. The simulator mimicked the production workload of the e-learning application HP LIFE. Carver provided the best incomes for the data centre provider by balancing the SLOs the smart-city authority penalties with regards to the workload and the availability of renewable energies.

References

1. Aksanli, B., Venkatesh, J., Zhang, L., Rosing, T.: Utilizing Green Energy Prediction to Schedule Mixed Batch and Service Jobs in Data Centers. SIGOPS OSR (January 2012)
2. Goiri, I.n., Le, K., Haque, M.E., Beauchea, R., Nguyen, T.D., Guitart, J., Torres, J., Bianchini, R.: Greenslot: Scheduling energy consumption in green datacenters. In: Proceedings of International . Conference for High Performance Computing, Networking, Storage and Analysis, ACM (2011)
3. Goiri, I.n., Le, K., Nguyen, T.D., Guitart, J., Torres, J., Bianchini, R.: Green-Hadoop: Leveraging Green Energy in Data-processing Frameworks. In: Proceeding of the 7th ACM Eurosys, ACM (2012)
4. Li, Y., Orgerie, A.C., Menaud, J.M.: Opportunistic scheduling in clouds partially powered by green energy. In: IEEE Intl. Conference GreenCom. (2015)
5. Liu, Z., Chen, Y., Bash, C., Wierman, A., Gmach, D., Wang, Z., Marwah, M., Hyser, C.: Renewable and Cooling Aware Workload Management for Sustainable Data Centers. In: Proceeding of the 12th ACM SIGMETRICS, ACM (2012)
6. de Oliveira, Jr., F.A., Ledoux, T.: Self-management of cloud applications and infrastructure for energy optimization. SIGOPS OSR (July 2012)
7. Dupont, C., Sheikhalishahi, M., M. Facca, F., Hermenier, F.: An energy aware framework for virtual machine placement in cloud federated data centres. In: 8th IEEE/ACM Intl. Conference on Utility and Cloud Computing. (December 2015)
8. Hasan, M.S., d. Oliveira, F.A., Ledoux, T., Pazat, J.L.: Enabling green energy awareness in interactive cloud application. In: 2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). (Dec 2016) 414–422
9. Sharma, N., Sharma, P., Irwin, D., Shenoy, P.: Predicting solar generation from weather forecasts using machine learning. In: IEEE Intl. Conference on Smart Grid Communications. (Oct 2011)
10. Colmant, M., Kurpicz, M., Felber, P., Huertas, L., Rouvoy, R., Sobe, A.: Process-level Power Estimation in VM-based Systems. In: Proceeding of the Tenth Eurosys, ACM (2015)
11. Rossi, F., van Beek, P., Walsh, T., eds.: Handbook of Constraint Programming. Volume 2 of Foundations of Artificial Intelligence. Elsevier (2006)
12. Menana, J., Demassey, S.: Sequencing and counting with the multicost-regular constraint. In: Proceedings of the 6th Intl. Conference CPAIOR, Springer-Verlag (2009)
13. Cano, I., Aiyar, S., Krishnamurthy, A.: Characterizing private clouds: A large-scale empirical analysis of enterprise clusters. In: Proceedings of the Seventh ACM Symposium on Cloud Computing. SoCC '16, New York, NY, USA, ACM (2016)
14. The DC4Cities Consortium: D6.3 - Report on the experimentation phase 2. Evaluation report on the second trial cycle. `http://www.dc4cities.eu/` (2015)
15. Goiri, I.n., Katsak, W., Le, K., Nguyen, T.D., Bianchini, R.: Parasol and GreenSwitch: Managing Datacenters Powered by Renewable Energy. SIGARCH Comput. Archit. News **41**(1) (March 2013)
16. Wang, C., Nasiriani, N., Kesidis, G., Urgaonkar, B., Wang, Q., Chen, L.Y., Gupta, A., Birke, R.: Recouping energy costs from cloud tenants: Tenant demand response aware pricing design. In: Proceedings of the 6th Intl. Conference on Future Energy Systems, ACM (2015)