Heuristiques d'ordonnancement pour les centres de données alimentés par énergies renouvelables

Léo Grange, Patricia Stolf, Georges da Costa, Amal Sayah

Compas'2017 : Parallélisme / Architecture / Système
Sophia Antipolis, France, du 27 au 30 juin 2017

Heuristiques d'ordonnancement pour les centres de données alimentés par énergies renouvelables

Léo Grange, Patricia Stolf, Georges Da Costa, Amal Sayah

IRIT, Université de Toulouse
118 Route de Narbonne
F-31062 Toulouse Cedex 9, France
{leo.grange, stolf, dacosta, sayah}@irit.fr

Résumé
La consommation d'énergie des centres de données est un sujet qui est devenu, au cours des dernières années, de plus en plus important. L'utilisation de sources d'énergies renouvelables pour alimenter ces derniers est une perspective intéressante pour réduire, à terme, leur coût d'exploitation et leur impact écologique. Cependant, la nature intermittente des principales énergies renouvelables pose de nouveaux problèmes. Dans cet article, nous proposons une approche d'ordonnancement de tâches prenant en compte la disponibilité des énergies renouvelables pour maximiser leur utilisation. Nous avons implémenté cette approche dans un simulateur de centre de données et l'avons évaluée en considérant un centre alimenté par des panneaux photovoltaïques et connecté au réseau électrique. Les résultats indiquent une réduction de la consommation d'énergie non renouvelable jusqu'à 49% comparé à un ordonnancement classique ne prenant pas en compte la disponibilité de l'énergie renouvelable.

Mots-clés : énergies renouvelables, ordonnancement en ligne, centre de données

---

1. Introduction

De nos jours plus que jamais, la consommation d'énergie générée par l'utilisation des technologies informatiques est une problématique majeure, tant d'un point de vue écologique qu'économique. Avec l'émergence et le développement des paradigmes du *grid computing* et du *cloud computing*, durant la dernière décennie, le nombre et la taille des centres de données ont considérablement augmenté. En consommant au niveau mondial, pour l'année 2012, environ 270 TW h [19], ils sont responsables d'une part significative de la consommation d'énergie humaine. Cela correspond, d'après la même étude, à 1.4% de la consommation électrique mondiale. De plus, elle est en hausse de près de 5% par an en moyenne depuis 2006.
Afin de réduire l'impact écologique de leurs centres de données, tout en réduisant les coûts d'exploitation, plusieurs opérateurs ont déjà construit des centres partiellement ou totalement alimentés par des sources d'énergies renouvelables (ou EnR) [1, 3]. La nature intermittente et difficilement prévisible de la plupart des EnR, telles que les énergies solaires et éoliennes, soulève de nouveaux problèmes relatifs à la gestion de l'énergie.
Dans l'approche que nous proposons, nous considérons le problème au niveau d'un unique centre de données. Celui-ci, disposant de sources renouvelables à proximité, est utilisé pour exécuter une charge de travail composée de tâches ponctuelles, de type *batch*, ayant une contrainte

de date d'échéance. Contrairement aux approches existantes, détaillées en section 2, nous proposons de séparer la gestion de l'infrastructure électrique de celle des ressources informatiques. Nous proposons une heuristique gloutonne d'ordonnancement, nommée Attractiveness-Based Blind Scheduling Heuristic, ou ABBSH, qui échange des informations avec le système de gestion électrique afin de prendre en compte la disponibilité de l'énergie. Cette approche a été implémentée dans un simulateur de centre de données de la communauté, que nous avons étendu pour gérer différents composants électriques.
Cet article est organisé de la façon suivante. En premier lieu, une présentation des approches similaires est donnée en section 2. Notre approche est décrite en section 3, ainsi que les concepts utilisés par celle-ci. La section 4 détaille la méthodologie utilisée pour valider notre heuristique, dont les résultats sont décrits en section 5. Enfin, la section 6 présente nos conclusions et la direction de nos travaux futurs.

2. État de l'art

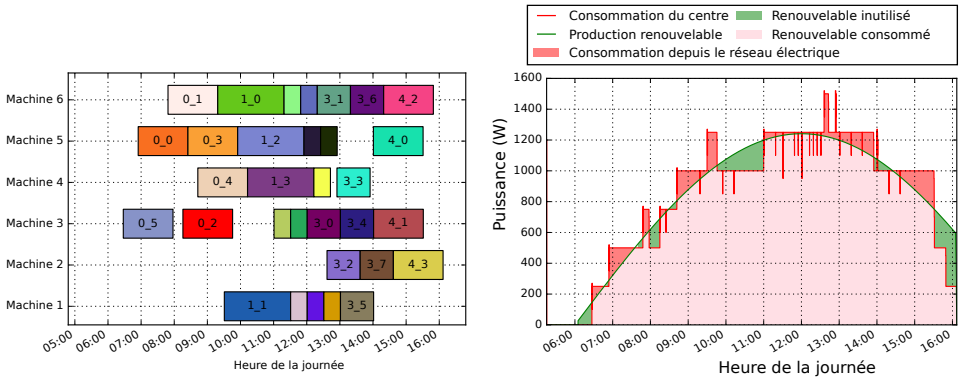
On peut catégoriser les travaux existants selon deux directions différentes. La première consiste à profiter de la distribution géographique de plusieurs centres pour répartir la charge là où l'énergie est disponible à un moment donné [10, 16].
La seconde considère un centre de données unique pour lequel on adapte la charge de travail, et donc la consommation des machines, à la puissance disponible via les EnR. Si la charge de travail est composée de *services interactifs*, comme des serveurs web, il est possible d'utiliser des techniques classiques de gestion de l'énergie pour faire varier le compromis entre qualité de service (QoS) et consommation [4, 15]. Dans le cas de tâches ponctuelles, il est possible de retarder leur exécution afin de les traiter quand l'énergie est effectivement disponible [8, 9, 12].
Quelques travaux considèrent un modèle similaire au nôtre, en particulier deux approches de Goiri et al. qui traitent l'ordonnancement de tâches ponctuelles ayant des dates d'échéance.
Avec GreenSlot [9], les auteurs proposent une heuristique de placement prenant en compte des prédictions de production solaire et le coût d'achat de l'électricité au réseau. Dans cette approche, le temps est représenté sous forme de *slots* de durée fixe, valués avec la production prédite et le coût d'achat. L'algorithme consiste alors à chercher, pour chaque tâche soumise, le premier slot qui minimise le coût, sans risquer de faire dépasser la date d'échéance.
L'autre approche, GreenSwitch [8], propose un algorithme d'ordonnancement conçu spécifiquement pour des charges de type MapReduce. Une formulation en programme linéaire en nombre entiers (MILP) est utilisée pour optimiser l'exécution des tâches et l'utilisation des différentes sources d'énergie.

3. Approche proposée

Nous allons donner une vue d'ensemble du but recherché, exposer l'infrastructure considérée, ainsi que les modèles des parties électriques et informatiques utilisées dans notre approche, puis détailler l'algorithme d'ordonnancement lui-même.

3.1. Principe

Pour donner une meilleure vue d'ensemble de l'approche, nous proposons d'illustrer la manière dont la disponibilité des EnR est mise à profit. Dans l'exemple de la figure 1, un centre de données est alimenté par des panneaux photovoltaïques en plus d'une alimentation au réseau électrique. Des tâches sont soumises au cours du temps, chacune ayant une date d'échéance à laquelle elle doit être terminée.



(a) Une exécution possible des tâches. (b) Profil de puissance correspondant.

FIGURE 1 – Ordonnancement et profil de puissance associé dans une approche considérant l'intermittence des sources d'énergies renouvelables.

Une approche classique, sans prise en compte des EnR, cherchera à minimiser l'énergie totale consommée tout en assurant la qualité de service. Une approche adaptée à l'utilisation d'EnR doit, en plus, adapter la consommation à la disponibilité des sources. La figure 1 présente un ordonnancement possible et l'utilisation de l'électricité correspondante. Dans la figure 1a, l'exécution des différentes tâches et l'utilisation des machines sont représentées au cours du temps. La production d'EnR et la consommation des machines sont données dans la figure 1b. Un tel ordonnancement permet d'obtenir des courbes de consommation et de production similaires.

3.2. Infrastructure et modèles du centre de données

Un des objectifs de l'approche présentée ici est de réduire, autant que possible, le couplage entre les systèmes de gestion des ressources informatiques et de l'infrastructure électrique. Cela permet de concevoir ces systèmes séparément et de faciliter leur adaptation à des infrastructures différentes. Chacun des deux systèmes de gestion dispose de son propre modèle et peut effectuer ses optimisations indépendamment (telles que le choix des sources pour le système électrique, la gestion de l'allumage des machines ou l'ordonnancement des tâches pour le système informatique).

Modèle des ressources informatiques. On considère un ensemble $\mathcal{J}$ de J tâches soumises au système. Pour chaque tâche j, on dispose de sa date de soumission S_j, sa date d'échéance D_j, le nombre de processeurs requis n_j, la quantité de RAM nécessaire r_j et son temps d'exécution T_j. La date de début d'exécution, déterminée par l'ordonnanceur, est notée B_j. La *flexibilité* d'une tâche, donnant une idée de la marge disponible pour son placement, vaut $D_j - B_j - T_j$.

Les ressources informatiques sont représentées par un ensemble $\mathcal{M}$ de M machines. Pour chaque machine m, on note N_m son nombre de processeurs et R_m la quantité de mémoire installée. La consommation des machines est donnée par sa part statique ps_m (consommation au repos) et par la consommation additionnelle maximale $pmax_m$ engendrée par chaque processeur. Ce modèle simplifie le calcul du temps d'exécution des tâches en supposant que leur comportement est identique sur chacune des machines. Il est cependant possible d'ajouter un coefficient de vitesse d'exécution à chaque machine, par rapport à une machine de référence, sans changement majeur dans l'approche proposée, afin de modéliser des centres de données hétérogènes.

Il est possible d'allumer ou d'éteindre des machines. Les transitions entre ces deux états sont prises en compte dans le modèle, nécessitant un temps $tshut_m$ pour l'extinction et $tboot_m$ pour le démarrage. De plus, la consommation des machines durant ces phases est différente de leur consommation au repos. On note $p_m(t)$ la fonction qui donne la consommation d'une machine

au temps t. Lorsqu'elle est allumée, $p_m(t)$ vaut $ps_m + up_m(t)pmax_m$, et 0 lorsqu'elle est éteinte, avec $up_m(t)$ le nombre de processeurs en cours d'utilisation. Pendant l'extinction et l'allumage, la consommation $p_m(t)$ est respectivement $pshut_m$ et $pboot_m$. La consommation totale $P(t)$ du centre de données peut alors être obtenue par $P(t) = \sum_m p_m(t)$.
Une tâche est exécutée sur une seule machine, sans migration et sans interruption. Elle utilise entièrement les ressources requises (mémoire et nombre de processeurs) durant toute la durée de son exécution. Sa date d'échéance fait partie du contrat (SLA) passé entre le client et l'opérateur du centre. La tâche viole le SLA si $B_j + T_j > D_j$.
Modèle de l'infrastructure électrique. L'infrastructure électrique considérée ici est composée d'un ensemble de sources d'énergie renouvelable et d'une connexion au réseau électrique. Pour chaque source w, la production effective au temps t est donnée par $available_w(t)$. On considère que, à chaque source, est également associée une fonction de prédiction de la production future, $predicted_w(t)$, supposée définie dans une fenêtre de durée $window_{prediction}$ à partir de l'instant actuel. Cette fonction peut être implémentée en se basant sur des travaux existants, que cela soit pour des panneaux photovoltaïques [5, 11] ou pour des éoliennes [14, 17].
Le modèle du réseau électrique prend en compte une variation des prix au cours du temps, connue à l'avance pour la même durée $window_{prediction}$. Ce coût par kilowatt-heure est donné par $gridp(t)$.
Attractivité. Chaque système de gestion, électrique et informatique, n'a une connaissance complète que de son modèle respectif, décrit précédemment. De plus, leurs objectifs sont assez différents : réduire le coût et la quantité d'énergie achetée depuis le réseau pour le premier, et exécuter les tâches en respectant les SLA pour le second. Dans notre approche, les informations échangées entre eux sont valuées de manière abstraite et normalisée. Nous nommons *attractivité* cette valeur, comprise entre -1 et 1, qui représente le bénéfice (>0) ou le coût (<0), pour une *proposition* donnée, du point de vue de l'un des systèmes.
Une fonction d'attractivité est définie pour chacun des deux systèmes. L'attractivité informatique est calculée, pour un placement possible, en fonction de la fin d'exécution prévue. Plus ce moment se rapproche de la date d'échéance, plus l'attractivité diminue, jusqu'à atteindre -1 si le SLA est violé.
Pour le système électrique, une proposition consiste en une puissance totale demandée entre deux instants. La fonction d'attractivité utilisée ici prend en compte les EnR et le réseau. Si toute l'énergie provient des EnR, la valeur sera d'autant plus proche de 1 que l'excès de production est important. Sinon, elle se rapproche de -1 en fonction du recours au réseau et du prix moyen durant la période.
Algorithme d'ordonnancement. L'algorithme proposé ici, ABBSH (pour Attractiveness-Based Blind Scheduling Heuristic), est une heuristique gloutonne en ligne. Une version simplifiée du placement d'une tâche est donné par l'algorithme 1. Le pas de temps utilisé est calculé en fonction de la durée d'exécution de la tâche considérée. Sa valeur pour une tâche j est donnée par $\min(\alpha_{step}.T_j, step_{max})$, avec $\alpha_{step} = 0.3$ et $step_{max} = 0.5$ h. En plus du placement des tâches, l'algorithme gère l'extinction des machines inutilisées. Une machine éteinte est rallumée $tboot_m$ avant la prochaine tâche planifiée sur celle-ci.
Afin de choisir parmi ces placements possibles, nous sommes confrontés à un problème multi-objectifs : d'une part la qualité de l'énergie utilisée (représentée par l'attractivité électrique) et d'autre part le respect des critères de qualité de service (représenté par l'attractivité informatique). Nous avons utilisé deux méthodes multi-objectifs différentes pour déterminer la meilleure solution. La première est une simple somme pondérée des valeurs d'attractivité. La seconde est une variante de l'approche *fuzzy-based* présentée par Sun et al. [18].

Function *placeTask*(j, *multiObjectiveFun)*
  placements ← liste vide ;
  foreach $t_{cur} \in$ *pas de temps dans la fenêtre temporelle* **do**
    foreach *machine* $\in \mathcal{M}$ **do**
      $t_s \leftarrow$ premier instant $\geq t_{cur}$ avec (c_j, r_j) disponible sur machine durant T_j ;
      $P_{required} \leftarrow$ moyenne de $P(t)$ entre $[t_s, t_s + T_j]$, incluant le placement étudié ;
      $a_{elec} \leftarrow$ attractivité électrique pour la proposition $(P_{required}, t_s, T_j)$;
      $a_{it} \leftarrow$ attractivité informatique pour la proposition (j, t_s) ;
      place ← (machine, t_s, a_{it}, a_{elec}) ;
      placements.ajouter(place) ;
    end
  end
  (bestMachine, bestTime) ← multiObjectiveFun(placements) ;
  planifier l'exécution de j sur bestMachine avec B_j = bestTime ;

Algorithm 1: Pseudo-code du placement d'une tâche pour l'algorithme ABBSH. Le paramètre *multiObjectiveFun* est la fonction multi-objectifs utilisée.

Pour chaque tâche à placer, notre heuristique nécessite un nombre d'opérations proportionnel au nombre M de machines, répétées pour chaque pas de temps considéré pour une tâche donnée. Ce nombre de pas de temps ne dépend ni du nombre de machines ni du nombre de tâches. La complexité de notre algorithme pour un ordonnancement complet est $\mathcal{O}(M \cdot J)$, avec J le nombre de tâches.

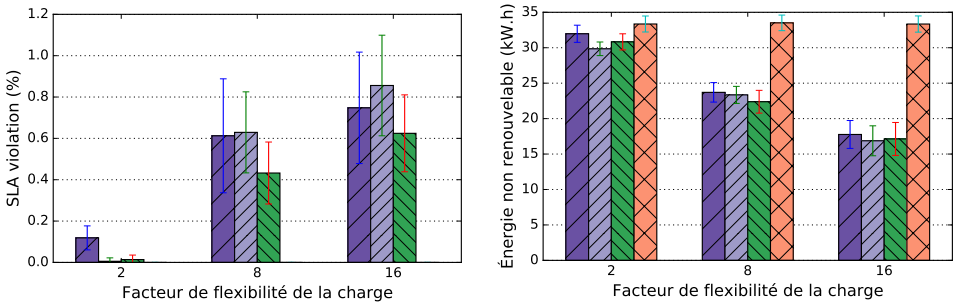
4. Méthodologie et validation

Nous avons évalué notre approche en utilisant DCworms [13], un simulateur de centre de données créé principalement pour étudier la consommation d'énergie. Nous l'avons étendu afin de permettre la simulation de composants électriques, tels que des sources d'énergies renouvelables ou des éléments de stockage de l'énergie.
Nous avons utilisé un générateur de charge synthétique, basé sur une précédente étude de traces issues de grappes de serveurs de Google [7]. Afin d'étudier le comportement des algorithmes en fonction de la qualité de service demandée, nous avons fait varier la flexibilité des tâches (et donc leurs dates d'échéance). La flexibilité est générée aléatoirement via une loi normale, dont les paramètres dépendent de la priorité de la tâche, telle que définie dans [7]. Nous pondérons alors la valeur obtenue par le *facteur de flexibilité* utilisé.
Les charges générées pour la validation de l'algorithme contiennent des tâches soumises durant 72 heures, correspondant à 3600 tâches en moyenne. Dix charges ont été générées pour chacune des valeurs du facteur de flexibilité étudiées.
L'environnement simulé est un centre de données de petite taille, assez similaire en termes de puissance de calcul et de consommation à celui utilisé par les auteurs de GreenSlot [9]. Les serveurs simulés correspondent aux caractéristiques du Dell PowerEdge R210 II, un serveur rack commercial équipé d'un processeur Intel Xeon à quatre cœurs (chacun étant considéré comme un processeur dans ce modèle) et de 32 Go de RAM. Sa consommation en *idle* est ps_m = 44 W et environ 130 W en pic de charge [2]. Dans notre modèle, la totalité de la consommation dynamique, soit 86 W ici, est associée aux processeurs. Le système ayant quatre cœurs, on trouve $pmax_m$ = 21.5 W. Le centre de données est homogène, composé de 10 de ces machines, soit 40 processeurs disponibles.
L'infrastructure électrique est composée de deux sources. La première est un ensemble de panneaux photovoltaïques, fournissant un pic de production de 1500 W. Cela correspond, en supposant une efficacité de 15% [6], à 10 m^2 de panneaux. Nous avons considéré une prédiction

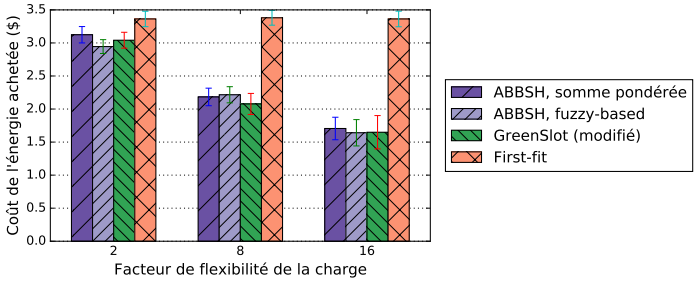
parfaite de la production future ($predicted_{solar}(t) = available_{solar}(t)$). L'ensoleillement simulé durant les expériences correspond à des journées sans nuage, aux moments des équinoxes. La seconde source est le réseau électrique, avec une variation du prix de l'énergie sur un principe d'heures creuses et d'heures pleines, utilisant les tarifs et créneaux de [9].
Pour évaluer notre approche, nous avons comparé différentes variantes de celle-ci à d'autres algorithmes d'ordonnancement. En premier lieu, nous avons utilisé un algorithme *first-fit*, sans prise en compte de la disponibilité des EnR. L'algorithme de placement de GreenSlot [9] a également été utilisé. Les variantes de notre approche présentées ici concernent la méthode multi-objectifs utilisée, somme pondérée ou *fuzzy-based*.
L'approche de GreenSlot considère un modèle assez proche de celui utilisé ici. Cependant, cet algorithme s'est révélé peu adapté au type des charges de travail utilisées, composées d'une majorité de tâches courtes. GreenSlot utilise des pas de temps fixes (*slots*) pour procéder à la réservation et au placement. Cela cause deux problèmes principaux. Le premier est dû au fait que l'algorithme est exécuté à chaque début de slot. Cela peut conduire à la violation des tâches ayant une flexibilité faible, qui seraient soumises au milieu d'un slot. Le second problème vient du caractère indivisible des slots temporels. En conséquence, une tâche très courte engendrera la réservation des ressources CPU et mémoire durant toute la durée d'un slot.
Aussi, nous avons effectué quelques modifications dans cet algorithme, afin d'améliorer ses résultats dans notre scénario. Le premier problème est résolu par la possibilité de placer une tâche urgente à n'importe quel moment, sans attendre le prochain slot. Pour le second problème, nous avons implémenté la réservation de fraction de slots, et la possibilité pour une tâche d'être placée immédiatement après une réservation existante, dans le même slot temporel.

5. Résultats

Pour chacune des approches et de ses variantes étudiées, nous avons simulé son utilisation dans le centre de données, avec les différentes charges de travail générées. Les résultats pour plusieurs métriques sont visibles dans la figure 2.
Les violations de SLA engendrées par chaque approche sont données dans la figure 2a. L'algorithme first-fit n'engendre aucune violation sur les charges utilisées, montrant qu'il est possible d'exécuter toutes les tâches en respectant leurs dates d'échéance. Pour toutes les heuristiques, le taux moyen de violation du SLA est inférieur à 1%, GreenSlot modifié étant légèrement meilleur. On observe cependant une augmentation du nombre de violations avec l'augmentation de la flexibilité de la charge. Ce résultat contre-intuitif s'explique par le fait que, en étant en mesure de déplacer plus de tâches vers des moments où l'énergie est disponible, ces algorithmes utilisent toutes les machines durant ces périodes. Par conséquent, une tâche ayant une date d'échéance proche qui serait soumise durant ces pics de production risque de ne pas pouvoir être exécutée à temps, faute de machines disponibles.
La figure 2b indique l'énergie non-renouvelable consommée, tandis que la figure 2c présente le coût total de celle-ci. À l'exception du first-fit, toutes les heuristiques tirent parti de l'augmentation de la flexibilité en réduisant l'énergie consommée depuis le réseau. Pour des flexibilités faibles ou élevées, la variante *fuzzy-based* de ABBSH est sensiblement plus efficace que la version adaptée de GreenSlot. Dans le cas de la flexibilité la plus élevée, notre approche réduit l'énergie non-renouvelable consommée de 49.4% et le coût de 51.2%, contre respectivement 48.2% et 51% pour GreenSlot. Au contraire, dans le cas d'une charge avec une flexibilité modérée, GreenSlot est légèrement plus efficace, atteignant 33.2% de réduction de l'énergie du réseau et 38.6% pour le coût, alors que notre approche atteint respectivement 31% et 33.2%.



(a) Taux de violation de SLA. (b) Énergie consommée depuis le réseau.



(c) Coût de l'énergie achetée.

FIGURE 2 – Comparaison des résultats des différentes heuristiques, pour des charges de travail avec différentes valeurs de flexibilité (moyenne et écart-type sur 10 expériences).

6. Conclusion

Dans cet article, nous avons présenté ABBSH, un nouvel algorithme d'ordonnancement de tâches permettant d'adapter la consommation d'un centre de données à la disponibilité des énergies renouvelables. Contrairement aux approches existantes, notre algorithme n'utilise pas directement de modèle de l'infrastructure électrique, mais exploite plutôt des informations abstraites et partielles provenant du système en charge des sources électriques. Notre approche parvient, malgré cette connaissance partielle, à des résultats proches de ceux d'une approche centralisée, et s'avère même légèrement meilleure dans certains cas, permettant d'économiser jusqu'à 3.2% supplémentaire.

Cela motive la direction de nos travaux futurs, l'exploration d'aspects décentralisés dans la résolution du problème : plutôt qu'une simple coopération unilatérale, nous voulons étudier de véritables stratégies de *collaboration* entre les systèmes de gestion électrique et informatique, dans lesquelles les deux parties cherchent activement à négocier un compromis.

7. Remerciements

Les expériences présentées dans cet article ont été effectuées en utilisant la plateforme Grid'5000, soutenu par un groupement d'intérêt scientifique organisé par Inria, le CNRS, RENATER et plusieurs universités ainsi que d'autres organisations (voir https ://grid5000.fr). Ces travaux sont soutenus par le projet DATAZERO, ANR-15-CE25-0012.

Bibliographie

1. Solar Power at Data Center Scale. – http://www.datacenterknowledge.com/archives/2009/06/16/solar-power-at-data-center-scale/, juin 2009.
2. Dell PowerEdge R210 II review. – http://alphr.com/dell/31582/dell-poweredge-r210-ii-review, 2011.
3. Green Data Center - AISO.Net. – http://www.aiso.net/technology, 2016.
4. Aksanli (B.), Venkatesh (J.), Zhang (L.) et Rosing (T.). – Utilizing Green Energy Prediction

to Schedule Mixed Batch and Service Jobs in Data Centers. *SIGOPS Oper. Syst. Rev.*, vol. 45, n3, janvier 2012, pp. 53–57.

5. Chen (C.), Duan (S.), Cai (T.) et Liu (B.). – Online 24-h solar power forecasting based on weather type classification using artificial neural network. *Solar Energy*, vol. 85, n11, novembre 2011, pp. 2856–2870.
6. Chu (Y.) et Meisen (P.). – Review and comparison of different solar energy technologies. *Global Energy Network Institute (GENI), San Diego, CA*, 2011.
7. Da Costa (G.), Grange (L.) et De Courchelle (I.). – Modeling and Generating large-scale Google-like Workload. – In *Proceedings of the Senventh International Green and Sustainable Computing Conference, First International Workshop on Resilience and/or Energy-Aware Techniques for High-Performance Computing, Forthcoming*. IEEE, 2016.
8. Goiri (Í.), Katsak (W.), Le (K.), Nguyen (T. D.) et Bianchini (R.). – Parasol and GreenSwitch : Managing Datacenters Powered by Renewable Energy. – In *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, *ASPLOS '13*, ASPLOS '13, pp. 51–64, New York, NY, USA, 2013. ACM.
9. Goiri (Í.), Le (K.), Haque (M. E.), Beauchea (R.), Nguyen (T. D.), Guitart (J.), Torres (J.) et Bianchini (R.). – GreenSlot : Scheduling Energy Consumption in Green Datacenters. – In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, *SC '11*, SC '11, pp. 20 :1–20 :11, New York, NY, USA, 2011. ACM.
10. Hatzopoulos (D.), Koutsopoulos (I.), Koutitas (G.) et Heddeghem (W. V.). – Dynamic virtual machine allocation in cloud server facility systems with renewable energy sources. – In *2013 IEEE International Conference on Communications (ICC)*, pp. 4217–4221, juin 2013.
11. İzgi (E.), Öztopal (A.), Yerli (B.), Kaymak (M. K.) et Şahin (A. D.). – Short–mid-term solar power prediction by using artificial neural networks. *Solar Energy*, vol. 86, n2, février 2012.
12. Krioukov (A.), Alspaugh (S.), Mohan (P.), Dawson-Haggerty (S.), Culler (D. E.) et Katz (R. H.). – Design and evaluation of an energy agile computing cluster. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2012-13*, 2012.
13. Kurowski (K.), Oleksiak (A.), Piątek (W.), Piontek (T.), Przybyszewski (A.) et Węglarz (J.). – DCworms – A tool for simulation of energy efficiency in distributed computing infrastructures. *Simulation Modelling Practice and Theory*, vol. 39, décembre 2013, pp. 135–151.
14. Li (S.), Wunsch (D. C.), O'Hair (E. A.) et Giesselmann (M. G.). – Using neural networks to estimate wind turbine power generation. *IEEE Transactions on Energy Conversion*, vol. 16, n 3, septembre 2001, pp. 276–282.
15. Liu (Z.), Chen (Y.), Bash (C.), Wierman (A.), Gmach (D.), Wang (Z.), Marwah (M.) et Hyser (C.). – Renewable and Cooling Aware Workload Management for Sustainable Data Centers. – In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, *SIGMETRICS '12*, SIGMETRICS '12, pp. 175–186, New York, NY, USA, 2012. ACM.
16. Liu (Z.), Lin (M.), Wierman (A.), Low (S. H.) et Andrew (L. L.). – Geographical Load Balancing with Renewables. *SIGMETRICS Perform. Eval. Rev.*, vol. 39, n3, décembre 2011.
17. Sideratos (G.) et Hatziargyriou (N. D.). – An Advanced Statistical Method for Wind Power Forecasting. *IEEE Transactions on Power Systems*, vol. 22, n1, février 2007, pp. 258–265.
18. Sun (H.), Stolf (P.), Pierson (J.-M.) et Da Costa (G.). – Energy-efficient and thermal-aware resource management for heterogeneous datacenters. *Sustainable Computing : Informatics and Systems*, vol. 4, n4, décembre 2014, pp. 292–306.
19. Van Heddeghem (W.), Lambert (S.), Lannoo (B.), Colle (D.), Pickavet (M.) et Demeester (P.). – Trends in worldwide ICT electricity consumption from 2007 to 2012. *Computer Communications*, vol. 50, septembre 2014, pp. 64–76.