Compas'2023 : Parallélisme / Architecture/ Système
LISTIC/USMB - Annecy France, 4-7 juillet 2023

Evaluation de la consommation d'énergie nécessaire à l'exécution d'un workload dans un datacenter vert

Louis-Claude Canon✧, Damien Landré✦✧, Laurent Philippe✧, Jean-Marc Pierson✦ et Paul Renaud-Goud✦

✦IRIT, Univ. de Toulouse 3, Toulouse, France

✧Institut FEMTO-ST, Univ. de Bourgogne Franche-Comté, Besançon, France

Résumé
Les datacenters sont un élément essentiel de l'internet mais leur développement continu nécessite de produire des solutions durables afin de limiter leur impact sur le changement climatique. Le projet DATAZERO2 vise à concevoir des datacenters fonctionnant uniquement avec des énergies renouvelables locales. Dans cet article, nous abordons le problème d'évaluation de la puissance minimale nécessaire au traitement d'un workload sous la contrainte d'une qualité de service dans un datacenter vert. Pour résoudre ce problème, nous proposons un algorithme dichotomique nécessitant la détermination de la configuration des machines avec une puissance de calcul maximale. Lorsque les machines sont hétérogènes, nous sommes confrontés au problème du choix des machines à allumer et de leur état DVFS. Pour cela, un MILP (Mixed-Integer Linear Programming) fournissant la solution optimale, et trois heuristiques donnant des résultats satisfaisants en un temps raisonnable sont proposés. Ces heuristiques ont un écart relatif moyen par rapport à la solution optimale de 0,03% à 0,65%.

Mots-clés : Datacenter vert, Consommation de puissance, Optimisation

1. Introduction

Depuis une décennie, les datacenters sont devenus une partie essentielle de l'internet. Leur nombre et leur taille ne cessent d'accroître, tout comme leur consommation d'énergie. Ils représentaient en 2018 1% de la consommation mondiale d'électricité, soit 6% de plus qu'en 2010 [11]. Il est estimé qu'en 2025, leur consommation d'énergie sera multipliée par 2.9 et leur émission de gaz à effet de serre par 3.1. Pour réduire l'impact des datacenters sur le changement climatique, plusieurs travaux de recherche proposent des solutions pour optimiser leur consommation d'énergie. [3], [15]. Ces solutions sont essentielles en terme d'efficacité, mais ne permettent pas de réduire radicalement leur empreinte carbone. D'autres projets et travaux de recherche prétendent réduire leur consommation d'énergie [7], [2]. L'objectif du projet DATAZERO [18] (2015-2019) et DATAZERO2 (2020-2024) est d'étudier les solutions permettant de concevoir et d'exploiter un datacenter alimenté uniquement par des énergies renouvelables locales. Par conception, ce projet s'appuie sur une négociation [21] entre la partie électrique (production d'énergie et stockage) et la partie IT (traitement du workload) du datacenter afin de déterminer un profil de puissance qui sera appliquée à l'infrastructure sur une fenêtre temporelle à venir, typiquement sur les prochaines 72 heures.

La négociation est un processus qui demande les prévisions de consommation de puissance de la partie IT et de production d'électricité pour converger, après plusieurs itérations, vers une solution acceptable à la fois pour la partie IT et pour la partie électrique. Le résultat de la négociation est un profil de puissance, sur une fenêtre temporelle, représenté par une série de valeurs de puissance, une sur chaque intervalle de temps de la fenêtre. Dans cet article, nous abordons le problème du calcul de la prévision de consommation, un profil de puissance, minimum requise pour traiter une prévision de workload. Nous n'abordons pas le problème de prévision de workload, qui a déjà été largement étudié dans la littérature [12]. Nous étudions le problème de la transformation d'une prévision de workload en une utilisation optimisée d'une infrastructure donnée qui minimise les besoins en énergie. Le processus de négociation étant interactif, le calcul des besoins en puissance électrique doit se faire en un temps raisonnable [21]. Nous considérons un workload consolidé, et non des jobs individuels. Un workload est représenté sous la forme d'une quantité totale d'opérations qui doivent être traitées pour chaque pas de temps. Un workload peut ainsi regrouper les opérations de plusieurs jobs utilisant simultanément l'infrastructure et partageant les machines.
Cet article présente plusieurs alternatives d'un algorithme calculant un profil de puissance minimal en plusieurs étapes. L'étape principale est le calcul de la puissance minimale sur chaque intervalle de temps de la fenêtre temporelle grâce à un algorithme dichotomique. A chaque itération, l'algorithme dichotomique calcule la capacité de traitement maximale qui peut être atteinte avec la valeur de puissance, puis ordonnance le workload sous la contraintes d'une qualité de service (QoS). Pour obtenir la capacité de traitement maximale des machines pour une valeur de puissance donnée, plusieurs solutions sont proposées, à savoir un MILP (Mixed-Integer Linear Programming) et différentes heuristiques.

2. Travaux annexes

Afin de minimiser la consommation de puissance ou d'énergie tout en respectant éventuellement d'autres critères, différentes approches en online sont envisagées dans la littérature.
Dans [24], une méthode online gère la consommation d'énergie et l'ordonnancement du workload pour des datacenters hybrides géo-distribués. De manière similaire, dans [17], un algorithme online optimise la consommation d'énergie, la dégradation des performances et le coût du réseau électrique pour de multiples datacenters hybrides. Zhang et al. [23] propose PoDD, un algorithme de plafonnement de la puissance fonctionnant en online maximisant les performances des serveurs homogènes pour les workloads d'applications dépendantes. Une méthode similaire est également proposée pour les machines hétérogènes d'un datacenter. [5]. Dans [22], Différents algorithmes online sont présentés pour minimiser la dégradation des performances et l'énergie totale consommée. Ces algorithmes migrent les machines virtuelles des serveurs surchargés vers les serveurs sous-chargés. D'autres méthodes utilisant l'allocation et la migration de machines virtuelles sont proposées [13], [9], [14], [4]. Dans [10], une approche holistique online ordonnance les machines virtuelles pour minimiser la consommation totale d'énergie du datacenter. Mais ces travaux se concentrent uniquement sur l'objectif de minimiser la consommation d'énergie en online par l'allocation et la migration des machines virtuelles. Dans notre cas, nous cherchons à minimiser une prévision de la demande en énergie.
Dans [14], les auteurs proposent un algorithme multi-objectif online qui optimise la consommation d'énergie. Une méthode similaire dans [6] est utilisée en prenant en compte différents paramètres. Mais l'optimisation est également réalisée en online, avec une approche non-clairvoyante. Dans [8], une méthode clairvoyante online pour prédire la consommation totale d'énergie du datacenter est proposée pour améliorer la gestion de l'énergie. Un réseau de neu-

rones calcule la prévision de la consommation totale d'énergie, une valeur unique pour les 20 minutes à venir, le temps nécessaire pour que le cooling atteigne la température désirée.
Finalement, tous ces travaux traitent du problème en online et, à notre connaissance, aucun travail traite la minimisation en offline de la consommation d'énergie dans le cas de machines homogènes ou hétérogènes avec différentes quantités d'opérations à traiter, sous la contrainte d'une qualité de service.

3. Définition du problème, modèle et objectifs

Le système de gestion de l'énergie, fonctionnant uniquement à l'aide d'énergies renouvelables, doit planifier l'utilisation de ses sources d'énergie et de son stockage afin d'alimenter correctement les machines du datacenter. En raison de contraintes techniques [18], la puissance délivrée aux machines doit être constante sur un intervalle de temps, allant de 15 minutes à une heure. Pour donner la consommation d'énergie sur une fenêtre de temps, nous devons donc définir un profil de puissance donnant, pour chaque intervalle de temps, une valeur de puissance constante. L'alimentation électrique ne reposant que sur des énergies renouvelables intermittentes, il est essentiel de stocker autant d'énergie que possible pour alimenter le datacenter lors des périodes de sous-production. Le problème auquel nous sommes confrontés est donc de calculer le profil de puissance minimum nécessaire au traitement d'un workload donné.
D'autre part, les intervalles de temps sont indépendants. Pour cette raison, nous nous concentrons dans la suite sur la recherche de la valeur minimale P pour un seul intervalle de temps. Notez que le problème est abordé comme un problème en offline mais qu'il est également contraint par l'interactivité de la négociation. Le problème doit donc être résolu en un temps raisonnable. Dans ce qui suit, nous définissons d'abord le problème et le modèle qui lui est associé, puis nous définissons les objectifs.
Dans le contexte de ce travail, la prévision du workload est une donnée d'entrée du problème. Étant donné que les variations de puissance et de workload sont différentes dans le temps, un intervalle de temps est subdivisé en plusieurs pas de temps, dont la durée Δt est comprise entre une seconde et une minute, et les opérations à traiter du workload varient sur ces pas de temps. On note T le nombre de pas de temps d'un intervalle de temps, et on normalise de telle sorte que le t^{eme} pas de temps commence à $t-1$, pour $t \in \mathcal{T} = \{1, \ldots, T\}$.
On définit le workload comme étant un ensemble de W parts de charge, l_k pour $k \in \mathcal{W} = \{1, \ldots, W\}$. Une part l_k est définie par sa date d'arrivée r_k, son nombre d'opérations à traiter p_k et sa deadline d_k. Une part l_k, qui arrive à $t = r_k$ avec une deadline d_k doit être traitée au plus tard à $t = r_k + d_k$. Toutes les opérations à traiter d'une part l_k ont la même deadline d_k.
Le datacenter est composé de M machines, notées i, avec $i \in \mathcal{M} = \{1, \ldots, M\}$. Une machine i consomme $static_i$ lorsqu'elle est inactive. Chaque machine i peut être configurée dans S_i états DVFS différents [6]. L'état DVFS d'une machine est noté $j \in \mathcal{S}^{(i)} = \{0, \ldots, S_i\}$. L'ensemble des machines et de leurs états DVFS définit la configuration du datacenter. On note $\mathcal{S}$ l'ensemble des états DVFS de toutes les machines, $\mathcal{S} = \{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(M)}\}$. Un état DVFS j définit $g^{(i)}_{max_j}$ la quantité maximum d'opérations par seconde pouvant être traitée par la machine i et $power^{(i)}_j$ la puissance consommée par opération par seconde. Si la machine i est allumée, elle traite $g^{(i)}$ opérations par seconde avec $0 \leq g^{(i)} \leq g^{(i)}_{max_j}$ en consommant une puissance de $power^{(i)}_j$ par opération par seconde. Par conséquent, si une machine i traite plusieurs parts l_k dans l'état DVFS j durant un temps Δt avec une quantité d'opérations à traiter $g^{(i)}\Delta t = \sum_{k \in \mathcal{W}} p_k \leq g^{(i)}_{max_j}\Delta t$, elle consomme une puissance de $P_i = static_i + g^{(i)} \times power^{(i)}_j$. On considère que l'état DVFS d'une machine éteinte est $j = 0$ et qu'elle ne consomme rien et ne traite aucune opération

dans cet état. Enfin, la consommation de puissance P d'une configuraiton est $P = \sum_{i \in \mathcal{M}} P_i$ et sa puissance de calcul maximale $w^{(p)}$ est $w^{(p)} = \sum_{i \in \mathcal{M}} g^{(i)} \Delta t$.
L'objectif de ce problème d'optimisation est de trouver une configuration des machines fournissant suffisamment de puissance de calcul pour traiter le workload tout en consommant le moins de puissance possible P. De plus, on définit opk le nombre total d'opérations ne respectant pas leur deadline, ainsi que le ratio D la quantité d'opérations tuées par rapport à la quantité d'opérations à traiter sur l'intervalle de temps (1). Afin de donner aux utilisateurs une garantie de traitement de leurs opérations, ce ratio ne doit pas dépassé un seuil fixé D_{max} (2).

$$D = \frac{opk}{\sum_{k \in \mathcal{W}} p_k} \qquad (1) \qquad\qquad D \leq D_{max} \qquad (2)$$

4. Détermination de la valeur minimale de puissance

Lee problème de minimisation de la valeur de puissance pour un intervalle de temps sous la contrainte de violation de deadline est résolu en deux étapes imbriquées. L'étape principale utilise un algorithme dichotomique présenté en annexe. A chacune des itérations, pour la valeur de puissance donnée P, l'algorithme calcule d'abord (fonction *config*) une configuration de machine qui maximise la puissance de calcul sans consommer plus que la puissance courante, puis ordonnance le workload pour déterminer la valeur opk, le nombre d'opérations tuées. L'ordonnanceur utilise simplement un algorithme EDF (Earliest Deadline First) pour ordonnancer les parts de charge sur les pas de temps. Enfin, si le ratio des deadlines non respectées ne dépasse pas le seuil D_{max}, alors la puissance de calcul est suffisante et la valeur de puissance P peut être réduite, sinon elle est augmentée.
La fonction *config* calcule la configuration des machines la plus puissante pouvant être alimentée avec la puissance P, donnée par l'algorithme dichotomique. Dans le cas hétérogène, ce problème est NP-Difficile. La difficulté réside dans le choix des machines à allumer et de leur état DVFS. La preuve de complexité est donnée en annexe. Dans la suite, nous considérons le cas des machines hétérogènes, avec plusieurs états DVFS, qui inclut le cas homogène. Nous proposons un MILP et, en raison de son temps de calcul et de la nature interactive de la négociation qui doit se faire en un temps raisonnable, différentes heuristiques sont proposées.

maximiser $\sum_{i=1}^{M} g^{(i)}$, sous les contraintes. :

$$\begin{cases} \sum_{j=0}^{S_i} x_{i,j} & = & 1 \\ g^{(i)} & \leq & \sum_{j=0}^{S_i} (x_{i,j} \times g^{(i)}_{max_j}) \\ P_i & = & \sum_{j=1}^{S_i} x_{i,j} (static_i + g^{(i)} power^{(i)}_j) \\ \sum_{i=1}^{M} P_i & \leq & P \end{cases} \qquad \begin{cases} avec : & \\ \forall i \in \mathcal{M}, \forall j \in \mathcal{S}^{(i)} & x_{i,j} \in \{0,1\} \\ \forall i \in \mathcal{M} & g^{(i)} \geq 0 \\ \forall i \in \mathcal{M} & P_i \geq 0 \end{cases} \qquad (3)$$

Mixed Integer Linear Programming : Nous définissons la variable de décision $x_{i,j}$ pour déterminer les machines à allumer et leur état DVFS. Pour chaque machine i et pour chaque état DVFS j, $x_{i,j} = 1$ si l'état DVFS j de la machine i est sélectionné, sinon $x_{i,j} = 0$. Ces variables définissent donc la configuration des machines. Pour une machine donnée, un seul état DVFS peut être sélectionné et reste le même durant tout l'intervalle de temps.
Le MILP est décrit par l'équation 3. L'objectif est de maximiser la puissance de calcul des machines. En utilisant la variable de décision binaire $x_{i,j}$, la première contrainte fixe un seul état

DVFS pour toute machine $i \in \mathcal{M}$. La seconde contrainte impose pour toute machine $i \in \mathcal{M}$ de ne pas dépasser la capacité de calcul maximale de la machine. La troisième contrainte limite la consommation d'énergie de la machine, pour toute machine $i \in \mathcal{M}$. Enfin, la quatrième contrainte impose que la consommation totale d'énergie des machines n'excède pas la valeur de puissance P donnée par l'algorithme dichotomique.

Balance Power-Performance (BPP) : Cette heuristique évalue les meilleurs machines et états DVFS à allumer en fonction de deux métriques : (i) la puissance de calcul et (ii) le ratio de performance (puissance consommée par unité de puissance de calcul). Ces deux métriques sont utilisées pour calculer un score pour chaque type de machine et chaque état DVFS selon une puissance donnée et un paramètre α. Le paramètre α (avec $0 \leq \alpha \leq 1$), donné en entrée, contrôle le compromis entre la puissance de calcul et le ratio de performance. Plus α est proche de 0, plus la puissance de calcul est prise en compte dans le calcul du score, et inversement. L'heuristique BPP allume alors la machine ayant le score le plus élevé. En fonction de la valeur du paramètre α, les configurations proposées peuvent être différentes. Pour cette raison, plusieurs valeurs α sont évaluées afin de produire différentes configurations de machines et obtenir celle qui maximise la puissance de calcul totale.

Best State Redistribute Without Static (BSRWS) : Cette heuristique allume d'abord les machines ayant le meilleur ratio de performance. Si plus aucune machine ne peut être allumée, soit parce qu'elles sont toutes allumées, soit parce qu'il n'y a pas assez de puissance pour en allumer d'autres, alors la puissance restante est redistribuée aux machines allumées. Cette redistribution permet d'augmenter l'état DVFS des machines et donc leur puissance de calcul.

Best State Redistribute Without Static And Removing (BSRWS-AR) : Cette heuristique est une extension de l'heuristique BSRWS qui explore davantage de configurations de machines. Elle éteint une machine de la configuration trouvée précédemment afin de tester des configurations avec moins de machines allumées et davantage de puissance à redistribuer pour augmenter l'état DVFS et donc la puissance de calcul des machines restantes.

5. Expérience et résultats

Nous présentons ici une expérience qui prend l'exemple d'un datacenter d'une taille de 267 kW [18] et 10 types de machine. A noter que différentes tailles de datacenters sont expérimentées dans le rapport de recherche [19] pour évaluer nos heuristiques dans différents cas. Nous avons implémenté le MILP et les heuristiques en Python [1] et les avons exécutés [2] avec des données d'entrées comprenant 1241 machines divisées en 10 types. Notez qu'aucun workload n'est utilisé pour l'expérience car nous évaluons uniquement le MILP et les heuristiques dans la détermination de la puissance de calcul. Les données relatives aux caractéristiques des machines sont connues à l'avance [16], [20]. Ce sont des valeurs moyennes calculées à partir d'expériences réalisées sur Grid5000 par le projet ANR ENERGUMEN [1]. Pour chacun des 10 types de machines nous avons calculé le ratio de performance en W/GFlops, en tenant compte de leur puissance statique et de leurs états DVFS (voir la figure 2 en annexe). Nous avons pu constaté une continuité des courbes de rapport de performance, même lorsque les états DVFS changent, pour tous les types. Par ailleurs nous constatons que, plus la puissance statique et le ratio de performance d'une machine sont faibles, plus il est avantageux d'allumer cette machine.

Les figures 1 (a) et (b) montrent la puissance de calcul maximale des solutions données par le MILP et les heuristiques pour des valeurs de puissance de 63 W à 267 kW par pas de 100 W.

1. Le code source en fichier zip est disponible ici.
2. Les expériences ont été menées sur Ubuntu 22.04.1 LTS, processeur Intel Core i7-11850H, 32.0 Go de mémoire RAM, Python 3.10 et PulP 2.6.0 avec le solveur Gurobi 9.5.1.

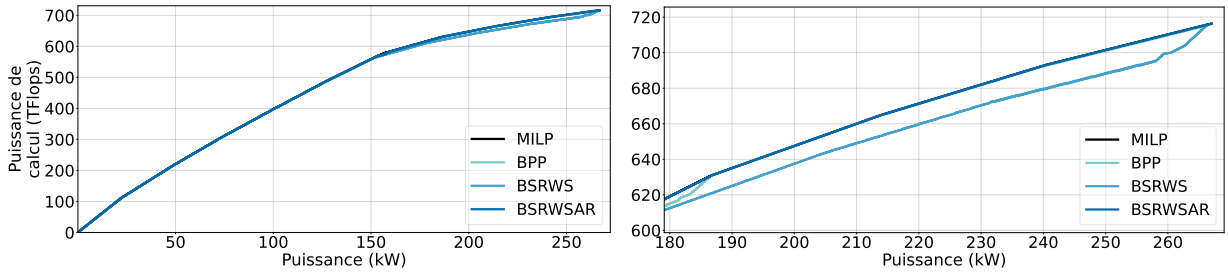

FIGURE 1 – Puissance de calcul en fonction de la puissance ((b) est un grossissement de (a))

L'heuristique BPP et l'heuristique BSRWS-AR sont les plus proches de la solution optimale. Leur écart relatif moyen à l'optimal est de 0.12% et 0.03% respectivement. L'heuristique BSRWS-AR est plus performante que l'heuristique BSRWS car elle explore davantage de configurations. L'heuristique BSRWS a un écart relatif moyen par rapport à l'optimal de 0.65%. En termes de précision, BPP et BSRWS-AR sont les heuristiques les plus satisfaisantes, mais le temps de calcul de BPP est nettement meilleur que celui de BSRWS-AR.

Pour vérifier la compatibilité des propositions avec nos contraintes de temps nous avons mesuré les temps de calcul du MILP et des heuristiques en fonction de la puissance (voir la figure 3 en annexe). Tous les temps de calcul augmentent avec la puissance car le nombre de machines à allumer est plus grand. Le MILP a un temps de calcul moyen de 2,83 s, ce qui le rend inutilisable dans notre contexte. Les heuristiques sont plus rapides pour fournir une configuration. Le temps de calcul de l'heuristique BPP est de l'ordre de quelques millisecondes. Il augmente légèrement en fonction de la puissance. En revanche, celui de l'heuristique BSRWS-AR augmente considérablement : de 0,4 ms à plus de 4 s. L'heuristique BPP semble donc la plus adaptée à nos besoins.

6. Conclusion

Nous abordons ici le problème de minimisation d'une valeur de puissance pour allumer un nombre suffisant de machines pour traiter un workload sur un intervalle de temps sous la contrainte d'une qualité de service. Plusieurs variantes d'un algorithme dichotomique sont proposés pour résoudre ce problème. Cet algorithme utilise deux fonctions, l'une fournissant la puissance de calcul maximale qui peut être obtenue sachant une puissance donnée, et l'autre qui ordonnance le workload sur les machines allumées. Comme le calcul de la puissance de calcul maximale est NP-Hard dans le cas hétérogène, nous proposons un MILP et 3 heuristiques non triviales et comparons leur performance et leur temps de calcul. Les heuristiques donnent des résultats satisfaisants en un temps raisonnable, avec un écart relatif moyen par rapport à l'optimal de 0,12%, 0,65% et 0,03%. Au vu des résultats, l'heuristique BPP (Balance Power-Performance) semble la plus appropriée pour résoudre ce problème en un temps raisonnable.

Remerciements

Les travaux présentés ici sont financés par le projet DATAZERO2 (contract "ANR-19-CE25-0016") et par la Graduate school EIPHI (contract "ANR-17-EURE-0002"). Nous tenons à remercier le projet ANR ENERGUMEN (contract "ANR-18-CE25-0008") ainsi que Grid5000 pour avoir fourni certaines des données utilisées dans cet article.

Bibliographie

1. Energy saving in large scale distributed platforms – Energumen. – https://anr.fr/Project-ANR-18-CE25-0008, 2018. [Online ; accessed 20-November-2022].
2. Apple : Environmental Progress Report (2022). – https://www.apple.com/environment/, 2022. [Online ; accessed 20-May-2022].
3. Chaithra (P.). – Eco friendly green cloud computing. *Journal of Research Proceedings*, vol. 1, n 2, 2021, p. 41–52.
4. Chang (K.), Park (S.), Kong (H.) et Kim (W.). – Optimizing energy consumption for a performance-aware cloud data center in the public sector. *Sustainable Computing : Informatics and Systems*, vol. 20, 2018, pp. 34–45.
5. Ciesielczyk (T.), Cabrera (A.), Oleksiak (A.), Piątek (W.), Waligóra (G.), Almeida (F.) et Blanco (V.). – An approach to reduce energy consumption and performance losses on heterogeneous servers using power capping. *Journal of Scheduling*, vol. 24, 2021, pp. 489–505.
6. Fang (Q.), Wang (J.), Gong (Q.) et Song (M.). – Thermal-aware energy management of an hpc data center via two-time-scale control. *IEEE Transactions on Industrial Informatics*, vol. 13, n5, 2017, pp. 2260–2269.
7. Greendatanet research project. – http://www.greendatanet-project.eu/, 2013–2016. Accessed : 2021-05-28.
8. Hsu (Y.), Matsuda (K.) et Matsuoka (M.). – Self-aware workload forecasting in data center power prediction. – In *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pp. 321–330. IEEE, 2018.
9. Kurdi (H.), Alismail (S.) et Hassan (M.). – Lace : A locust-inspired scheduling algorithm to reduce energy consumption in cloud datacenters. *IEEE Access*, vol. 6, 2018, pp. 35435–35448.
10. Li (X.), Garraghan (P.), Jiang (X.), Wu (Z.) et Xu (J.). – Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy. *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, n6, 2017, pp. 1317–1331.
11. Masanet (E.), Shehabi (A.), Lei (N.), Smith (S.) et Koomey (J.). – Recalibrating global data center energy-use estimates. *Science*, vol. 367, n6481, 2020, pp. 984–986.
12. Masdari (M.) et Khoshnevis (A.). – A survey and classification of the workload forecasting methods in cloud computing. *Cluster Computing*, vol. 23, n4, 2020, pp. 2399–2424.
13. Mazumdar (S.) et Pranzo (M.). – Power efficient server consolidation for cloud data center. *Future Generation Computer Systems*, vol. 70, 2017, pp. 4–16.
14. Nikzad (B.), Barzegar (B.) et Motameni (H.). – Sla-aware and energy-efficient virtual machine placement and consolidation in heterogeneous dvfs enabled cloud datacenter. *IEEE Access*, vol. 10, 2022, pp. 81787–81804.
15. Pahlevan (A.), Rossi (M.), Garcia del Valle (P.), Brunelli (D.) et Atienza Alonso (D.). – *Joint computing and electric systems optimization for green datacenters*. – Rapport technique, Springer, 2017.
16. Pedretti (K.), Grant (R.), III (J. L.), Levenhagen (M.), Olivier (S.), Ward (L.) et Younge (A.). – A comparison of power management mechanisms : P-states vs. node-level power cap control. – In *International Parallel and Distributed Processing Symposium Workshops*. IEEE, 2018.
17. Peng (Y.), Kang (D.), Al-Hazemi (F.) et Youn (C.). – Energy and qos aware resource allocation for heterogeneous sustainable cloud datacenters. *Optical Switching and Networking*, vol. 23, 2017, pp. 225–240.
18. Pierson (J.), Baudic (G.), Caux (S.), Celik (B.), Da Costa (G.), Grange (L.), Haddad (M.), Lecuivre (J.), Nicod (J.), Philippe (L.), Rehn-Sonigo (V.), Roche (R.), Rostirolla (G.), Sayah

(A.), Stolf (P.), Thi (M.) et Varnier (C.). – Datazero : Datacenter with zero emission and robust management using renewable energy. *IEEE Access*, vol. 7, 2019, pp. 103209–103230.

19. Research report : Assessing power needs to run a workload with quality of service constraint on green datacenters. – `https://members.femto-st.fr/Laurent-Philippe/sites/femto-st.fr.Laurent-Philippe/files/content/articles/rr-1-2023.pdf`, 2023.
20. Standard performance evaluation corporation. – `http://spec.org/`. Accessed : 2023-02-28.
21. Thi (M.), Pierson (J.), da Costa (G.), Stolf (P.), Nicod (J.), Rostirolla (G.) et Haddad (M.). – Negotiation Game for Joint IT and Energy Management in Green Datacenters. *Future Generation Computer Systems*, vol. 110, 2020, pp. 1116–1138.
22. Yadav (R.), Zhang (W.), Li (K.), Liu (C.), Shafiq (M.) et Karn (N.). – An adaptive heuristic for managing energy consumption and overloaded hosts in a cloud data center. *Wireless Networks*, vol. 26, 2020, pp. 1905–1919.
23. Zhang (H.) et Hoffmann (H.). – Podd : Power-capping dependent distributed applications. – In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–23, 2019.
24. Zhao (M.), Wang (X.) et Mo (J.). – Workload and energy management of geo-distributed datacenters considering demand response programs. *Sustainable Energy Technologies and Assessments*, vol. 55, 2023, p. 102851.

Annexes

A. Algorithme dichotomique

Algorithm 1: Algorithme dichotomique minimisant la puissance nécessaire à l'exécution d'un workload sur un intervalle de temps.

Data: $\mathcal{M}, \mathcal{S}, \mathcal{W}, \mathcal{T}, \epsilon, D_{max}, \Delta t$
Result: minimize P

begin
  $P_{min} \leftarrow 0$; $P_{max} \leftarrow \sum_{i \in \mathcal{M}} (static_i + power^{(i)}_{S_i} \times g^{(i)}_{max_{S_i}})$
  while $P_{max} - P_{min} \geq \epsilon$ **do**
    $P \leftarrow (P_{min} + P_{max})/2$
    $w^{(p)} \leftarrow config(\mathcal{M}, \mathcal{S}, P, \Delta t)$
    $opk \leftarrow 0$; $\bar{\mathcal{W}} \leftarrow \mathcal{W}$
    for $t \in \mathcal{T}$ **do**
      $\bar{w}^{(p)} \leftarrow w^{(p)}$
      $\bar{\mathcal{W}}, opk \leftarrow schedule(\bar{\mathcal{W}}, opk, \bar{w}^{(p)}, t)$
    $D \leftarrow opk / \sum_{k \in \mathcal{W}} p_k$
    if $D \leq D_{max}$ **then** $P_{max} \leftarrow P$ **else** $P_{min} \leftarrow P$

B. Preuve de complexité

D'un point de vue complexité, le problème du calcul de la puissance de calcul maximale $w^{(p)}$ avec des machines hétérogènes est NP-Hard puisqu'il est au moins aussi difficile que le problème de partition. Le problème est trivialement dans NP puisque nous pouvons vérifier une solution à partir des variables de décision $g^{(i)}$ en temps polynomial. En outre, toute instance du problème de partition peut être directement réduite à notre problème : pour chaque entier z_i, nous considérons une machine telle que $static_i = 0$ et $g^{(i)}$ avec deux valeurs possibles, *i.e.* 0 ou $g^{(i)}_{max_j} = z_i$. Notez que, dans le cas général, $g^{(i)}$ est une variable discrète comprise entre 0 et $g^{(i)}_{max_j}$. Dans ce cas particulier, nous donnons simplement la valeur la plus basse possible à $g^{(i)}_{max_j}$. Ainsi, la puissance consommée par une machine devient constante, $P_i = static_i + g^{(i)}_{max_j} = z_i$. En outre, nous fixons la puissance totale disponible $P = \frac{1}{2}\sum_i z_i$. Il existe un ordonnancement avec une puissance de calcul maximale $w^{(p)} = P$ si et seulement si le problème de partition a une solution valide.

C. Figure 2 : Performances des machines

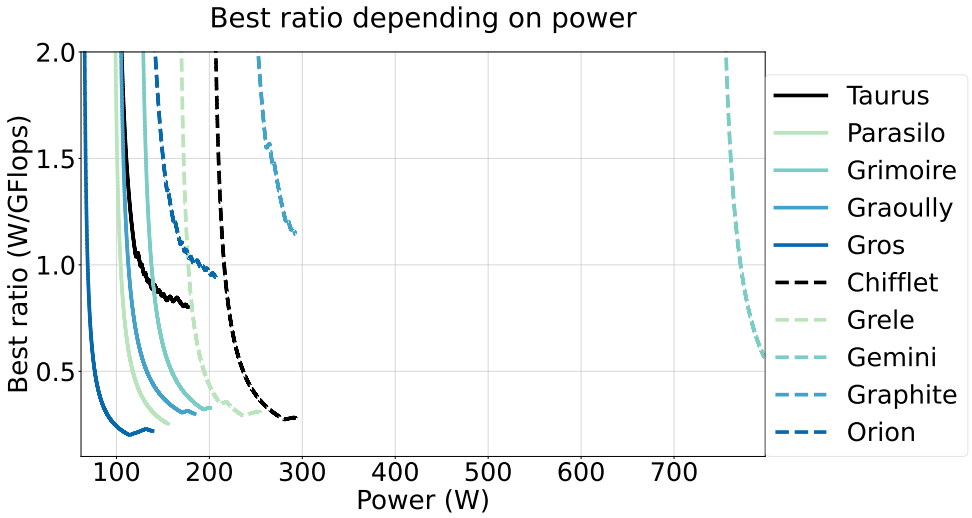

FIGURE 2 – Meilleur ratio de performance en W/GFlops en fonction du type de machine et de la puissance.

D. Table 1 : Ecarts relatifs moyen et médian par rapport à l'optimal

TABLE 1 – Temps de calcul moyen, écart relatif moyen et écart relatif médian des heuristiques par rapport à l'optimal.

	MILP	BPP	BSRWS	BSRWS-AR
Ecart relatif moyen (%)	-	0.12	0.65	**0.03**
Ecart relatif median. (%)	-	0.04	0.09	**0.00**
Temps de calcul moyen (s)	2.83	9.07×10^{-3}	$\mathbf{1.03 \times 10^{-3}}$	1.61

E. Figure 3 : Temps d'exécution du MILP et des heuristiques

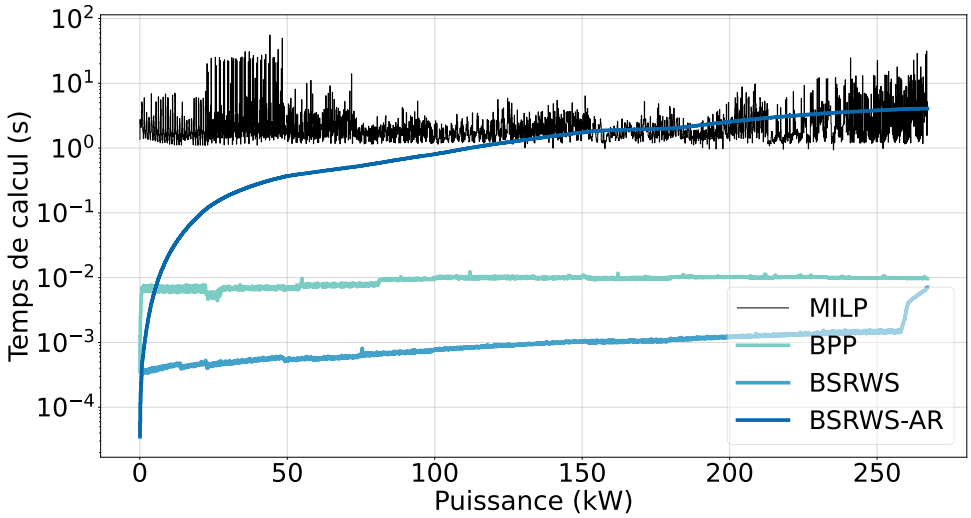

FIGURE 3 – Temps de calcul du MILP et des heuristiques en fonction de la puissance (l'axe des ordonnées utilise une échelle logarithmique).