Stratégie de réplication de données statique pour prendre en compte la consommation énergétique des fournisseurs de Cloud

Morgan Séguéla, Riad Mokadem, Jean-Marc Pierson

Compas'2020 : Parallélisme/ Architecture/ Système / Temps Réel
MILC - Lyon, France, du 30 juin au 3 juillet 2020

Stratégie de réplication de données statique pour prendre en compte la consommation énergétique des fournisseurs de Cloud

Morgan Séguéla, Riad Mokadem, Jean-Marc Pierson

Université Toulouse 3 Paul Sabatier,
Institut de Recherche en Informatique de Toulouse - 118 Route de Narbonne
31062 TOULOUSE CEDEX 9
prénom.nom@irit.fr

Résumé
L'accès et le traitement de gros volume de données est un challenge important, d'autant plus que ces données sont répartis à travers le monde. Dans ce contexte, les entreprises d'analyse de données se tournent vers des solutions tels que les fournisseurs de Cloud, qui offrent des ressources en apparence illimitées avec un faible coût d'investissement. La réplication de données est une technique bien connue qui permet de répondre à de tels problèmes. Cependant, peu de stratégies de réplication de données prennent en compte simultanément la consommation énergétique et le profit du fournisseur. Dans ce papier, nous proposons un placement initial, qui sera par la suite intégré à une stratégie de réplication de données dynamique. Ce placement est nommé 2EARM pour *Energy and Expenditure Aware Replica Management*. D'abord, elle offre la possibilité d'éteindre les noeuds inactifs pour pouvoir diminuer la consommation énergétique. De plus, elle profite de l'hétérogénéité des noeuds entre les centres de données pour réduire les dépenses. cela permet un placement initial de données en prévision d'une réplication dynamique lors du traitement des requetes d'utilisateurs.

Mots-clés : Cloud, Réplication de données, Profit économique, Consommation énergétique

---

1. Introduction

Depuis la démocratisation d'Internet et l'avènement des réseaux sociaux, la production de données ne cesse de croître. Cela est due à l'augmentation continue du nombre de sources de données qui génèrent de plus en plus de volume de données (photos, vidéos, etc.). L'accès à ces données est également en forte croissance, par exemple une augmentation de 11% de personnes actives par jour a été constatée dans l'ensemble des applications fournis par Facebook [12]. Au travers d'applications de plus en plus nombreuses, les utilisateurs produisent des données et accèdent aux données d'autres personnes. Ainsi, certains utilisateurs peuvent être suivis par plusieurs milliers d'autres utilisateurs dans le monde, ce qui génère l'accès simultanée à des données de partout dans le monde. La réplication de données est une technique qui permet de répondre à ces problématiques. Elle permet entre autres une meilleure disponibilité de données, de meilleures performances [9] et d'assurer une tolérance aux fautes [13]. La réplication de données a été largement utilisée dans les systèmes classiques tels que les systèmes de grilles de données [7, 16]. Néanmoins, les stratégies proposées dans de tels système ne s'adaptent

pas aux systèmes Cloud. En effet, d'autres contraintes doivent être prises en compte tel que les contraintes énergétiques et financières en plus de la satisfaction des objectifs des utilisateurs, par exemple les performances. De plus, le fournisseur devant être rentable, ils mettent en place un modèle économique qui s'appuie sur la gestion élastique des ressources [8]. Ce modèle économique est le *Pay-as-you-go model* ce qui signifie que le client paie uniquement ce qu'il consomme. Dans ce contexte, le *Service Level Agreement* [14], un contrat qui lie le fournisseur et le locataire, doit être satisfait. Certaines stratégies s'intéressent à la consommation énergétique et à la bande passante afin d'en réduire la consommation. [2]. D'autres optimisent le rapport qualité/prix de chaque serveur afin de fournir les meilleurs performances avec le prix le plus bas [6].

Peu de stratégies de réplication de données dynamique s'intéressent à leur placement initial. En effet, dans leur description soit les données ont 2 répliques stockées aléatoirement [17], soit ils n'ont pas de réplique et sont stockée sur un seul serveur [2]. L'objectif était donc d'avoir un placement initial pour une stratégie dynamique qui essaie de répondre aux objectifs de consommation énergétique et de profit sans supposer une future charge de travail. Ainsi, s'il y a une augmentation de la charge de travail, ce sera à la partie dynamique d'adapter la réplication et le placement pour répondre à ce changement de contexte. Dans le reste de l'article, nous décrirons ce placement comme une stratégie de réplication de données statique car elle répond aux questions de quoi répliquer ? combien de répliques ? où répliquer ? et quand répliquer ? Dans le reste de ce papier, la stratégie proposée est nommée 2EARM (*Energy and Expenditure Aware Replica Management*).

Le papier s'organise de la manière suivante. Dans un premier temps, nous passons en revue un état de l'art portant sur les principales stratégies de réplication de données qui prennent en compte la consommation énergétique et/ou le profit du fournisseur. Puis, nous présentons notre stratégie de réplication de données que nous comparons, par la suite, à une autre stratégie. Enfin, nous terminerons cet article avec une conclusion et la présentation de futurs travaux.

2. État de l'art

La stratégie proposée dans ce papier a pour objectif de minimiser la consommation énergétique ainsi que la dépense du fournisseur. Cependant, très peu de stratégies s'intéressent à ces deux problématiques simultanément, tendance qui s'inverse peu à peu. Nous allons dans un premier temps s'intéresser aux stratégies qui ont pour objectif de réduire la consommation énergétique. [2] est une stratégie reconnue pour sa modélisation de la consommation énergétique et de la consommation de bande-passante afin de minimiser les deux. Une autre stratégie [18] regroupe la charge de travail sur quelques noeuds afin de pouvoir mettre en veille voire éteindre les noeuds inutilisés. Enfin une dernière stratégie que l'on peut mettre en avant, est [10], qui est une stratégie statique qui cherche à prendre en compte plusieurs objectifs dont la consommation énergétique parmi des problématiques de disponibilité, de temps de réponse, de charge de travail et de latence.

Cependant, les objectifs de consommation énergétique se couple de plus en plus à des objectifs de coûts. Cela se voit avec [1] qui s'intéresse au profit à travers le prisme de la réduction de la consommation énergétique. Une stratégie plus récente [4] cherche à minimiser la consommation énergétique et le coût à l'aide d'un algorithme d'optimisation par essaim. Cependant, contrairement à nous, il prend aussi en considération la possibilité de modifier les données.

Enfin nous finissons cet état de l'art avec les stratégies qui s'intéressent uniquement à la dépense. Avec dans un premier temps, [5] qui prend en compte la disponibilité des fichiers ainsi que l'hétérogénéité des prix et des capacités des serveurs de stockage. Un trait intéressant de

cette stratégie est son objectif de minimiser l'impact de son algorithme sur la consommation énergétique et la charge des serveurs. Une seconde stratégie qui s'intéresse à l'hétérogénéité des prix et des performances est [6] qui de son côté considère considères 3 couches de centres de données, avec au centre les plus chers et performants, et au bords les moins chers et moins performant. Son objectif est donc de minimiser les coûts tout en gardant un minimum de qualité de service. Enfin, [11] cherche à minimiser le coût stockage et de lecture de données chaudes et froides, en décidant de répliquer ou de migrer la donnée dans des centre de données moins chères en coût de stockage ou de lecture.

3. Stratégie statique de Réplication de données

Nous proposons ici une nouvelle stratégie de réplication de données statique, pour prendre en compte la consommation énergétique et le profit. Ainsi, dans le cadre d'un problème Multi-Objectif, nous avons mis en place un modèle par objectif. Nous allons avant tout, définir la notation, puis exposer les modèles pour chaque fonction objectif avec leurs contraintes, puis nous présenterons de façon globale la stratégie.

3.1. Notation

Nous allons dans un premier temps, définir la notation pour les modèles suivants. Nous avons donc les fichiers $f_1...f_i...f_n$ qui peuvent être stockés sur les noeuds $N_1...N_j...N_m$ qui sont des noeuds de calcul avec du stockage. Nous supposons ici que tous les noeuds peuvent stocker des données. La notation permettant d'exprimer la présence ou non d'une données sur un noeud est $\phi(i,j)$, qui est égal à 1 si le fichier i est présent sur le noeud j, et 0 sinon. Enfin, certaines variables impactent la prise en compte de certains critères, comme le temps de location (t), où encore le nombre de lectures de l'ensemble des fichiers ($nbRead$), qui seront les mêmes pour les modèles de consommation énergétique et de dépense.

3.2. Définition des fonctions objectifs

Dans cet article, nous prenons en compte deux objectifs distincts, la réduction de la consommation énergétique et la réduction de la dépense afin de maximiser le profit. Ces deux objectifs ne sont pas complètement corrélés, car dans les dépenses, la consommation énergétique n'est qu'une partie des coûts pour le fournisseur. Nous allons dans un premier temps nous intéresser à la consommation énergétique.

Consommation énergétique pour le stockage : Dans cette modélisation, nous avons considéré qu'il était plus efficace d'un point de vue énergétique de consolider les données dans peu de noeuds dans un centre de données afin de pouvoir éteindre les autres noeuds si la technologie du centre le permet. Nous avons donc :

$$EC_{Stock}(i,j,t) = (Pw_{Node}(j,i) - Pw_{Node}(j,0)) * t \quad (1)$$

Qui correspond d'un coté à la consommation statique du noeud j si le fichier i est stocké dessus, et de l'autre la consommation statique du noeud j si la donnée n'y est pas stockée. Ce qui signifie que si aucune donnée n'est stockée dessus, $Pw_{Node}(j,0) = 0$, donc cela correspond à la consommation énergétique liée au fait de garder allumé un autre noeud pour le stockage. A l'inverse, si une donnée est déjà stockée sur le noeud j, alors $Pw_{Node}(j,0) = Pw_{Node}(j,i)$, ce qui n'aura pas d'impact sur la consommation pour le stockage.

Consommation énergétique pour l'écriture : Ici, nous allons modéliser la consommation énergétique entre le noeud d'origine (j_o) de la donnée (i) et le noeud (j) choisi pour la stocker.

$$EC_{Write}(i,j,j_o) = \phi(i,j) * s(i) * [\frac{Pw_{Network}(j_o,j)}{Bw_{Network}(j_o,j)} + \frac{Pw_{Write}(j)}{Bw_{Write}(j)}] \quad (2)$$

Consommation énergétique pour la lecture : La lecture du fichier sera la principale source de consommation énergétique, nous avons donc estimé la consommation moyenne par lecture de fichier, pour l'ensemble des noeuds.

$$EC^{EC}_{Rd}(i,j) = \frac{1}{m}\sum_{j'=1}^{m} s(i) * [\frac{Pw_{Network}(j',j'')}{Bw_{Network}(j',j'')} + \frac{Pw_{Read}(j'')}{Bw_{Read}(j'')}] \quad (3)$$

$$avec\{j'' \in j, \max_{j''}(Bw_{Network}(j',j'') * \phi(i,j'')\} \quad (4)$$

Nous obtenons ainsi la fonction objectif pour la consommation énergétique suivante :

$$ECG(i,j,j_0,t,nbRead) = EC_{Stock}(i,j,t) + nbRead * EC_{Rd}(i,j) + EC_{Wr}(i,j,j_O) \quad (5)$$

Dans un second temps, nous nous intéressons aux dépenses liées au placement, mais aussi au nombre de répliques.
Dépense pour le stockage : La dépense pour le stockage du fichier i sur le noeud j correspond au modèle suivant :

$$Exp_{Stock}(i,j,t) = \phi(i,j) * s(i) * Pr_{stock}(j) * t \quad (6)$$

Dépense d'écriture : Comme pour la consommation énergétique, nous estimons le prix de transfert entre le noeud d'origine du fichier, et le noeud sélectionné.

$$Exp_{Wr}(i,j,j_o) = \phi(i,j) * s(i) * Price_{Net}(j,j_o) \quad (7)$$

Dépense de lecture : Enfin, comme pour la consommation énergétique, nous estimons le prix moyen de lecture d'un fichier entre tout les noeuds. Cela nous donne le modèle suivant :

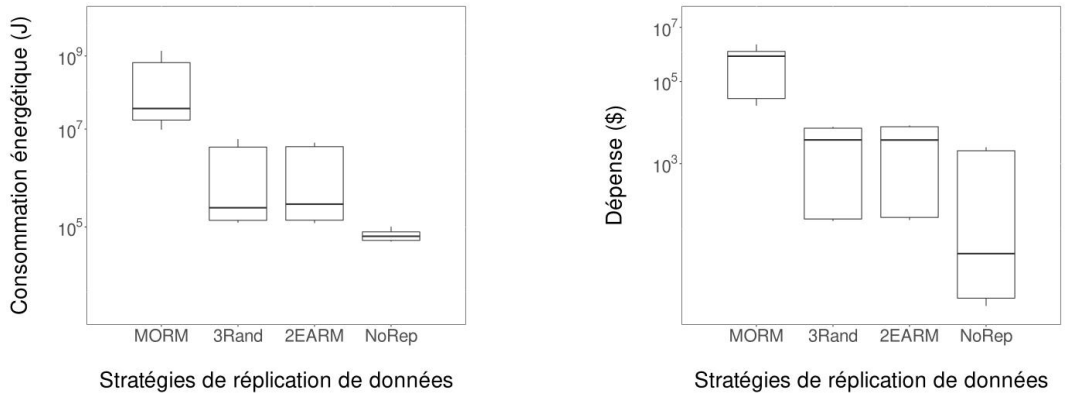
$$Exp_{Rd}(i,j) = \sum_{j'=1}^{m} \phi(i,j') * s(i) * Pr_{Net}(j',j'') \quad (8)$$

$$avec\{j'' \in j, \max_{j''}(Bw_{Network}(j',j'') * \phi(i,j'')\} \quad (9)$$

La seconde fonction objectif pour la dépense est donc la suivante :

$$ExpG(i,j,j_o,t,nbRead) = Exp_{Stock}(i,j,t) + nbRead * Exp_{Rd}(i,j) + Exp_{Wr}(i,j,j_o) \quad (10)$$

Pour la suite de l'article nous fixerons le temps de location à 1 mois et le nombre de lecture à 100.



(a) Consommation énergétique (b) Dépense

FIGURE 1 – Résultats des fonctions objectifs

3.3. Présentation de la stratégie

La stratégie de réplication proposée se divise en deux parties. La première partie correspond au nombre de répliques et au choix du centre de données où seront stockées les répliques, ces décisions se font à travers l'algorithme SPEA2. Ce choix est le plus important, car les centres de données sont plus hétérogènes entre eux que les noeuds au sein d'un même centre de données. Dans cette partie, nous avons imposé la contrainte d'au moins 2 répliques pour chaque donnée. Ainsi, dans une seconde partie nous avons mis en place un First Fit pour le placement des répliques car il permet de répondre à la problématique de consolidation des noeuds. C'est à dire que le stockage se fera sur le minimum de noeuds possibles afin de pouvoir éteindre les autres si la technologie est présente dans le centre de données.
Dans la suite l'article, cette stratégie sera nommée 2EARM, pour *Energy and Expenditure Aware Replica Management*.

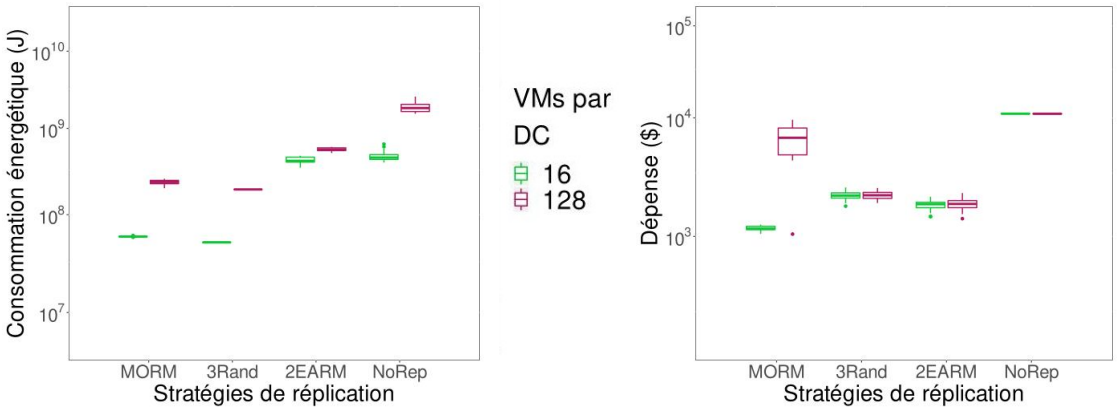
4. Expérience

4.1. Mise en place

Afin d'étudier l'efficacité de la stratégie proposée, nous l'avons comparée à la stratégie MORM [10], qui est une stratégie statique qui s'intéresse aux objectifs de consommation énergétique et de performance entre autres. Deux autres méthodes ont été utilisées pour comparer les résultats : l'absence de réplication et le placement aléatoire de 3 répliques.
Pour comparer ces stratégies, nous avons mis en place des expériences sur le simulateur CloudSim [3]. Ce dernier ayant été étendu pour la thèse de Uras Tos [15] afin de prendre en compte une architecture à large echelle, la réplication de données et le coût monétaire des ressources. Il a ensuite été étendu de nouveau pour prendre en compte la consommation énergétique, et l'hétérogénéité entre les centres de données. Nous avons ainsi comparé les stratégies en se basant sur les paramètres présentés dans la Table 1 en annexe. Cependant, nous avons trouvé intéressant de faire l'expérience avec d'un coté 16 noeuds par centre de données, et de l'autre 128 noeuds par centre de données.

4.2. Résultats

Dans un premier temps, nous avons comparé les différentes stratégies de réplication de données du point de vue de la fonction objectif (Figure 1). Pour cette première comparaison, nous nous sommes concentrés ici sur le cas de 128 noeuds par centre de données, car la différence



(a) Consommation énergétique (b) Dépense

FIGURE 2 – Résultats des exécutions

entre les deux étaient minimes. Cette fonction objectif à l'avantage de prendre en compte un grande temporalité (1 mois) et la possibilité d'éteindre les noeuds. Cependant, il ne prend pas en compte une grande quantité de requêtes. L'idée est donc de voir l'impact des différentes stratégies de réplication de données dans un contexte avec peu d'activités. Nous voyons ainsi que lorsqu'il y a peu d'activités, MORM consomme le plus d'énergie et coûte le plus cher, alors qu'à l'inverse, l'absence de réplication est ce qui coûte et consomme le moins. Cela s'explique par le nombre de répliques créées, car plus il y a de répliques plus cela ĉoûtera cher et aura tendance à consommer plus. Entre ces deux stratégies, nous trouvons les stratégies 3Rand et 2EARM. Du fait que, 2EARM ait créé légèrement plus de répliques, il coûte en moyenne un peu plus cher que 3Rand, mais il permet de consommer légèrement moins.
Dans un second temps, nous avons comparé ces stratégies sur le simulateur CloudSim dans le but de comprendre le comportement de chaque stratégie (Figure 2). Cette expérience à l'avantage de prendre en compte les problématiques de performance, et de mettre en avant des problématiques liées au réseau. Nous voyons ainsi que lorsqu'il y a peu de noeuds MORM est la stratégie la moins chère et consomme peu d'énergie. De même, nous voyons que 3Rand est celui que qui consomme le moins d'énergie mais il reste plus cher que 2EARM. Cette dernière consomme presque autant d'énergie que l'absence de réplication. Cela s'explique par le fait que les données sont concentrées sur peu de noeuds, ce qui entraîne un goulot d'étranglement pour accéder aux données, et cela entraîne une augmentation de la consommation de la carte réseau. Cependant, quand on augmente le nombres de noeuds par centre de données, nous voyons que la stratégie 2EARM se rapproche de la consommation énergétique des stratégies les moins énergivore. De plus, du point de vue de la dépense, MORM devient la stratégie la plus chère, alors que 2EARM devient la moins chère. Enfin, à l'aide de la simulation, nous pouvons voir l'impact de la stratégie sur les performances, notamment du point de vu du nombre de violations (Tableau 2). Nous voyons d'abord avec ce tableau en annexe, que MORM a tendance à créer énormément de répliques, ce qui explique le fait que cette stratégie est la plus chère. De plus, nous voyons qu'avec plus de 20 fois plus de répliques que 2EARM, ce dernier a seulement 3 à 4 fois plus de violations.

5. Conclusion et perspectives

Dans ce papier, Nous avons proposé un placement initial de répliques et de données dans le cadre d'une stratégie dynamique de réplication de données. Nous pouvons la considérer comme une stratégie de réplication de données statique car elle répond à des problématique de consommation énergétique et de baisse des dépense en utilisant la réplication. Cette stratégie de réplication de données statique, nommée 2EARM (*Energy and Expenditure Aware Replica Management*), profite bien de la possibilité d'éteindre les noeuds inutilisés, et permet de diminuer les coûts lorsqu'il y a beaucoup de noeuds. Nous voyons cependant, que lors des simulations la stratégie proposée consomme plus que les autres mais réduit la dépense pour le fournisseur. Ce placement initial sera inclus dans une stratégie de réplication de données dynamique, qui en plus des considération énergétique et économique s'adaptera à des problématiques de performance pour minimiser le nombre de violations.

Bibliographie

1. Alghamdi (M.), Tang (B.) et Chen (Y.). – Profit-based file replication in data intensive cloud data centers. – In *2017 IEEE International Conference on Communications (ICC)*, pp. 1–7, Paris, France, mai 2017. IEEE.
2. Boru (D.), Kliazovich (D.), Granelli (F.), Bouvry (P.) et Zomaya (A. Y.). – Energy-efficient data replication in cloud computing datacenters. *Cluster Computing*, vol. 18, n1, mars 2015, pp. 385–402.
3. Calheiros (R. N.), Ranjan (R.), Beloglazov (A.), De Rose (C. A. F.) et Buyya (R.). – CloudSim : a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software : Practice and Experience*, vol. 41, n1, janvier 2011, pp. 23–50.
4. Ebadi (Y.) et Navimipour (N. J.). – An energy-aware method for data replication in the cloud environments using a Tabu search and particle swarm optimization algorithm. *Concurrency and Computation : Practice and Experience*, vol. 31, n1, janvier 2019, p. e4757.
5. Edwin (E. B.), Umamaheswari (P.) et Thanka (M. R.). – An efficient and improved multi-objective optimized replication management with dynamic and cost aware strategies in cloud computing data center. *Cluster Computing*, vol. 22, n5, septembre 2019, pp. 11119–11128.
6. Gill (N. K.) et Singh (S.). – A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers. *Future Generation Computer Systems*, vol. 65, décembre 2016, pp. 10–32.
7. Hamrouni (T.), Slimani (S.) et Charrada (F. B.). – A survey of dynamic replication and replica selection strategies based on data mining techniques in data grids. *Engineering Applications of Artificial Intelligence*, vol. 48, février 2016, pp. 140–158.
8. Herbst (N. R.), Kounev (S.) et Reussner (R.). – Elasticity in Cloud Computing : What It Is, and What It Is Not. *ICAC*, vol. 13, 2013, pp. 23–27.
9. Kumar (K. A.), Quamar (A.), Deshpande (A.) et Khuller (S.). – SWORD : workload-aware data placement and replica selection for cloud data management systems. *The VLDB Journal*, vol. 23, n6, décembre 2014, pp. 845–870.
10. Long (S.-Q.), Zhao (Y.-L.) et Chen (W.). – MORM : A Multi-objective Optimized Replication Management strategy for cloud storage cluster. *Journal of Systems Architecture*, vol. 60, n2, février 2014, pp. 234–244.
11. Mansouri (Y.) et Buyya (R.). – Dynamic replication and migration of data objects with hot-

spot and cold-spot statuses across storage data centers. *Journal of Parallel and Distributed Computing*, vol. 126, avril 2019, pp. 121–133.

12. Park (M.). – *Facebook Reports Fourth Quarter and Full Year 2019 Results*. – Rapport technique, janvier 2020.
13. Selvi (S. A. E.) et Anbuselvi (R.). – RAAES : Reliability-Assured and Availability-Enhanced Storage for Cloud Environment. *International Journal of Pure and Applied Mathematics*, vol. 118, février 2018, pp. 103–112.
14. Serrano (D.), Bouchenak (S.), Kouki (Y.), de Oliveira Jr. (F. A.), Ledoux (T.), Lejeune (J.), Sopena (J.), Arantes (L.) et Sens (P.). – SLA guarantees for cloud services. *Future Generation Computer Systems*, vol. 54, janvier 2016, pp. 233–246.
15. Tos (U.). – *Data replication in large-scale data management systems*. – Theses, Université Paul Sabatier - Toulouse III, juin 2017.
16. Tos (U.), Mokadem (R.), Hameurlain (A.), Ayav (T.) et Bora (S.). – Dynamic replication strategies in data grid systems : a survey. *The Journal of Supercomputing*, vol. 71, n11, novembre 2015, pp. 4116–4140.
17. Tos (U.), Mokadem (R.), Hameurlain (A.), Ayav (T.) et Bora (S.). – Ensuring performance and provider profit through data replication in cloud systems. *Cluster Computing*, décembre 2017.
18. Zhang (L.), Deng (Y.), Zhu (W.), Zhou (J.) et Wang (F.). – Skewly replicating hot data to construct a power-efficient storage cluster. *Journal of Network and Computer Applications*, vol. 50, avril 2015, pp. 168–179.

Paramètres	Valeurs	Paramètres	Valeurs
Nombre de fichiers	30	Délai entre les région	160 ms
Taille moyenne des fichiers	5600 MB	Délai dans une région	30 ms
Nombre de tâches	55400	Nb de VMs par CD	16 - 128
Taille minimale de la tâche	1,000 MI	Nb de CDs par région	4
Taille maximale de la tâche	7,500 MI	Nb de régions par Cloud	3
Durée de la simulation	1h30	Revenu par tâche	0.0205$
Capacité de calcul d'une VM	1,600 MIPS	Coût du calcul d'une tâche	3-4$*$10^{-9}$ MI
BP entre les région	100 Gbit/s	Coût de stockage	2-4*$10^{-2}$$/GB
BP dans une région	10 Gbit/s	Coût de transfert entre régions	0.014$/GB
BP dans un centre	10 Gbit/s	Coût de transfert d'une région	0.0078$/GB
Délai dans un CD	1 µs	Coût de transfert dans un CD	7.8*$10^{-4}$$/GB
Objectif de temps de réponse	15s	Pénalité par violation de SLO	0.0016$

TABLE 1 – Annexe - Paramètres de la simulation

Stratégie	Nombre de VMs	Nombre de répliques	Nombre de violations
NoRep	16	0	53257
	128	0	49269
3Rand	16	60	7665
	128	60	1309
2EARM	16	80	6867
	128	82	302
MORM	16	3955	1775
	128	22283	105

TABLE 2 – Annexe - Nombre de réplications et de violations