Effet du placement de données

sur les coûts de communication dans les grilles

Cherif Haddad

Département Informatique
Faculté des Sciences de Tunis
1060, Tunis
Tunisie
cherif.haddad@gmail.com

RÉSUMÉ. La méthode avec laquelle les bases de données sont placées affecte sensiblement les performances d'un Système de Gestion de Bases de Données Distribuées dans un système à large échelle. En effet, une faible distribution des données peut générer un flux important de données qui risque de saturer le réseau entre sites et aussi diminuer les performances globales d'un SGBDD. Dans cet article, nous étudions l'effet du placement de BD sur les coûts de communication dans les grilles de calcul. Nous proposons une approche de placement de bases de données basée sur un modèle économique qui tient compte de la fragmentation et de la réplication de données. L'effet du placement de données est évalué avec des simulations en utilisant le simulateur de grilles, OptorSim.

ABSTRACT. Database placement strategies are a key issue because appropriate placement of database fragments reduces bandwidth consumption and improves response time. In fact, the scheduling of database placement across a Grid is a critical aspect, since the access to data has the potential to become the main bottleneck for data intensive applications. In this paper, we address the problem of database placement in Grid architecture. For this purpose, we propose a database placement approach based on an economic model. We use the communication cost as a main criterion to evaluate a database placement. We experiment our proposed cost model via simulations using OptorSim simulator.

MOTS-CLÉS : Grille de calcul, Bases de données distribuées, Placement de bases de données, Modèle économique, Coût de communication.

KEYWORDS : Grid computing, Distributed databases, Database placement, Economic model, Communication cost.

1. Introduction

L'un des principaux apports des architectures à large échelle est d'avoir découplé les calculs des infrastructures nécessaires pour prendre en charge ces calculs. Paradoxalement, on dispose d'infrastructures complexes permettant d'ordonnancer, de manière transparente, des calculs répartis à large échelle, alors que le stockage et le transfert de données nécessaires aux applications sont laissés à la charge de l'utilisateur. Dans le meilleur des cas, des fonctionnalités peu évoluées de type transfert de fichiers sont proposées, comme par exemple le protocole GridFTP [2] du middleware GLOBUS. Or, les applications modernes (fouilles de données, bioinformatique, algorithmique génétique, etc.) utilisent de grandes masses de données réparties à une très large échelle, dont la gestion est très complexe.

Les Systèmes de Gestion de Bases de Données Distribués (SGBDD) offrent des facilités importantes pour les architectures à large échelle [3]. Ces SGBDD exploitent les ressources disponibles pour intégrer, partager et traiter des volumes importants de données, qui sont à la fois distribuées et hétérogènes. L'apport majeur des ces SGBDD a été de permettre la manipulation des données par des requêtes, exprimées dans un langage de haut niveau (SQL) [4], qui sont ensuite traduites automatiquement en plans d'exécution optimisés. Cette optimisation permettra de traiter et de transférer des données d'une manière optimale, selon une certaine fonction de coût. Cette optimalité ne peut être obtenue que par l'utilisation d'une technique de placement des données assurant la fragmentation et la réplication des données. Ceci permet d'augmenter le degré de parallélisme, notamment lors de l'exécution de requêtes, et de minimiser le coût de transfert des données.

Dans cet article, nous étudions l'effet du placement de données sur les coûts de communication dans les systèmes à large échelle de type grille de calcul. Nous proposons une approche de placement de données basée sur un modèle économique. Ce choix est motivé par le fait qu'une grille est composée de plusieurs organisations virtuelles qui sont indépendantes les unes par rapport aux autres [5]. Etant donné que ces organisations disposent de politiques de gestion de ressources qui leurs sont propres, le modèle économique est plus approprié car il permet d'intégrer, dans l'approche de placement, les caractéristiques propres à chaque organisation qui fait partie d'une grille.

Le reste de cet article est structuré comme suit : la section 2 présente les nouveaux challenges de placement des bases de données dans les grilles de calcul. La section 3 décrit l'architecture de la grille que nous avons utilisé pour la définition de notre approche de placement. La section 4 présente le modèle économique que nous avons défini pour résoudre le problème de placement de bases de données dans les grilles de calcul. La section 5 présente et analyse les résultats d'une série d'expérimentations de notre approche. Finalement la section 6 conclut cet article et donne quelques orientations pour des recherches futures.

2. Placement de bases données dans les grilles

Le domaine des bases de données distribuées a connu de grandes mutations ces dernières années, sous l'influence de l'évolution des architectures à large échelle, d'une part, par l'augmentation des capacités de calcul et de stockage, et d'autre part par les progrès en matière de technologie réseaux. Parallèlement à cette évolution, de nouvelles applications ont émergé et ont créé de nouveaux besoins en terme de gestion des données [3]. Ceci a mis en évidence les limites des approches de réplication et de placement de données

utilisées dans le cadre des systèmes distribués classiques.
Le placement de bases de données est rendu difficile par l'hétérogénéité des plate-formes, le caractère dynamique des ressources et la grande échelle [6]. Toutes ces spécificités font que le placement de bases de données dans une grille doit être capable de réagir aux éléments suivants :

– *Changement de l'environnement* : Stabilité des sites et disponibilité des ressources ;

– *Domaines administratifs multiples* : Les ressources d'une grille sont propriétés de plusieurs domaines administratifs, qui utilisent différentes stratégies de gestion et de sécurité qui peuvent être parfois contradictoires ;

– *Grande échelle* : Une grille est constituée par un grand nombre de sites qui varie dans le temps.
En raison de ces challenges, le placement de bases de données dans les grilles de calcul devient un problème complexe [7]. Cette complexité rend les stratégies traditionnelles de placement de bases de données inutilisables ou inadaptées dans le cas des grilles de calcul. Ces stratégies tentent d'optimiser des paramètres globaux tels que le temps de réponse, en utilisant des algorithmes centralisés ou par un consensus décentralisé [8]. Ce type d'approche est difficilement applicable pour les grilles dans lesquelles les ressources sont la propriété d'organismes multiples.

3. Description de l'architecture d'une grille

Dan cette section, nous décrivons l'architecture de la grille que nous avons utilisé pour définir notre approche de placement, puis nous présenterons les paramètres considérés dans le cadre du placement de bases de données.

3.1. Architecture

Nous modélisons une grille comme un ensemble de sites, chacun comportant un nombre différent d'*éléments de calcul* (CEs) et d'*éléments de stockage* (SEs), un ensemble d'utilisateurs - chacun associé à un site - et un ensemble de fragments répliqués. Les sites sont reliés entre eux par des réseaux étendus (WAN) avec des bandes passantes limitées, tandis que les éléments de calcul et de stockage, dans un même site, sont reliés à travers un réseau local. Les sites d'une grille décident s'ils doivent placer ou non de nouvelles répliques, d'un fragment donné d'une base de données, en utilisant une approche dite *approche de placement*.

3.2. Paramètres de placement

Les paramètres de placement considérés dans notre approche sont définis comme suit :

1) *Paramètres des éléments de stockage* : Les éléments de stockage fournissent aux utilisateurs d'une grille des capacités de stockage pour leurs données. Chaque élément de stockage, noté SE_i, est situé dans un site particulier, noté $Site(SE_i)$, de la grille. La capacité de stockage ainsi que la disponibilité de ces éléments de stockage changent selon les politiques locales de chaque site. Pour chaque élément de stockage SE_i, $SR(SE_i)$ représente sa capacité de stockage, $DBW(SE_i)$ sa bande passante disque et $Stab(SE_i) \in [0, 1]$ son taux de disponibilité défini comme étant la probabilité qu'il soit en ligne ;

2) *Paramètres de réseau* : Nous définissons $CC(GS_a, GS_b)$ comme étant le coût de communication entre deux sites GS_a et GS_b. Ce coût représente le délai moyen d'envoi d'une unité de données (1KB) de GS_a vers GS_b ;

3) *Paramètres des bases de données* : Nous considérons une base de données DB comme étant une collection de m fragments répliqués, $\{F_1, F_2, ..., F_m\}$. F_k^l représente la réplique l du fragment F_k et $||F_k^l||$ définit sa taille ;

4) *Paramètres de requêtes* : Des requêtes d'accès aux bases de données sont soumises par les utilisateurs de la grille. Nous avons considéré les requêtes d'écriture et de lecture. Chaque requête q_z peut être soumise de n'importe quel site avec des fréquences différentes.

4. Approche de placement

Dans cette section, nous proposons un modèle économique qui vise à resoudre le problème de placement de bases de données dans les grilles de calcul. Dans ce modèle, les fragments de bases de données représentent les biens du marché et sont demandés par *les courtiers de ressources* sur chaque site, selon les besoins des requêtes. L'adoption d'un tel modèle a deux motivations principales. La première est de pouvoir prendre des décisions de placement d'une manière distribuée. La seconde est que la grille est un environnement dynamique dans lequel la disponibilité des ressources peut changer à tout instant et ce de manière imprévisible.

4.1. Agents du modèle économique

Dans notre modèle économique, nous utilisons deux types d'agents : les *Fournisseurs de Données* (FDs) jouant le rôle de *fournisseurs* et les *Courtiers de Ressources* (CRs) représentant les *consommateurs*. Les utilisateurs interagissent avec leurs propres *courtiers de ressources* dans le même site pour ordonnancer leurs requêtes sur la grille. Les *fournisseurs de données* rendent leurs ressources (fragments de bases de données) disponibles pour l'exécution de requêtes des utilisateurs contrôlées par les *courtiers de ressources*. La négociation entre les *fournisseurs de données* et les *courtiers de ressources* est gérée par un protocole d'interaction.
Les *fournisseurs de données* et les *courtiers de ressources* ont leurs propres fonctions d'utilité qui doivent être satisfaites et maximisées. Les *courtiers de ressources* effectuent une analyse par coût pour choisir le *fournisseur de données* approprié ayant des fragments de base de données qui répondent aux exigences des requêtes des utilisateurs, tandis que les *fournisseurs de données* utilisent une fonction d'utilité pour augmenter les qualités de service du système.

4.2. Protocole d'interaction

Le but du protocole d'interaction est de choisir la *meilleure* réplique d'un fragment requis par une requête. Pour ce faire, nous avons utilisé une forme renversée de vente aux enchères (reverse auctioning) pour prendre les décisions concernant les coûts des fragments (représentant les coûts de transfert). Ceci signifie que chaque *fournisseur de données* offre le fragment au prix demandé s'il peut le faire, étant donné le coût de transfert d'un fragment vers le site qui en a fait la demande.
Dans notre cas, le rôle du commissaire-priseur (auctioneer) est joué par le *courtier de ressources* tandis que les *fournisseurs de données* jouent le rôle des soumissionnaires

(bidders). Le courtier de ressources, gérant une requête q_z, reçoit un plan d'exécution contenant un ensemble de sous-requêtes, $q_z^1, ..., q_z^n$. Le choix d'une réplique d'un fragment est effectué dynamiquement pendant l'exécution d'une requête par un mécanisme d'enchère. La réplique choisie est celle qui réduit au minimum le coût d'accès. Celui-ci représente le coût de communication produit par une requête, quand nous utilisons une réplique particulière d'un fragment F_k. Ceci inclut le coût d'envoi des fragments réduits aux emplacements où les opérations de la requête auront lieu et le coût d'envoi des résultats à l'utilisateur.
Dans le cas où le fragment n'est pas stocké localement, le *courtier de ressources* pourrait déclencher une *enchère récursive* pour créer une réplique locale. Cette *enchère récursive* sera initialisée sous certaines conditions : (i) il faut tout d'abord que le *courtier de ressources* soit autorisé à répliquer le fragment ; et, (ii) il faut que le *courtier de ressources* décide qu'avoir une réplique locale du fragment est bénéfique économiquement pour les enchères futures. Pour étudier la valeur (au sens économique du terme) future d'un fragment, nous utilisons une fonction de prévision pour estimer les avantages économiques futures d'un fragment.

4.3. Valeur économique future des fragments

Afin d'étudier la nécessité d'une enchère récursive, nous utilisons une fonction de prévision pour estimer la valeur économique future des fragments. Cette fonction estime la valeur économique future d'un fragment comme étant le nombre de ses demandes futures qui seront reçues par les *courtiers de ressources* durant une période donnée. La fonction de prévision exploite des informations sur les anciennes demandes de fragment (*modes d'accès*) reçues par le *fournisseur de données*.
Nous définissons une fonction pour une demande future d'un fragment F_k durant une période t, $FR(F_k, t)$, faite par un *fournisseur de données* particulier. Cette demande future d'un fragment est donnée par la somme pondérée de ses demandes, dans le passé, reçues par le *fournisseur de données*.

$$FR(F_k, t) = \sum_{p=1}^{t} c_{t-p}.FR(F_k, t-p) + c_t$$

Les coefficients c_t sont utilisés pour ajuster graduellement la mesure de demandes futures. Cet ajustement permet d'améliorer progressivement les estimations. Ainsi, à chaque estimation, nous comparons la valeur estimée, donnée par la fonction $FR(F_k, t)$, à la valeur réelle des demandes. Si nous remarquons des différences significatives, nous ajustons et enregistrons les nouvelles valeurs des coefficients c_t.

4.4. Placement des fragments

L'objectif principal du placement d'une nouvelle réplique de fragment dans un site d'une grille est de servir les demandes des requêtes provenant de ce même site. Etant donné que la stabilité des éléments de stockage est très importante dans les grilles, les éléments de stockage, ayant une haute stabilité, seront sélectionnés en priorité pour le placement des nouvelles répliques des fragments. S'il y a beaucoup d'éléments de stockage avec le même taux de stabilité, nous choisissons ceux qui disposent d'une large bande passante disque et d'un grand espace de stockage disponible. Le placement d'une nouvelle réplique d'un fragment, dans un site d'une grille, est guidé par l'algorithme 1 décrit ci-dessous. Le résultat de cet algorithme est un ensemble P représentant le placement

proposé, formé par le couple (F_k, SE_i), où F_k est le fragment à placer et SE_i désigne l'élément où il sera stocké.

Algorithme 1 PLACEMENT D'UNE RÉPLIQUE D'UN FRAGMENT DANS UN SITE

Entrée: F_k : Fragment ; GS_a : Site
Sortie: Placement P_{F_k,GS_a}
1: $P_{F_k,GS_a} = \emptyset$
2: Générer une liste $L_{GS_a} = \{SE_i, SE_i \in GS_a\}$ d'éléments de stockage du site GS_a
3: Trier L_{GS_a} selon l'ordre décroissant de $(STAB(SE_i), DBW(SE_i), SR(SE_i))$
4: $SE_i = PremierElem(L_{GS_a})$
5: **Tantque** (F_k n'est pas placé) **Faire**
6: **Si** $SR(SE_i) >= ||F_k||$ **Alors**
7: /* Placer F_k */
8: $P_{F_k,GS_a} = P_{F_k,GS_a} \cup \{(F_k, SE_i)\}$
9: $SR(SE_i) = SR(SE_i) - ||F_k||$
10: **Sinon**
11: $SE_i = ElemSuivant(L_{GS_a})$
12: **Fin Si**
13: **Fin Tantque**
Résultat : P_{F_k,GS_a}

La complexité de l'algorithme du placement d'une réplique d'un fragment dans un site d'une grille est de l'ordre de $O(slog_2(s))$, où s est le nombre moyen d'éléments de stockage du site.

5. Etude expérimentale

Dans cette section, nous allons présenter et analyser les résultats d'expérimentation de l'approche de placement proposée dans cet article. Tout d'abord, nous commençons par présenter l'objectif des simulations, puis nous analyserons les résultats que nous avons obtenus.

5.1. Objectif

L'objectif des simulations effectuées est d'évaluer la qualité de la distribution de données et l'effet de l'utilisation de notre modèle économique sur les coûts de communication. Pour chaque simulation, nous avons calculé le coût moyen de communication engendré par l'exécution d'un ensemble de requêtes. Celui-ci comprend le coût de transfert de fragments et le coût d'envoi des résultats.

5.2. Résultats

Nous avons effectué un ensemble d'expériences pour évaluer les placements proposés par notre modèle économique. Pour réaliser ces expériences, nous avons utilisé le simulateur de grille de données OptorSim [1], qui est largement utilisé dans le domaine des grilles de données. Pour évaluer notre approche de placement, nous l'avons comparé à l'approche *Best Client* proposée dans [9]. Dans cette approche, chaque site de la grille maintient un historique détaillé pour chaque fragment qu'il contient, indiquant le nombre de demandes de ce fragment ainsi que les sites demandeurs. L'approche fonc-

tionne comme suit : à un intervalle de temps donné, chaque site vérifie si le nombre de demandes d'un fragment quelconque a excédé un seuil prédéfini. Si oui, le *Best Client*, pour ce fragment, est identifié comme étant le site le plus demandeur. Puis, une réplique de ce fragment sera créée sur ce site.
Pour chaque expérience, nous avons supposé un nombre donné de sites, de fragments et de requêtes. Les bases de données sont fragmentées verticalement. Le nombre initial de répliques par fragment est de un. Les requêtes sont soumises à partir de n'importe quel site d'une grille. Le transfert de données, d'un site vers un autre, engendre un coût de communication correspondant à la taille des données divisée par la largeur de la bande passante du lien du réseau de communication utilisé. Afin de simuler un réseau réel d'interconnexion d'une grille, nous avons utilisé différentes largeurs de bandes passantes (s'étendant de 100MB/s à 1GB/s) et différents coûts d'initialisation ou *Startup*(allant de 6 à 10μs).
Pour générer les requêtes, nous avons mis en oeuvre un algorithme de génération automatique. Tout d'abord, nous générons aléatoirement les graphes de connexion des prédicats pour chaque requête. La cardinalité de chaque relation, utilisée dans une requête, ainsi que la sélectivité sont choisies aléatoirement selon une distribution uniforme. Nous avons produit 100 requêtes, portant chacune sur 3 à 10 relations. Toutes les opérations d'une requête peuvent être exécutées sur tous les éléments de calcul. Pour ne pas avoir des exécutions très différentes des requêtes et pour tirer des conclusions pertinentes à propos des placements de données proposés, nous avons produit des plans d'exécution pour les requêtes générées.

Nous avons réalisé toute une série d'expériences avec des nombres différents de requêtes (allant de 20 à 100) et avec différentes bases de données fragmentées verticalement. A travers ces séries d'expériences, nous voulions étudier l'impact de l'utilisation des approches de placement sur les coûts de communication. Le tableau 1 synthétise les résultats obtenus, en terme des coûts de communication induits par l'exécution de requêtes en utilisant deux alternatives de placement : l'approche *Best Client* et notre approche, qui rappelons-le est basée sur un modèle économique.

# Requêtes		20	40	60	80	100
Coût de communication (sec)	Approche *Best Client* C_1	351	495	591	785	1220
	Modèle économique C_2	237	263	326	379	488
	$C_1 - C_2$	**+114**	**+232**	**+265**	**+406**	**+732**

Tableau 1. *Coût de communication par # requêtes : approche Best Client et modèle économique*

Les résultats du tableau 1 montrent clairement que le modèle économique fournit un coût de communication sensiblement meilleur pour l'exécution des requêtes. Ceci s'explique par l'amélioration significative des coûts de transfert de fragments comparé à ceux de l'approche *Best Client*. La différence des coûts de transfert des fragments, aux sites de traitement des opérations de requêtes, montre l'incapacité de la stratégie *Best Client* à optimiser les placements de répliques des fragments. Nous pouvons noter aussi que notre

approche de placement réduit le temps de communication de l'ordre de 80%, quand le nombre de requêtes excède 60. Ces résultats nous mènent à conclure que notre approche permet d'avoir une meilleure répartition des données dans une grille de calcul. Ceci est justifié par le fait que le modèle économique, que nous avons défini, favorise la réplicaion dynamique de fragments permettant ainsi la réduction des coûts de communication.

6. Conclusion

Dans cet article, nous nous sommes intéressés au problème de placement de bases de données dans les grilles de calcul. Etant donné les spécificités des grilles, nous avons proposé une approche de placement de données basée sur un modèle économique. Grace à ce modèle, nous arrivons à réguler l'offre des données et les demandes des requêtes.
Nous avons évalué, par la suite, notre proposition en faisant une série d'expériences en utilisant le simulateur de grilles OptorSim. Les résultats obtenus ont montré que l'approche proposée a permis de reduire, de manière sensible, les coûts de communication lors de l'exécution de requêtes de bases de données. Comme perspectives, nous envisageons d'étudier notre approche dans le cas d'une fragmentation horizontale et hybride d'une base de données.

7. Bibliographie

[1] D. G. CAMERON, R. CARVAJAL-SCHIAFFINO, A. P. MILLAR, C. NICHOLSON, K. STOCKINGER, F. ZINI,« Analysis of Scheduling and Replica Optimisation Strategies for Data Grids using OptorSim networks », *Journal of Grid Computing*, vol. 2, n° 1, 57-69, 2004.

[2] W. ALLCOCK, J. BRESNAHAN, R. KETTIMUTHU, M. LINK, C. DUMITRESCU, I. RAICU, I. FOSTER,« The Globus Striped GridFTP Framework and Server », *Proceedings of the ACM/IEEE SC 2005 Conference of SuperComputing (SC05)*, Washington, USA, 2005.

[3] S. NARAYANAN, U. CATALYUREK, T. KURC, X. ZHANG, J. SALTZ,« Applying Database Support for Large Scale Data Driven Science in Distributed Environments », *4th International Workshop on Grid Computing (Grid2003)*, pages 141-148, Phoenix, Arizona, November 2003.

[4] D. KOSSMANN,« The state of the art in distributed query processing », *ACM Computing*, vol. 32, n° 4, 422-469, 2000.

[5] I. FOSTER, C. KESSELMAN, S. TUECKE, « The anatomy of the grid : Enabling scalable virtual organizations », *International Journal of Supercomputer Applications*, vol. 15, n° 3, 200-222, 2001.

[6] T. KOSAR, M. LIVNY,« Stork : Making Data Placement a First Class Citizen in the Grid », *Proceedings of 24th IEEE Int. Conference on Distributed Computing Systems,(ICDCS2004)*, pages 151–160, Amsterdam, The pages 342-349, Tokyo, Japan, March 2004.

[7] A. GOUNARIS, NORMAN W. PATON, R. SAKELLARIOU, ALVARO A. FERNANDES, « Adaptive Query Processing and the Grid : Opportunities and Challenges », *DEXA Workshops*, pages 506-510, Zaragoza, Spain, 2004.

[8] Y. HUANG, N. VENKATASUBRAMANIAN,« Data Placement in Intermittently Available Environments », *9th International Conference of High Performance Computing (HiPC 2002)*, pages 367-376, Bangalore, India, December 2002.

[9] K. RANGANATHAN, I. FOSTER,« Design and Evaluation of Dynamic Replication Strategies for a High-Performance Data Grid », *International Conference on Computing in High Energy and Nuclear Physics*, Beijing, P.R. China, September 2001.