Extraction de motifs fréquents pour l'auto-administration des bases de données

Kamel Aouiche[1], Jérôme Darmont[1], Le Gruenwald[2]

[1]*Equipe BDD, Laboratoire ERIC*
Université Lumière – Lyon 2
5 avenue Pierre Mendès-France
69676 Bron Cedex
{kaouiche | jdarmont}@eric.univ-lyon2.fr

[2]*School of Computer Science*
University of Oklahoma
Norman, OK 73019
USA
ggruenwald@ou.edu

RÉSUMÉ. *Avec le développement des bases de données en général et des entrepôts de données (*data warehouses*) en particulier, il est devenu très important de réduire la fonction d'administration de base de données. Les systèmes auto-administratifs ont pour objectif de s'administrer et de s'adapter eux-mêmes, automatiquement, sans perte (ou même avec un gain) de performance. L'idée d'utiliser des techniques de fouille de données (*data mining*) pour extraire des connaissances utiles pour leur administration à partir des données stockées est avancée depuis quelques années. Pourtant, aucune recherche n'a été entreprise dans ce domaine à notre connaissance. Cela reste néanmoins une approche très prometteuse, notamment dans le domaine des entrepôts de données, où les requêtes sont très hétérogènes et ne peuvent pas être interprétées facilement. L'objectif de ce travail est de rechercher une manière d'extraire des données elles-mêmes des connaissances utilisables pour appliquer de manière automatique des techniques d'optimisation des performances, et plus particulièrement d'indexation. Nous avons réalisé un outil qui effectue une recherche de motifs fréquents sur une charge donnée pour calculer une configuration d'index permettant d'optimiser le temps d'accès aux données. Les expérimentations que nous avons menées ont montré que les configurations d'index générées par notre outil permettent des gains de performance de l'ordre de 15% à 25% sur une base et un entrepôt de données tests.*

ABSTRACT. *With the development of databases in general and data warehouses in particular, it is now very important to reduce the database administration function. The aim of auto-administrative systems is to administrate and adapt themselves automatically, without loss (or even with a gain) in performance. The idea of using data mining techniques to extract useful knowledge for administration from the data themselves has been in the air for some years. However, no research has ever been achieved as far as we know. This nevertheless remains a very promising approach, notably in the field of data warehousing, where queries are very heterogeneous and cannot be interpreted easily. The aim of this study is to search for a way of extracting from stored data themselves useful knowledge to automatically apply performance optimization techniques, and more particularly indexing techniques. We have designed a tool that extracts frequent itemsets from a given workload to compute an index configuration that helps optimizing data access time. The experimentations we performed showed that the index configurations generated by our tool allowed performance gains of 15 to 25% on a test database and a test data warehouse.*

MOTS-CLÉS : *Bases de données, Entrepôts de données, Indexation automatique, Fouille de données, Motifs fréquents.*

KEYWORDS: *Databases, Data warehouses, Auto-indexing, Data mining, Frequent itemsets.*

Extraction de motifs fréquents pour l'auto-administration des bases de données

Kamel Aouiche, Jérôme Darmont
Laboratoire ERIC BDD, Lyon 2
{kaouiche, jdarmont}@eric.univ-lyon2.fr

Le Gruenwald
School of Computer Science, Oklahoma USA
ggruenwald@ou.edu

Objectif : Réduire la fonction d'administration en créant **automatiquement** des index.

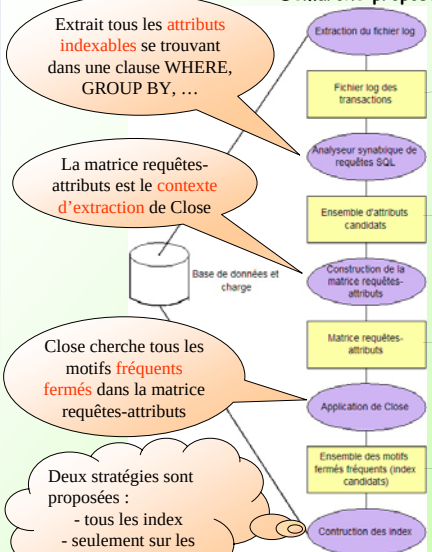
Démarche proposée

Extrait tous les attributs indexables se trouvant dans une clause WHERE, GROUP BY, …

La matrice requêtes-attributs est le contexte d'extraction de Close

Close cherche tous les motifs fréquents fermés dans la matrice requêtes-attributs

Deux stratégies sont proposées :
- tous les index
- seulement sur les "grosses" tables




```
Q1 SELECT * FROM T1, T2 WHERE A BETWEEN 1 AND 10 AND C=D
Q2 SELECT * FROM T1, T2 WHERE B LIKE '%this%' AND C=5 AND E<10
Q3 SELECT * FROM T1, T2 WHERE A=30 AND B>3 GROUP BY C HAVING Sum(E)>2
Q4 SELECT * FROM T1 WHERE B>2 AND E IN (3,2,5)
Q5 SELECT * FROM T1, T2 WHERE A = 30 AND B>3 GROUP BY C HAVING Sum(E)>2
Q6 SELECT * FROM T1, T2 WHERE B>3 GROUP BY C HAVING Sum(E)>2
```

Attributs candidats

A	B	C	D	E

Attributs

Requêtes	A	B	C	D	E
Q1	1	0	1	1	0
Q2	0	1	1	0	1
Q3	1	1	1	0	1
Q4	0	1	0	0	1
Q5	1	1	1	0	1
Q6	0	1	1	0	1

Motifs fermés fréquents

AC
BE
C
ABCE
BCE

Résultats encourageants (15 à 25% de gain de performances)

Perspectives

- Utiliser d'autres techniques d'optimisation de performances (vues matérialisées, regroupement, etc.)
- Comparer notre démarche avec celle de Microsoft[Chaudhuri et Narasayya, 1997]
- Étudier comment utiliser les algorithmes de découverte de dépendances fonctionnelles ou de dépendances d'inclusions.

http://bdd.univ-lyon2.fr