Quelques algorithmes pour la jointure

15 septembre 2014
christophe.cerin@iutv.univ-paris13.fr

1- Le problème

On suppose, à titre d'exemple, que l'on a les deux tables Employee et Dept suivantes :

Employee

Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

DeptName	Manager
Finance	George
Sales	Harriet
Production	Charles

Employee ⋈ *Dept*

Name	EmpId	DeptName	Manager
Harry	3415	Finance	George
Sally	2241	Sales	Harriet
George	3401	Finance	George
Harriet	2202	Sales	Harriet

Ces deux tables ont un attribut commun (DeptName) qui sert dans la jointure qui est l'ensemble des combinaisons des tuples de Employee et Dept qui sont égaux par leur attribut de jointure. Pour notre exemple la jointure est donnée par la table de droite. Comment la construire ?

2- Jointure par une bouble imbriquée

Le nom de cette méthode provient de la boucle imbriquée que l'on retrouve dans le pseudo code suivant :

```
Pour toutes les lignes de Employee faire
   Pour toutes les lignes de Dept faire
     si Employee.DeptName == Dept.DeptName alors
           Construire le tuple( Employee.Name,Employee.EmpId,Employee.DeptName,Dept.Manager)
           Ajouter ce tuble au résultat
     finsi
  finpour
finpour
```

La complexité du traitement est liée au produit entre le nombre de lignes de Employee et de Dept, soit une complexité quadratique.

3- Jointure par tri puis fusion

L'idée consiste à trier les deux tables (séparément) selon le critère de l'attribut commun, puis de fusionner. L'intérêt de cet algorithme réside dans sa complexité qui est de N.log M et N.log N pour les tris plus une complexité linéaire pour la fusion. Au final on obtient une complexité en O(P.log P).

Exercice : appliquer cette méthode sur l'exemple.

Attention : il y a aussi un tri qui s'appele le Mergesort (tri fusion) qui, conceptuellement, s'arrange pour fusionner des listes à 1 élément (nécessairement triées), puis des listes triées à 2 élements, puis des listes triées à 4 élements etc.

3- Jointure par hachage

Rappel : Une **table de hachage** est, en informatique, une structure de données qui permet une association clé-élément à l'image de ce que l'on a avec un dictionnaire. L'idéee est accéder à la valeur d'une clé en un coût O(1). Concernant la fonction de hachage h on pourrait prendre la fonction qui associe la somme des valeurs ascii des caractères composant la clé, modulo un entier, ce qui donne une valeur entière qui est une position dans un tableau. On voit bien que h('ab') = h('ba') et dans cet exeple on a un problème de « collision », c'est-à-dire que deux clés différentes, voire davantage, pourront se retrouver associées à la même valeur de hashage et donc à la même case dans le « tableau » (la fonction n'est pas injective). Pour diminuer les risques de collisions, il faut donc premièrement choisir avec soin sa fonction de hachage. Ensuite, un mécanisme de résolution des collisions sera à implémenter.

La méthode de calcul de la jointure par hachage se déroule comme suit:

1. Commencez par préparer une table de hachage de la plus petite relation. Les entrées de la table de hachage sont constitués de l'attribut de jointure et son rang. *Pour notre exemple on pourrait avoir que h('Finance') nous indique que 'Fiance' est sur la première ligne de la table ; que h('Sales') nous indique que 'Sales' est sur la deuxième ligne de la table et enfin que h('Production') nous indique que 'Production' est sur la troisième ligne de la table Dept.*
2. Une fois que la table de hachage est construite, parcourir la plus grande relation pour trouver les lignes pertinentes de la plus petite relation qui font partie de la solution en regardant dans la table de hachage. *Mettre en oeuvre cette idée pour notre exemple.*

D'un point de vue complexité, nous remarquons que cette troisième méthode introduit deux parcours linéaires et que cela est meilleur que les autres méthodes. On pourrait en conclure qu'il suffit d'utiliser cette troisième méthode. En fait n'oubliez pas qu'un calcul de complexité vous donne une vue asymptotique... et qu'il y a des constantes cachées qui peuvent jouer, sous certaines contidions, en faveur des deux premières méthodes et dans la pratique.

Le TP permet d'apréhender les résultats effectifs de performance dans des cas réels.