Cost Hash Join

July 2023

1 Notation

B : Nombre de partitions.
F : Fudge factor
s : Facteur de selectivité

$\left|M\right|$: Nombre de pages disponibles en mémoire.
$\left|R\right|$: Nombre de pages occupées par R.
$\left|S\right| \geq \left|R\right|$: Nombre de pages occupées par S.
$\left\{R\right\}$: Nombre de tuples dans R.
$\left|H\right|$: Taille de la table de hachage.
$A_r = \lceil \frac{\left|R\right|}{B} \rceil$: Nombre de pages par partition de R.

comp : Temps pour comparer des clé dans la mémoire.
hash : Temps pour hacher une clé dans la mémoire.
move : Temps pour déplacer un tuple dans la mémoire.
swap : Temps pour échanger deux tuples dans la mémoire.
IO : Temps pour lire ou écrire une page entre le disque et la mémoire.

2 Simple Hash Join

2.1 Cas 1 : La table de hachage rentre dans la mémoire

Condition : $F * \left|R\right| = \left|H\right| \leq \left|M\right| - 2$

2.1.1 I/O

Lecture : $\left|R\right| + \left|S\right|$
Ecriture : $\lceil \left|R\right| * s \rceil$

2.1.2 Coût

$$
\begin{aligned}
temps = & \left\{R\right\} * (hash + move) && (1)\\
& + \left\{S\right\} * (hash + F * comp) && (2)\\
& + \left[Lecture + Ecriture\right] * IO + \left\lceil \left\{R\right\} * s \right\rceil && (3)
\end{aligned}
$$

(1) Tuples de R que l'on hache et déplace pour créer la table H en mémoire.
(2) Tuples de S que l'on hache et compare pour chercher dans la table H en mémoire.
(3) Entrées/Sorties et le déplacement des tuples satisfaisant la jointure.

2.2 Cas 2 : La table de hachage ne rentre pas dans la mémoire

Condition : $F * \left|R\right| = \left|H\right| > \left|M\right| - 2$

2.2.1 Partitionnage

Condition : $F * \left|R\right| = \left|H\right| \leq \left|M\right| - 3$

$B = \left\lceil \frac{\left|R\right| * F}{\left|M\right| - 3} \right\rceil$

$\left|H\right| = \left\lceil \frac{\left|R\right| * F}{B} \right\rceil$

2.2.2 I/O

Lecture : $\left|R\right| + \left|S\right| + \frac{B * (B-1)}{2} * (A_r + A_s)$

Ecriture : $\left\lceil \left|R\right| * s \right\rceil + \frac{B * (B-1)}{2} * (A_r + A_s)$

2.2.3 Coût

$$
\begin{aligned}
temps = & \left[\frac{B * (B+1)}{2}\right] * \left[\left\{A_r\right\} + \left\{A_S\right\}\right] * (hash + move) && (4)\\
& - \left\{S\right\} * move && (5)\\
& + \left\{R\right\} * hash && (6)\\
& + \left\{S\right\} * (hash + F * comp) && (7)\\
& + \left[Lecture + Ecriture\right] * IO + \left\lceil \left\{R\right\} * s \right\rceil && (8)
\end{aligned}
$$

(4) Tuples de R et S qu'on hache en partition et réécrit sur le disque ou déplace dans la table H_i.
(5) Tuples de S qui appartiennent à la partition B_i et que l'ont ne déplacent

pas et qui on été comptabilisées dans (4).
(6) Tuples de R que l'on hache et avons déplacé en (4) pour créer la table H_i en mémoire.
(7) Tuples de S que l'on hache et compare pour chercher dans la table H_i en mémoire.
(8) Entrées/Sorties et le déplacement des tuples satisfaisant la jointure.

3 Grace Hash Join

3.1 Cas 2 : La table de hachage ne rentre pas dans la mémoire

3.1.1 Partitionnage

$B = \big|M\big| - 1$

Condition : $\Big\lceil \dfrac{\big|R\big| * F}{B} \Big\rceil = \big|H\big| \leq \big|M\big| - 2$

3.1.2 I/O

Lecture : $\big|R\big| + \big|S\big| + B * (A_r + A_s)$

Ecriture : $\big\lceil \big|R\big| * s \big\rceil + B * (A_r + A_s)$

3.1.3 Coût

$$
\begin{aligned}
temps = & \Big[\big\{R\big\} + \big\{S\big\} \Big] * (hash + move) && (9)\\
& + \big\{R\big\} * (hash + move) && (10)\\
& + \big\{S\big\} * (hash + F * comp) && (11)\\
& + \Big[Lecture + Ecriture \Big] * IO + \big\lceil \big\{R\big\} * s \big\rceil && (12)
\end{aligned}
$$

(9) Tuples de R et S que l'on hache en partition et qu'on écrit sur la partition correspondante en mémoire.
(10) Tuples de R que l'on hache et déplace pour créer la table H_b en mémoire.
(11) Tuples de S que l'on hache et compare pour chercher dans la table H_b en mémoire.
(12) Entrées/Sorties et le déplacement des tuples satisfaisant la jointure.

4 Hybrid Hash Join

4.1 Cas 2 : La table de hachage ne rentre pas dans la mémoire

4.1.1 Partitionnage

Condition : $\Big\lceil \dfrac{\big|R\big| * F}{B} \Big\rceil = \big|H\big| \leq \big|M\big| - 2 - (B - 1)$

4.1.2 I/O

Lecture : $\big|R\big| + \big|S\big| + (B-1) * (A_r + A_s)$

Ecriture : $\lceil \big|R\big| * s \rceil + (B-1) * (A_r + A_s)$

4.1.3 Coût

$$
\begin{aligned}
temps = & \Big[\big\{R\big\} + \big\{S\big\} \Big] * (hash + move) && (13)\\
& - \big\{A_s\big\} * move && (14)\\
& + \big\{R\big\} * hash && (15)\\
& + \big\{S\big\} * (hash + F * comp) && (16)\\
& + \Big[Lecture + Ecriture \Big] * IO + \lceil \big\{R\big\} * s \rceil && (17)
\end{aligned}
$$

(13) Tuples de R et S que l'on hache en partition et qu'on écrit sur la partition correspondante en mémoire ou déplace dans la table H_0.
(14) Tuples de S de la partition 0 pour qui n'ont pas a être déplacés et qui ont été comptabilisés en (13).
(15) Tuples de R que l'on hache pour créer la table H_b.
(16) Tuples de S que l'on hache et compare pour chercher dans la table H_b en mémoire.
(17) Entrées/Sorties et le déplacement des tuples satisfaisant la jointure.

5 Sort-Merge Join

5.1 Tri

5.1.1 Passe

$$B_R = 1 + \left\lceil \log_{M-1} \left\lceil \frac{\big|R\big|}{\big|M\big|} \right\rceil \right\rceil$$

$$B_S = 1 + \left\lceil \log_{M-1} \left\lceil \frac{\big|S\big|}{\big|M\big|} \right\rceil \right\rceil$$

5.1.2 I/O

Lecture : $\big|R\big| * B_R + \big|S\big| * B_S$
Ecriture : $\big|R\big| * B_R + \big|S\big| * B_S$

5.1.3 Coût

$$
\begin{aligned}
temps = & \Big[\big\{R\big\} * \log_2 \big\{R\big\} + \big\{S\big\} \log_2 \big\{S\big\} \Big] * (swap + move) && (18)\\
& + \Big[Lecture + Ecriture \Big] * IO && (19)
\end{aligned}
$$

(18) Tri fusion des tuples R et S.
(19) Entrées/Sorties.

5.2 Fusion

5.2.1 I/O

Lecture : $\big|R\big| + \big|S\big|$
Ecriture : $\big|R\big| + \big|S\big|$

5.2.2 Coût

$$temps = Tri_R + Tri_S \tag{20}$$
$$+ \Big[\big\{R\big\} + \big\{S\big\}\Big] * comp \tag{21}$$
$$+ \Big[Lecture + Ecriture\Big] * IO + \Big\lceil\big\{R\big\} * s\Big\rceil \tag{22}$$

(20) Cout des tris de R et S.
(21) Comparaisons des tuples R et S lors de la fusion.
(22) Entrées/Sorties et le déplacement des tuples satisfaisant la jointure.

6 Index nested loop join

6.1 Build

6.1.1 Niveaux

lg = nombre de pointeurs/tuples par page
$n = 1 + \lceil \log_{lg} \big|R\big| \rceil$: nombre de niveaux de l'index I
$\big|I_i\big| = \dfrac{\big|R\big|}{lg^i}$: nombre de pages de par niveaux de I
$\big|I\big| = \sum_{i=0}^{i=N} \big|I_i\big|$: nombre de pages de I
n_M : nombre de niveaux de l'index rentrant dans la mémoire.

6.1.2 I/O

Lecture : $IO_{Tri_R} + \big|R\big|$
Ecriture : $IO_{Tri_R} + \big|I\big|$

6.1.3 Coût

$$temps = Tri_R \tag{23}$$
$$+ \big\{R\big\} * (move + hash) \tag{24}$$

(23) Tri fusion des tuples R .
(24) Détermination de la clé d'indexation et déplacement des tuples de R.

6.2 Probe

6.2.1 I/O

Lecture : $IO_{Build_I} + \big|S\big| + \big\{S\big\} * \Big[(n+1) - n_M\Big]$
Ecriture : $IO_{Build_I} + \lceil \big|R\big| * s \rceil$

6.2.2 Coût

$$
\begin{aligned}
temps = {} & Build_I && (25)\\
& + \big\{S\big\} * comp * \Big[(n+1)\Big] && (26)\\
& + \Big[Lecture + Ecriture\Big] * IO + \lceil \big\{R\big\} * s \rceil && (27)
\end{aligned}
$$

(25) Cout du build de l'index.
(26) Comparaisons des tuples S et des plusieurs niveaux de l'index.
(27) Entrées/Sorties et le déplacement des tuples satisfaisant la jointure.