Réseaux de tri

Résumé: Le premier exercice est facile. Le deuxième exercice sur le tri bitonique est un classique du domaine (voir par exemple [2] ou [1]). Le troisième exercice nous permet d'aborder un réseau de tri plus sophistiqué; on trouvera de nombreux autres exemples difficiles dans [3].

1 Principe du 0-1

On démontre ici, une propriété des réseaux de tris particulièrement utile : le principe du 0-1.

▷ **Question 1** *Montrer qu'un réseau de comparateurs implante correctement le tri si et seulement si il calcule bien cette fonction pour toutes suites de type 0-1 en entrée.*

2 Réseau de tri bitonique

Définition 1. On appelle **séquence bitonique** une séquence qui est soit croissante puis décroissante, soit décroissante puis croissante. Ainsi, les séquences $\langle 2,3,7,7,4,1\rangle$ et $\langle 12,5,10,11,19\rangle$ sont bitoniques. Les séquences binaires bitoniques sont de la forme $0^i1^j0^k$ ou de la forme $1^i0^j1^k$.

Définition 2. Un **réseau de tri bitonique** est un réseau de comparateurs triant toute séquence binaire bitonique.

Définition 3. On appelle **séparateur** un réseau à n entrées, avec n pair, composé d'une colonne de comparateurs dans lequel chaque entrée i est comparée à l'entrée $i+\frac{n}{2}$ pour $i \in \{1,2,...,\frac{n}{2}\}$.

▷ **Question 2** *Comment construire un réseau de tri bitonique à partir de séparateurs ? Quelle est sa profondeur et le nombre de comparateurs utilisés ?*

▷ **Question 3** *En utilisant des trieuses bitoniques, construire un réseau fusionnant deux listes triées. En déduire la construction d'un réseau général de tri dont on déterminera la profondeur et le nombre de comparateurs.*

3 Tri sur une grille 2D

Cet exercice étend le tri par transposition pair-impair, déjà étudié sur un réseau linéaire, au cas d'une grille à deux dimensions.

Définition 4. Un tableau carré $A = ((a_{i,j}))$ de taille $n \times n$, $\quad n = 2^m$ est ordonné en serpent si les éléments du tableau sont ordonnés comme suit :

$$\begin{aligned}
a_{2i-1,j} &\leqslant a_{2i-1,j+1}, && \text{si} \quad 1 \leqslant j \leqslant n-1, 1 \leqslant i \leqslant n/2,\\
a_{2i,j+1} &\leqslant a_{2i,j}, && \text{si} \quad 1 \leqslant j \leqslant n-1, 1 \leqslant i \leqslant n/2,\\
a_{2i-1,n} &\leqslant a_{2i,n}, && \text{si} \quad 1 \leqslant i \leqslant n/2,\\
a_{2i,1} &\leqslant a_{2i+1,1}, && \text{si} \quad 1 \leqslant i \leqslant n/2-1.
\end{aligned}$$

On peut noter que ce serpent induit un réseau linéaire à l'intérieur de la grille (voir figure 1).

Définition 5. Un «shuffle» transforme la séquence de $n = 2p$ éléments $\langle z_1,\dots,z_n\rangle$ en la séquence $\langle z_1, z_{p+1}, z_2, z_{p+2},\dots, z_p, z_{2p}\rangle$. Par exemple le «shuffle» de $(1,2,3,4,5,6,7,8)$ est $(1,5,2,6,3,7,4,8)$.

On se propose d'étudier l'algorithme suivant, qui réalise la fusion de 4 tableaux de taille $2^{m-1} \times 2^{m-1}$ ordonnés en serpent en un tableau de taille $2^m \times 2^m$ ordonné en serpent :

1. «shuffle» de chaque ligne du tableau (en utilisant des transpositions pair-impair sur les indices des éléments), ce qui revient à appliquer la transformation «shuffle» sur les colonnes.

$$\begin{array}{ccccccc}
a_{1,1} & \rightarrow & a_{1,2} & \rightarrow & a_{1,3} & \rightarrow & a_{1,4} \\
 & & & & & & \downarrow \\
a_{2,1} & \leftarrow & a_{2,2} & \leftarrow & a_{2,3} & \leftarrow & a_{2,4} \\
\downarrow & & & & & & \\
a_{3,1} & \rightarrow & a_{3,2} & \rightarrow & a_{3,3} & \rightarrow & a_{3,4} \\
 & & & & & & \downarrow \\
a_{4,1} & \leftarrow & a_{4,2} & \leftarrow & a_{4,3} & \leftarrow & a_{4,4}
\end{array}$$

FIG. 1 – L'ordre serpent sur une grille 4×4.

2. Trier les paires de colonnes, c'est-à-dire les tableaux de taille $n \times 2$ en ordre serpent, en utilisant $2n$ étapes de transpositions pair-impair sur le réseau linéaire induit sur chaque serpent de longueur $2n$.
3. Appliquer $2n$ étapes de transpositions pair-impair sur le réseau linéaire induit par le serpent de taille n^2.

▷ **Question 4** *Faire tourner l'algorithme de tri induit avec* $n = 4$ *et* $a_{i,j} = 21 - 4i - j$ *pour* $1 \leqslant i, j \leqslant 4$.

▷ **Question 5** *Montrer que la première étape de l'algorithme peut s'effectuer en temps* $2^{m-1} - 1$, *l'unité étant un échange entre voisins (plusieurs échanges entre voisins pouvant être effectués en parallèles pour le même coût). On pourra effectuer les transpositions pair-impair sur un ensemble d'indices astucieusement choisis. En déduire que l'algorithme global de fusion s'effectue en temps* $\leqslant \frac{9}{2}n$.

▷ **Question 6** *En supposant l'algorithme de fusion correct, construire un algorithme qui trie une séquence de longueur* 2^{2m} *sur une grille* $2^m \times 2^m$. *Estimer sa complexité.*

▷ **Question 7** *Montrer que le tri par transposition pair-impair sur une grille est correct (il s'agit de montrer que* $2n$ *étapes de transposition pair-impair dans la troisième phase de l'algorithme de fusion suffisent à obtenir un serpent correctement ordonné).*

Références

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, 2 edition, 1990. Traduction française publiée chez Dunod, Introduction à l'algorithmique, 2002.

[2] A. Gibbons and W. Rytter. *Efficient Parallel Algorithms*. Cambridge University Press, 1988.

[3] F.T. Leighton. *Introduction to parallel algorithms and architectures : arrays, trees, hypercubes*. Morgan Kaufmann, 1992.