14b Tris et permutations

14b.1 Permutation aléatoire en place

Avant de procéder au tri rapide avec une sélection de pivot déterministe (p.e., médiane-de-trois), on fait une permutation aléatoire du tableau (afin de se rendre au cas moyen). L'algorithme suivant met les éléments d'un tableau dans un ordre aléatoire *en place*, selon la distribution uniforme sur toutes $(n!)$ permutations.

```
    SHUFFLE(A[0..n − 1])                    // met les éléments de A dans un ordre aléatoire
P1  for i ← 0, 1, . . . , n − 1 do
P2      j ← i + rnd(n − i)                  // rnd(m) donne un nombre entier (pseudo-)aléatoire de {0, 1, . . . , m − 1}
P3      échanger A[i] ↔ A[j]                // échange de A[i] et un élémént du suffixe A[i..n − 1] ; i = j est possible
```

14b.2 Le minimum de comparaisons dans un tri

Un **tri par comparaison** n'utilise que des comparaisons entre les éléments d'un tableau (genre $A[i] < A[j]$) pour le trier. L'exécution de l'algorithme ne dépend que l'ordre initial du tableau. On définit l'**arbre de décision** qui montre la séquence de comparaisons pour toute exécution possible. Un nœud interne de l'arbre contient une comparaison entre les éléments de A ; il a toujours deux enfants qui correspondent au branchement de l'exécution selon le résultat de la comparaison. Tout nœud externe correspond à une permutation ce qui est l'ordre final des éléments. La Figure 1 montre l'exemple du tri par insertion de $A[0..3]$.

Théorème 14b.1. *Pour tout algorithme déterministe de tri par comparaison, il existe un arrangement initiel de n éléments distincts qui prend au moins $\lg(n!)$ comparaisons à trier.*

Démonstration. Par définition, chaque chemin de la racine jusqu'à un nœud externe correspond à la séquence de comparaisons performées pour établir l'ordre final. Le tableau doit être trié à la sortie, et chaque ordre est possible : il faut avoir au moins un nœud externe pour toute permutation. Le nombre de nœuds externes dans cet arbre binaire est donc $\geq n!$. En conséquence, la hauteur de l'arbre est au moins $\lg(n!)$ (voir Théorème 6.3). Dans d'autres mots, il existe un ordre à l'entrée pour lequel l'algorithme fait au moins $\lceil \lg(n!) \rceil$ comparaisons. ■

REMARQUE. Le théorème montre qu'aucun algorithme déterministe ne peut trier n éléments avec moins que $\lg(n!) \sim n \lg n$ comparaisons ; donc le temps de calcul doit être $\Omega(n \log n)$ au pire. Le tri par fusion utilise ce nombre minimal de comparaisons. Par contre, on peut exploiter parfois la distribution des éléments dans le tableau et dépasser la borne : c'est le cas avec le tri binaire qui prend $\Theta(n)$ temps car il utilise le fait qu'il y a juste deux clés différentes parmi les éléments.

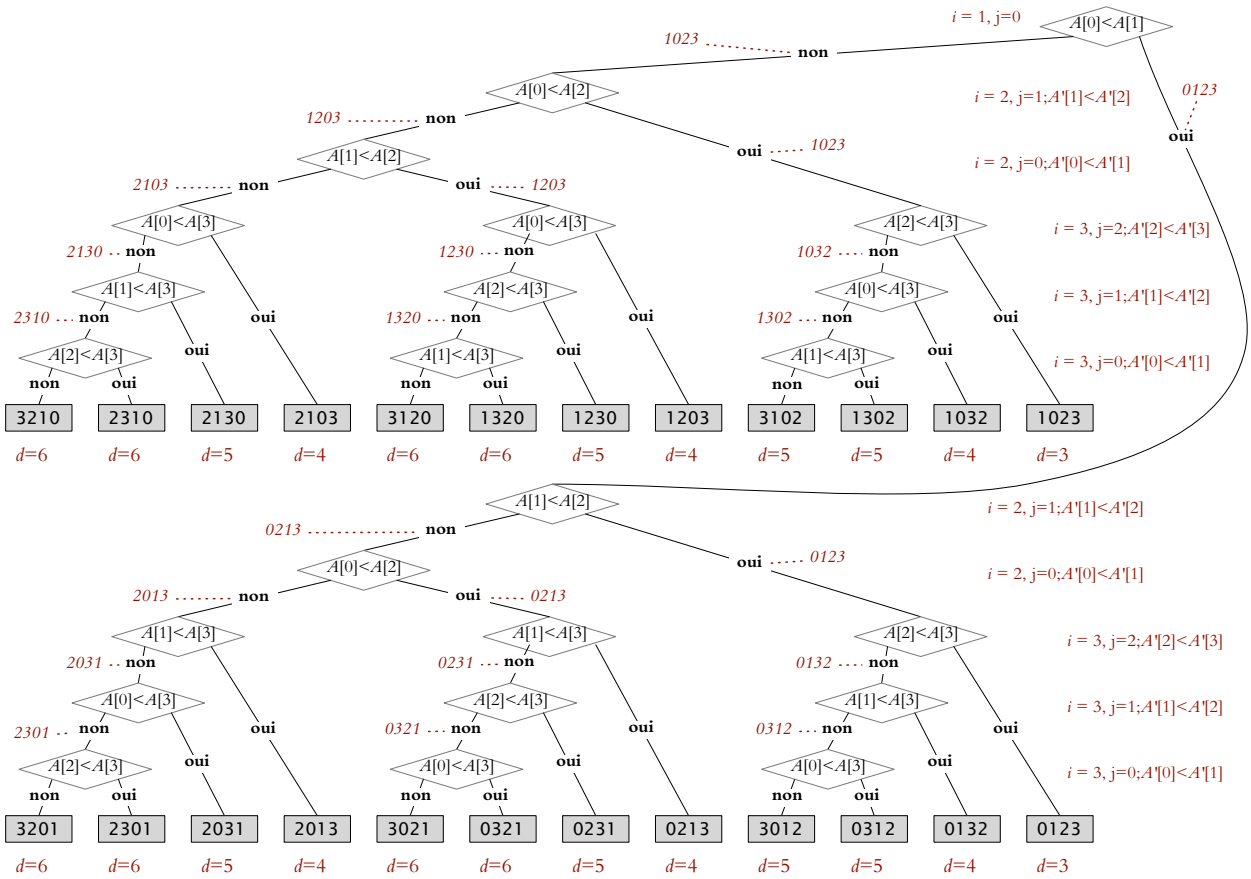

FIG. 1 – Arbre de décision du tri par insertion appliqué à 4 éléments. Pour clarté, les nœuds internes montrent les indices originaux des éléments. A' est le tableau réarrangé pendant l'exécution, le résultat des échanges sur l'ordre initial est indiqué à chaque branche. C'est un arbre de hauteur 6 — l'algorithme utilise 6 comparaisons au pire. La profondeur des nœuds externes ($d =$) est entre 3 (meilleur cas) et 6 (pire cas).