République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumédiène

Faculté d'Electronique et d'Informatique
Département Informatique

Master Systèmes Informatiques intelligents

Module : Conception et Complexité des Algorithmes

Rapport de projet 2 de TP

Réalisé par :
AIT AMARA Mohamed, 181831072170
BOUROUINA Rania, 181831052716
CHIBANE Ilies, 181831072041
HAMMAL Ayoub, 181831048403

Année universitaire : 2021 / 2022

Table des matières

0.1 Modélisation de la solution

Dans ce problème, chaque anneau porte un numéro séquentiel unique $a_i \in [1, n]$ qui représente sa taille tel que n est le nombre maximal d'anneaux (e.g. l'anneau avec le nombre 1 est plus petit que l'anneau avec le nombre 3).

De plus, on modélise chaque tour par un tableau T_j d'une taille égale au nombre maximum d'anneaux n. Si un niveau i de la tour j contient un anneau $a_{i'}$, $T[j, i] = a_{i'}$, sinon $T[j, i] = 0$. Le niveau le plus bas de la tour (la base de la tour) est placé à la dernière case du tableau ; $\forall j\ T[j, n]$ est le niveau le plus bas de la tour (voir Figure 1).

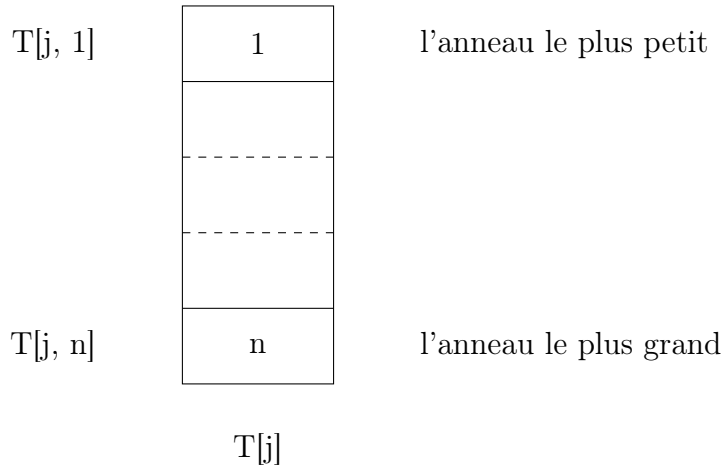

FIGURE 1 – Exemple d'une tour avec tous les anneaux

Par concéquent, le bord de jeu peut être représenté par une matrice colonne par colonne ou chaque colonne est en réalité une tour du jeu.

$$\textbf{bord} = \begin{pmatrix} T[1,1] & T[2,1] & ... \\ ... & & \\ T[1,n] & T[2,n] & ... \end{pmatrix}$$

Un exemple d'initialisation classique de trois tours avec trois anneaux placés sur la première tour :

$$\textbf{bord} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 3 & 0 & 0 \end{pmatrix}$$

Règle de changement d'état Pour passer d'un état de bord vers le suivant, on ne peut bouger qu'un seul anneau du haut d'une tour vers une autre tour, à condition que l'anneau supérieur de la tour destination a un nombre supérieur à l'anneau qu'on veut bouger.

Plus formellement, on peut déplacer le permier élément non nul d'une colonne s'il existe vers une autre colonne, s'il y a encore de la place et que le premier element non nul de la colonne destination est supérieur à celui qu'on déplace.

0.2 Algorithme de résolution

Cet algorithme récursif permet de produire la séquence exacte d'actions pour résoudre le problème des tours de Hanoi.

On commence d'abord par déplacer les $n-1$ disques de la tour de départ vers la tour intermédiaire par un appel récursif. Puis, le plus grand disque restant est transporté vers la tour d'arrivée. Ensuite, les $n-1$ disques qui se trouvaient sur la tour intermédiaire sont déplacés vers la tour d'arrivée par le même processus récursif.

Algorithme 1 : Hanoi

Données : bord : matrice [1, 3][1, n] d'entiers, depart, arrivee, intermediaire : 1..3,
nbdisques : entier
Résultat : bord : matrice [1, 3][1, n] d'entiers
début
  si *nbdisques* $\neq 0$ **alors**
    Hanoi(bord, depart, intermediaire, arrivee, nbdisques - 1);
    deplacer(board, depart, arrivee); `// Deplacer le disque supérieur de la tour depart vers la tour arrivee`
    Hanoi(bord, intermediaire, arrivee, depart, nbdisques - 1);
  fin si
fin

La fonction *deplacer* permet de déplacer le disque de niveau supérieur d'une tour vers un autre.

Fonction deplacer(Entrée/sortie : bord : matrice [1, 3][1, n] d'entiers, Entrée : depart, arrivee : 1..3)

Variable :
i, j : entier;
début
    `// Trouver le disque supérieur de la tour depart`
    $i \leftarrow 1$;
    tant que $i \leq n$ *et* $bord[depart][i] = 0$ **faire**
        $i \leftarrow i + 1$;
    fin tq
    `// Trouver le disque supérieur de la tour arrivee`
    $j \leftarrow 1$;
    tant que $j \leq n$ *et* $bord[arrivee][j] = 0$ **faire**
        $j \leftarrow j + 1$;
    fin tq
    $bord[arrivee][j-1] \leftarrow bord[depart][i]$;
    $bord[depart][i] \leftarrow 0$;
fin

Calcul de la complexité La complexité est exprimée en terme de nombre d'opérations de déplacement effectuées. En l'occurence, le nombre de déplacements est exprimé selon la suite numérique suivante :

$$h(1) = 1$$

$$h(n) = 2 * h(n-1) + 1$$

où n représente le nombre total de disque à déplacer.

En remplaçant $h(n-1)$ par la formule réccurente, on obtient :

$$h(n) = 2 * (2 * h(n-2) + 1) + 1$$

$$h(n) = 2 * (2 * (2 * h(n-3) + 1) + 1) + 1$$

$$\ldots$$

$$h(n) = 2^n - 1$$

Et ainsi la complexité est égale à $\mathcal{O}(2^n)$.

0.3 Algorithme de vérification

Le problème des tours de Hanoi à trois tours admet une unique solution, sous forme d'une suite de déplacements qui génèrent un séquencement d'états intermédiaires.

Pour vérifier la validité d'une solution quelconque, nous devons nous assurer que chaque déplacement est bien réglementaire (concerne le disque le plus haut et ne pose pas un disque sur un autre disque de taille plus petite) et que le dernier état engendré correspond effectivement à l'état but, c.à.d. que toutes les tours sont vides sauf la tour cible. De plus, la séquence de déplacement doit être exactement de longeur $2^n - 1$ tel que n représente le nombre de disques, car la solution est unique, donc égale à la solution calculée par l'algorithme exacte.

Nous nous assurons que l'algorithme de génération de solutions produit des séquences de déplacements qui respectent ces conditions susmentionnées. Par conséquent, nous devons vérifier que le dernier état qui doit correspondre à l'état but dans lequel les anneaux sont alignés sur la tour cible ou destination.

Fonction solutionestcorrecte(Entrée : bord : matrice [1, 3][1, n] d'entiers, arrivee : 1..3) : booleén

```
Variable :
début
    pour i ← 1àn faire
        si bord[arrivee][i] ≠ i alors
            retourner faux;
        fin si
    fin pour
    retourner vrai;
fin
```

Calcul de complexité La complexité de l'algorithme de vérification est triviale, elle est égale à la complexité du parcours séquentiel des éléments d'un tableau (la tour d'arrivée).

La complexité temporelle est égale à $\mathcal{O}(n)$ tel que n est le nombre de disques.

Quant à la complexité spatiale, sachant que la fonction de vérification ne prend que le dernier état comme paramètre, elle est donc égale à la taille de la matrice d'entiers : $3 * n \cong \mathcal{O}(n)$.