Rapport de projet

Marco Freire, Clément Legrand-Duchesne

26 septembre 2017

Résumé

Dans ce rapport seront expliqués les codes du pavage de Penrose et des tours de Hanoi, ainsi que leurs extensions ; et les choix d'implémentation que nous avons été menés à faire.

1 Penrose

2 Hanoi

2.1 Principe : algorithme de base

Le problème des tours de Hanoi est essentiellement récursif. Pour déplacer n disques du premier poteau jusqu'au troisième il suffit de savoir en déplacer n-1 sur le deuxième poteau (étape 1), puis de déplacer le n-ième disque sur le troisième poteau (étape 2) et enfin de déplacer encore une fois les n-1 disques sur le troisième poteau (étape 3).

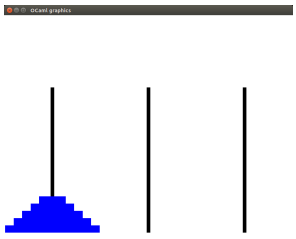

FIGURE 1 – État initial

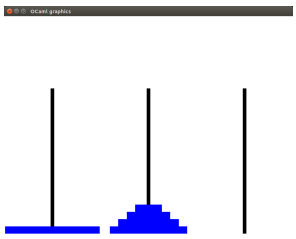

FIGURE 2 – Fin étape 1

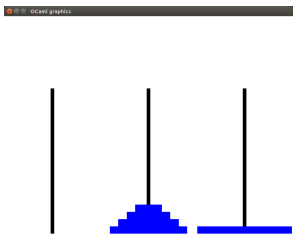

FIGURE 3 – Fin étape 2

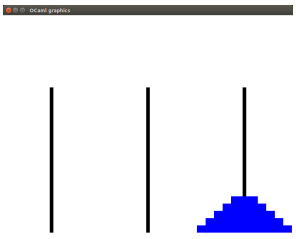

FIGURE 4 – État final

Nous avons choisi pour représenter la situation un tableau de piles `rods` : chaque pile du tableau représente chacun des poteaux et contient les disques qui y sont empilés.

La seule complication du code est le choix du poteau temporaire utilisé pour chaque déplacement de disques. La première implémentation réalisée repose sur la fonction suivante `choose` :

```
let choose a b =
  if a = b then failwith "a and b are equal"
  else if a = 0 then
        if b = 1 then 2
        else 1
  else if a = 1 then
        if b = 0 then 2
```

```
            else 0
    else if a = 2 then
            if b = 0 then 1
            else 0
    else failwith "a or b are not between 0 and 2"
;;
```

Cette fonction prend en argument deux éléments distincts de 0, 1, 2 et renvoie le troisième et est utilisée pour choisir automatiquement dans la fonction `move` le poteau temporaire à utiliser :

```
let rec move rods num_discs orig_rod dest_rod =
    if num_discs = 1 then
            move_disc rods orig_rod dest_rod
    else
            (
            let temp_rod = choose orig_rod dest_rod in
            move rods (num_discs - 1) orig_rod temp_rod;

            move_disc rods orig_rod dest_rod;

            move rods (num_discs - 1) temp_rod dest_rod;
            )
;;
```

Lors de l'implémentation de la fonction résolvant le problème des tours de Hanoi à n poteaux, nous nous sommes rendu compte que la fonction `choose` était difficilement généralisable. Nous avons donc choisi de passer le poteau intermédiaire en argument à la fonction `move` :

```
let rec move rods num_discs orig_rod dest_rod temp_rod=
    if num_discs = 1 then
            move_disc rods orig_rod dest_rod
    else
            (
            move rods (num_discs - 1) orig_rod temp_rod dest_rod;

            move_disc rods orig_rod dest_rod;

            move rods (num_discs - 1) temp_rod dest_rod orig_rod;
            )
;;
```

Cette modification permet de choisir comme l'on veut le poteau intermédiaire, ce qui est nécessaire pour la version généralisée de l'algorithme.

2.2 Principe : algorithme généralisé

Pour ce problème, l'implémentation choisie est la deuxième présentée.

Cette fois il s'agit d'utiliser au mieux les poteaux supplémentaires. Nous avons calculé le nombre de disques pouvant être sur les poteaux intermédiaires, pour répartir ces disques sur chacun des poteaux intermédiaires en utilisant comme poteau temporaire le dernier (étape 1), ensuite il faut déplacer le disque le plus grand sur le dernier poteau (étape 2), et finalement il suffit de réaliser le parcours inverse de celui effectué dans l'étape 1, et de regrouper les disques intermédiaires sur le dernier poteau (étape 3).

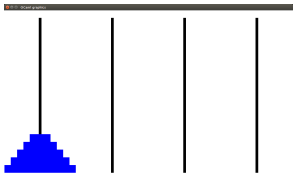

FIGURE 5 – État initial

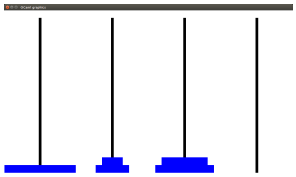

FIGURE 6 – Fin étape 1

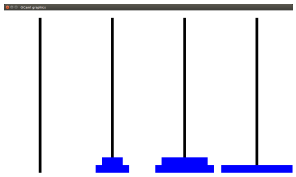

FIGURE 7 – Fin étape 2

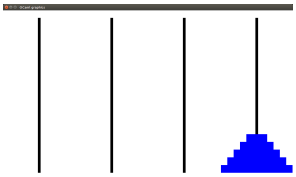

FIGURE 8 – État final

Ainsi beaucoup moins de déplacements sont nécessaires pour la résolution du problème.