

# Projet 1: Pavage de Penrose et Tours de Hanoi

Marco Freire, Clément Legrand-Duchesne

ENS de Rennes, Département Informatique 1A

29 septembre 2017

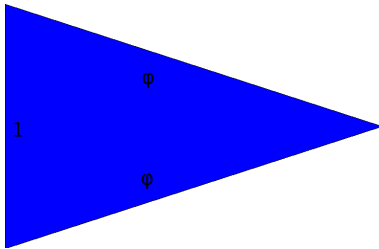
## 1 Pavage de Penrose

- Principe
- Implémentation
- Améliorations

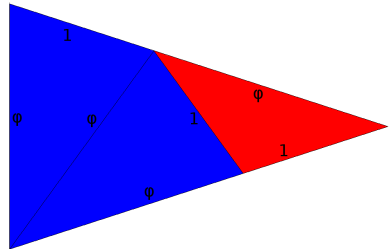
## 2 Tours de Hanoi

- Principe
- Implémentation
- Améliorations

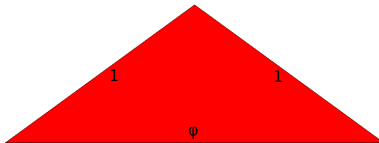
# Principe du pavage



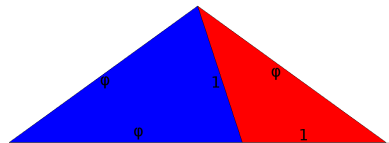
(a) Triangle d'or aigu



(b) Découpe en trois triangles d'or



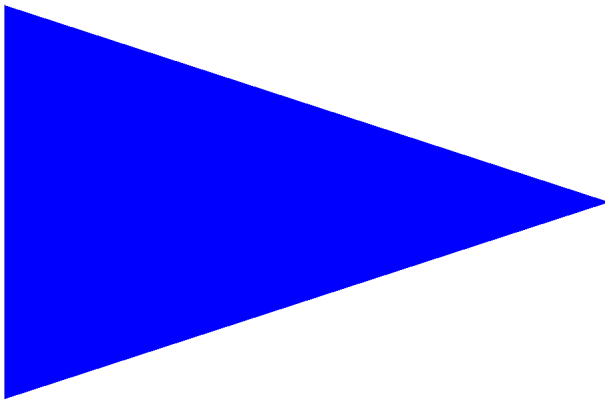
(c) Triangle d'or obtus



(d) Découpe en deux triangles d'or

# Un premier algorithme

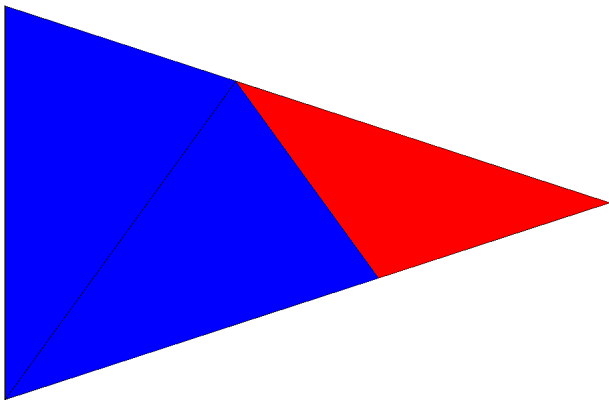
Découpe récursive en partant d'un grand triangle initial.



Éxecution de l'algorithme

# Un premier algorithme

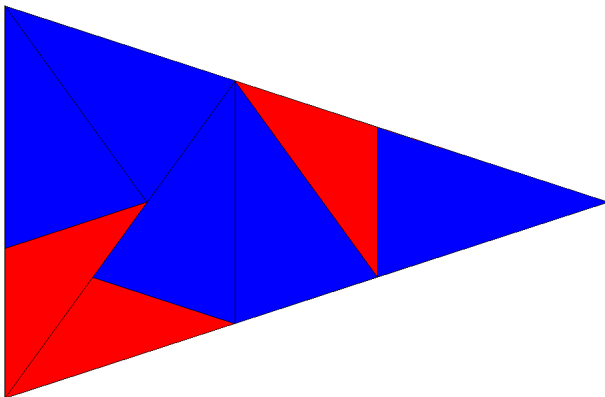
Découpe récursive en partant d'un grand triangle initial.



Éxecution de l'algorithme

# Un premier algorithme

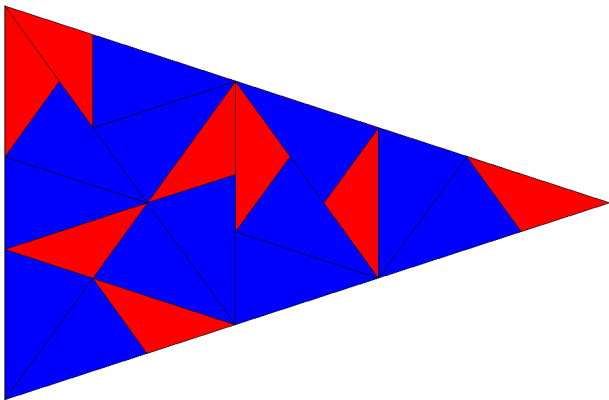
Découpe récursive en partant d'un grand triangle initial.



Éxecution de l'algorithme

# Un premier algorithme

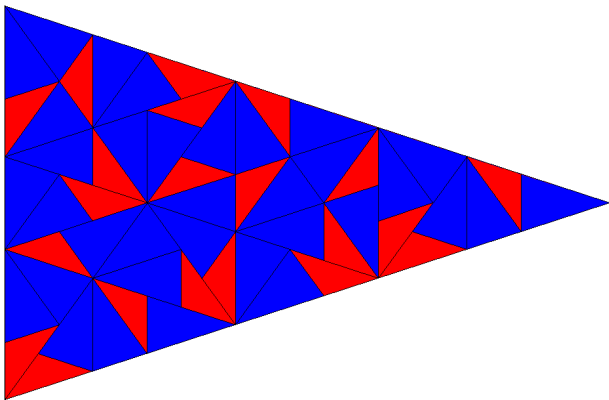
Découpe récursive en partant d'un grand triangle initial.



Éxecution de l'algorithme

# Un premier algorithme

Découpe récursive en partant d'un grand triangle initial.



Éxecution de l'algorithme



# Limite du premier algorithme

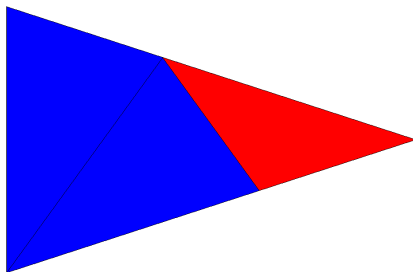
Algorithme assez limité :

- parcours en profondeur de l'arbre des découpes ;
- chaque côté des triangles est tracé deux fois ;
- le remplissage des triangles efface les bordures si celles ci ont été tracées avant.

Travail sur des flottants jusqu'à l'affichage : évite les problèmes d'arrondis.

# Tracé unique des côtés

Afin de ne tracer qu'une fois chaque côté du triangle, on ne dessine que les séparations internes à chaque nouvelle génération.



# Affichage successif de chacune des générations

## Parcours largeur :

- permet d'afficher chaque génération successivement sans tout recalculer ;
- complexité mémoire importante (nécessite une structure file) ;
- impossibilité de colorier les triangles à chaque génération sans effacer les cotés de ceux dessinés lors de la précédente génération.

## Plusieurs parcours profondeur successifs :

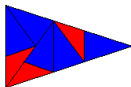
- complexité mémoire faible mais algorithme deux fois plus long ;
- possibilité de colorier les triangles et de ne tracer leurs côtés qu'une seule fois ;
- homothétie plus facilement implémentable.

# Version définitive avec homothétie



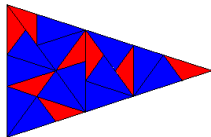
Éxecution de l'algorithme amélioré

# Version définitive avec homothétie



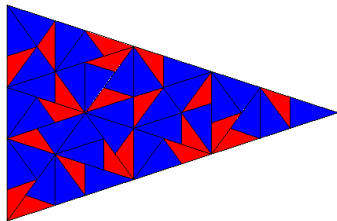
Éxecution de l'algorithme amélioré

# Version définitive avec homothétie



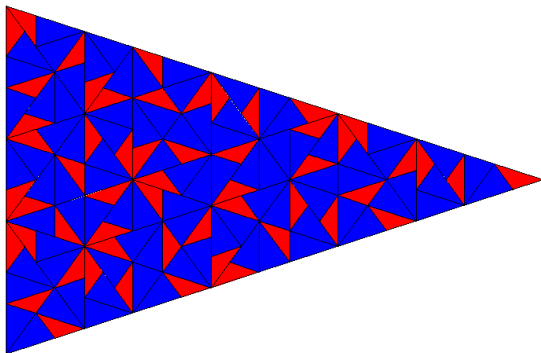
Éxecution de l'algorithme amélioré

# Version définitive avec homothétie



Exécution de l'algorithme amélioré

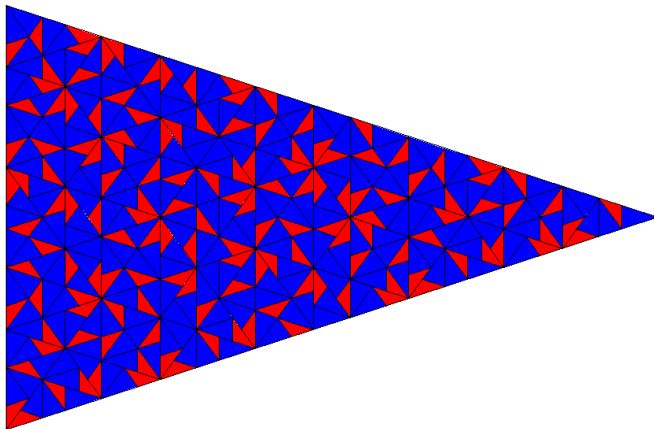
# Version definitive avec homothétie



Éxecution de l'algorithme amélioré

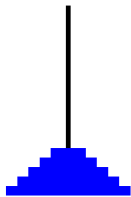


# Version definitive avec homothétie

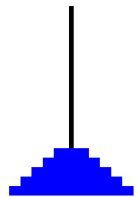


Éxecution de l'algorithme amélioré

# Principe du jeu



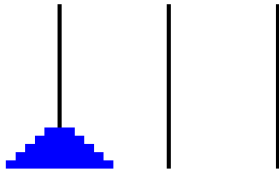
(a) État initial



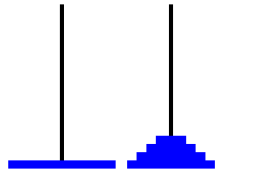
(b) État final

Le joueur doit déplacer les  $n$  disques initiaux du premier piquet au dernier en réalisant un seul mouvement à la fois, et avec la contrainte suivante : aucun disque ne peut être empilé à aucun moment sur un disque de taille inférieure.

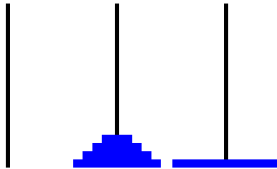
# Algorithme simple



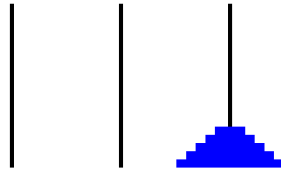
(a) État initial



(b) Étape 1



(c) Étape 2



(d) État final

# Calcul du nombre de déplacements

Initialisation  $M(0) = 1$

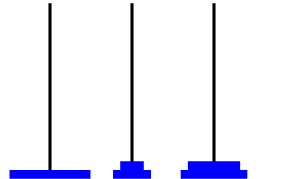
Relation de récurrence  $M(n) = 2M(n - 1) + 1$

Relation générale  $M(n) = 2^n - 1$

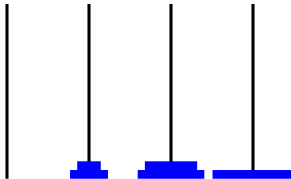
# Algorithme généralisé



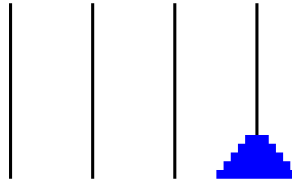
(a) État initial



(b) Étape 1



(c) Étape 2



(d) État final

# Choix et extensions

La situation globale est représentée par un tableau de piles.

Extensions créées :

- affichage graphique ;
- généralisation du problème à  $p$  piquets.

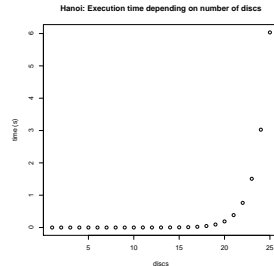
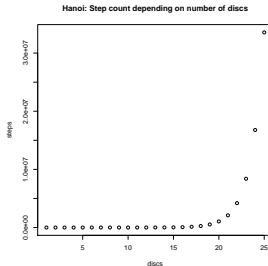
# Problèmes rencontrés

Différents problèmes :

- 1 premier algorithme non-généralisable ;
- 2 généralisation de l'affichage.

# Algorithme simple

Le nombre de déplacements de disques effectués par l'algorithme dans le cas où il n'y a que trois piquets est optimal.



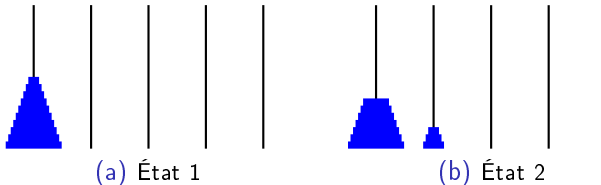
Les résultats expérimentaux sont en accord avec le calcul théorique : le nombre de mouvements le temps d'exécution sont fonction exponentielle du nombre initial de disques.



# Algorithme généralisé

À ce jour, la solution optimale des Tours de Hanoi à plus de quatre piquets est un problème ouvert.

L'algorithme ici utilisé n'utilise pas la totalité de piquets libres.



Dans cette situation, pour passer d'un état à l'autre, uniquement le dernier piquet est utilisé pour stocker des disques, alors que les autres pourraient être utilisés de façon à effectuer moins de mouvements.