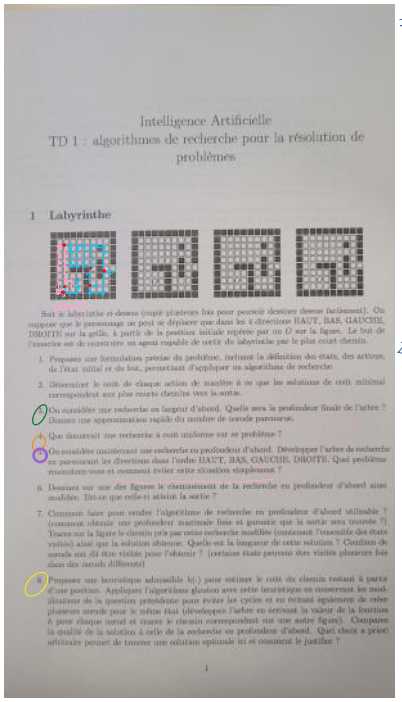


Intelligence Artificielle
TD 1 : algorithmes de recherche pour la résolution de problèmes

1 Labyrinthe

Soit le labyrinthe ci-dessus (copié plusieurs fois pour pouvoir dessiner dessus facilement). On suppose que le personnage ne peut se déplacer que dans les 4 directions HAUT, BAS, GAUCHE, DROITE sur la grille, à partir de la position initiale repérée par un *O* sur la figure. Le but de l'exercice est de construire un agent capable de sortir du labyrinthe par le plus court chemin.

1. Proposez une formulation précise du problème, incluant la définition des états, des actions, de l'état initial et du but, permettant d'appliquer un algorithme de recherche.
2. Déterminez le coût de chaque action de manière à ce que les solutions de coût minimal correspondent aux plus courts chemins vers la sortie.
3. On considère une recherche en largeur d'abord. Quelle sera la profondeur finale de l'arbre ? Donnez une approximation rapide du nombre de nœuds parcourus.
4. Que donnerait une recherche à coût uniforme sur ce problème ?
5. On considère maintenant une recherche en profondeur d'abord. Développez l'arbre de recherche en parcourant les directions dans l'ordre HAUT, BAS, GAUCHE, DROITE. Quel problème rencontrez-vous et comment éviter cette situation simplement ?
6. Dessinez sur une des figures le cheminement de la recherche en profondeur d'abord ainsi modifiée. Dit-ce que celle-ci atteint la sortie ?
7. Comment faire pour rendre l'algorithme de recherche en profondeur d'abord utilisable ? (comment obtenir une profondeur maximale finie et garantir que la sortie sera trouvée ?) Tracez sur la figure le chemin pris par cette recherche modifiée (contenant l'ensemble des états visités) ainsi que la solution obtenue. Quelle est la longueur de cette solution ? Combien de nœuds ont dû être visités pour l'obtenir ? (certains états peuvent être visités plusieurs fois dans des nœuds différents)
8. Proposez une heuristique admissible $h(\cdot)$ pour estimer le coût du chemin restant à partir d'une position. Appliquez l'algorithme glouton avec cette heuristique en conservant les modifications de la question précédente pour éviter les cycles et en évitant également de créer plusieurs nœuds pour le même état (développez l'arbre en écrivant la valeur de la fonction à pour chaque nœud et tracez le chemin correspondant sur une autre figure). Comparez la qualité de la solution à celle de la recherche en profondeur d'abord. Quel choix a priori arbitraire permet de trouver une solution optimale ici et comment le justifier ?

9. Donnez un exemple de labyrinthe dans lequel la recherche gloutonne ne trouvera jamais la solution optimale (la sortie peut aussi être un nœud ou milieu du labyrinthe).
10. Nous allons maintenant appliquer l'algorithme A* en conservant toutes les améliorations précédentes pour éviter les états répétés. Développez l'arbre de recherche en notant la valeur de la fonction de coût $f(n)$ à côté de chaque nœud (ou utilisez le coût de chemin $g(n)$ de la question 2 et l'heuristique $h(n)$ de la question 7). Lorsqu'un choix arbitraire se présente, choisissez toujours le nœud le plus profond. Quel est le nombre de nœuds parcourus par l'algorithme ? Quel est le coût de la solution ? Celle-ci est-elle optimale ? Comment justifier le fait de toujours choisir le nœud le plus profond en cas d'égalité ? Sans cette règle, la solution obtenue serait-elle optimale ?
11. Imaginons maintenant que la position initiale du personnage soit tirée aléatoirement. Est-ce que votre formulation du problème et les algorithmes permettent toujours de résoudre le problème ou doit-on considérer des algorithmes probabilistes ? Pourquoi ?
12. Idem mais si les positions des murs sont tirées aléatoirement.
13. Comment rendre le problème plus réaliste pour le personnage et de manière à ce que les algorithmes étudiés ne soient plus utilisables ?

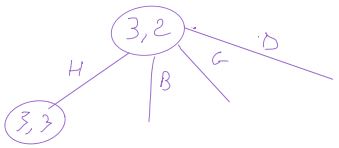
1/ Etats
→ position
↳ coord (x, y)
Etat initial
$O(3,2)$
Actions possibles
$\{H, B, G, D\}$
But
$\{(9,4)\}$

2/ coût
$\text{coût}(\text{act}) = 1 \quad \forall \text{act possible}$
↳ $\text{coût}(\text{solut}) = \text{nb act}$

3/ bcp trop de noeuds ...
($\hat{=} 4^{14}$)

4/ coût identiques, donc revient à

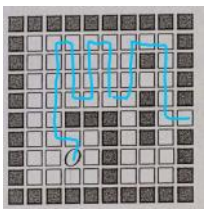
5/



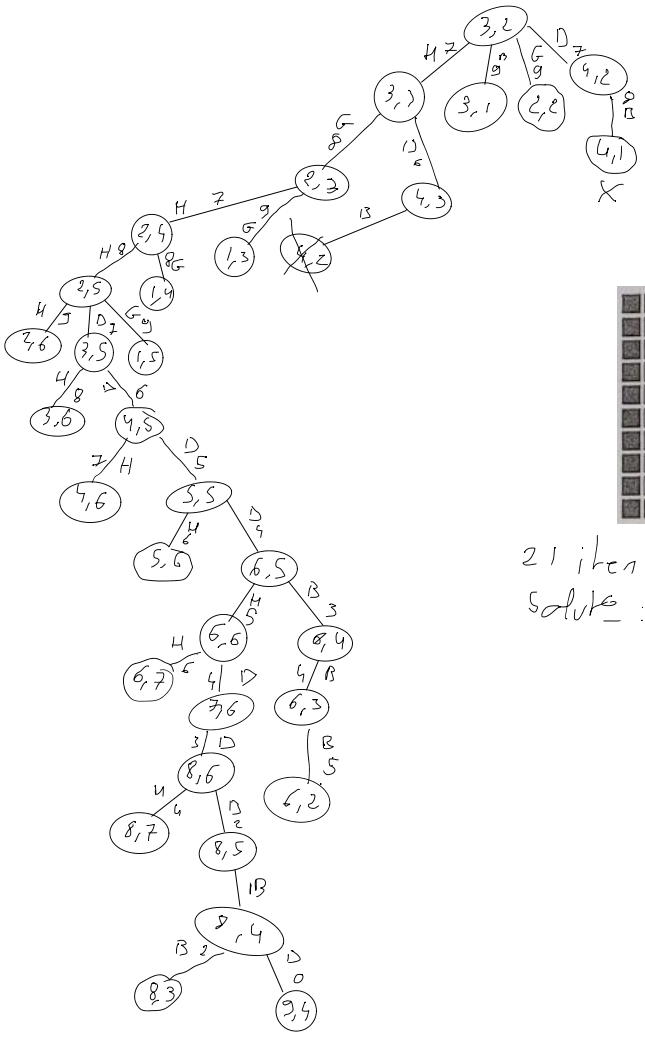
STOP !! on va tourner en rond !

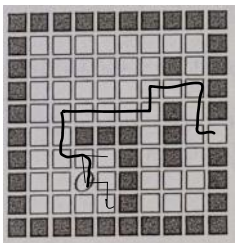
6+7/ légère modif : on vérifie la notion arrivée
⇒ Ne pas repasser au même endroit dans la même branche

53 coups, 78 itérat°, solut° : 30 coups



8/ heuristique :
$h^*(x,y) = \sqrt{(9-x)^2 + (4-y)^2}$
ou encore :
$h(x,y) = |9-x| + |4-y| \Rightarrow$ admissible





21 itérat°
Solut° : 14 (longueur)