République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université des Sciences et de la Technologie
HOUARI BOUMEDIENE
B. P. 32, El-Alia, 16111 Bab-Ezzouar, ALGER
Téléphone/Fax: +213 21 24 76 07

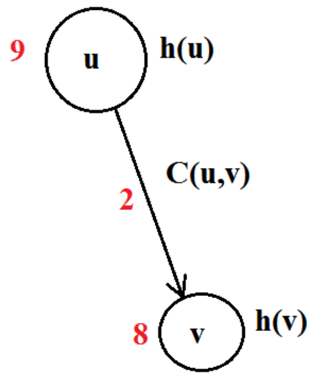
الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي
والبحث العلمي
جامعة هواري بومدين
للعلوم والتكنولوجيا
ص.ب. 32، العاليا، 16111، باب الزوار، الجزائر
الهاتف/ الفاكس : +213 21 24 76 07

Master Informatique Visuelle
USTHB, 2020/2021

Complément du cours
Travaux dirigés 1

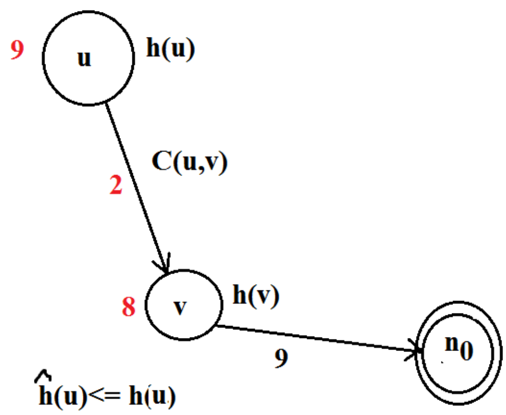
Définitions:

Une heuristique est **consistante (monotone)** si pour tout nœuds u, v connectés, si elle satisfait h(u) ≤ C(u, v) + h(v) où C(u, v) est le coût de transition de u à v.



h(u)<= h(v)+C(u,v)

Une heuristique est **admissible** si pour tout nœud n, $h(n) < \hat{h}(n)$, où $\hat{h}(n)$ est le coût réel de n au nœud objectif n_0.



Un algorithme A* est un algorithme de type A opérant avec une heuristique admissible.

Algorithme de type A : f(n)=g(n)+h(n)
Si h(n) <coût réel de n à no, h est admissible, A devient A*, il trouve le chemin optimal.

Théorème:

Si une heuristique est **consistante**, alors elle est aussi admissible.

L'algorithme A devient A*

Démonstration

Soit N le nombre de nœuds de n à no.
Comme h est consistante, alors $h(n) \leq C(n, n0) + h(n0)$
Comme : $\hat{h}(n) = C(n, n0)$, et $h(n0) = 0$, donc $h(n) \leq \hat{h}(n)$

Supposons que cette propriété pour les k-1 nœuds, montrons qu'elle est valable pour le nœud k.
Soit m le successeur du nœud n.

$$C(n, m) + h(m) \leq C(n, m) + \hat{h}(m) = \hat{h}(n)$$

Comme h est consistante, alors $h(n) \leq C(n, m) + h(m)$
Donc $h(n) \leq \hat{h}(n)$

Propriété:

L'admissibilité signifie que, même lorsque l'espace de recherche est infini, si des solutions existent, une solution sera trouvée et le premier chemin trouvé sera une solution optimale - un chemin à moindre coût d'un nœud de départ à un nœud de but.

Pour l'algorithme A∗, l'optimalité du chemin est garantie lorsque l'heuristique est admissible ou monotone ou consistante.

Quand un nœud est-il visité deux fois?
Considérons le graphique suivant, où l'heuristique satisfait la condition de sous-estimation de la longueur du chemin qui reste à parcourir, mais n'est pas monotone car h(b)>d(b,c)+h(c).
Lorsque nous visitons a, nous ajoutons b et c à la file d'attente avec les évaluations suivantes :

$$f(b)=d(a,b)+h(b)=1+8=9$$

$$f(c)=d(a,c)+h(c)=3+1=4$$

De toute évidence, **c** est visité en premier bien que le chemin le plus court vers c soit en fait via b. Plus tard, lorsque nous visitons b, c est à nouveau visité à partir de b et son poids total est mis à jour à 2.

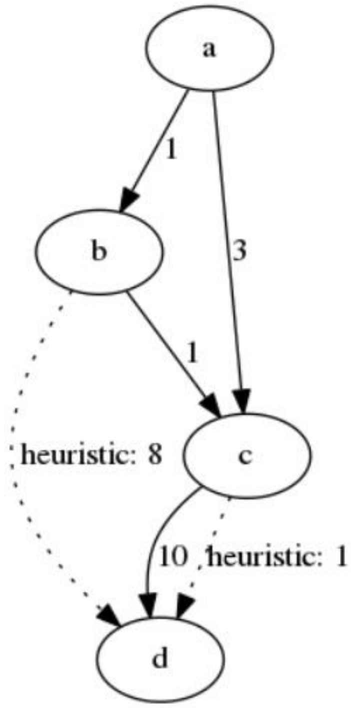


Figure 1. Espace d'états avec coûts et estimations h(n)

Stratégie pour trouver le chemin le plus court sans visiter un nœud deux fois.

Si a → b → c est plus court que a → c, nous voulons d'abord visiter b, donc nous atteignons c depuis b avant de l'atteindre depuis a. Si nous pouvons garantir cela, nous n'avons même pas besoin de le visiter plus tard, car a → b → c est de toute façon plus court.

Cela se résume à: Si

$$d(a,b)+d(b,c)<d(a,c)$$

Si c'est vrai, il faut d'abord visiter b. On peut ajouter h (c) des deux côtés sans rien changer.

$$d(a,b)+d(b,c)+h(c)<d(a,c)+h(c)$$

Si la contrainte monotone est remplie, on peut remplacer d (b, c) + h (c) par h (b), car elle est inférieure ou égale. L'équation

$$d(a,b)+h(b)<d(a,c)+h(c)$$

est toujours vrai. C'est aussi le test que l'algorithme A ∗ utilise pour décider s'il visitera ou non b avant c. Si c'est vrai, il visite d'abord b. C'est exactement ce que nous voulions réaliser.

Exercice 1 :

Un objet se déplace de l'état initial S vers un état final G comme indiqué par la figure 2 où l'on représente l'espace d'états. A chaque état n on associe une heuristique h(n) du chemin qui reste à parcourir de l'état courant vers G. Le coût du passage d'un état vers un autre est indiqué sur l'arc.
Donnez les nœuds visités et le chemin obtenu pour la recherche A avec f(n)=g(n)+h(n), où g(n) est le coût du chemin S vers l'état n. Le chemin trouvé est-il optimal ? Justifiez.
? A*
Réponse :

- Soit on démontre que h(n) est admissible, h(n)<=h*(n)
- Ou bien h est satisfait la contrainte de monotonie : h(n)<= h(n+1)+C(n, n+1)

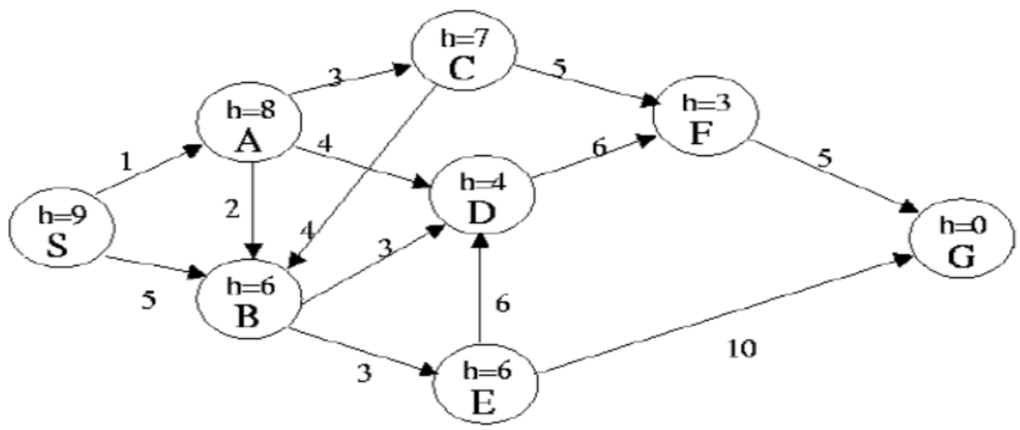


Figure 2. Graphe de recherche

Exercice 2 :

Il s'agit de trouver le chemin le plus court de l'entrée à la sortie d'un labyrinthe en utilisant un algorithme de type A* (voir figure 3). Nous supposons qu'aller d'une cellule à une autre se fait avec un coût égal à 1.

1- Si $h(n)$ est le coût estimé du chemin (de la cellule numéro n vers la cellule de sortie m) définie comme étant la distance de Manhattan (somme des distances horizontale et verticale entre n et m), l'algorithme est-il de type A* ? Cette distance est utile pour le déplacement vers les 4 voisins (horizontaux et verticaux).
2- Si $h(n)$ est le coût estimé du chemin (de la cellule numéro n vers la cellule de sortie m) définie comme étant la distance Diagonale (maximum des distances horizontale ou verticale entre n et m), l'algorithme est-il de type A* ? Cette distance est utile pour le déplacement vers les 8 voisins.
3- ~~Refaire la question (1) en utilisant la distance euclidienne entre n et m. Quelle distance choisir~~ ?

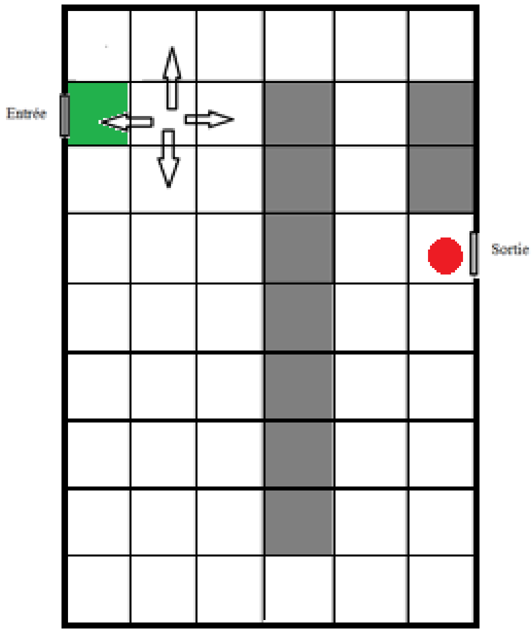


Figure 3. Exemple d'un labyrinthe (case crise représentent les obstacles).