L'algorithme A*

1 Introduction

Il s'agit de l'algorithme le plus célèbre de recherche de chemin. Formellement il s'agit de trouver le plus court chemin (succession d'arêtes à traverser) entre un noeud de départ à un noeud d'arrivée dans un graphe.
Un algorithme de recherche en largeur d'abord, qui consiste à examiner tous les noeuds à une certaine profondeur (nombre d'arêtes traverser depuis le noeud de départ) avant de passer à la profondeur suivante nous assurerait aussi de l'obtention d'un chemin le plus court. Or un tel algorithme est naïf et très couteux. L'idée qui a donnée naissance à l'algorithme itératif A* en 1968 est de donner une "orientation" à cette recherche en largeur. C'est-à-dire que l'algorithme va privilégier et exploiter le noeud le plus prometteur à une profondeur donnée grâce à une heuristique (une instruction plus ou moins exacte qui va nous aider à résoudre le problème). Cela évite ainsi de gaspiller des ressources pour des recherches qui concerneraient des noeuds dont la visite ne résolverait pas le problème. La puissance de l'algorithme A* est qu'il mélange une heuristique guidant "instinctivement" la recherche de la solution à une idée de base de recherche en largeur. En effet, si une solution de chemin envisagée aboutit à une impasse, il sera toujours possible de visiter un noeud voisin inspecté auparavant afin de rechercher une nouvelle solution.

2 Algorithme

A* repose sur l'attribution d'un coût à un déplacement vers un noeud donné. Le coût d'un noeud n noté f(n) d'un noeud est la somme de deux fonctions g et h (notations usuelles). On a donc f(n)=g(n)+h(n). Pour un noeud n, g(n) est le coût du déplacement du point de départ au noeud n. h(n) est notre fonction heuristique, notre estimation du coût qu'il faut encore dépenser afin d'atteindre le point d'arrivée depuis le noeud n. Les noeuds visités sont ensuite classés dans deux listes. La liste ouverte (open set) est la liste des noeuds qu'il reste encore à évaluer. La liste fermée (closed set) est la liste des noeuds évalués et choisis pour constituer le chemin solution.

Le déroulement de l'algorithme est le suivant pour une itération lors du déroulement de l'algorithme :
1) On choisit le noeud n au coût le plus faible de la liste ouverte (on note que au début de l'algorithme il n'y a que la case de départ dans la liste ouverte avec un coût par défaut de 0, ce noeud est donc naturellement choisi). On retire n de la liste ouverte.
2) On ajoute n à la liste fermée.
3) Si n est la case d'arrivée, on peut donner le chemin solution et l'algorithme se termine.
Sinon pour tous les noeuds successeurs de n on calcule leur coût par la fonction f. Et si un successeur donné se trouve déjà dans la liste ouverte, on met à jour son coût que l'on vient de calculer s'il est plus faible que le précédent. Ou sinon si ce successeur ne se trouve pas dans la liste ouverte (ni dans la liste fermée) on l'ajoute.

L'algorithme se termine avec une solution ou aucune (dans le cas où le chemin demandé n'existe pas).

Bibliographie :

http ://khayyam.developpez.com/articles/algo/astar/
http ://web.mit.edu/eranki/www/tutorials/search/
http ://ee.usc.edu/ redekopp/cs102/L15_AStar.pdf
http ://theory.stanford.edu/ amitp/GameProgramming/AStarComparison.html
http ://www.cse.lehigh.edu/ munoz/CSE497/classes/Astar.ppt