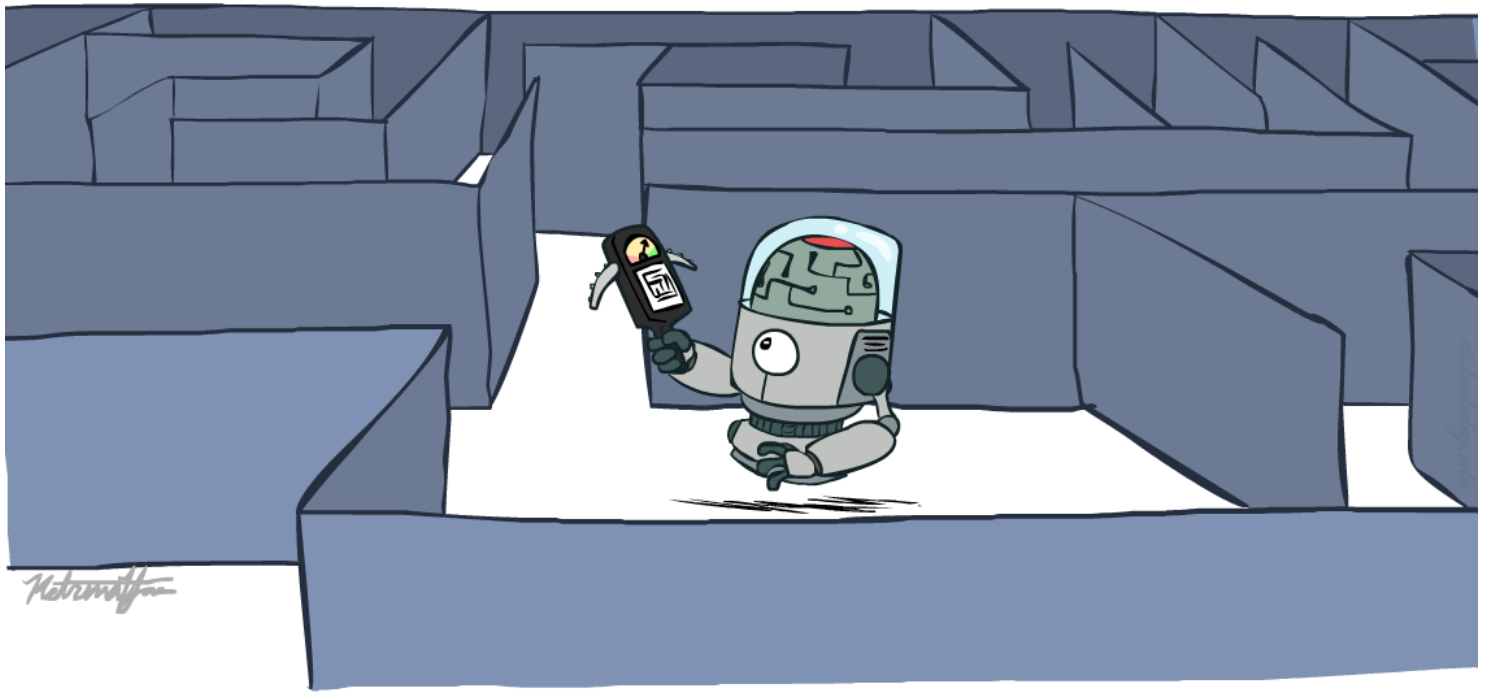
Intelligence Artificielle

Exploration informée



Professeur: A.Belcaid

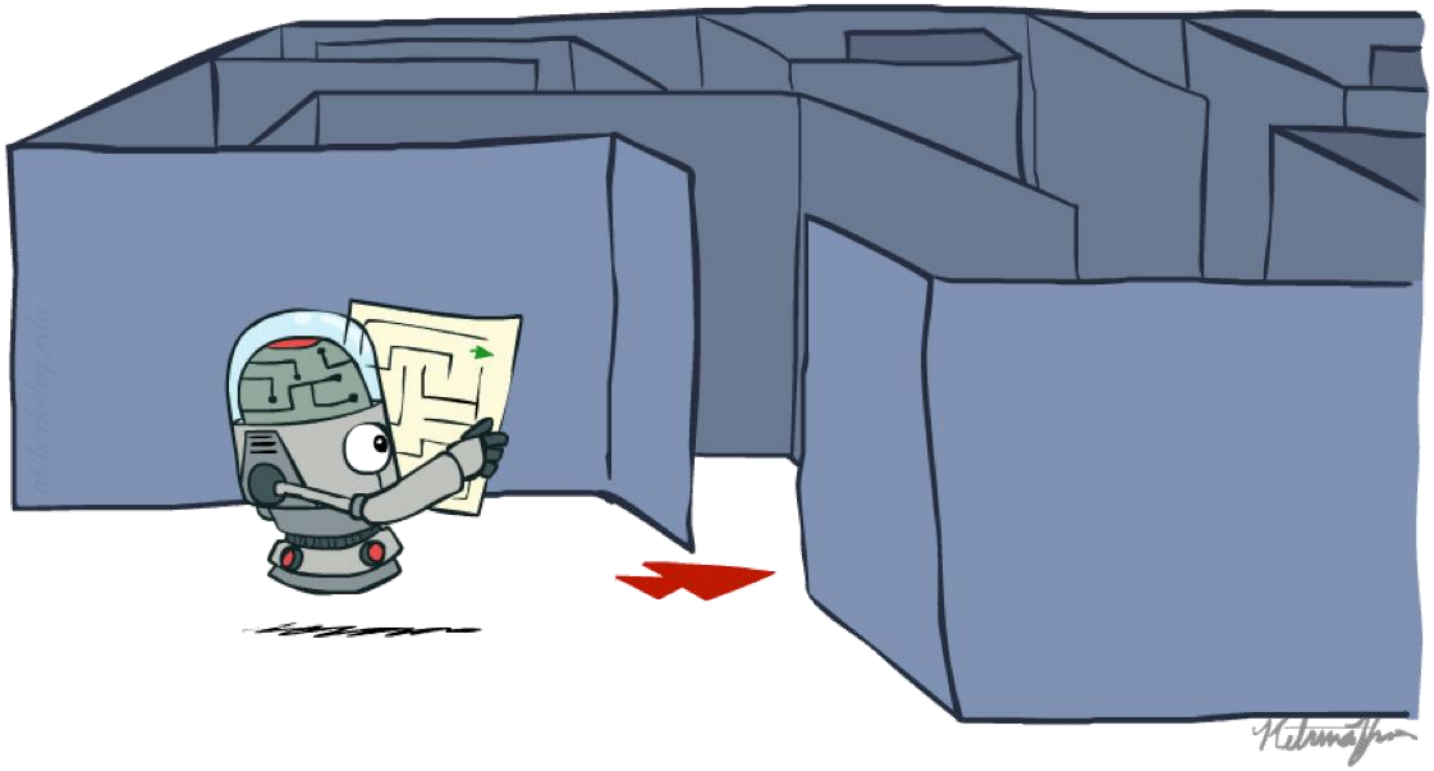
Ecole Nationale des Sciences Appliquées-Fès

[Contenu basé sur le cours CS188 Intro to AI at UC Berkeley.]

Contenu

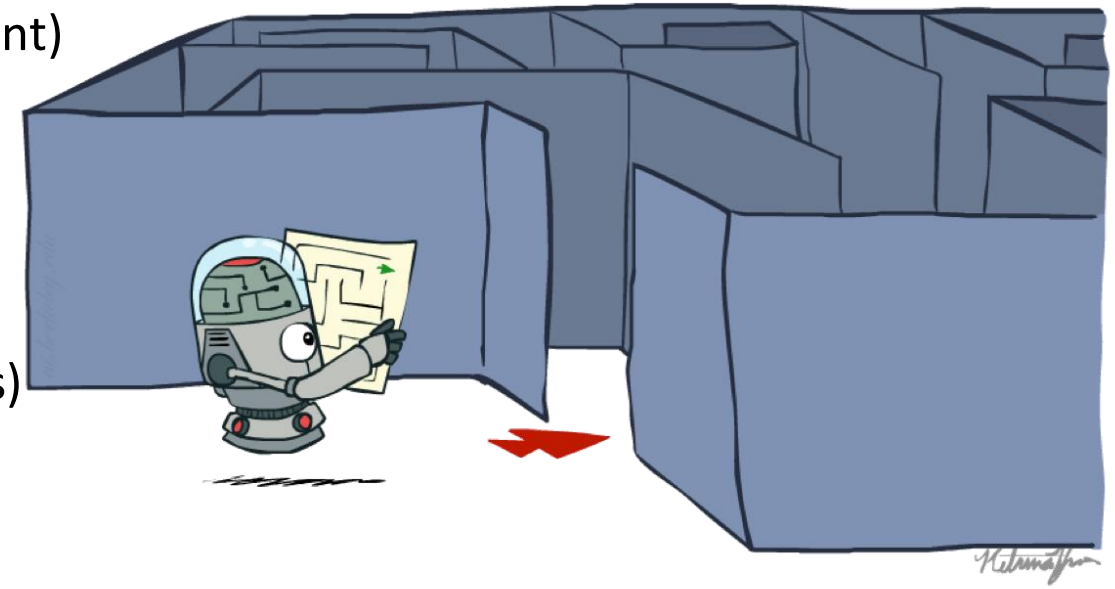
- Exploration informée
  - Heuristiques
  - Exploration Glouton
  - Exploration A*
- Exploration Graphe

Resumé: Exploration

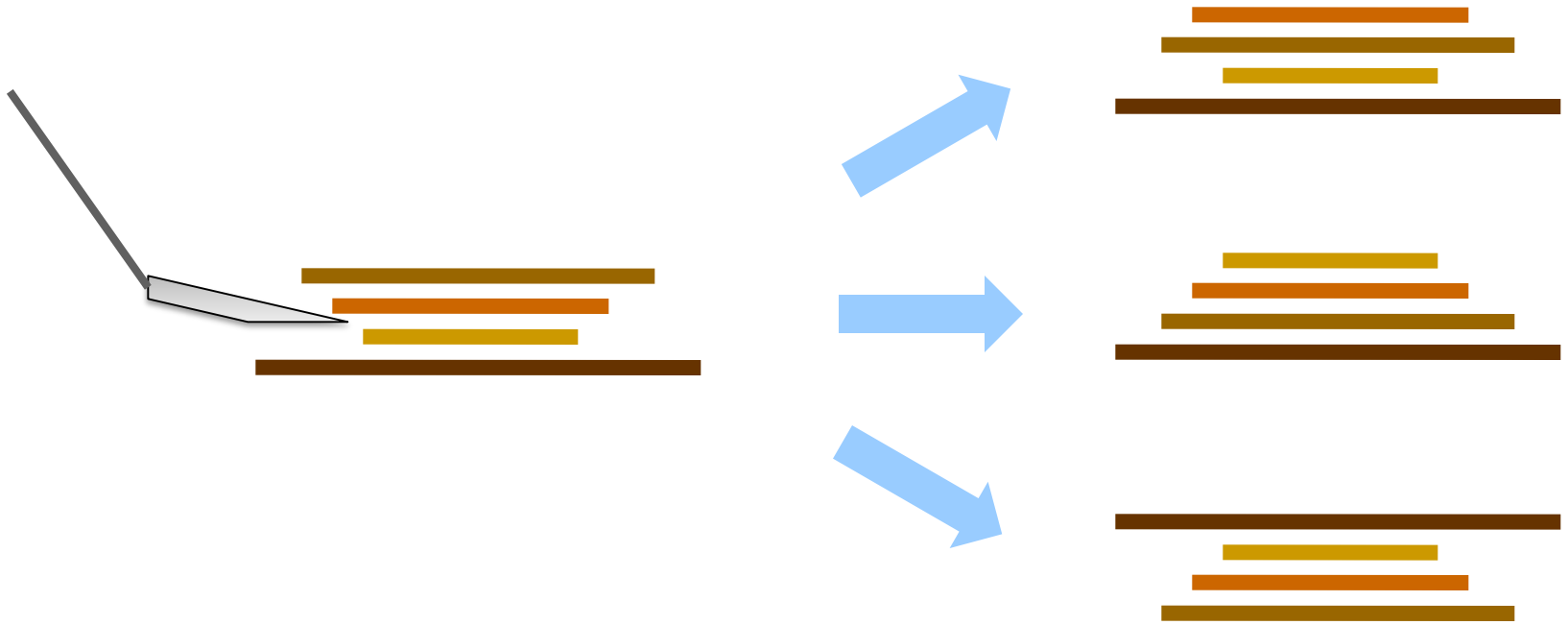


Resumé:Exploration

- Exploration non informée:
  - Etat (configurations de l'environnement)
  - Actions et leurs coûts
  - Fonction de succession (dynamique de l'environnement)
  - Etat final ou objectif
- Exploration en arbre:
  - Noueds representent des plans.
  - Plans possèdent un coût (somme des coût des actions)
- Algorithme d'exploration:
  - Construire systématiquement l'arbre d'exploration
  - Choix d'un ordre de devellopement des noeuds de la frontière
  - Optimalité: Plan le moins couteux.



Example: Pancake Problem



Coût: Nombre de crêpes tournée.

Exemple: Le Problème des crêpes

BOUNDS FOR SORTING BY PREFIX REVERSAL

William H. GATES
Microsoft, Albuquerque, New Mexico

Christos H. PAPADIMITRIOU*†
Department of Electrical Engineering, University of California, Berkeley, CA 94720, U.S.A.

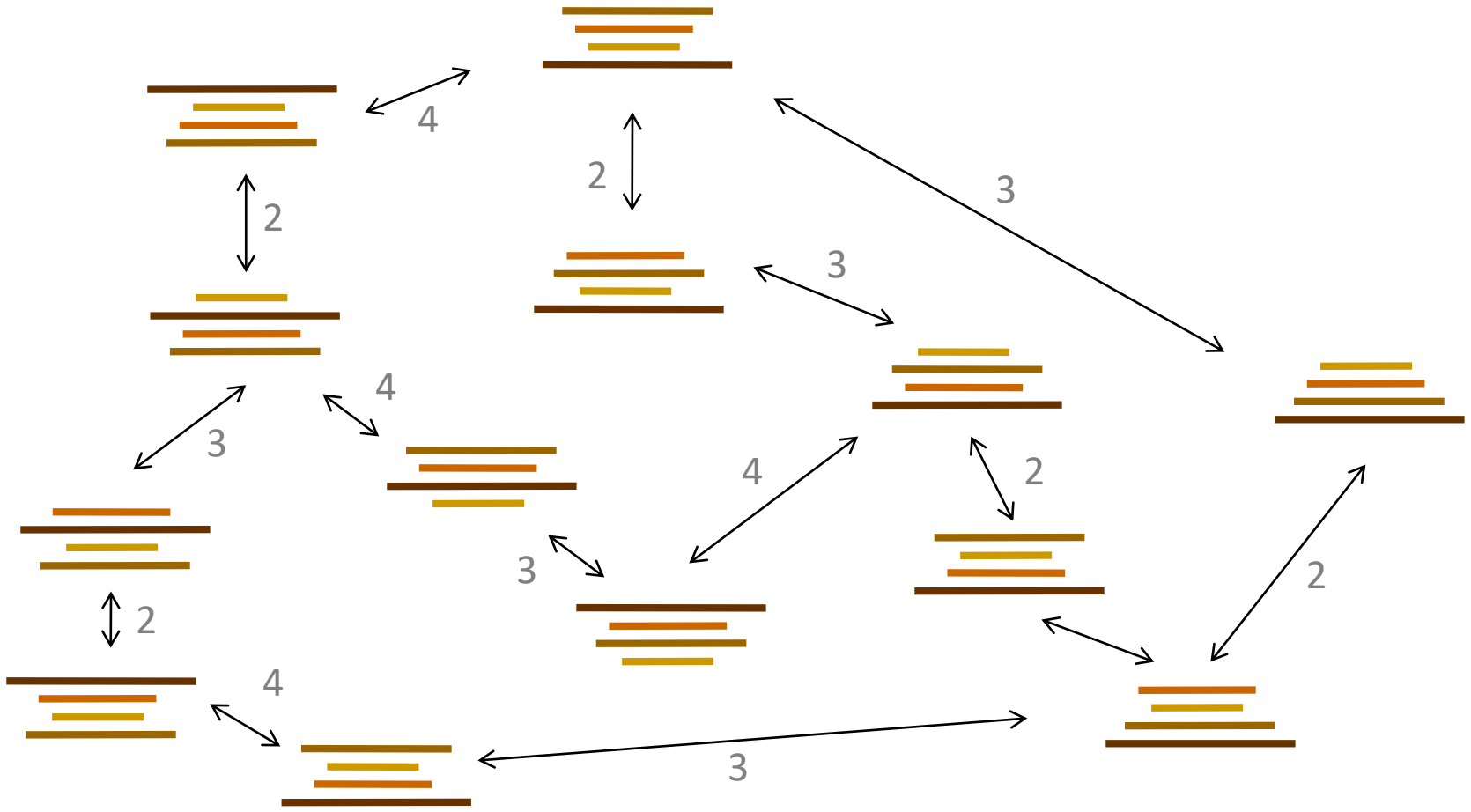
Received 18 January 1978
Revised 28 August 1978

For a permutation σ of the integers from 1 to n, let $f(\sigma)$ be the smallest number of prefix reversals that will transform σ to the identity permutation, and let $f(n)$ be the largest such $f(\sigma)$ for all σ in (the symmetric group) S_n. We show that $f(n) \leqslant (5n+5)/3$, and that $f(n) \geqslant 17n/16$ for n a multiple of 16. If, furthermore, each integer is required to participate in an even number of reversed prefixes, the corresponding function $g(n)$ is shown to obey $3n/2-1 \leqslant g(n) \leqslant 2n+3$.

Gates, W. and Papadimitriou, C., "Bounds for Sorting by Prefix Reversal.", Discrete Mathematics. 27, 47-57, 1979.

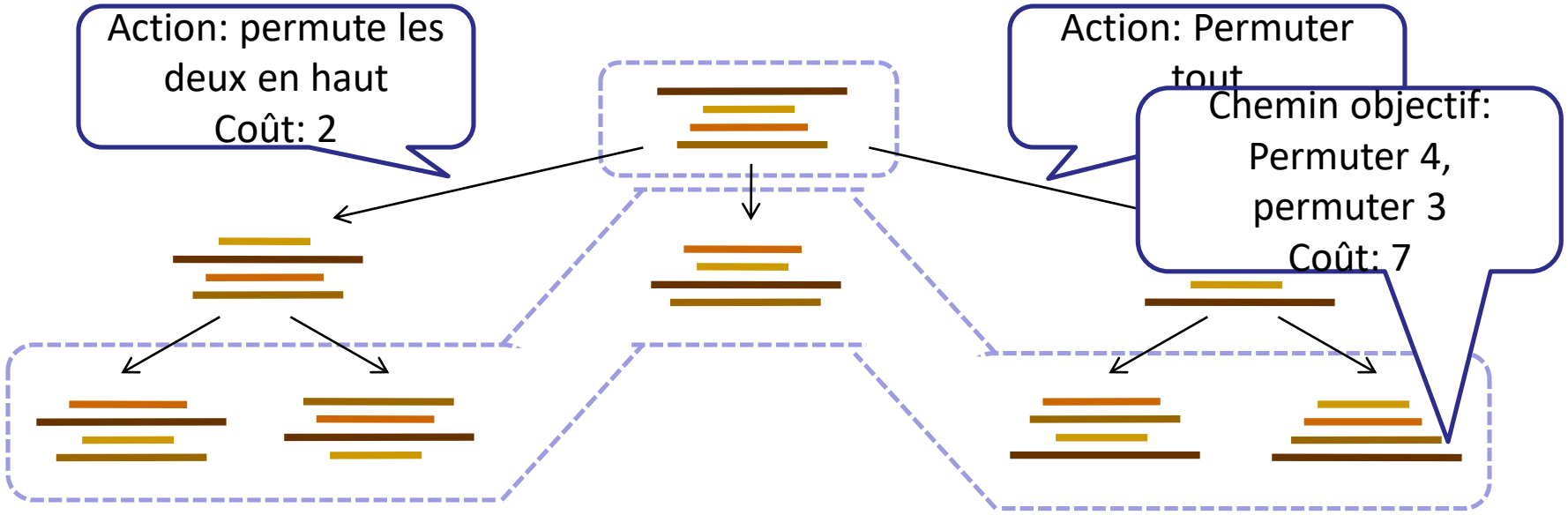
Example: Pancake Problem

Graphe pondéré des états

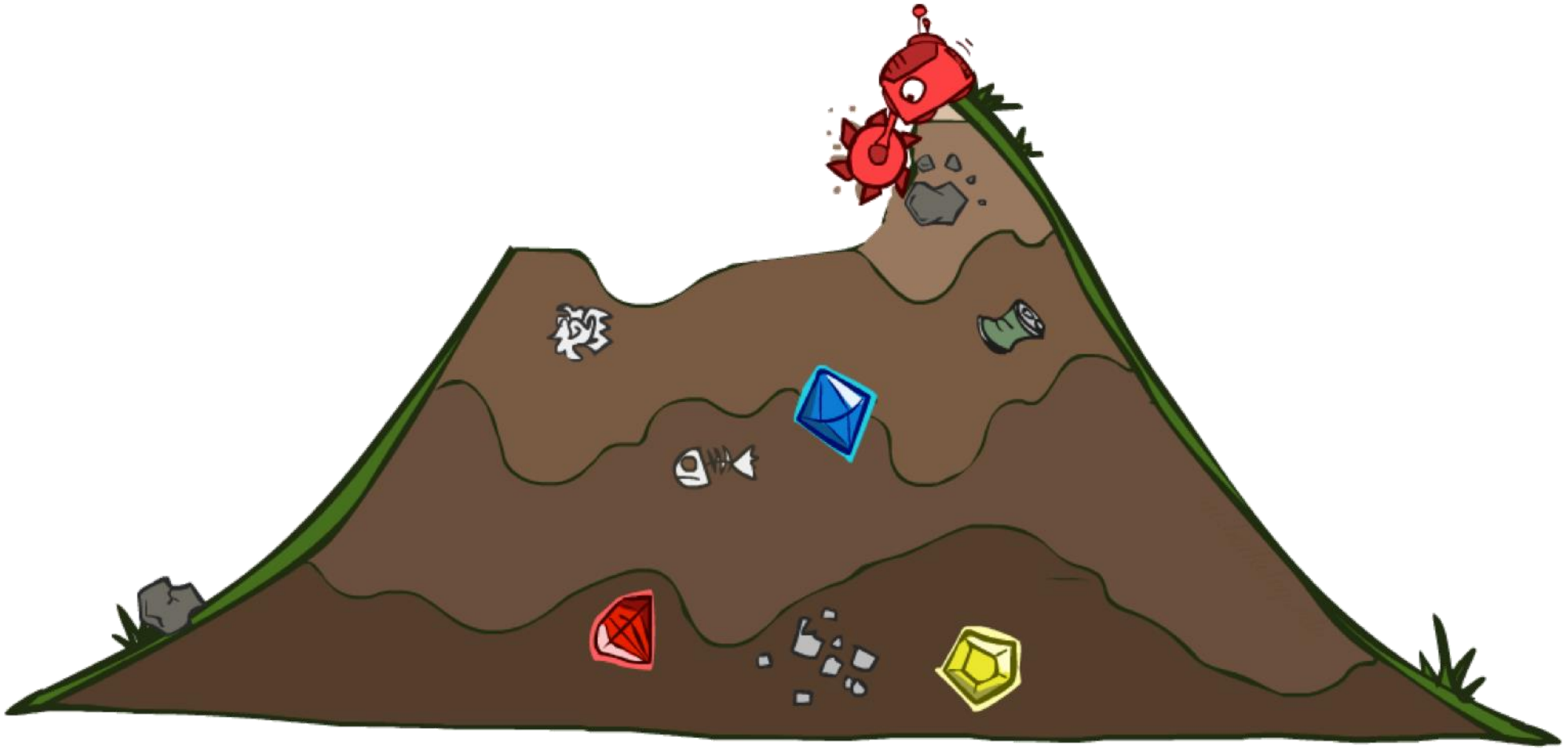


General Tree Search

```
function TREE-SEARCH( problem, strategy) returns a solution, or failure
    initialize the search tree using the initial state of problem
    loop do
        if there are no candidates for expansion then return failure
        choose a leaf node for expansion according to strategy
        if the node contains a goal state then return the corresponding solution
        else expand the node and add the resulting nodes to the search tree
    end
```

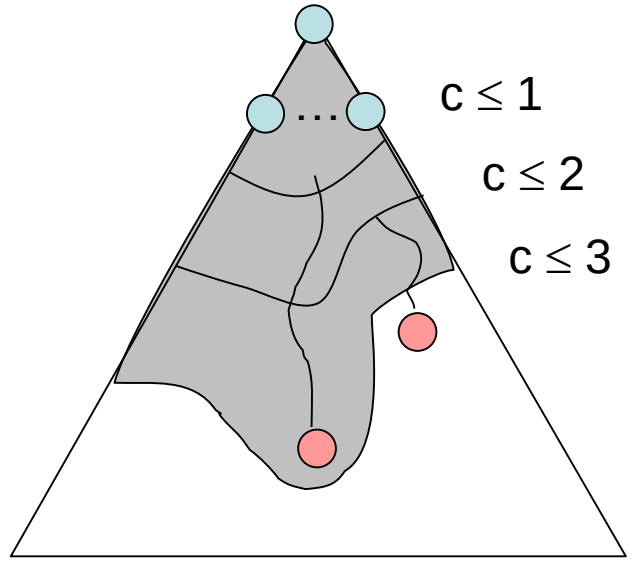


Exploration non informée



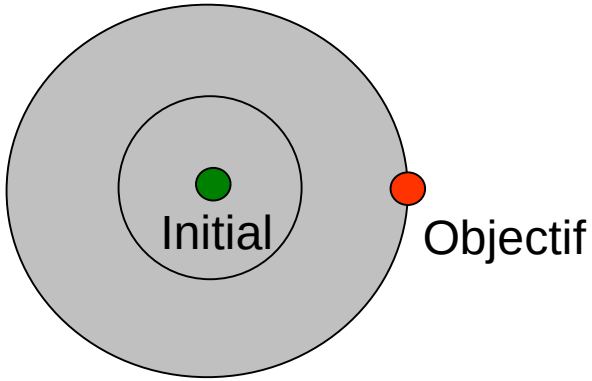
Uniform Cost Search (UCS)

- Stratégie: Développer les noeuds les moins couteux



- Avantages : UCS is complet et optimal

- Inconvénients:
  - Explore dans toutes les **directions** possibles.
  - Aucune information sur l'état objectif.

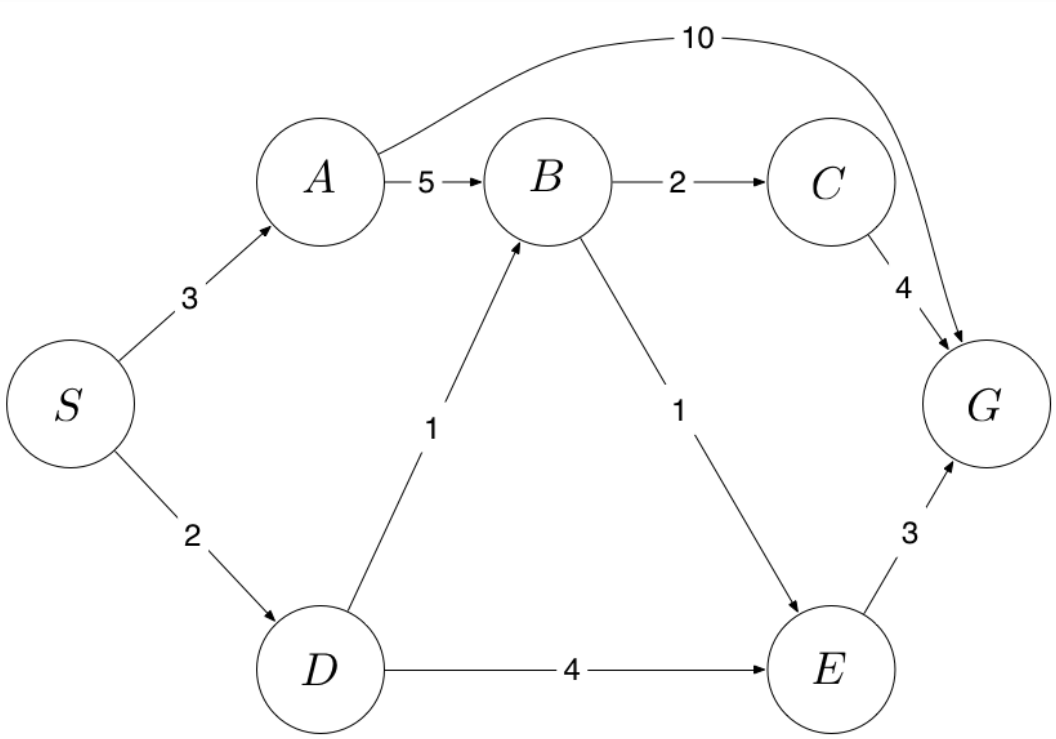


Quiz 1

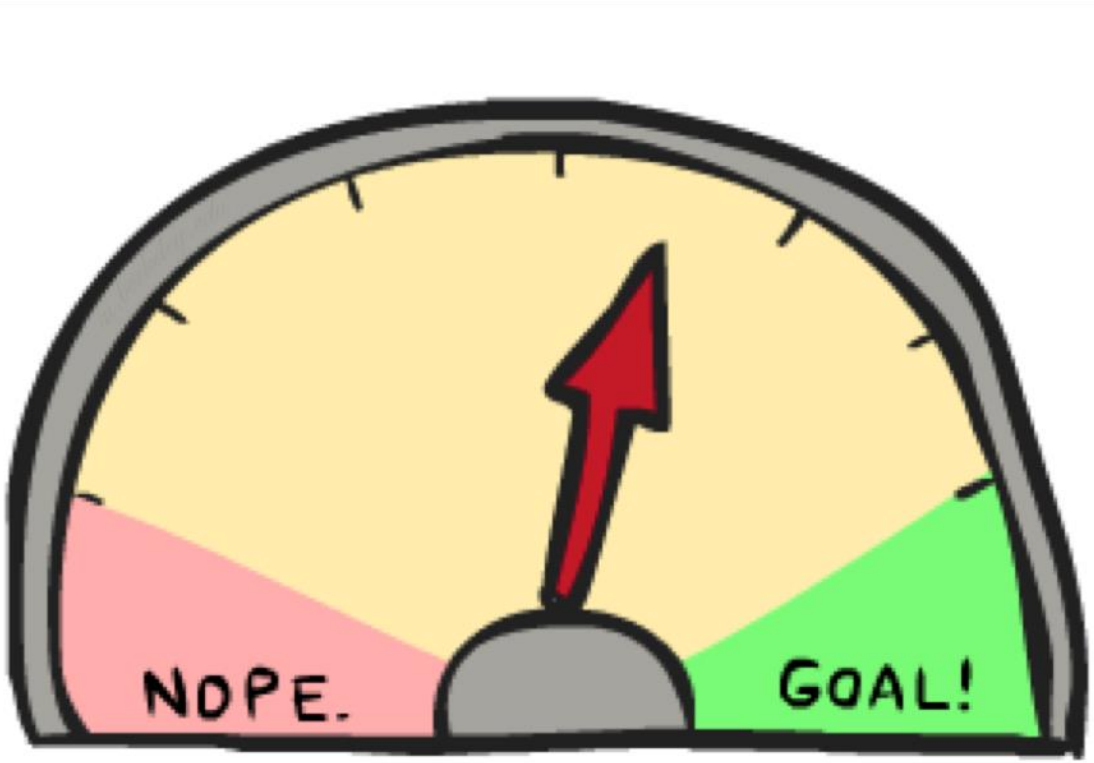
1. Donner la solution retrouvée par **DFS**.

2. Donner la solution calculée par **BFS**

3. Donner la solution calculée par **UCS**

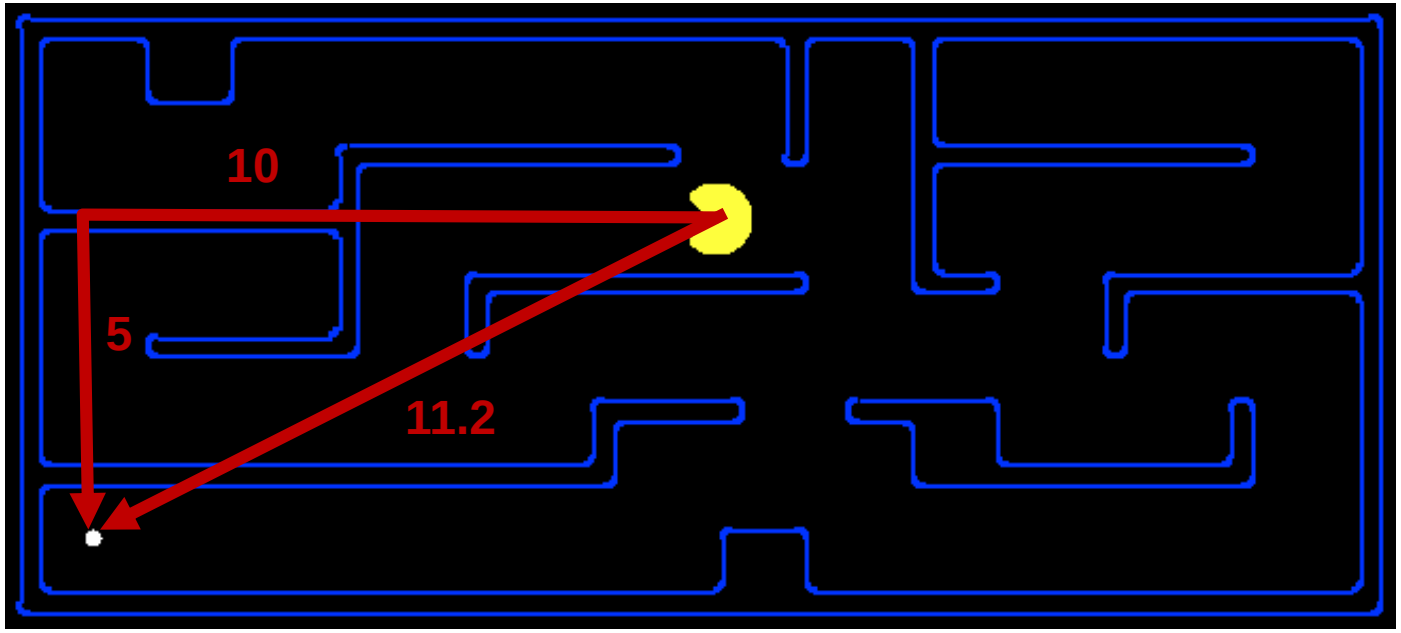


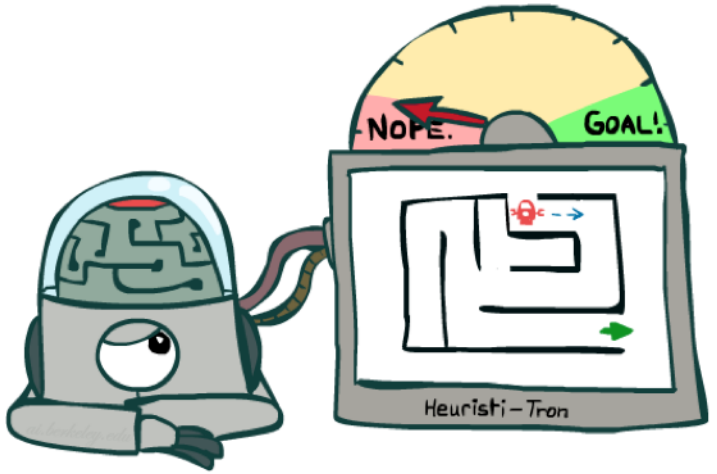
Exploration informée

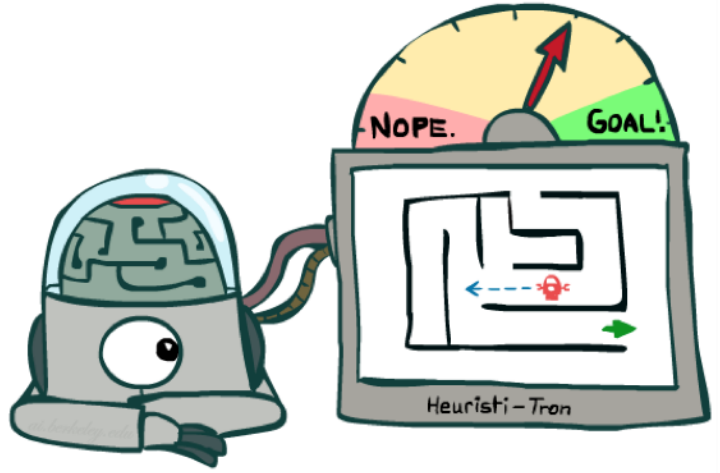


Heuristiques

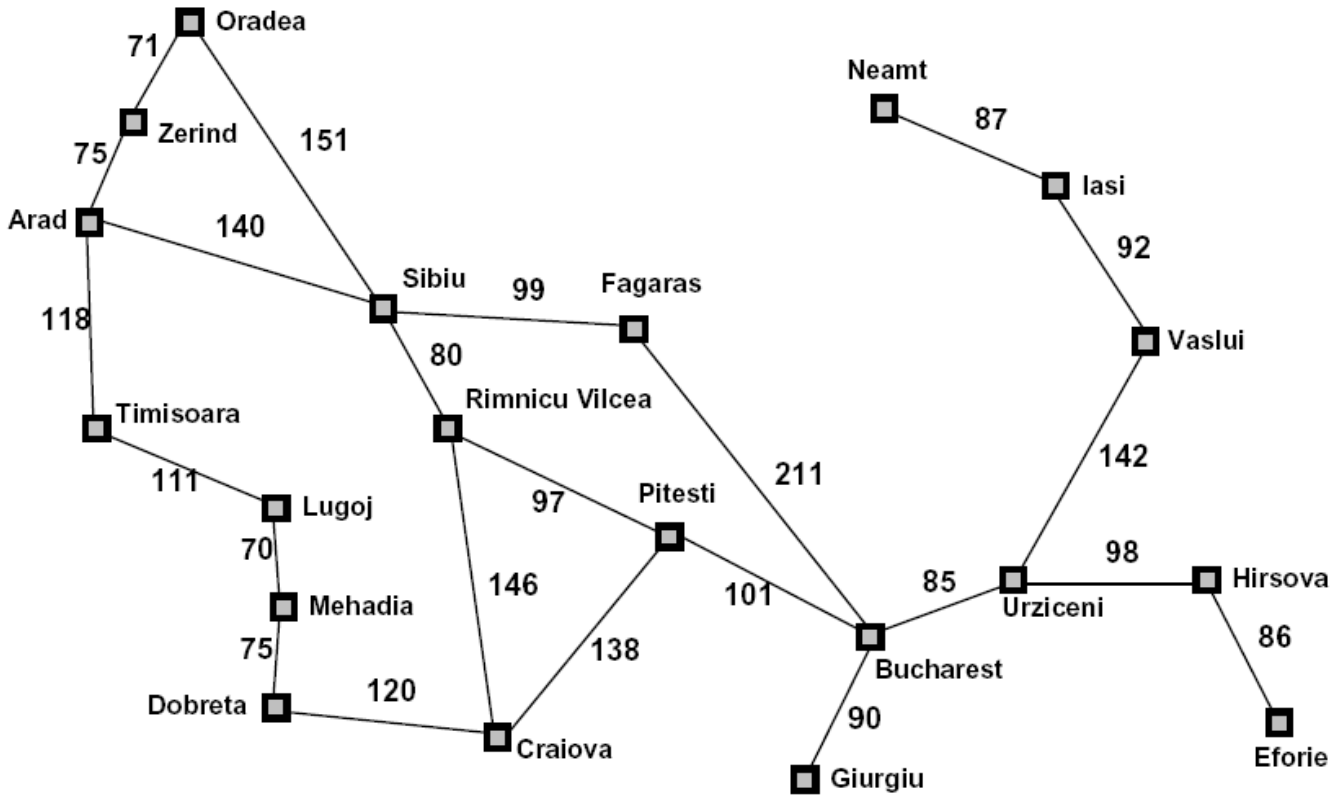
- Une heuristique est:
  - Une fonction pour *estimer le* ***coût*** d'un état à l'objectif.
  - Utilise les caractéristique du problème.
  - Exemples:
    - Distance Manhattan.
    - Distance Euclidienne







Exemple d'une heuristique

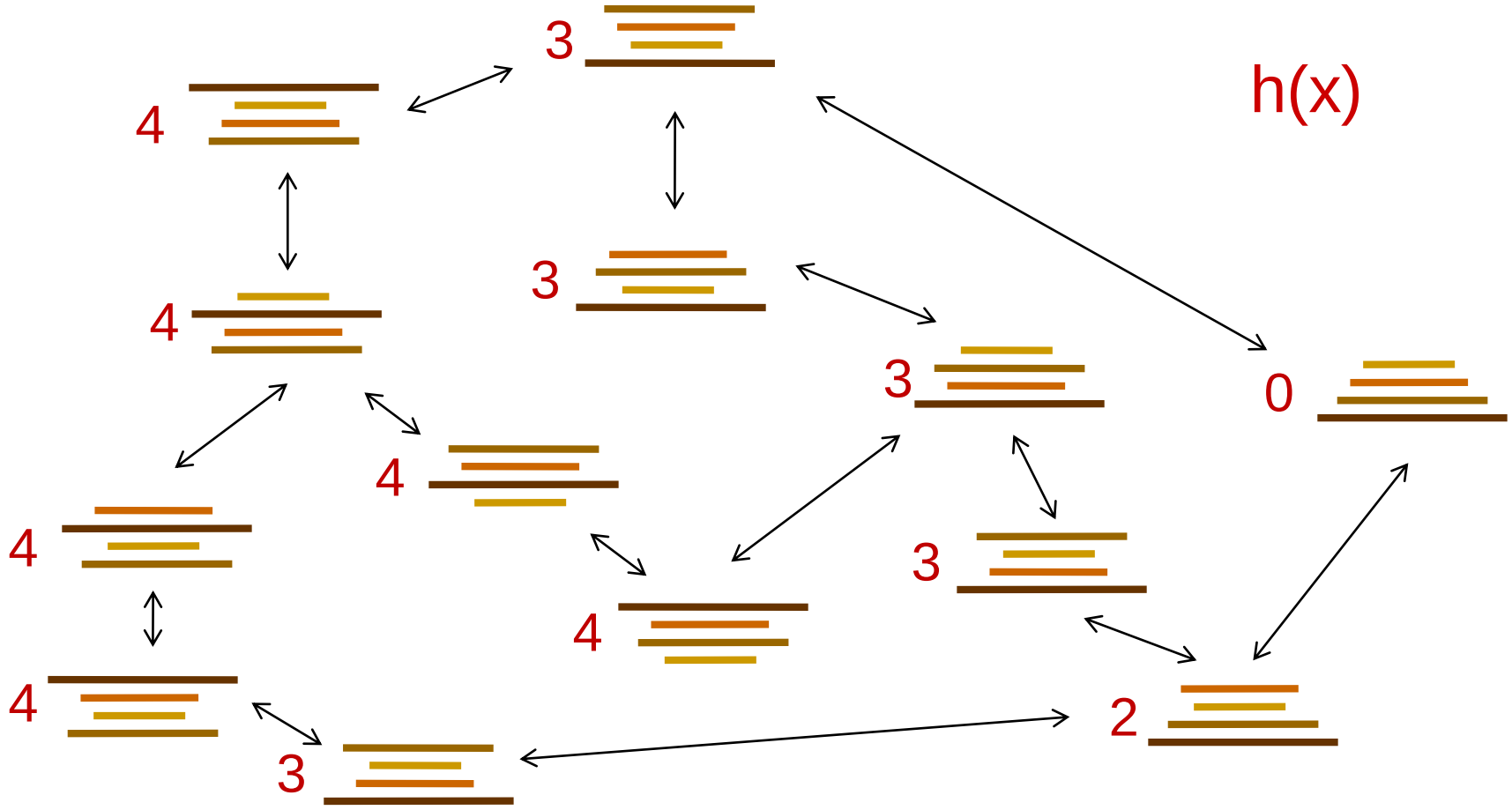


Straight−line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

h(x)

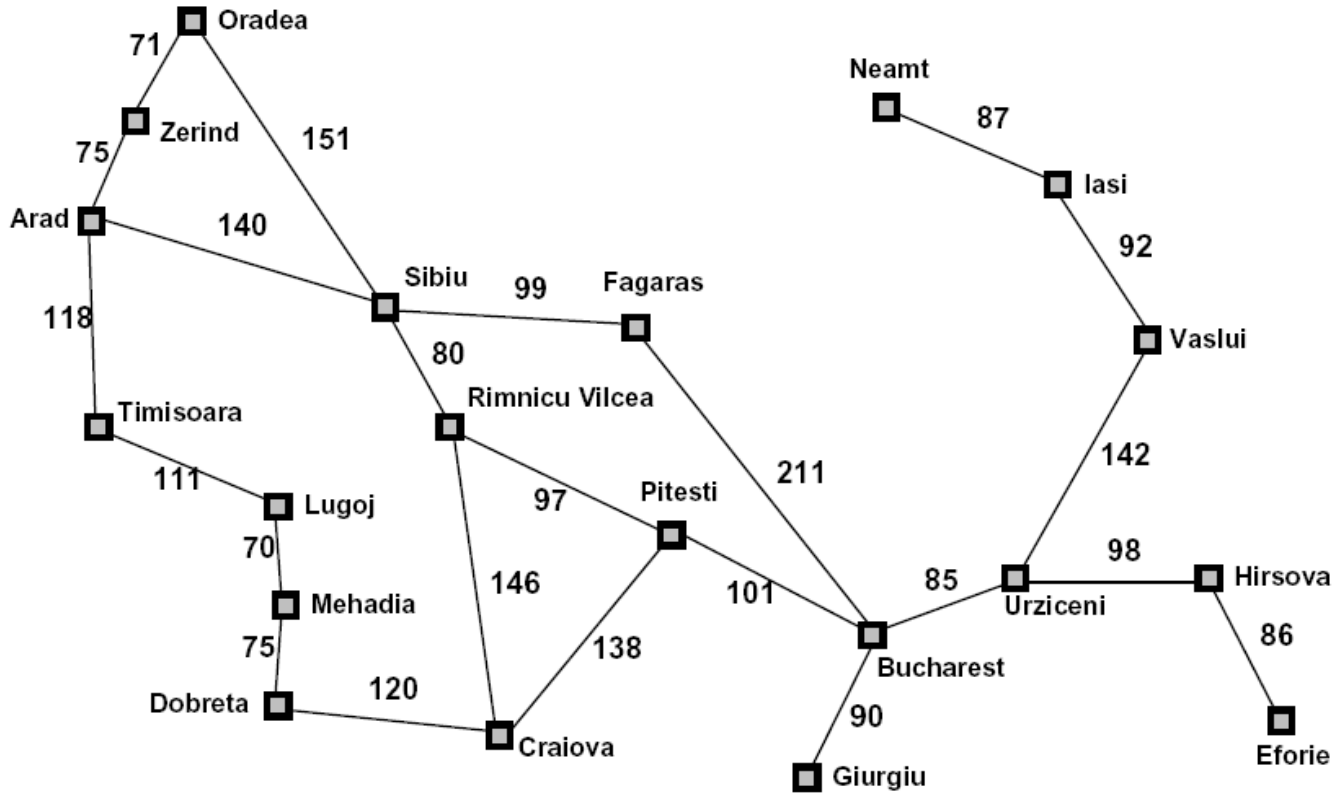
Exemple d'une function heuristique

Le numéro de la plus **large** crêpe mal placée.



Exploration Glouton

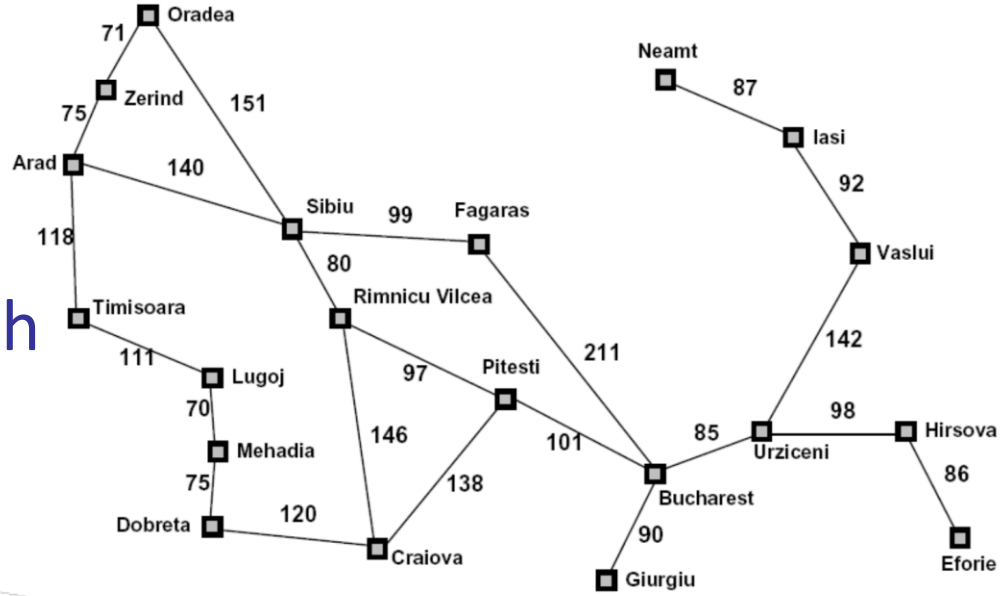
Exemple:Fonction heuristique



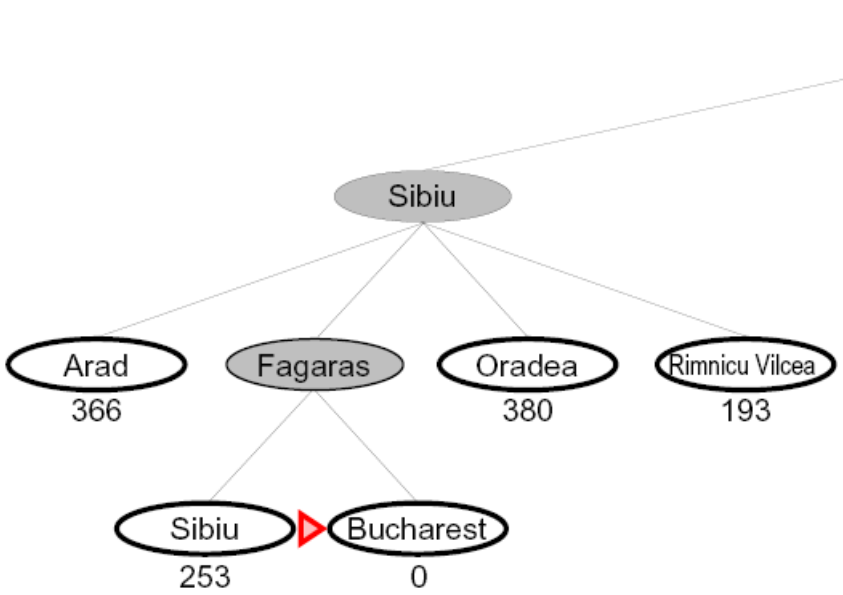
Straight−line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

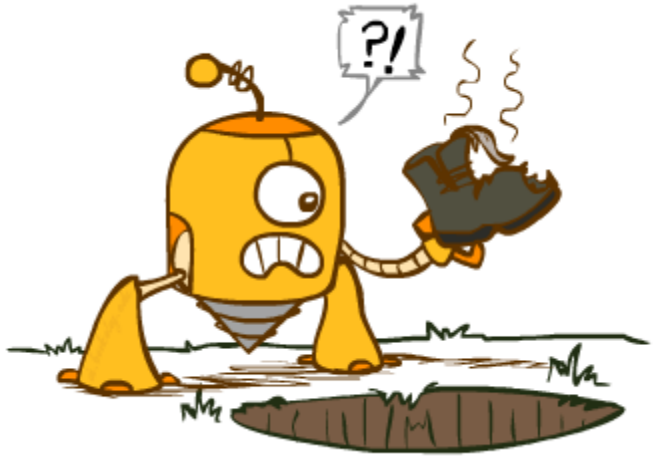
h(x)

Exploration Glouton



- Développer le noeud le plus proche selon h

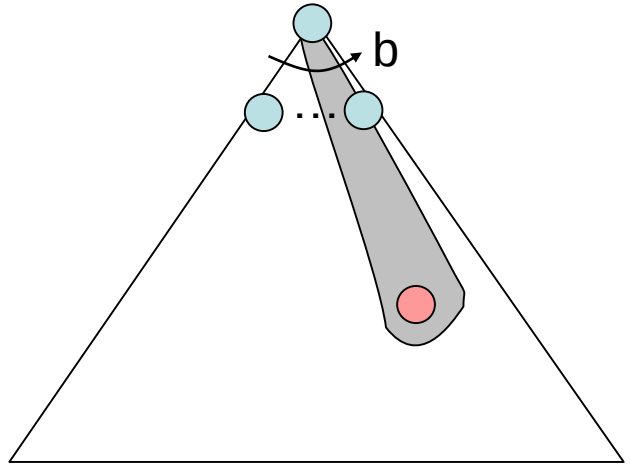




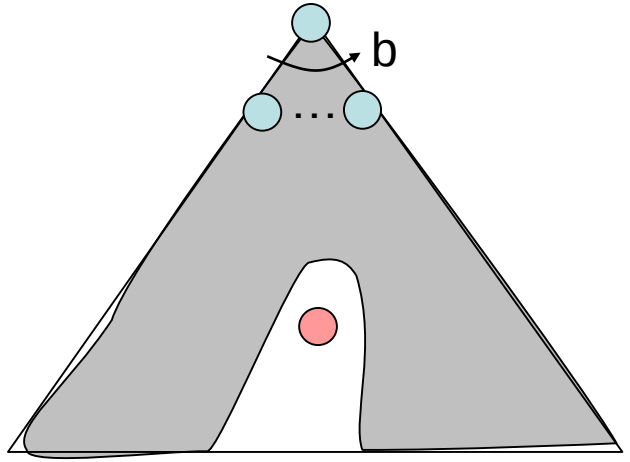
- Quel est le problèm?

Exploration Glouton (Greedy)

- Strategie: Explorer le noeud le plus proche selon l'heuristique
  - Heuristique: estimation de la distance de l'objectif.

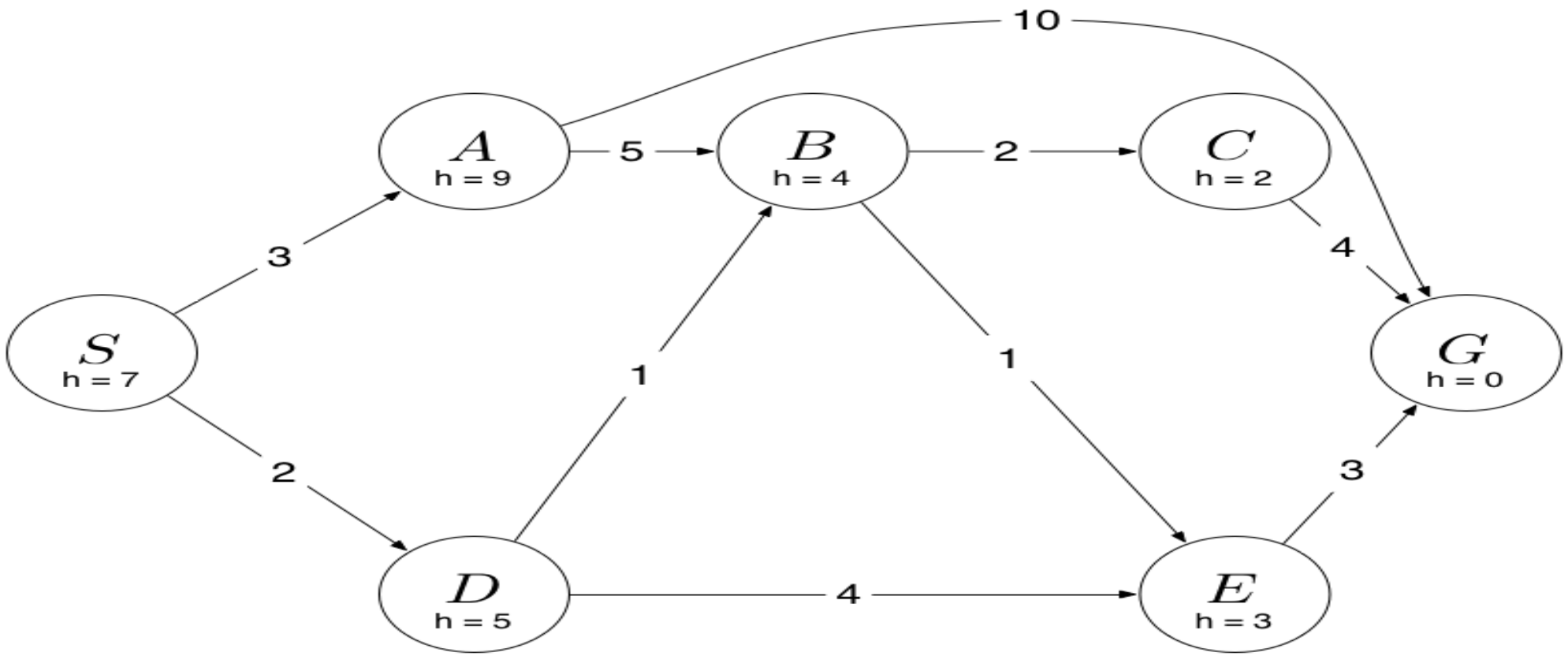


- Un cas commun:
  - Décision basé seulement sur une heuristique est fausse.



- Pire cas:
  - Explorer tous les noeuds

Quiz 2



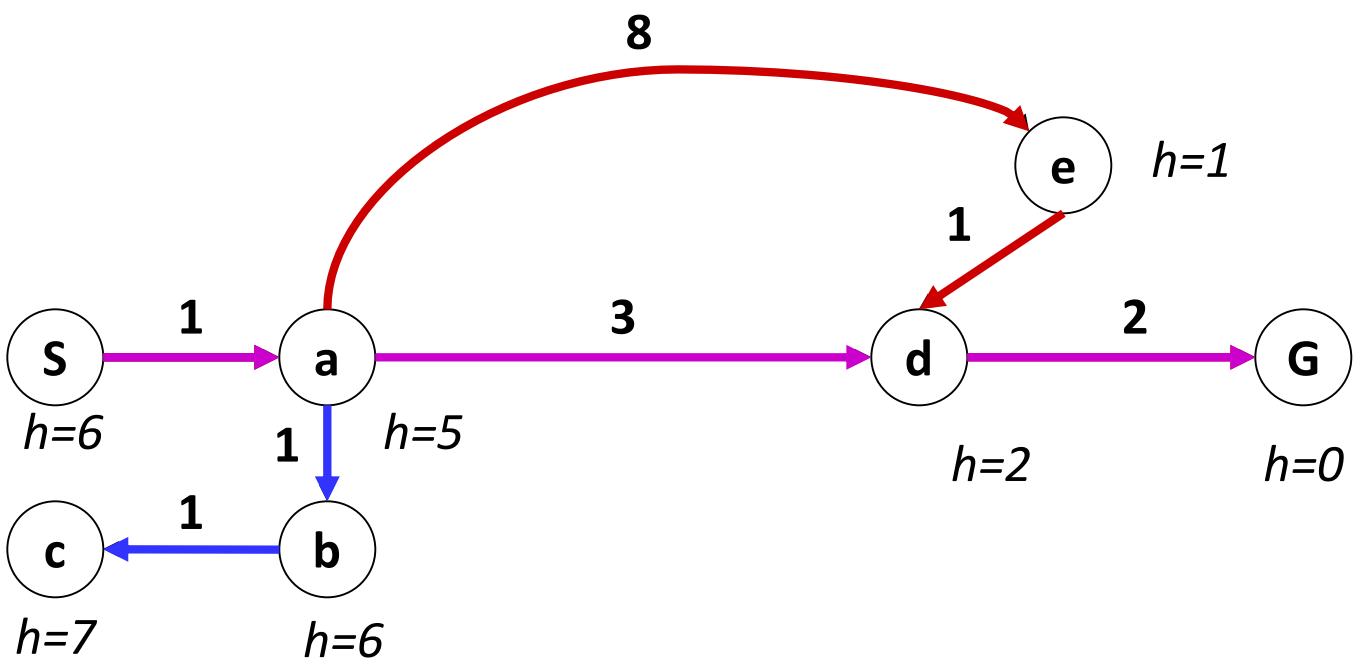
Donner le chemin calculé par l'exploration Glouton

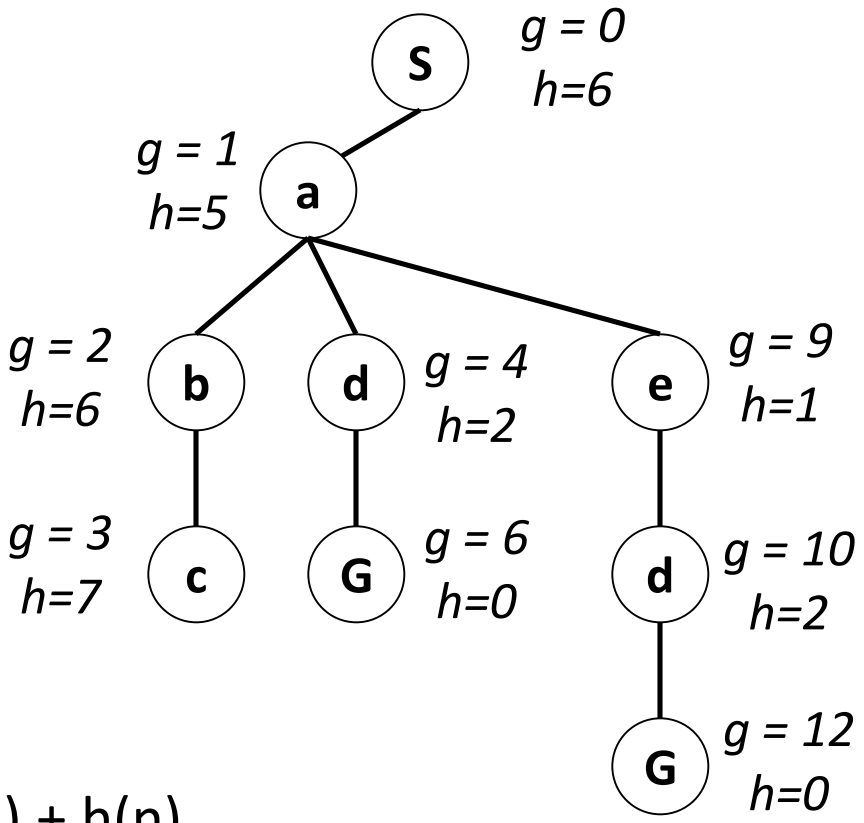
Exploration A*

A* Search

Combining UCS and Greedy

- Uniform-cost : priorité selon le chemin retrograde g(n)
- Greedy : priorité selon le chemin selon l'heuristique h(n)



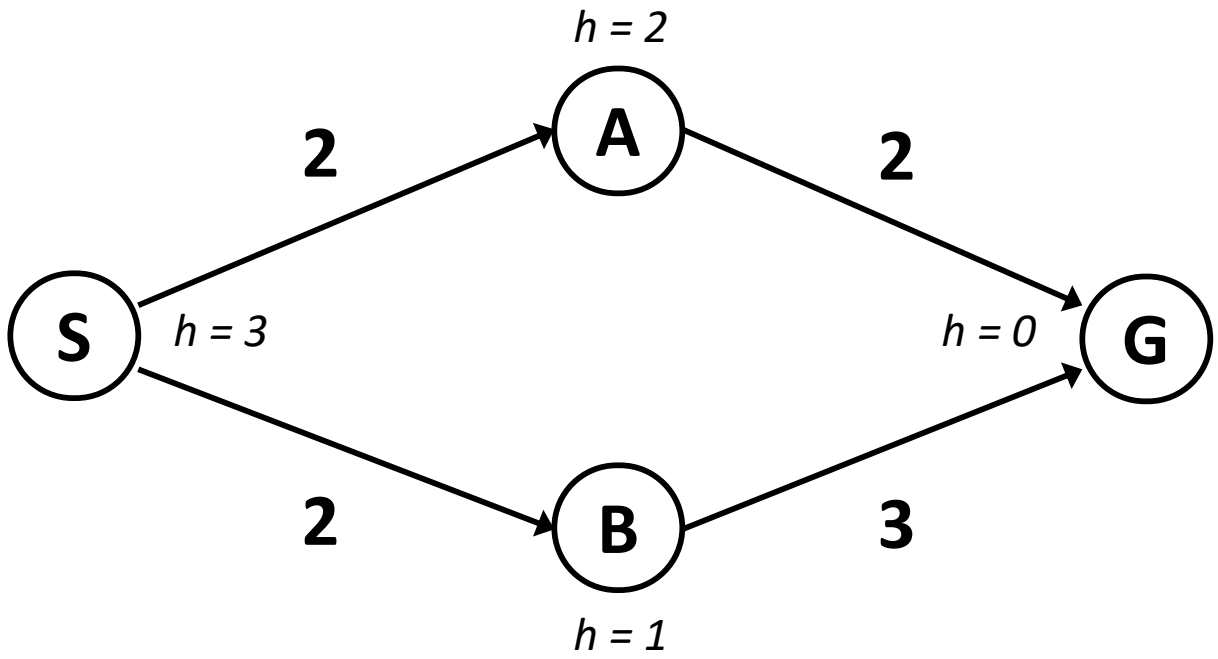


- A* Search priorité selon la somme: f(n) = g(n) + h(n)

Exemple: Teg Grenager

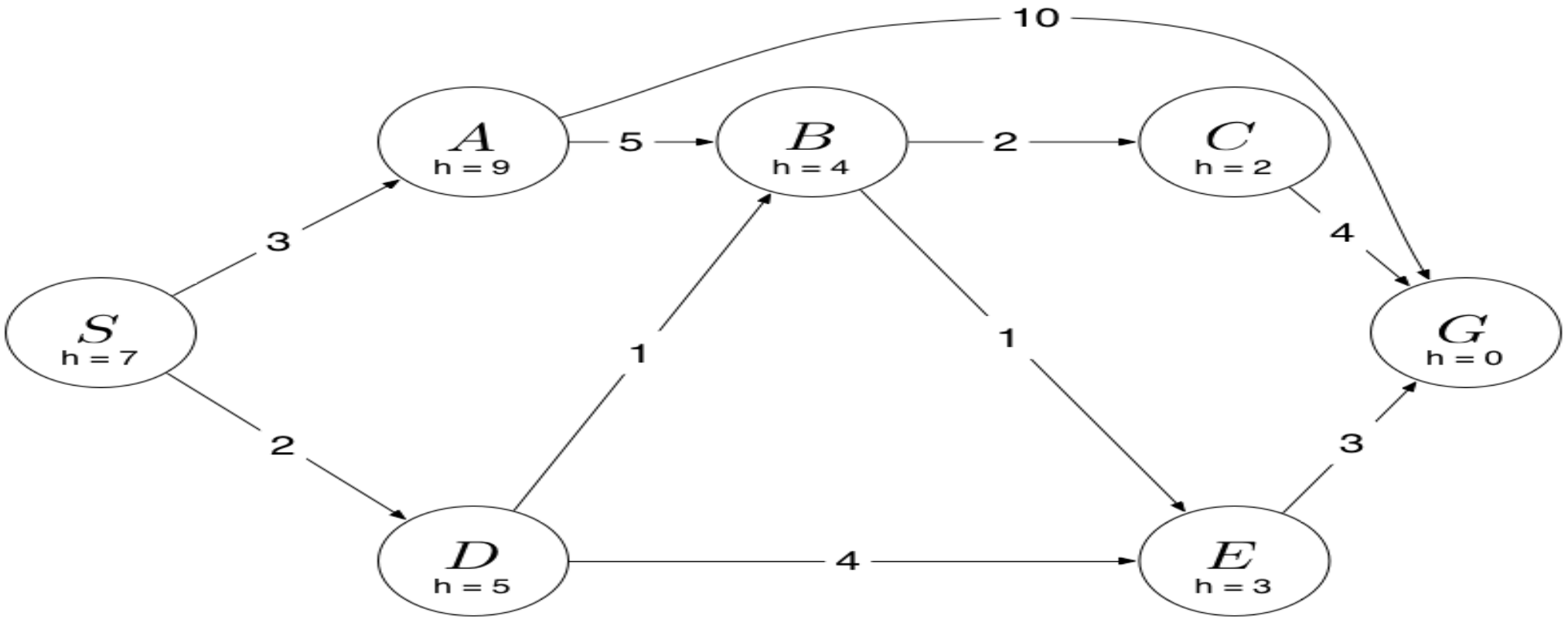
Quand est ce que A* doit terminer?

- Doit on s'arrêter quand on trouve l'objectif?



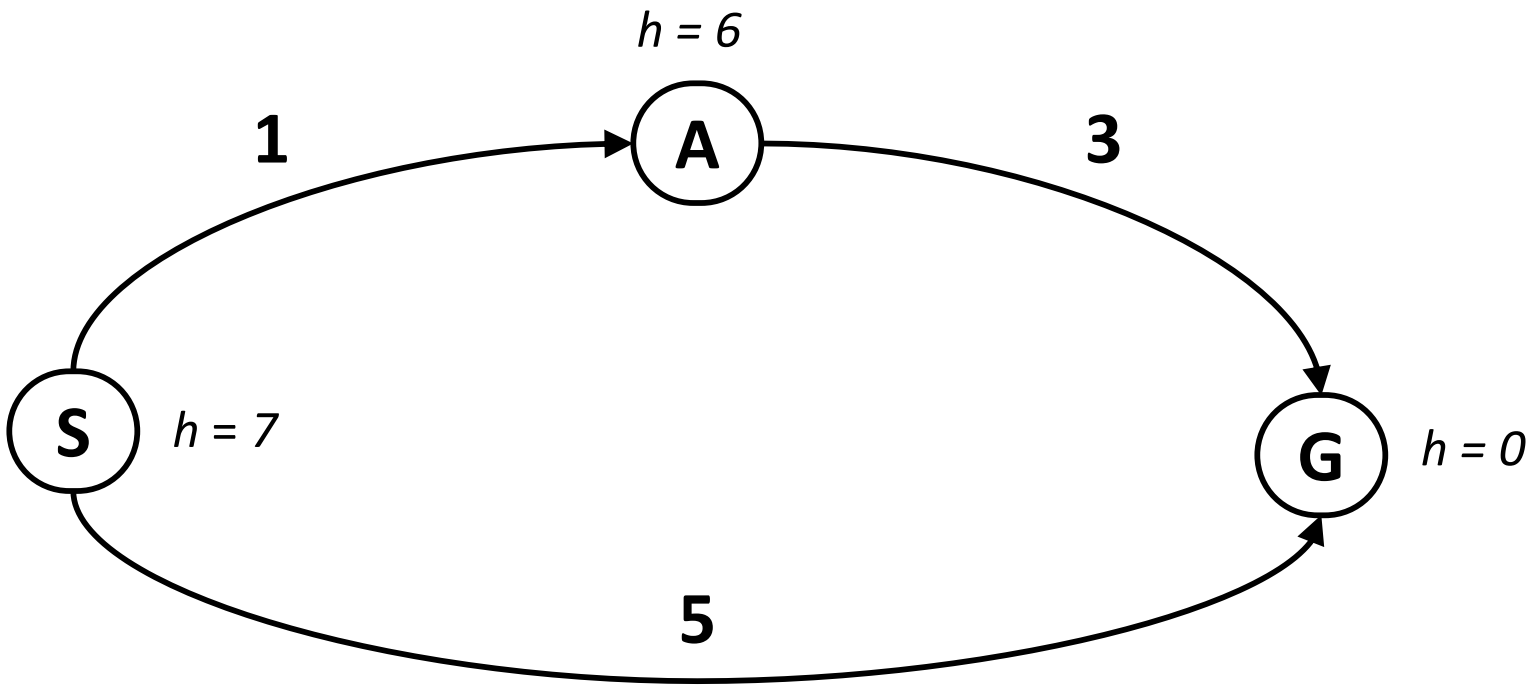
- Non: On s'arêtte quand on récupère l'objectif de la frontière.

Quiz 3



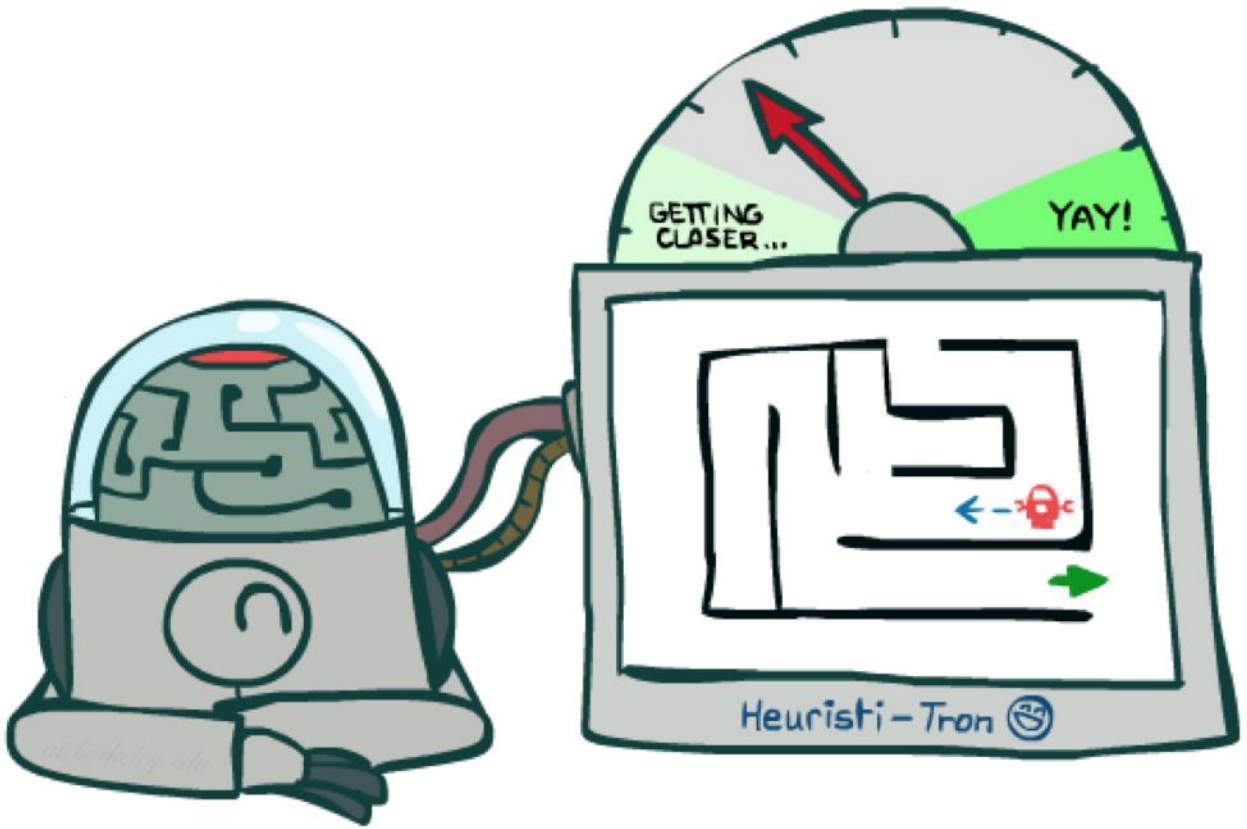
1. Calculer f(S→A)
2. Calculer f(S→D)
3. Quel est le nœud qui sera développé en premier **A** ou **B.**

Optimalité de A*?

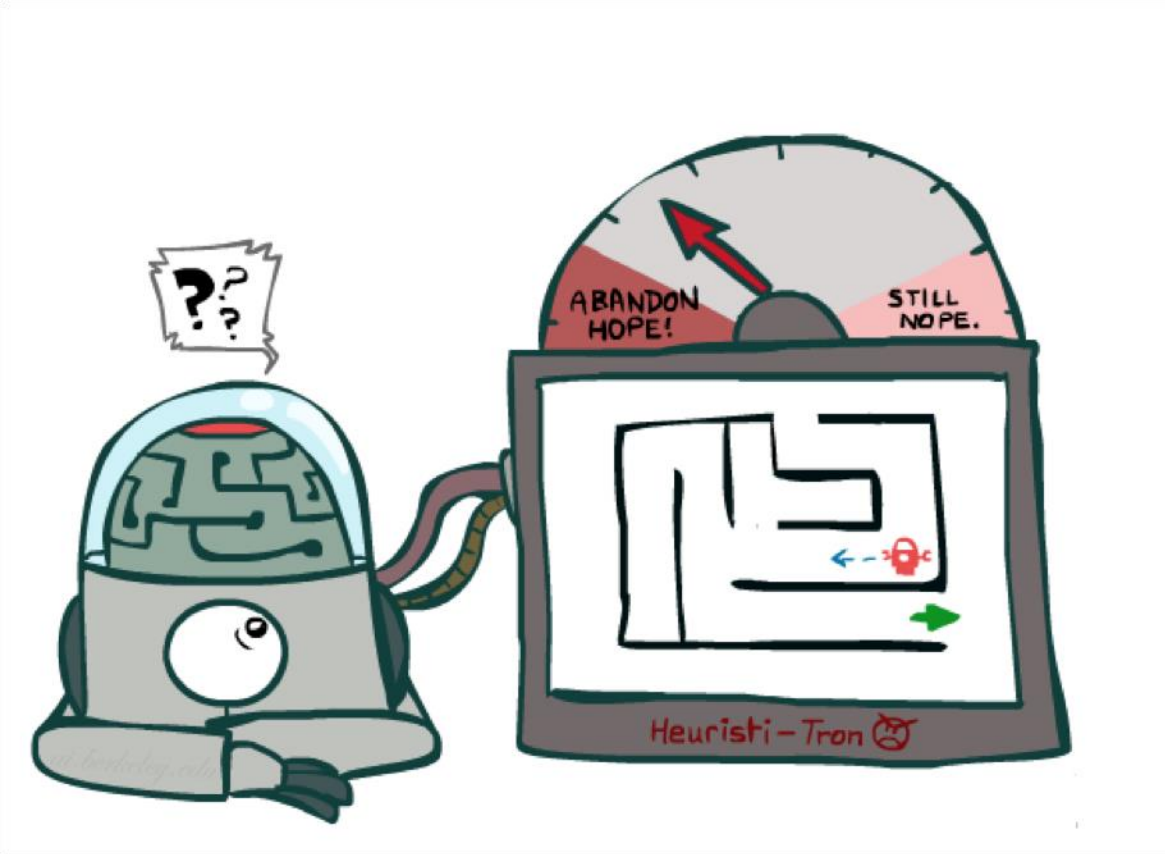


- Quelle est la source du problème?
- Coût de l'objectif < Coût estimé par l'heuristique
- Heuristique doivent être toujours inférieur au coût réel!

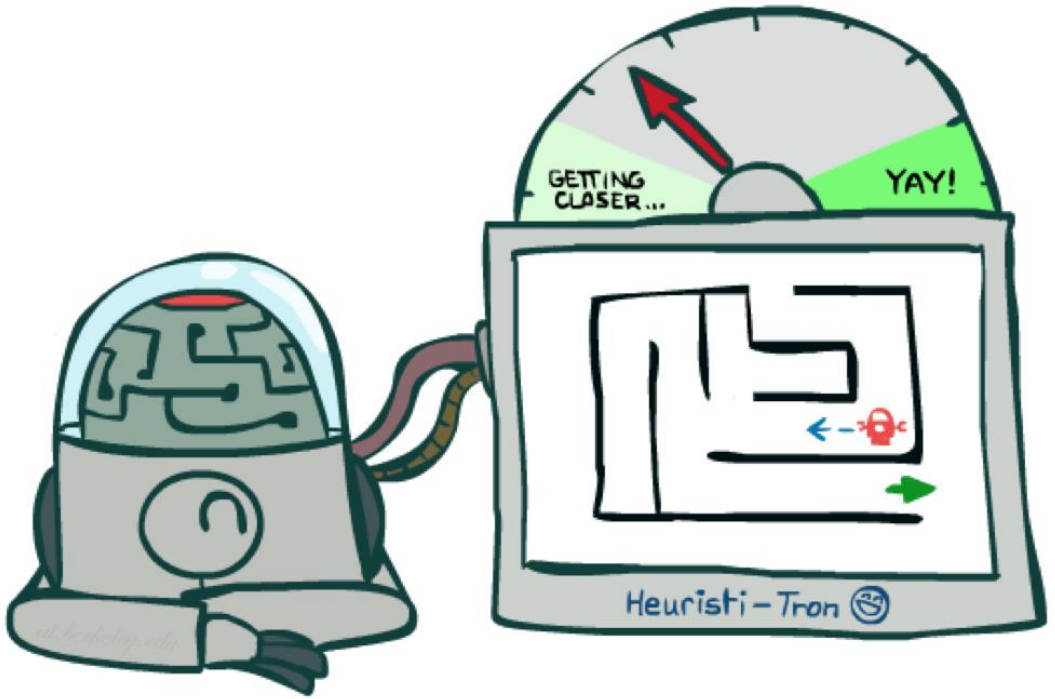
Admissible Heuristics



Idea: Admissibility



Inadmissible (pessimiste) heuristique où on perd l'optimalité



Admissible (optimiste) heuristique peut ralentir les bons plans mais jamais les surestimer.

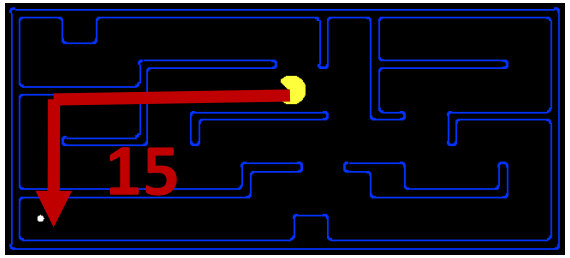
Heuristique admissible

- Une heuristique h est *admissible* (optimiste) si:

$$0 \leq h(n) \leq h^*(n)$$

où $h^*(n)$ est le coût reel à l'objectif le plus proche.

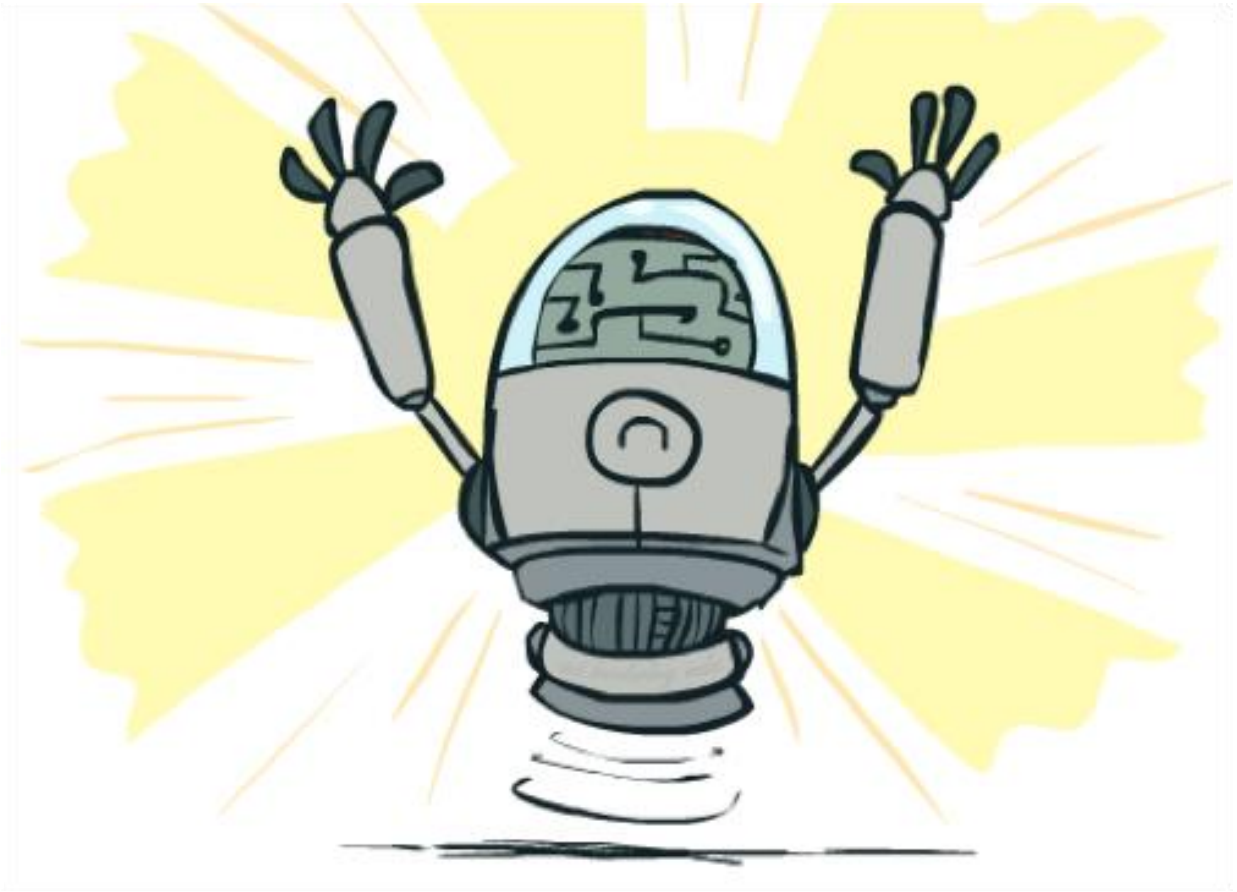
- Exemples:



4

- Construire une heuristique constitue le reel défi pour utiliser A*.

Optimalité de A*



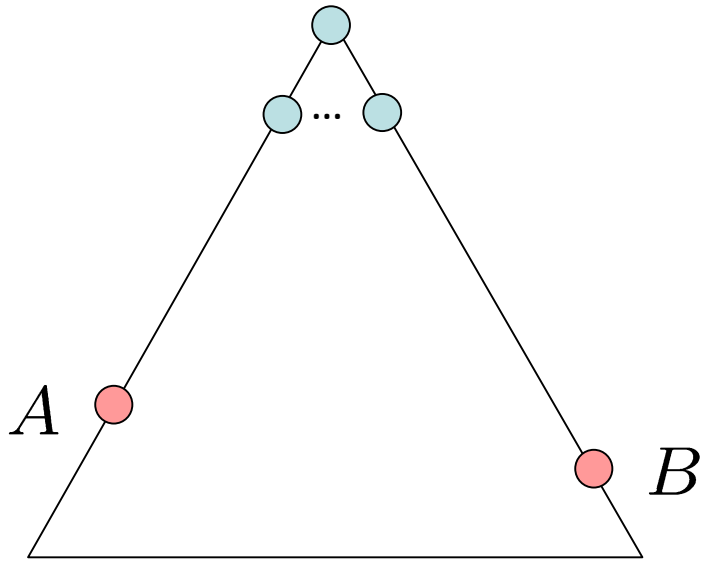
Optimalité de A*

On considère:

- A un objectif optimal
- B un objectif sous optimal
- h est admissible

Alors:

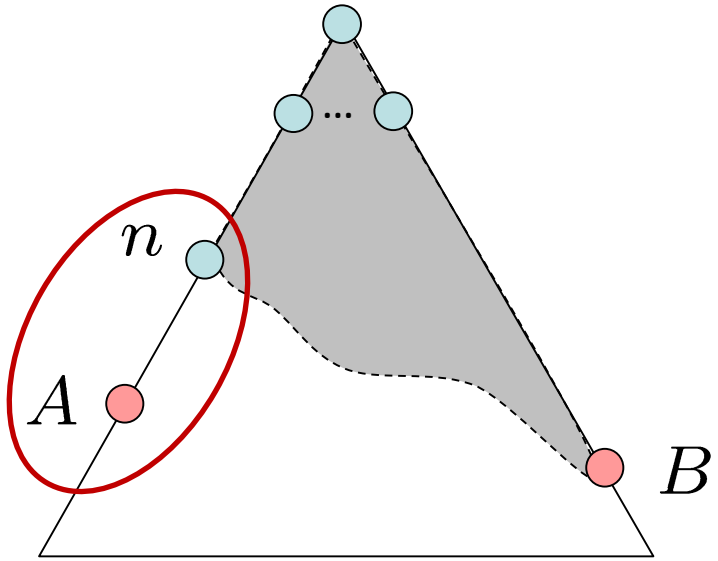
- A va être developpé avant B



Optimality of A*

Preuve:

- Imaginons que B est dans la frontière
- *Un antecedent n* de A est dans la frontière aussi.
- Alors: *n* va être devellopé avant B
  1. f(n) est inférieur à f(A)



$$f(n) = g(n) + h(n)$$ Definition du f

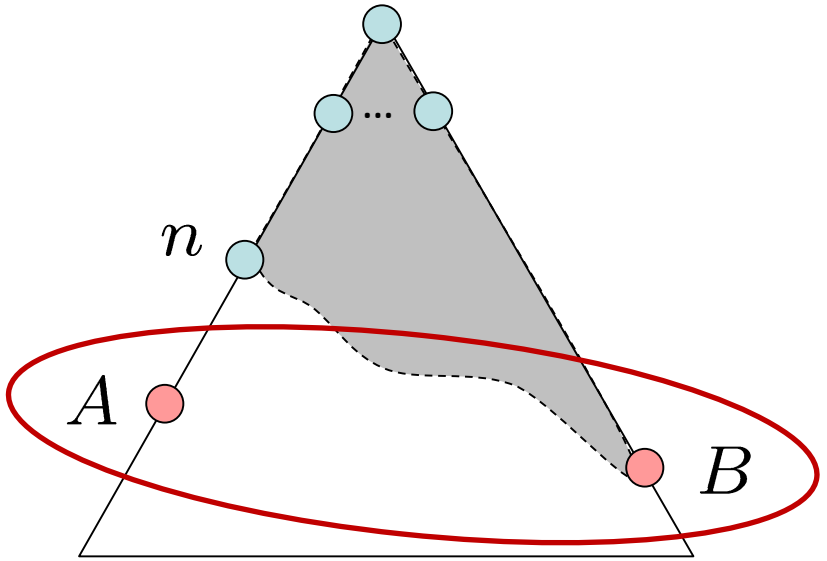
$$f(n) \leq g(A)$$ Admissibilité de h

$$g(A) = f(A)$$ h = 0 pour l'objectif

Optimalité de A*

Preuve:

- Imaginons que B est dans la frontière
- Un antécedant n de A est dans la frontière aussi.
- alors: *n* va être developé avantB
  1. f(n) est inférieur f(A)
  2. f(A) est inférieur f(B)



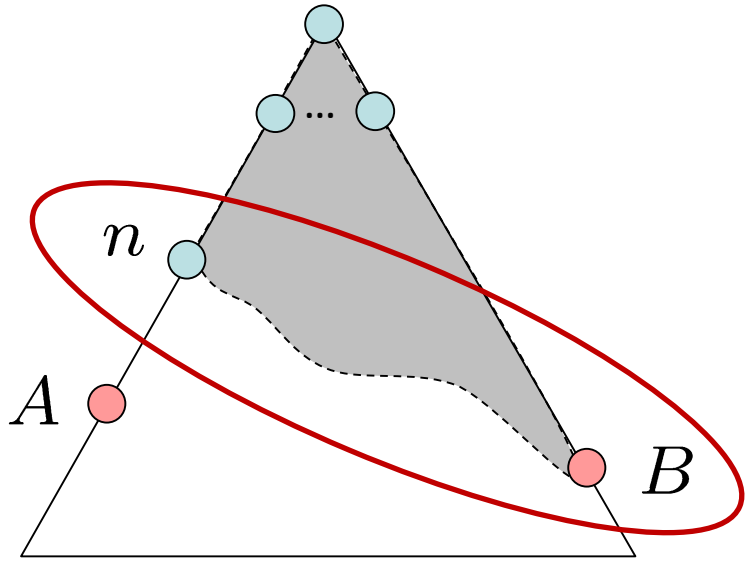
$g(A) < g(B)$ B est sous optimal

$f(A) < f(B)$ h = 0 pour l'objectif

Optimalité de A*

Preuve:

- Imaginons que B est dans la frontière
- *Un antécedant n* de A est dans la frontière aussi.
- Alors : *n* va être developé avant B
  1. f(n) est inférieur à f(A)
  2. f(A) est inférieur à f(B)
  3. *n* sera developé avant B
- Tous les antécédants de A seront développés avant B
- A sera developé avant B
- A* est optimal

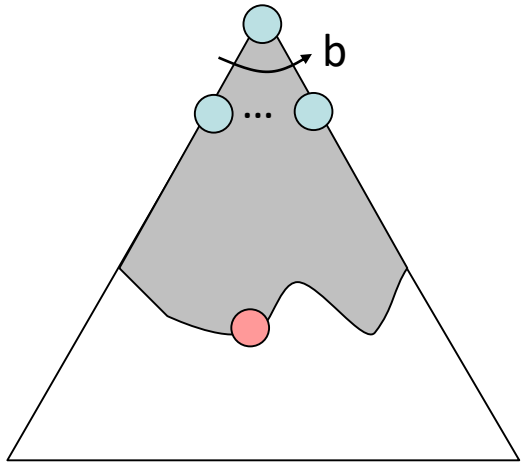


$$f(n) \leq f(A) < f(B)$$

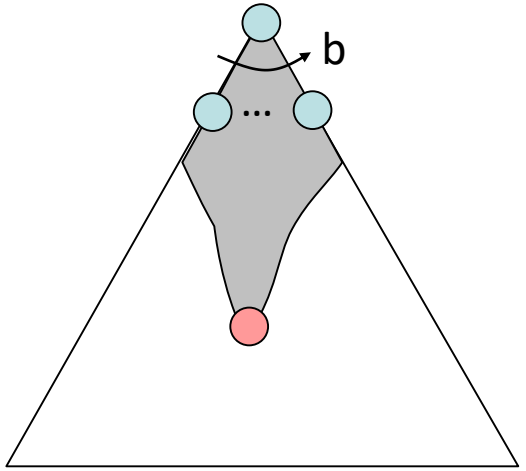
Propriétés de A*

Propriétés de A*

Uniform-Cost

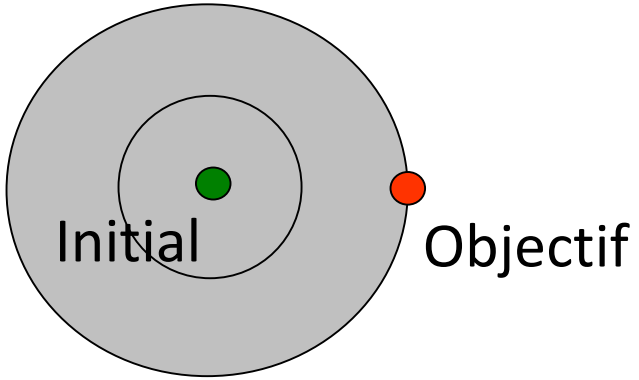


A*

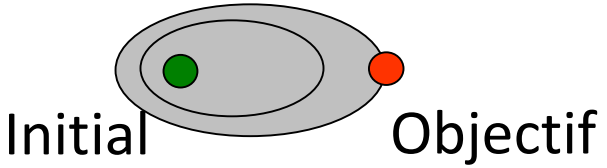


UCS vs A* Contours

- Uniform-cost develope les noeus uniformément dans toutes les directions.



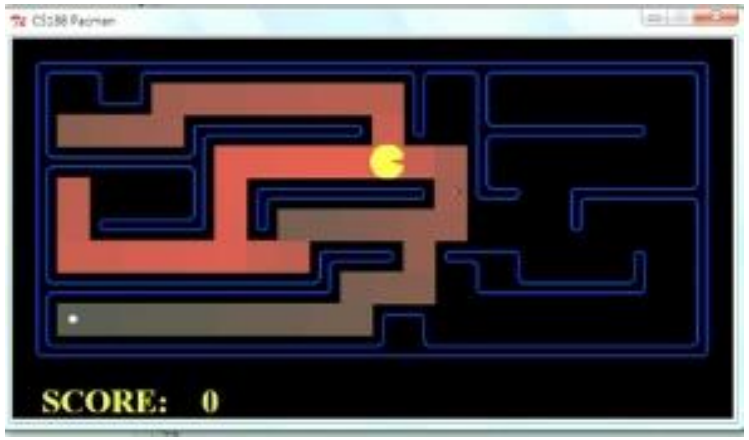
- A* dévelope principalement vers l'objecti, mais aussi dans les autres directions pour assurer l'optimalité.



Comparaison

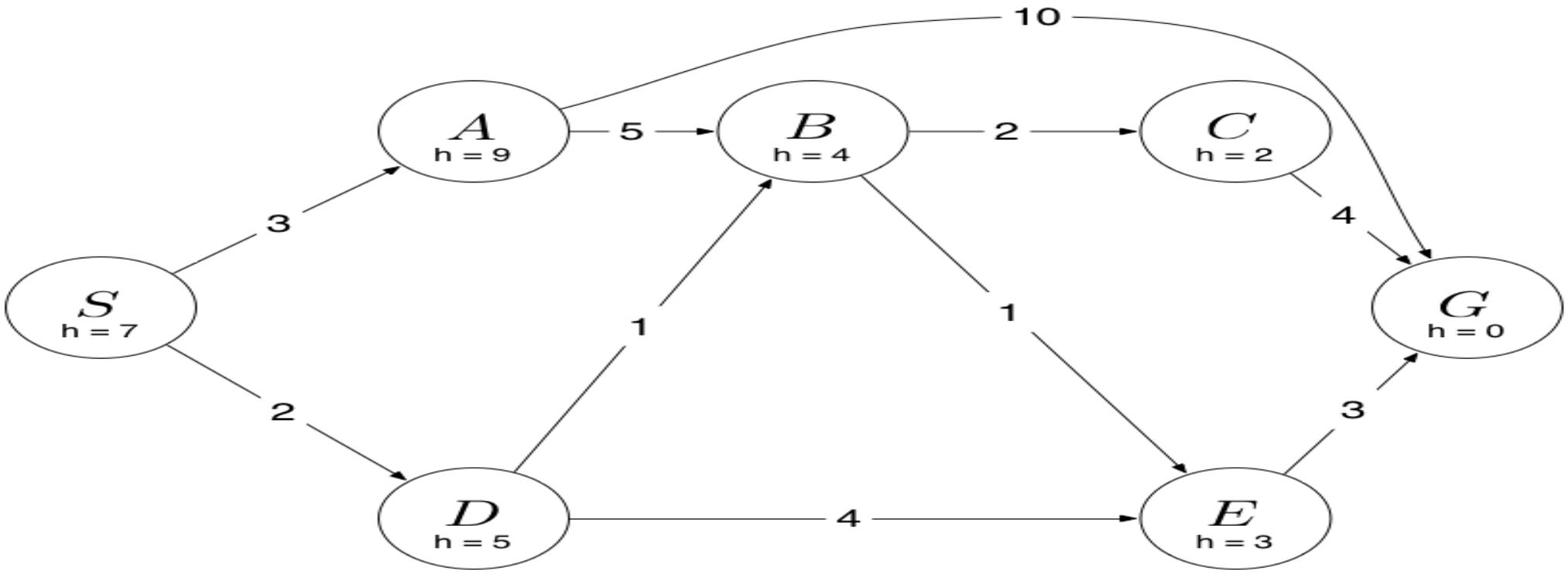
Greedy

Uniform Cost



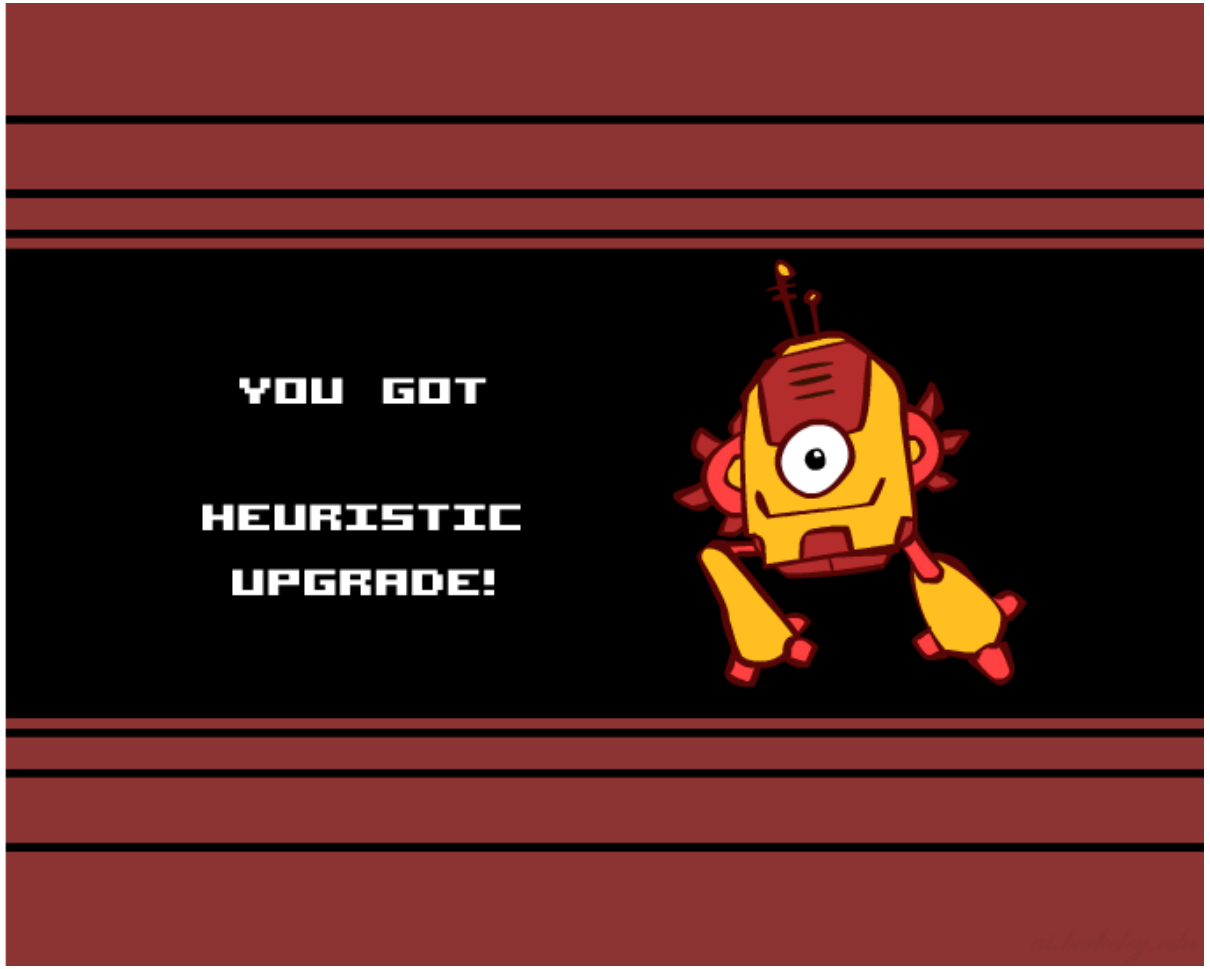
A*

Quiz 4



1. Est que l'heuristique est admissible?
2. Quelle est la solution calculé par A*?

Heuristiques

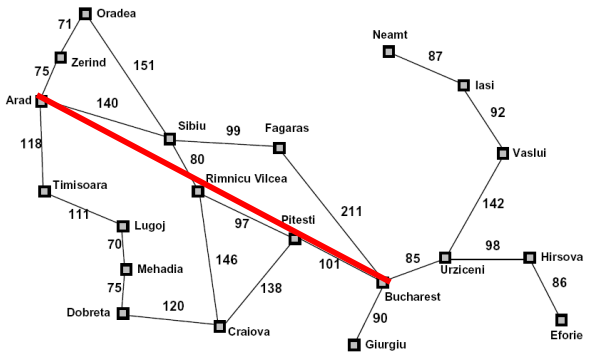


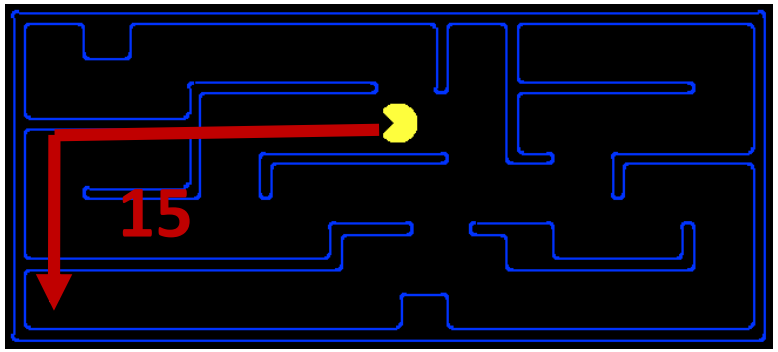
Creating Admissible Heuristics

- La tache la plus complexe dans un problème d'exploration, et trouver (créer) une heuristique efficace et admissible.

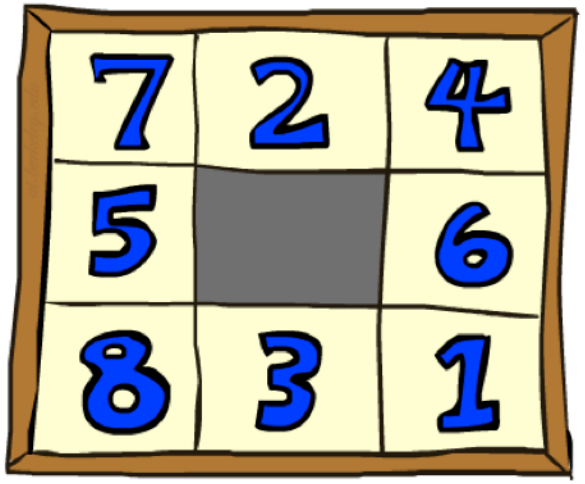
- Souvent, une heuristique est crée en **relaxant** le problème.

366

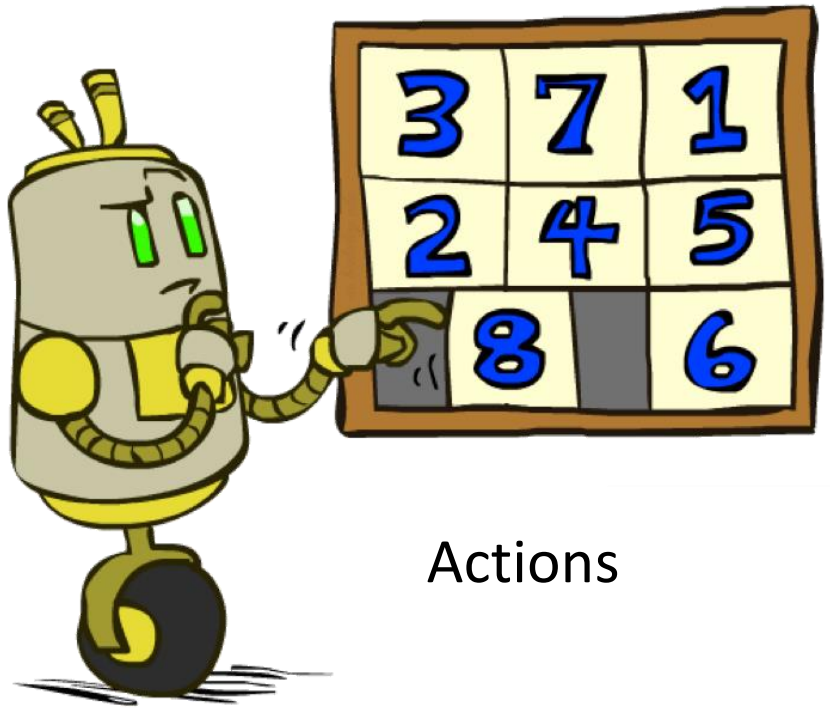




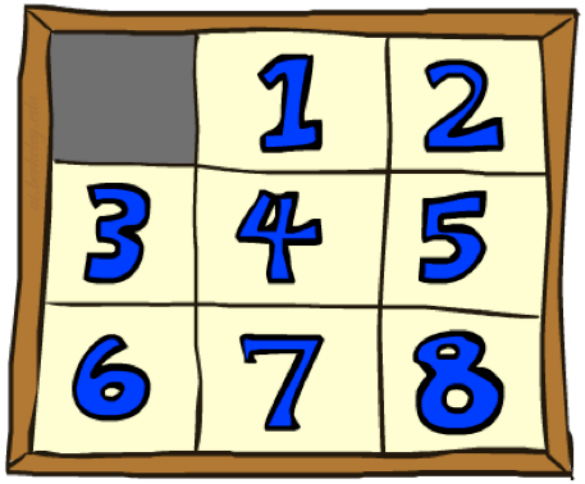
Exemple: 8 Puzzle



Etat initial



Actions

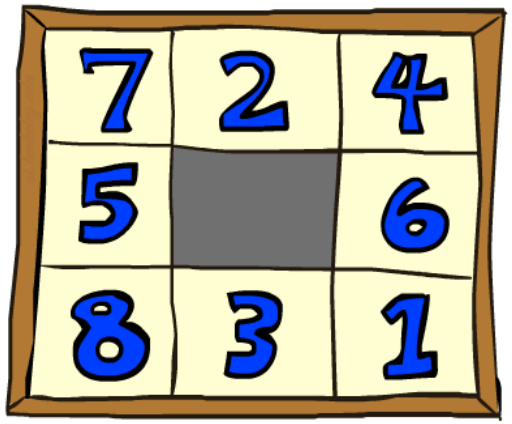


Objectif

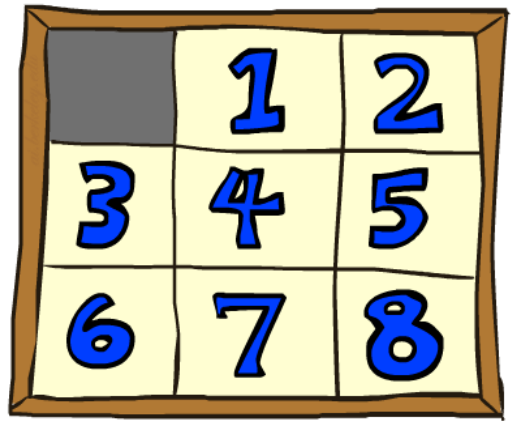
- Les configurations?
- Cardinal des configurations?
- Les actions possibles?
- Le nombre de successeurs d'un état?
- Comment définir le coût?

8 Puzzle I

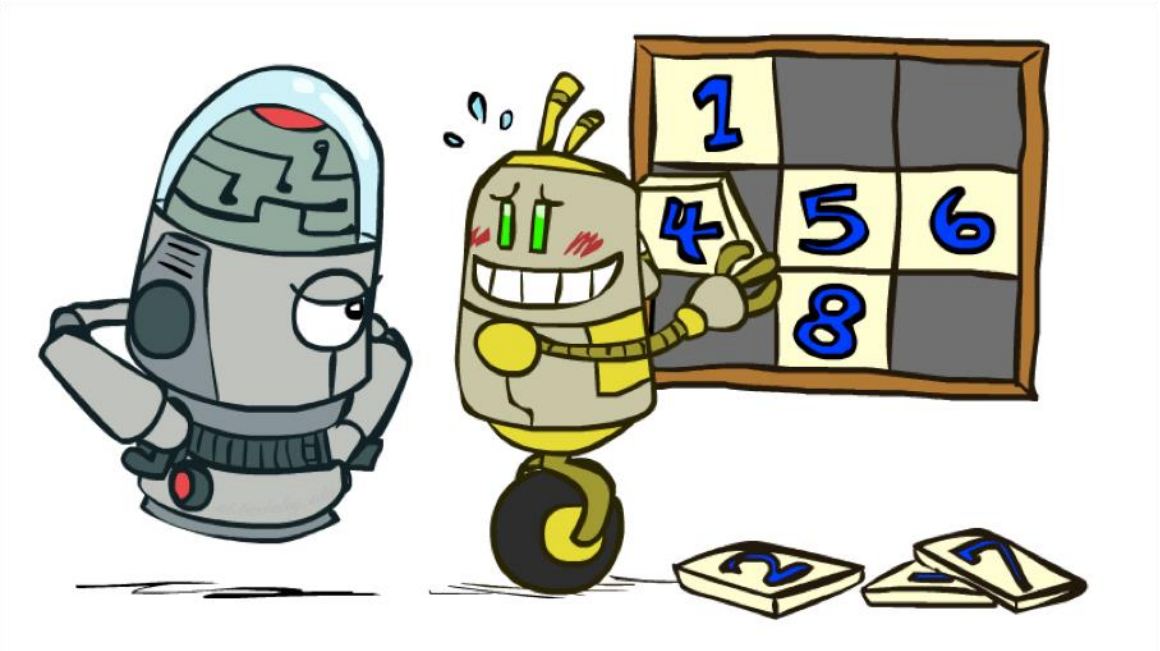
- Heuristique: Nombre des cases mal placés
- admissibilité?
- h(start) = 8
- Un problème relaxé



Etat initial



Etat objectif



	Noeuds de la frontière pour retrouver la solution		
	…4 iters	…8 iters	…12 iters
UCS	112	6,300	3.6×10^6
A*	13	39	227

Statistiques de Andrew Moore

Relations entre Heuristiques

Heuristique triviale, Dominance

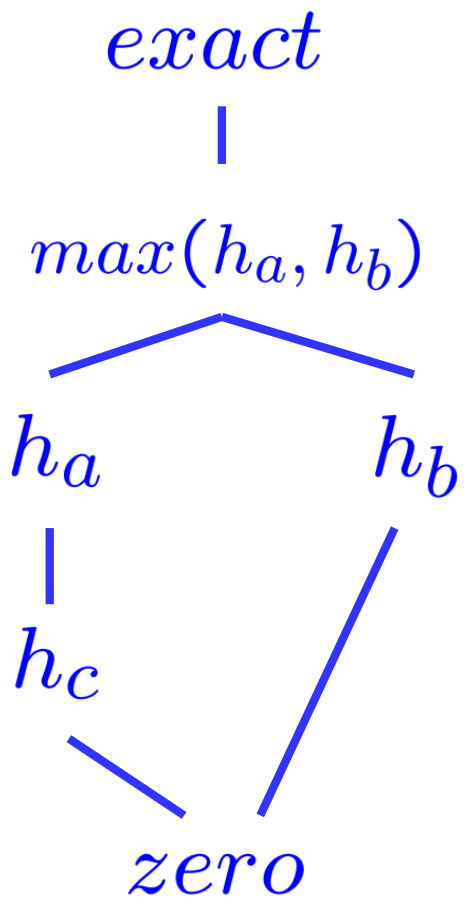
- Dominance: $h_a \geq h_c$ si

$$\forall n : h_a(n) \geq h_c(n)$$

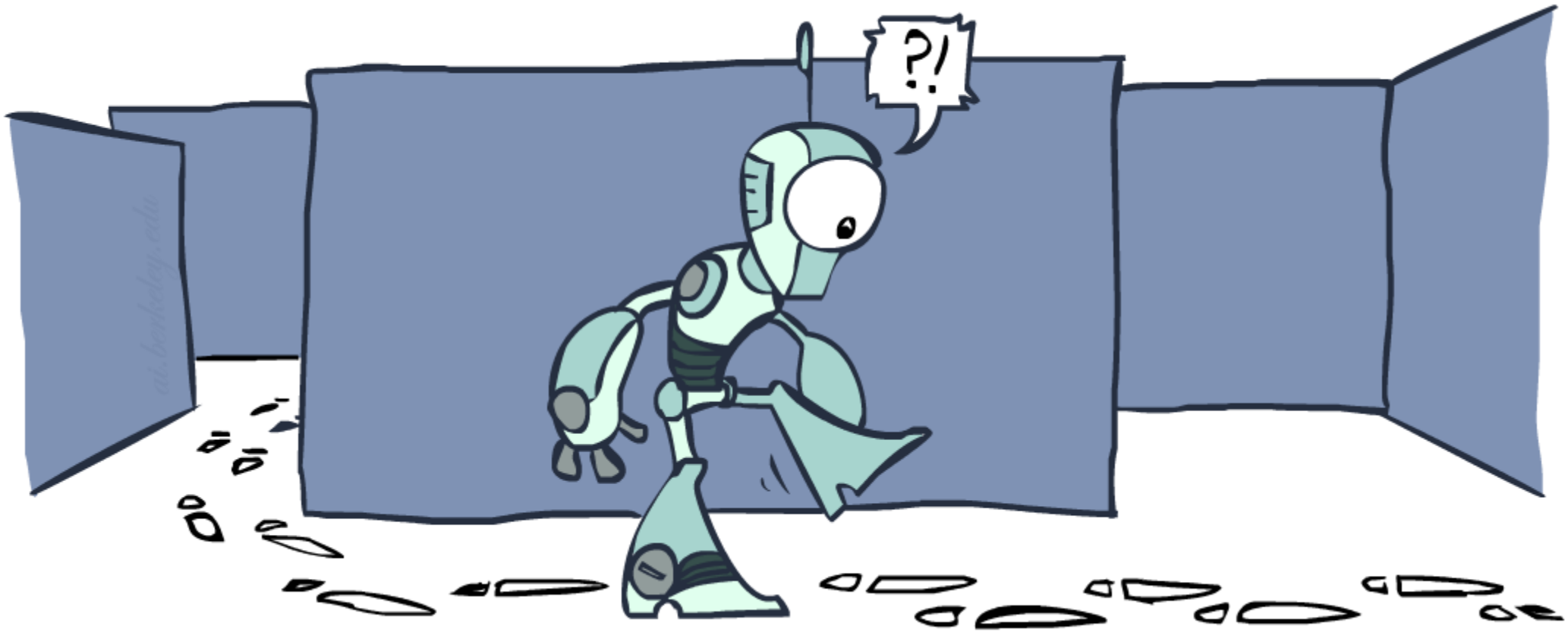
- Heuriststque incomparable:
  - Max est admissible aussi

$$h(n) = max(h_a(n), h_b(n))$$

- Heuristique triviale
  - Heuristique qui renvoie toujours 0.

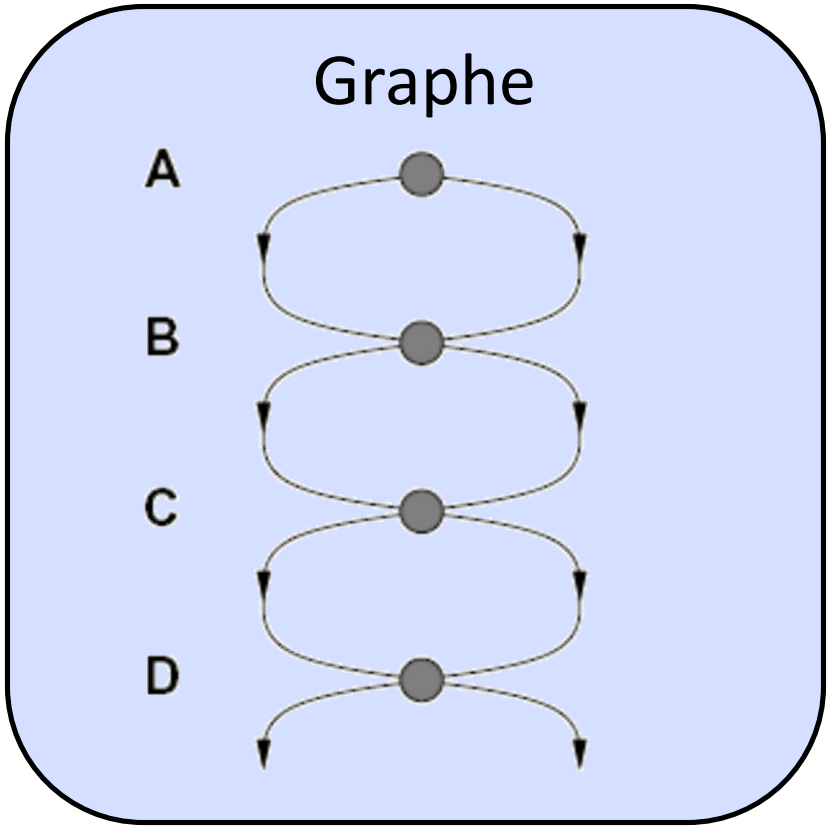


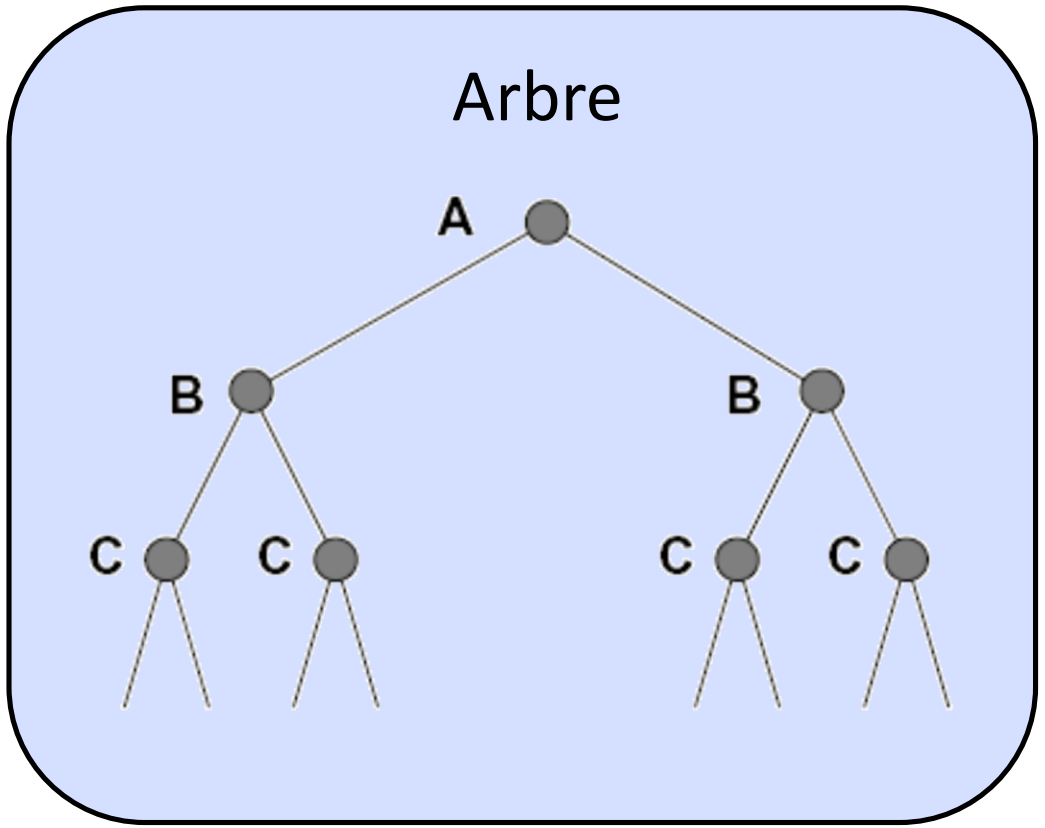
Exploration Graphe



Tree Search: Extra Work!

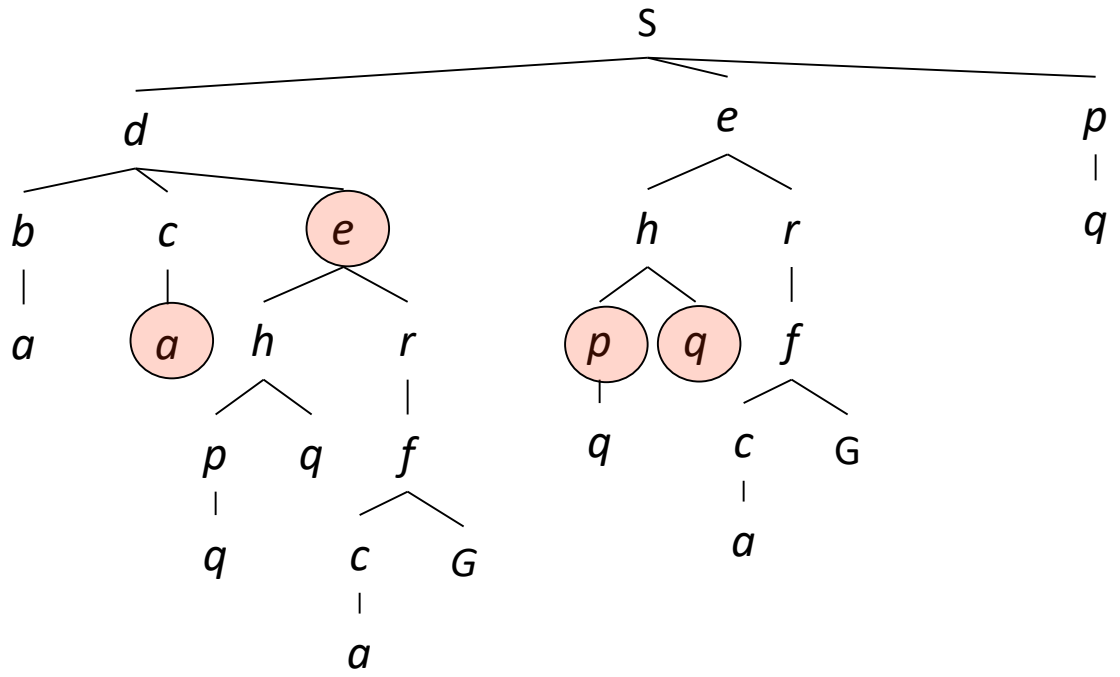
- Recherche en graphe, on ne détecte pas les répétitions.





Graph Search

- Dans BFS, par exemple, On doit pas developer les noeuds rouges (Pourqu'oi?)

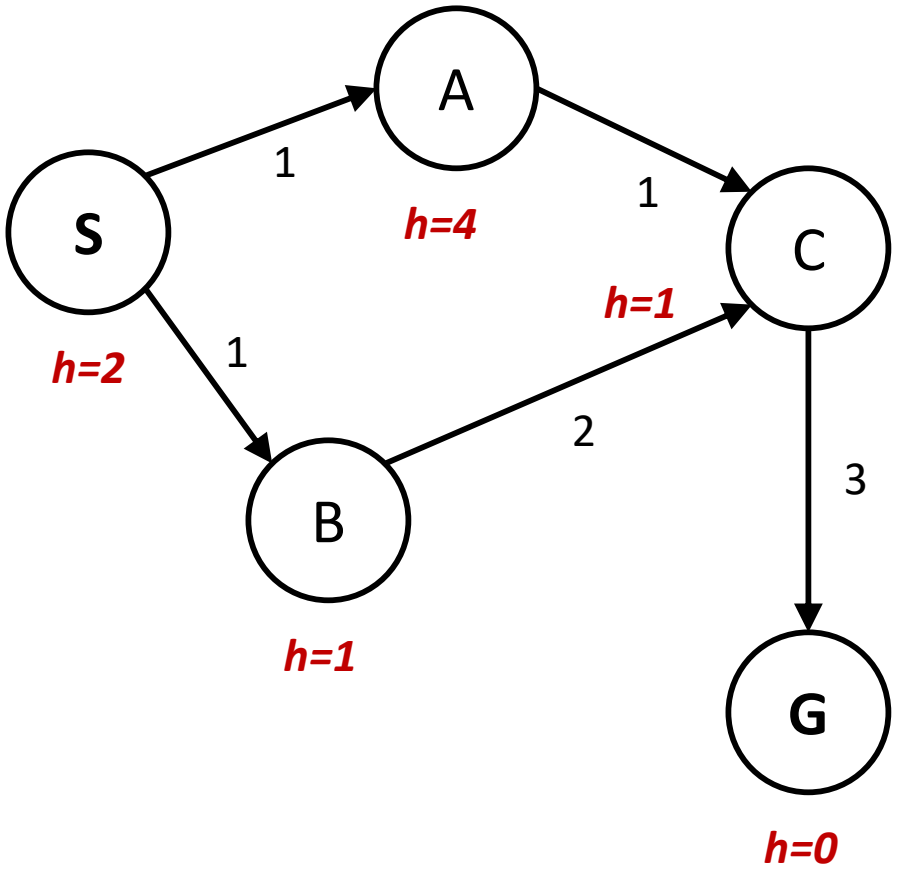


Exploration Graphe

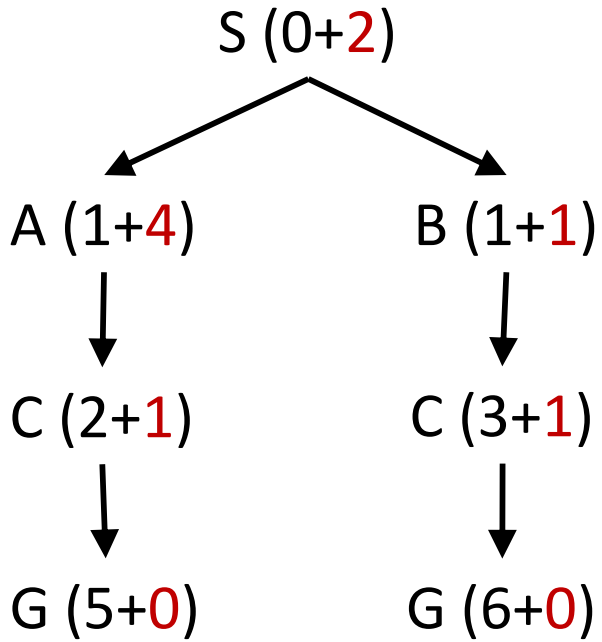
- Idée: Ne pas developper un noeuds deux fois.
- Implémentation:
  - Recherche en arbre + Un **ensemble E** pour les noeuds dévelopés
  - Avant de developer un noeud, verifier s'il existe déja dans l'ensemble E
- Exploration reste complete avec cette modification?
- Optimalité?

A* Recherche Graphe (optimalité)?

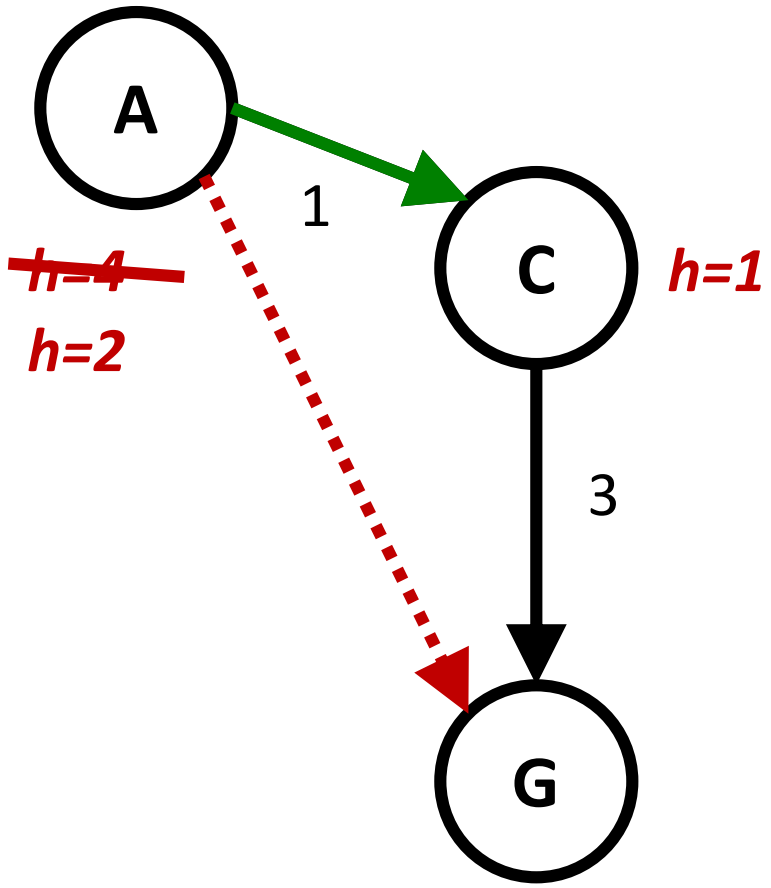
Graphe



Arbre



Consistence d'une Heuristique



- Idée principale: Coût estimé par une heuristique ≤ Cout réel
  - Admissibilité: coût heuristique ≤ coût réel

    h(A) ≤ coût actuel de A à G
  - Consistence: coût d'un arc selon l'heuristique ≤ cout d'un arc

    h(A) – h(C) ≤ cost(A to C)

- Conséquences de la consistance:
  - La valeur de f est toujours **croissante** dans un chemin.

    h(A) ≤ cost(A to C) + h(C)

Optimalité

- Exploration arbre:
  - A* est optimal si l'heuristique est admissible
  - UCS est un cas particulier avec (h = 0)
- Exploration graphe:
  - A* est optimal si l'heuristic is consistante
  - UCS est optimal car (h = 0 is consistante)
- Consistante implique admissibilité
- En general, La majorité des heuristiques admissibles sont consistants, surtout si ells parviennent d'une relaxation.