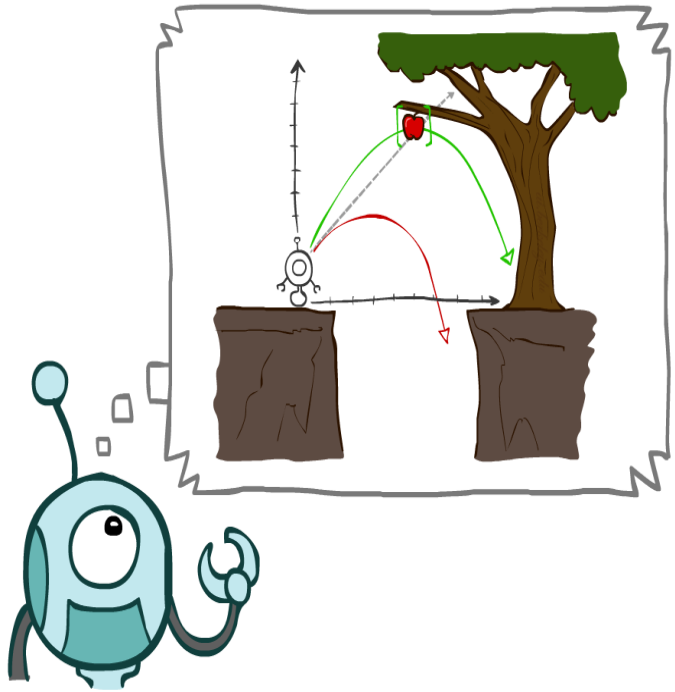
# Stratégie d'exploration non informée

**A.Belcaid**

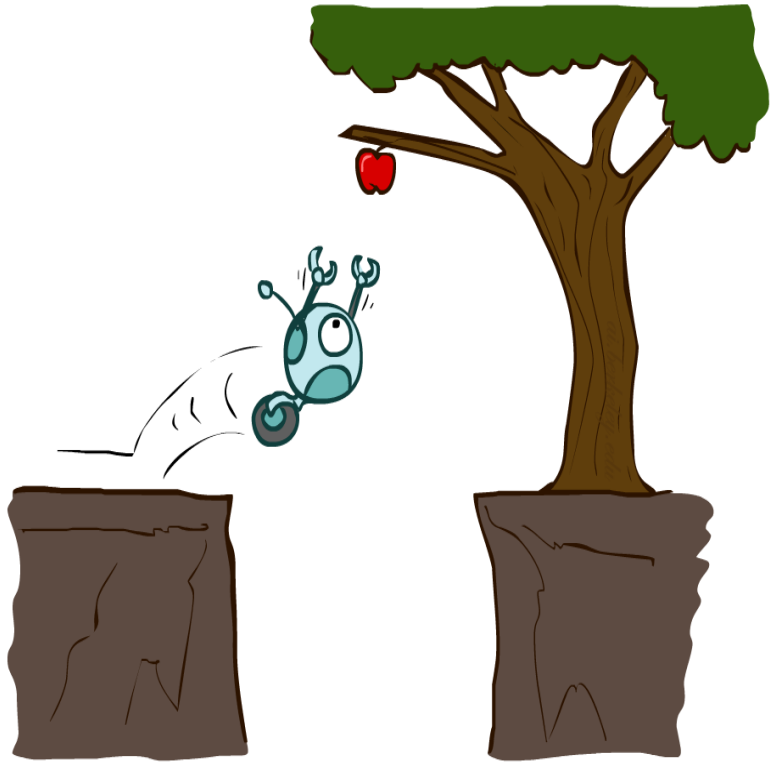
ENSA-Fès

March 25, 2019

1. Agent de résolution de problèmes.
2. Recherche des solutions
3. Exploration non informée
   - Recherche en profondeur (DFS)
   - Recherche en largueur (BFS)
   - Recherche uniforme en coût (UCS).

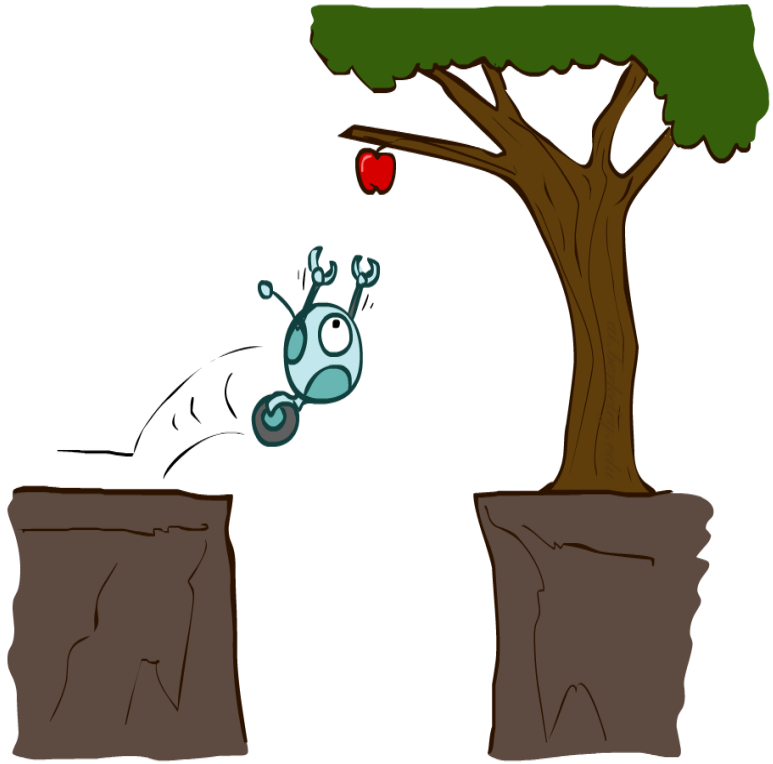


**Agents réflexes**:



## **Agents réflexes**:

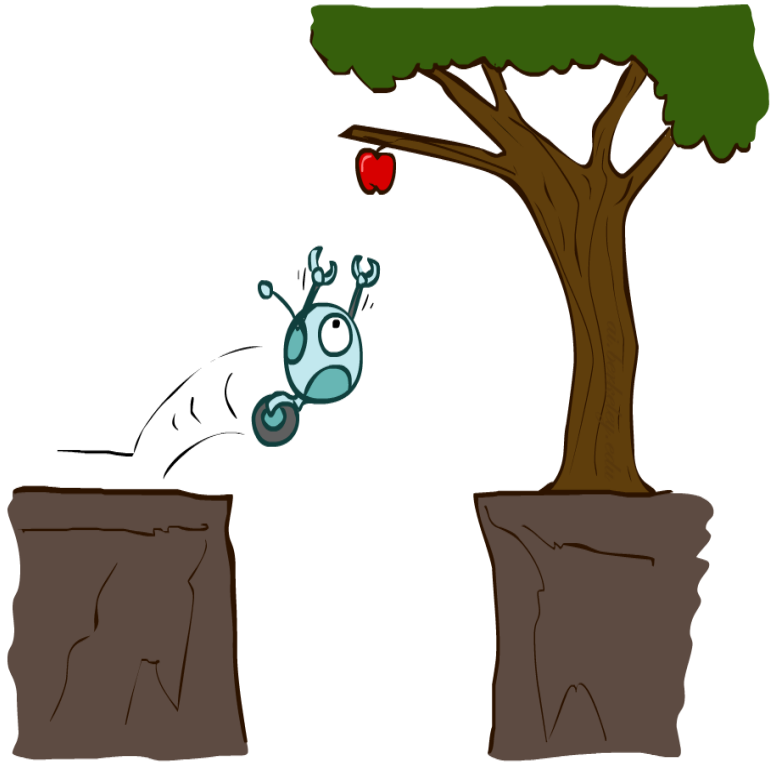
- Choix d'action sur l'état actuel des perceptions avec une mémorisation.



## **Agents réflexes**:

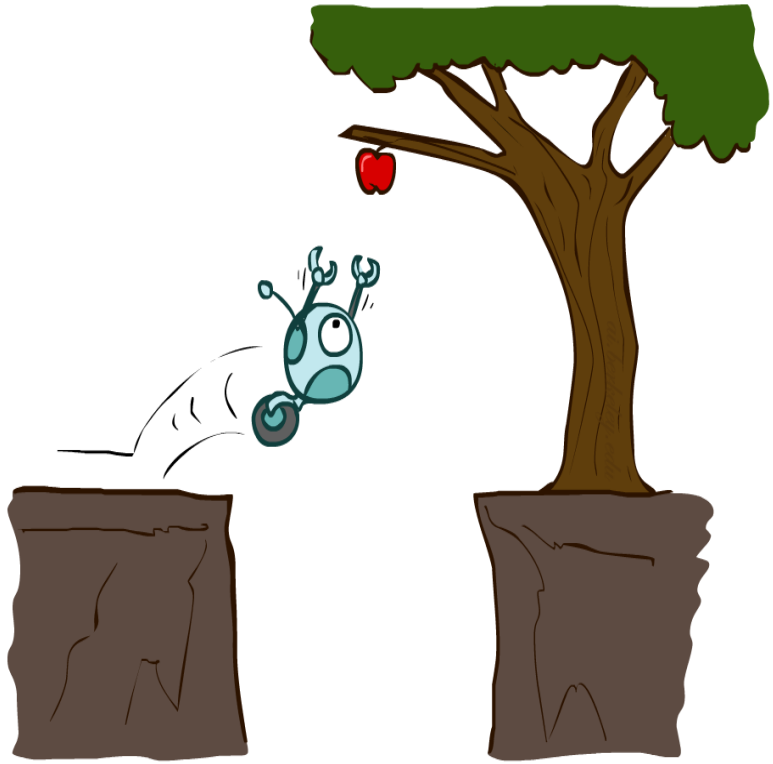
- Choix d'action sur l'état actuel des perceptions avec une mémorisation.

- L'environnement est soit stocké **entièrement** en mémoire, soit, on connait juste l'état actuel.



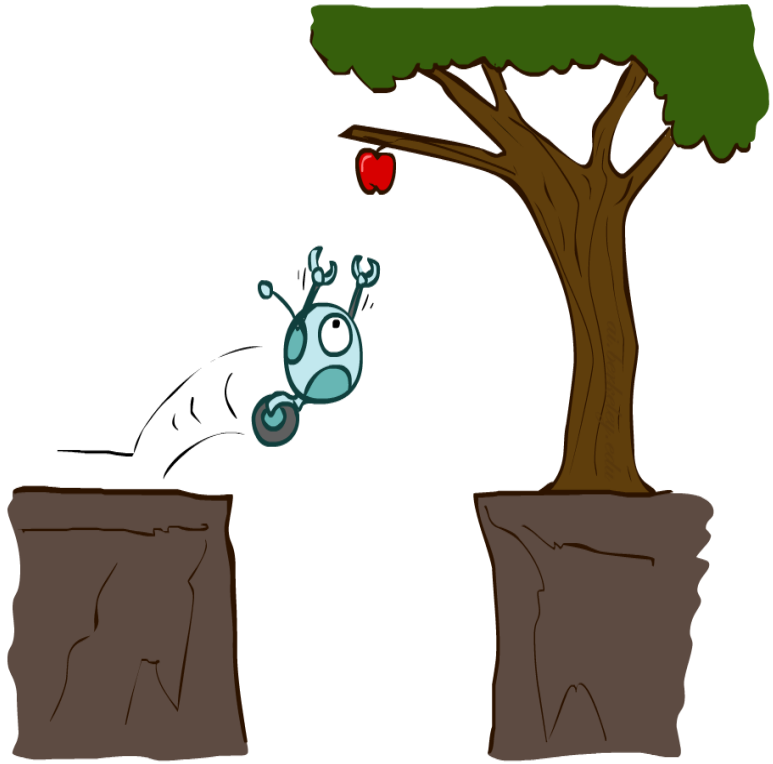
## **Agents réflexes**:

- Choix d'action sur l'état actuel des perceptions avec une mémorisation.
- L'environnement est soit stocké **entièrement** en mémoire, soit, on connait juste l'état actuel.
- Ne considèrent pas les **conséquences** de leurs actions.



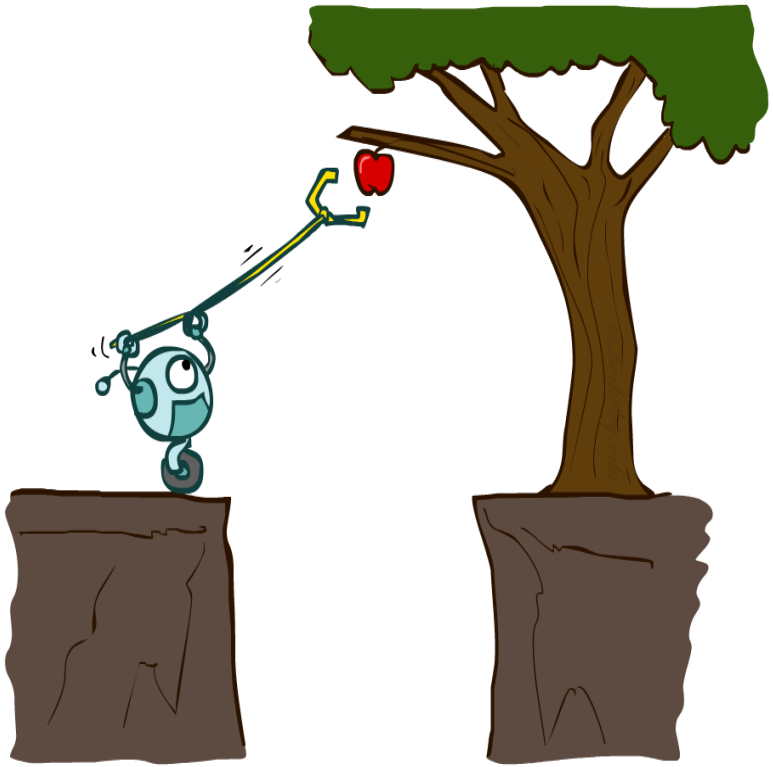
## **Agents réflexes**:

- Choix d'action sur l'état actuel des perceptions avec une mémorisation.
- L'environnement est soit stocké **entièrement** en mémoire, soit, on connait juste l'état actuel.
- Ne considèrent pas les **conséquences** de leurs actions.



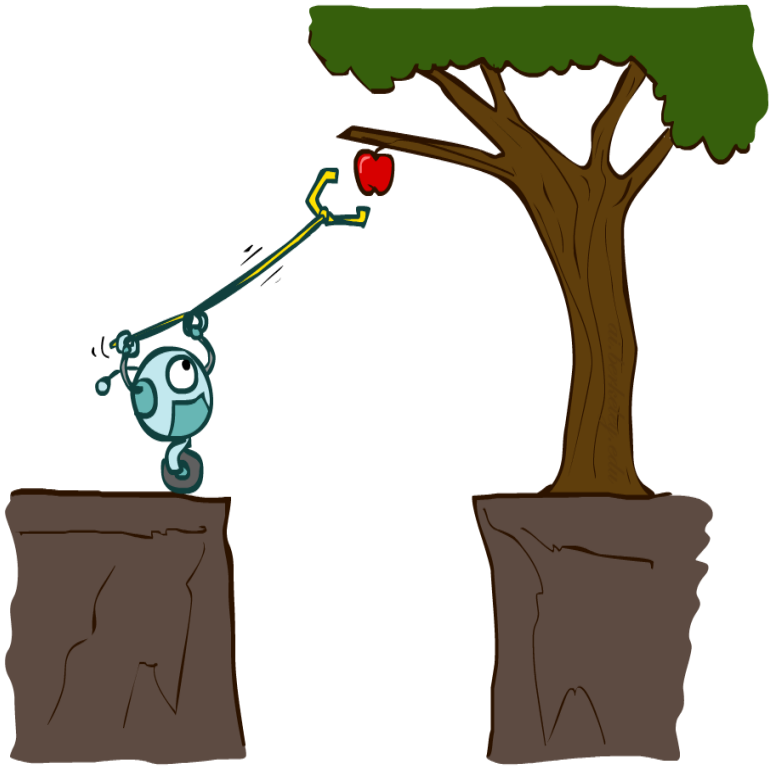
Est ce que les agents réflexes peuvent être **rationnels**?

**Agents de planification**:



## **Agents de planification**:

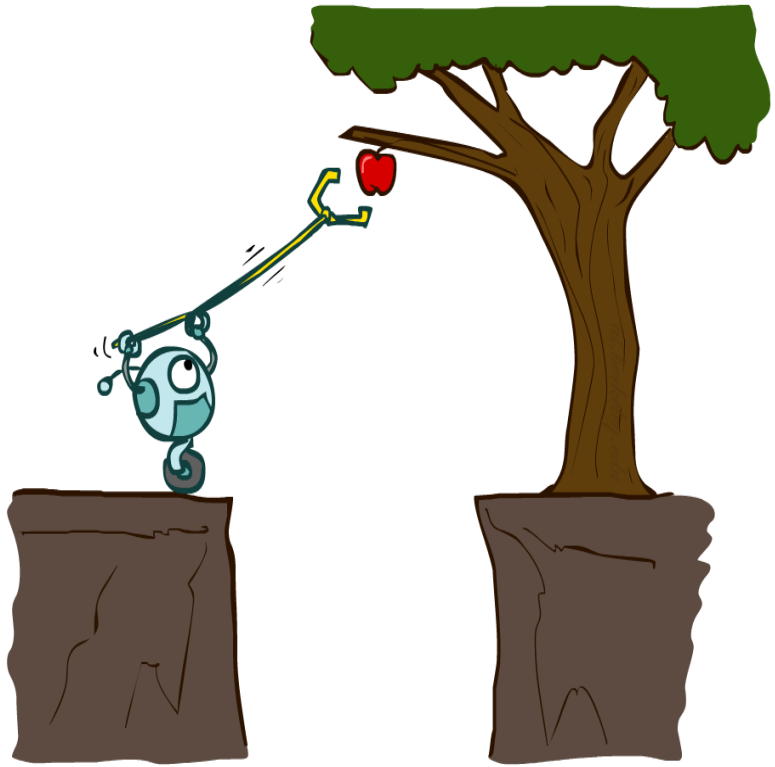
- Choix d'action selon une séquence d'actions qui permettent d'atteindre un **objectif**.



## **Agents de planification**:

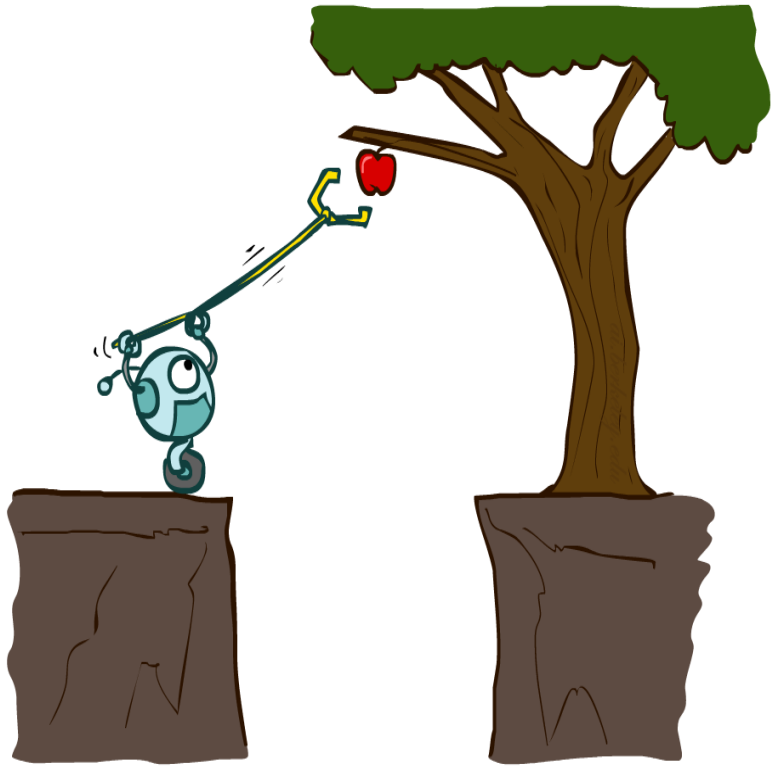
- Choix d'action selon une séquence d'actions qui permettent d'atteindre un **objectif**.

- Utilisent une représentation **atomique** de l'environnement (connaissent l'état après le choix d'une action).



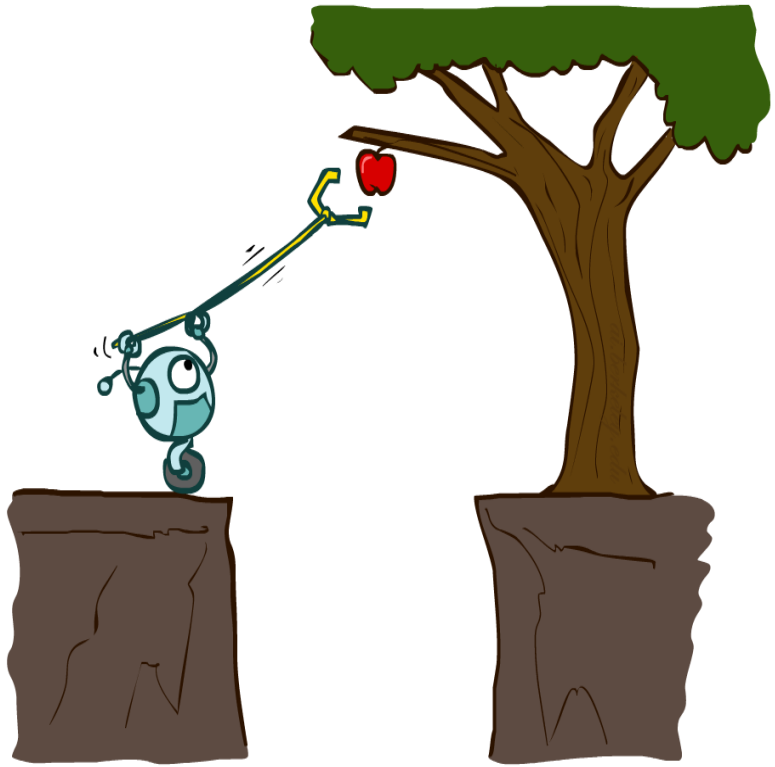
## **Agents de planification**:

- Choix d'action selon une séquence d'actions qui permettent d'atteindre un **objectif**.
- Utilisent une représentation **atomique** de l'environnement (connaissent l'état après le choix d'une action).
- Doivent formuler un **objectif** à atteindre.



## **Agents de planification**:

- Choix d'action selon une séquence d'actions qui permettent d'atteindre un **objectif**.
- Utilisent une représentation **atomique** de l'environnement (connaissent l'état après le choix d'une action).
- Doivent formuler un **objectif** à atteindre.
- Considèrent l'environnement dans l'état actuel et future.



Dans chacune des descriptions suivantes des **agents**, sélectionner leur **type**:

## Partie 1

Un agent Pacman qui est programmé à se déplacer toujours vers la **pellule** la plus proche.

- ☐ Agent réflexe
- ☐ Agent de planification

# Quiz 1

Dans chacune des descriptions suivantes des **agents**, sélectionner leur **type**:

## Partie 1

Un agent Pacman qui est programmé à se déplacer toujours vers la **pellule** la plus proche.

- ☐ Agent réflexe
- ☐ Agent de planification

## Partie 2

Un agent Pacman qui est programmé à se déplacer toujours vers la **pellule** la plus proche sauf s'il y as **fantôme** qui est à trois pas.

- ☐ Agent réflexe
- ☐ Agent de planification

# Quiz 1

Dans chacune des descriptions suivantes des **agents**, sélectionner leur **type**:

## Partie 1

Un agent Pacman qui est programmé à se déplacer toujours vers la **pellule** la plus proche.

- ☐ Agent réflexe
- ☐ Agent de planification

## Partie 2

Un agent Pacman qui est programmé à se déplacer toujours vers la **pellule** la plus proche sauf s'il y as **fantôme** qui est à trois pas.

- ☐ Agent réflexe
- ☐ Agent de planification

## Partie 3

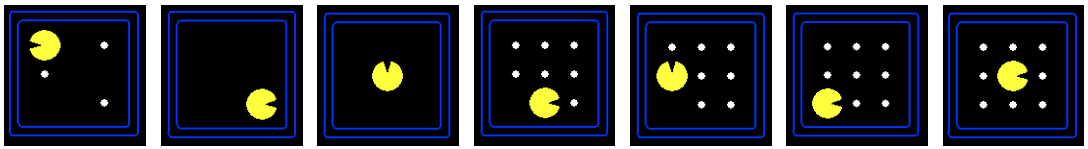
Un système de navigation qui considère toutes les voix vers une **destination**. Puis il choisit la plus courte.

- ☐ Agent réflexe
- ☐ Agent de planification

# Problème d'exploration

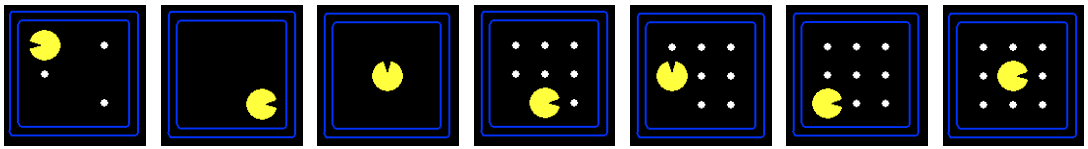
Un **Problème d'exploration** consiste de:

- **Un espace des états** :



Un **Problème d'exploration** consiste de:

- **Un espace des états** :

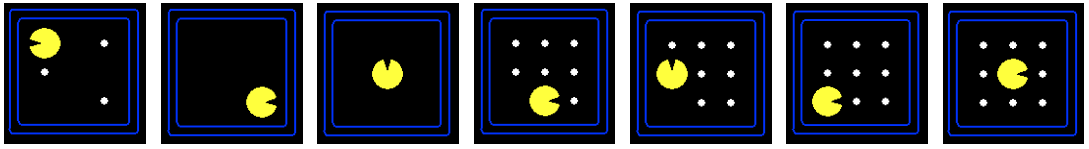


- Fonction de **Succession** (*Action*, *Coût*):

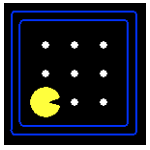
# Problème d'exploration

Un **Problème d'exploration** consiste de:

- **Un espace des états** :



- Fonction de **Succession** (*Action*, *Coût*):

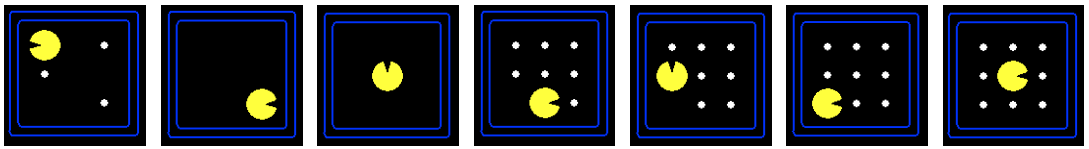


- Un état de **Départ** et un **Objectif**.

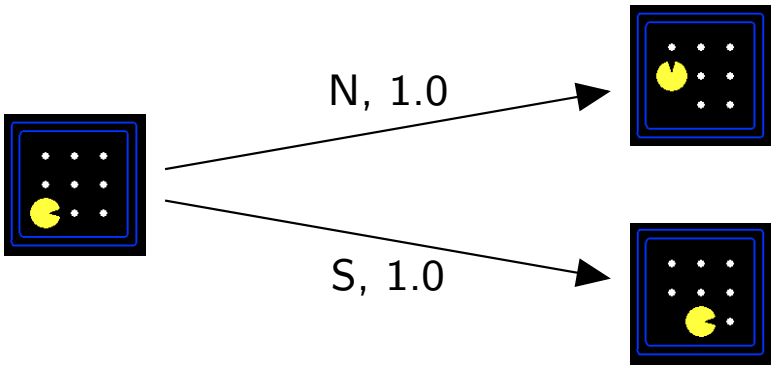
Une **Solution** est un séquence d'action ( **Plan**) pour atteindre l'objectif depuis l'état initial.

Un **Problème d'exploration** consiste de:

- **Un espace des états** :

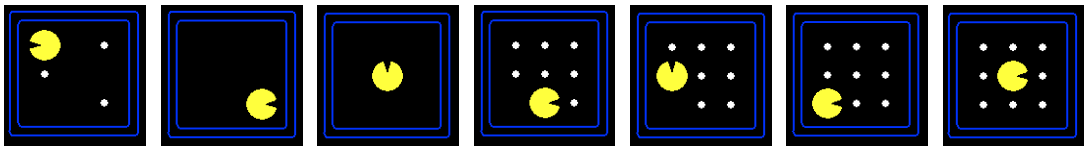


- Fonction de **Succession** (*Action*, *Coût*):

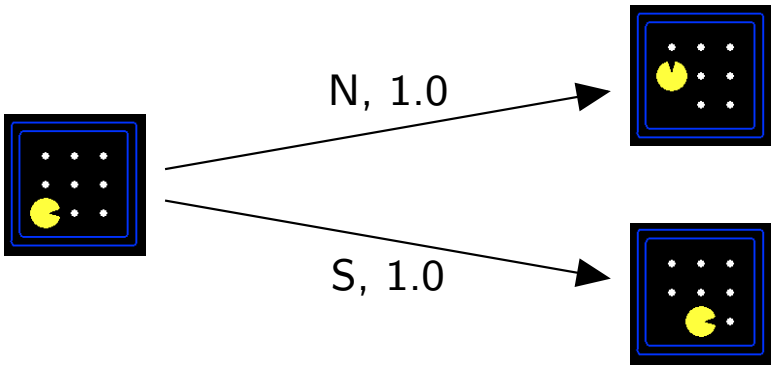


Un **Problème d'exploration** consiste de:

- **Un espace des états** :



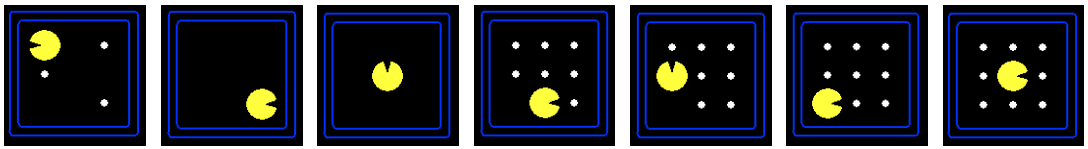
- Fonction de **Succession** (*Action*, *Coût*):



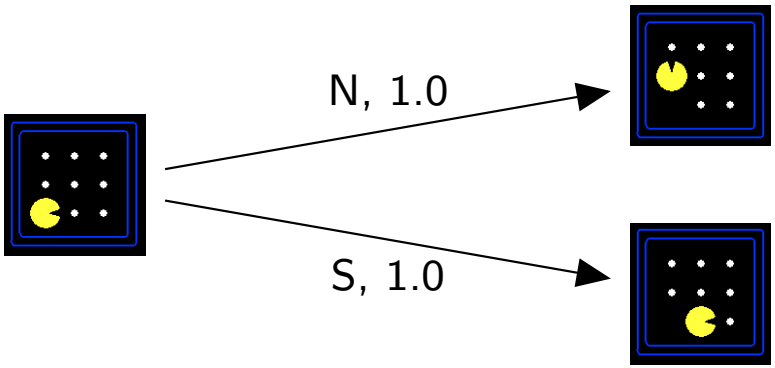

- Un état de **Départ** et un **Objectif**.

Un **Problème d'exploration** consiste de:

- **Un espace des états** :



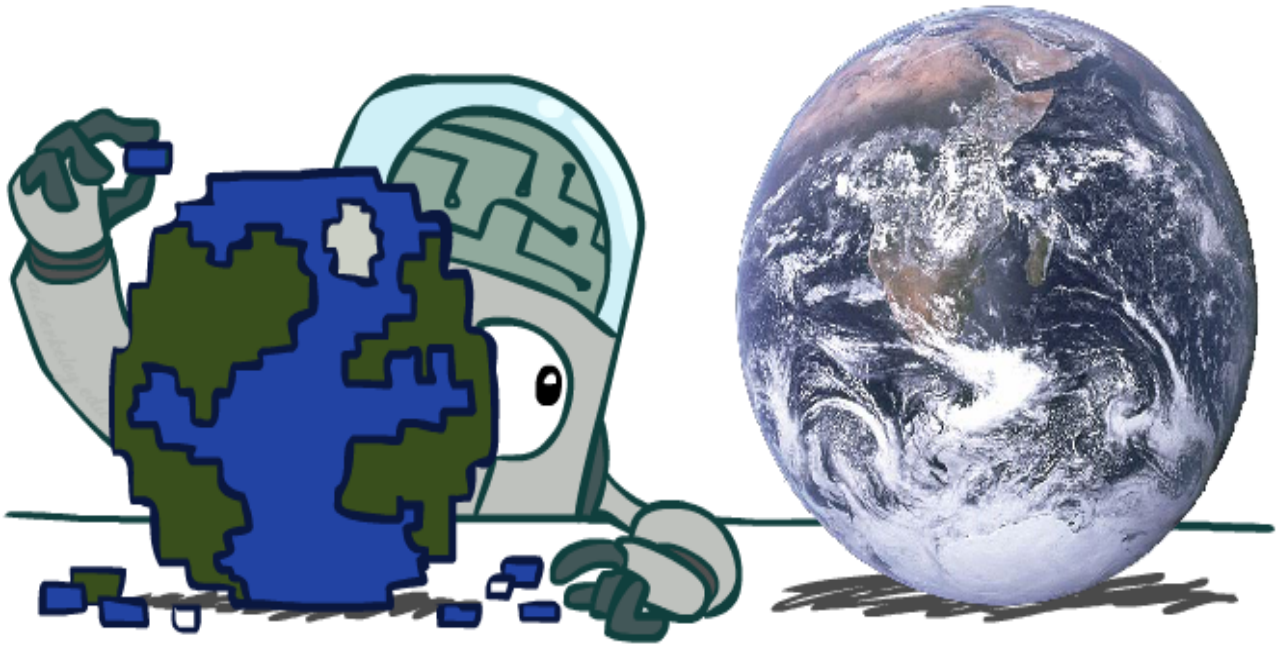
- Fonction de **Succession** (*Action*, *Coût*):



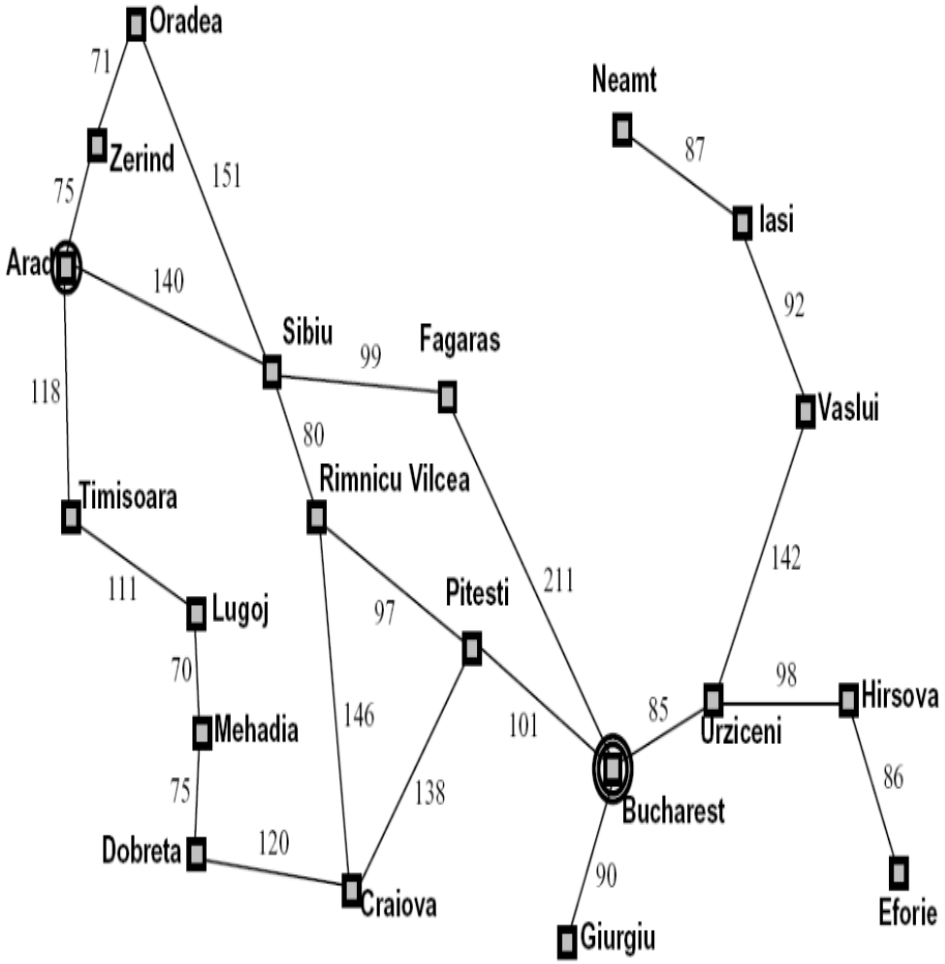

- Un état de **Départ** et un **Objectif**.

Une **Solution** est un séquence d'action ( **Plan**) pour atteindre l'objectif depuis l'état initial.

Un problème d'exploration doit considérer un **Modèle**:

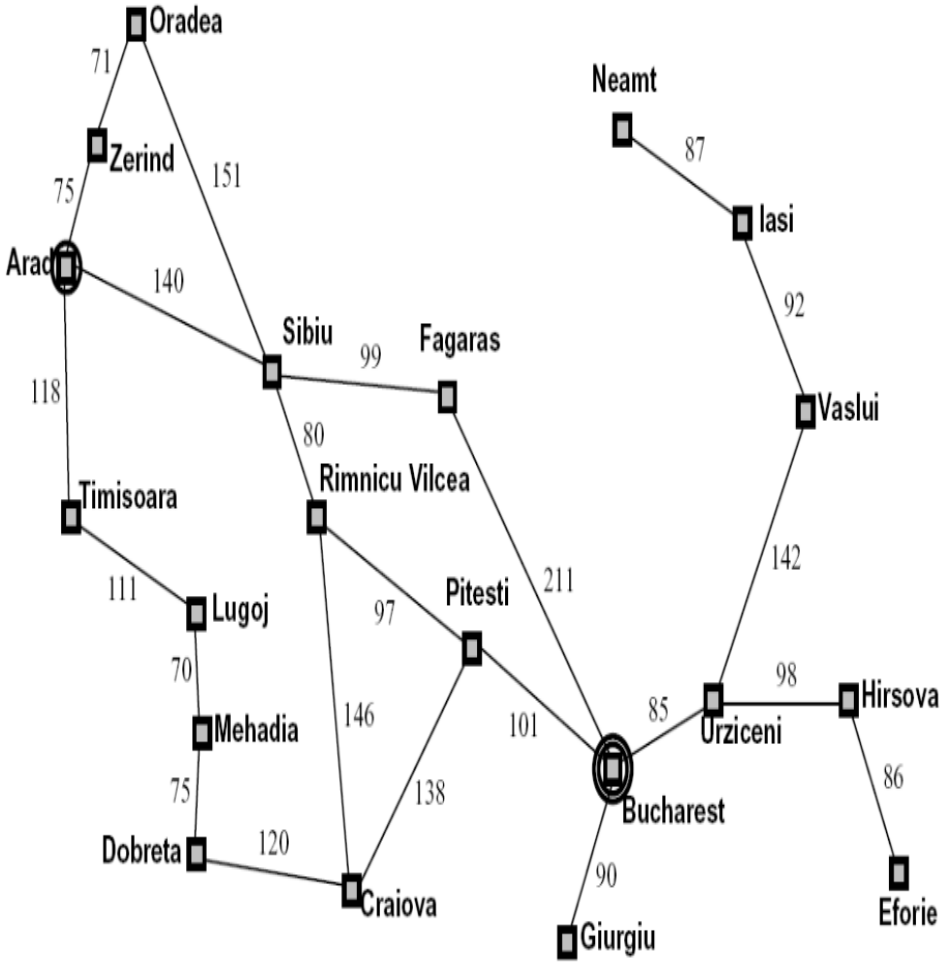


# exemple: Recherche de chemins



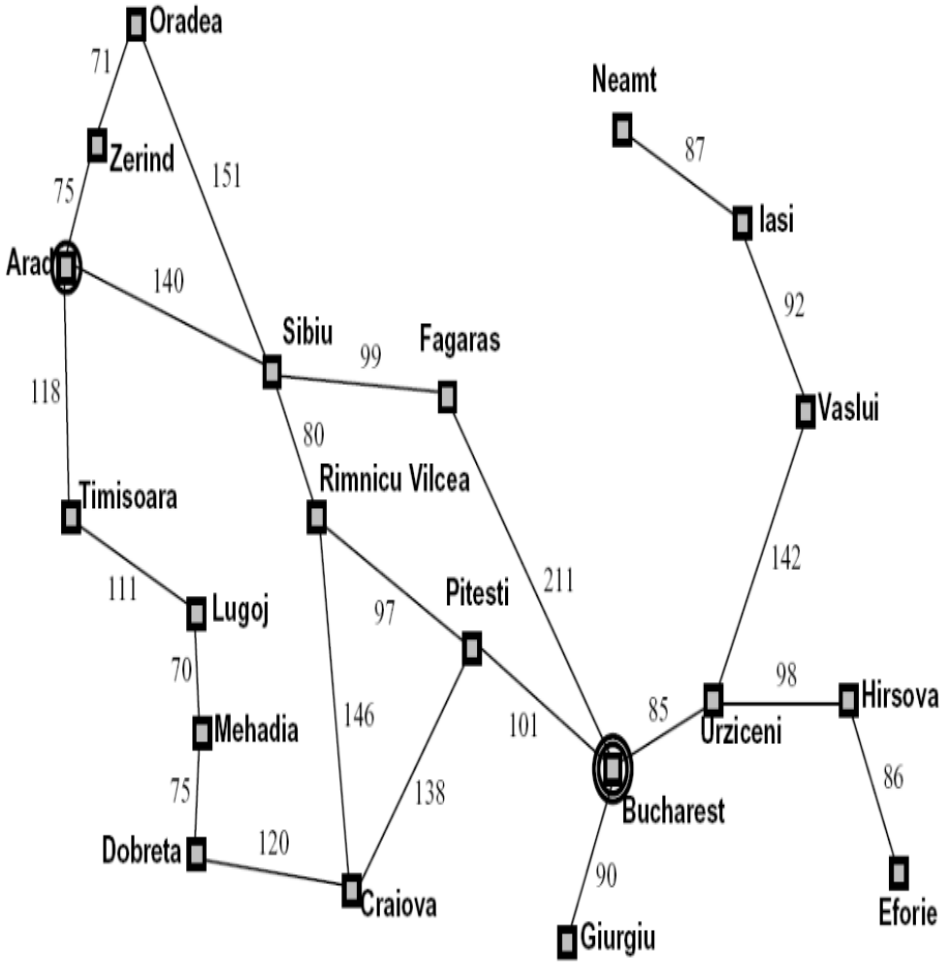

- **Espace d'états**

# exemple: Recherche de chemins



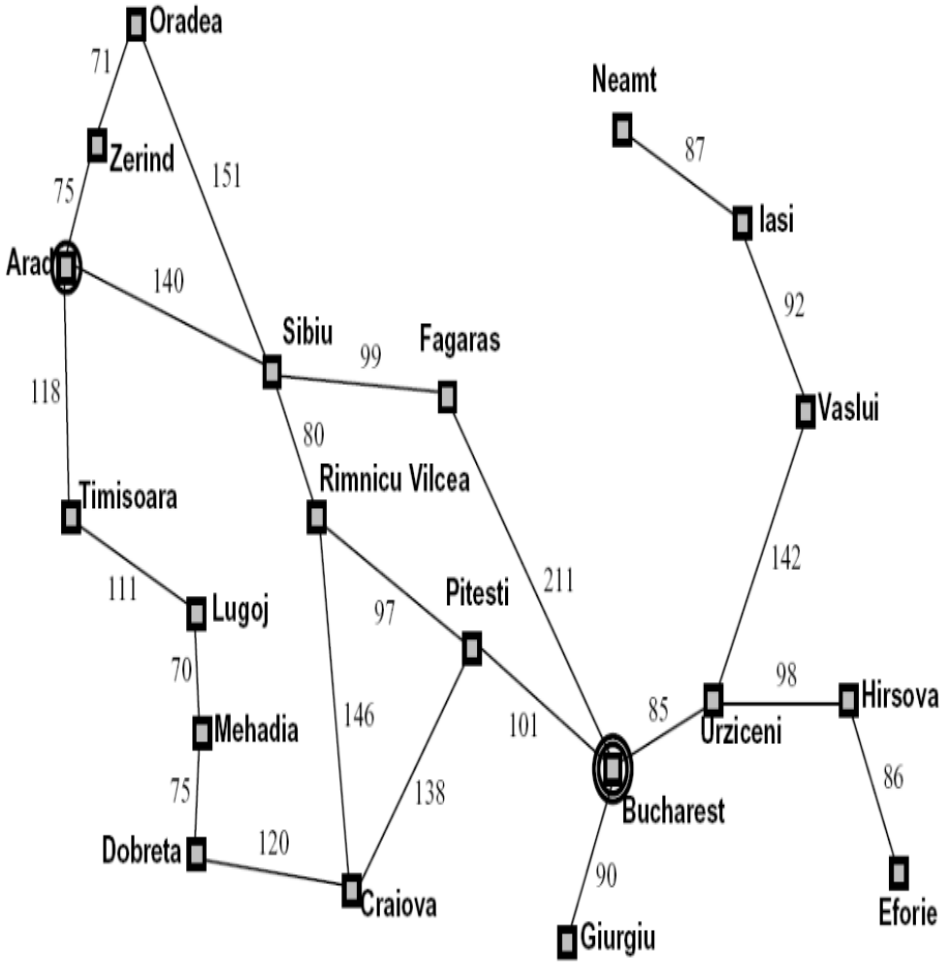

- **Espace d'états**
  - Villes

# exemple: Recherche de chemins



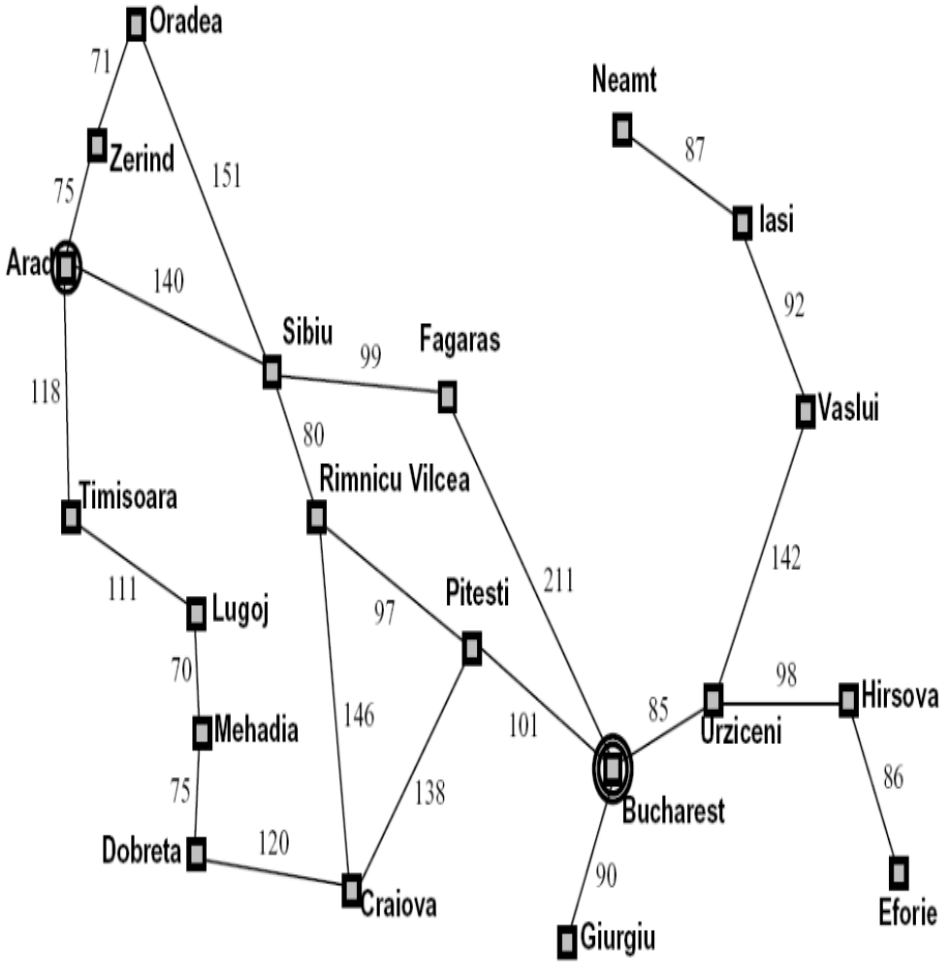

- **Espace d'états**
  - Villes
- **Fonction succession** :

# exemple: Recherche de chemins



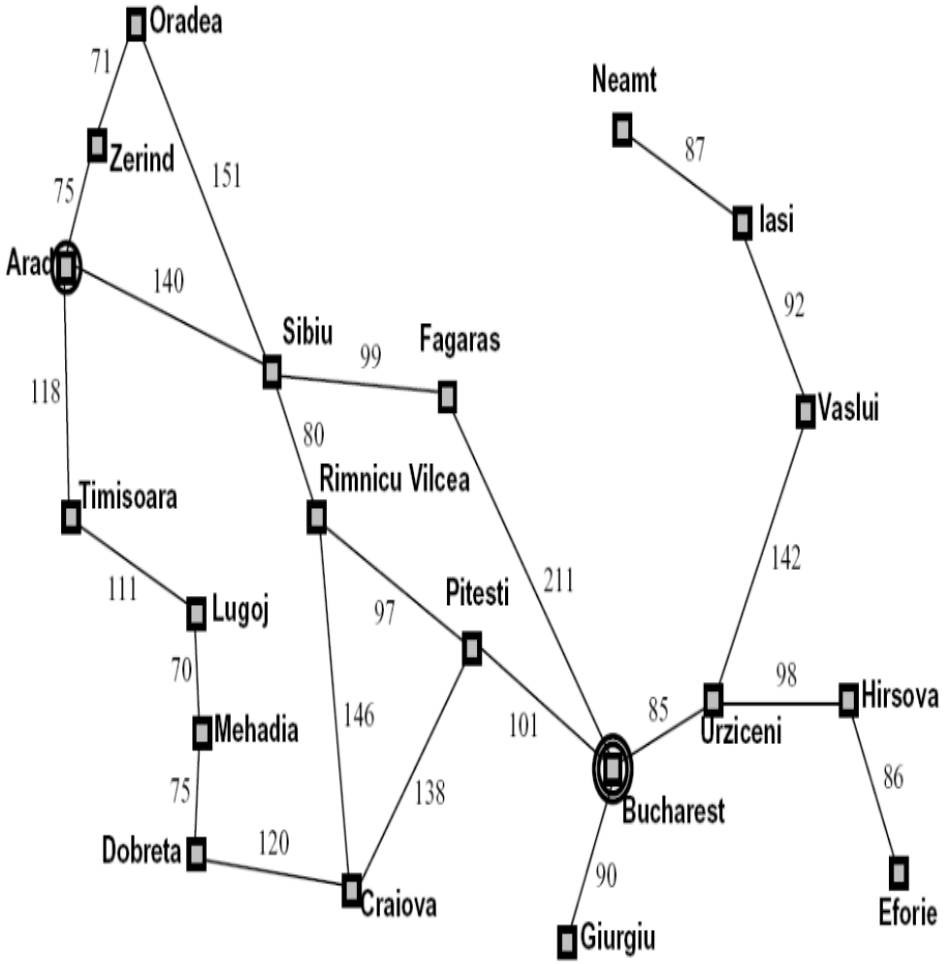

- **Espace d'états**
  - Villes
- **Fonction succession** :
  - Chemin sortant d'une ville, coût est la distance

# exemple: Recherche de chemins



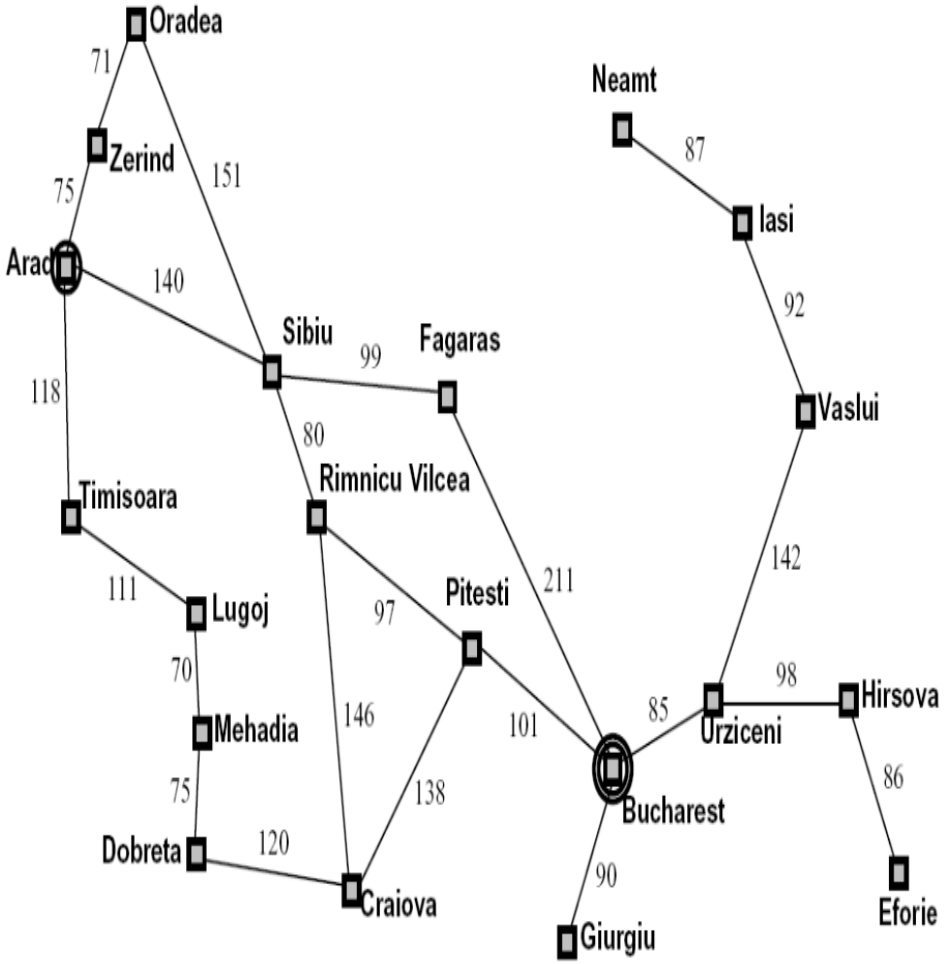

- **Espace d'états**
  - Villes
- **Fonction succession** :
  - Chemin sortant d'une ville, coût est la distance
- **Etat de départ et Objectif**:

# exemple: Recherche de chemins




- **Espace d'états**
  - Villes
- **Fonction succession** :
  - Chemin sortant d'une ville, coût est la distance
- **Etat de départ et Objectif**:
  - Argad, Bucharest
- **Solution ?**

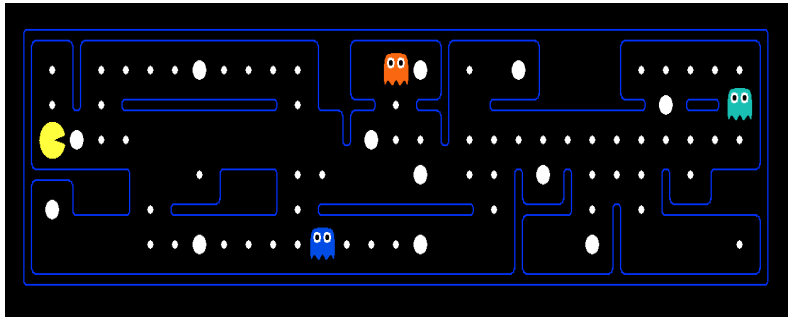
# exemple: Recherche de chemins




- **Espace d'états**
  - Villes
- **Fonction succession** :
  - Chemin sortant d'une ville, coût est la distance
- **Etat de départ et Objectif**:
  - Argad, Bucharest
- **Solution ?**
  - Chemin

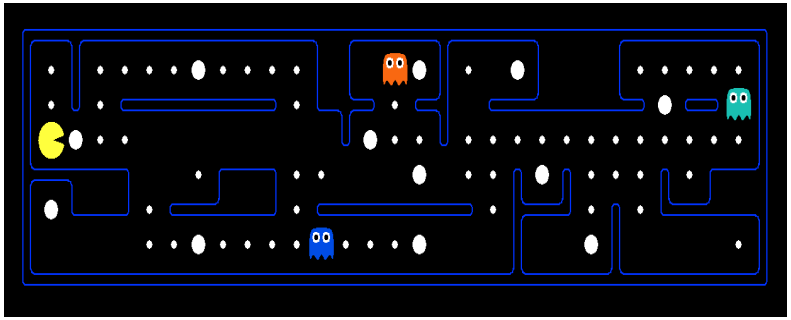
# Espace d'état

l'espace des états doit inclure **tous** les détails de l'environnement.



# Espace d'état

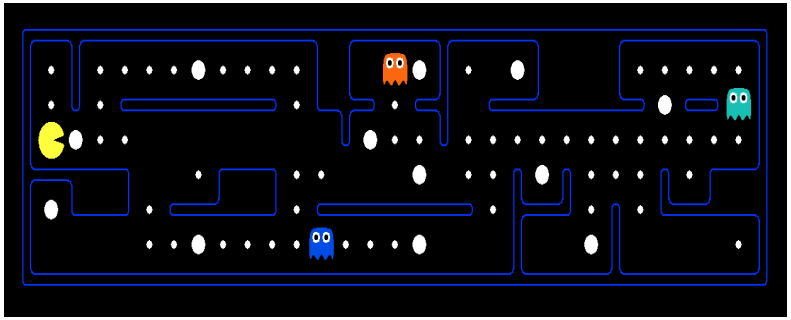
l'espace des états doit inclure **tous** les détails de l'environnement.



Un état d'exploration garde seulement les détails pour la planification (*abstraction*)

# Espace d'état

l'espace des états doit inclure **tous** les détails de l'environnement.

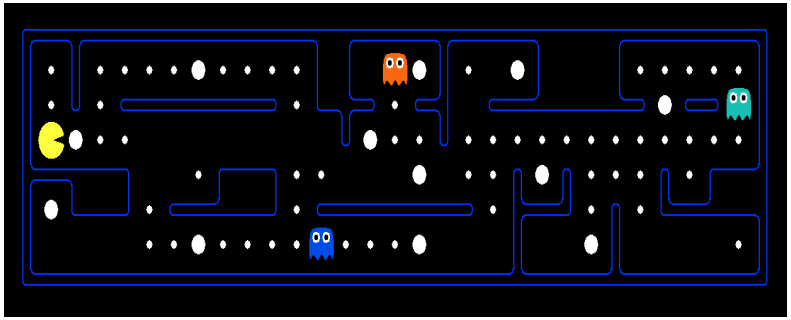


Un état d'exploration garde seulement les détails pour la planification (*abstraction*)

- Problem: chemin
  - États: positions $(x, y)$.
  - Actions: NSEW
  - Succession : nouvelle position
  - Objectif: $(x, y) = \text{END}$

# Espace d'état

l'espace des états doit inclure **tous** les détails de l'environnement.

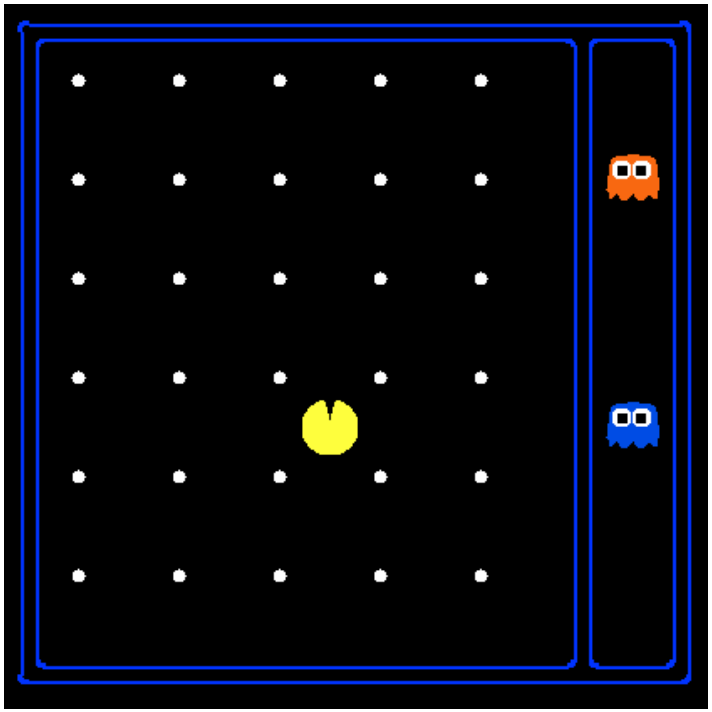


Un état d'exploration garde seulement les détails pour la planification (*abstraction*)

- Problem: chemin
  - États: positions $(x, y)$.
  - Actions: NSEW
  - Succession : nouvelle position
  - Objectif: $(x, y) = \text{END}$

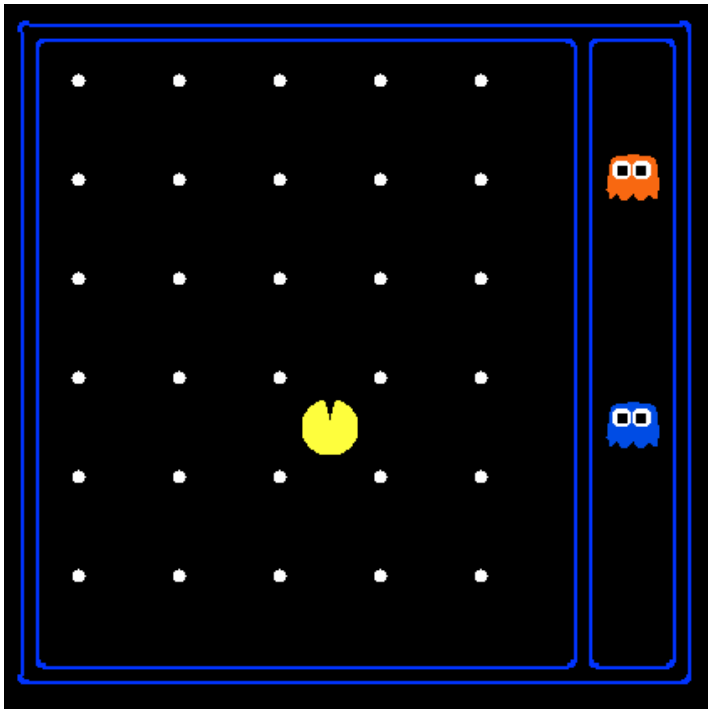
- Problem: Manger tous les points.
  - États: positions $\{(x, y), \text{booléens des points}\}$.
  - Actions: NSEW
  - Succession : nouvelle position, mettre à jour les booléens
  - Objectif: tous les booléens = Faux

# Cardinal de l'espace d'états



- Espace d'état:
  - Positions de l'agent: $\mathbf{120}$
  - Position des points : $\mathbf{30}$
  - Position des fantômes $\mathbf{12}$
  - Action de l'agent : **NSEW**
- Cardinal espace d'état

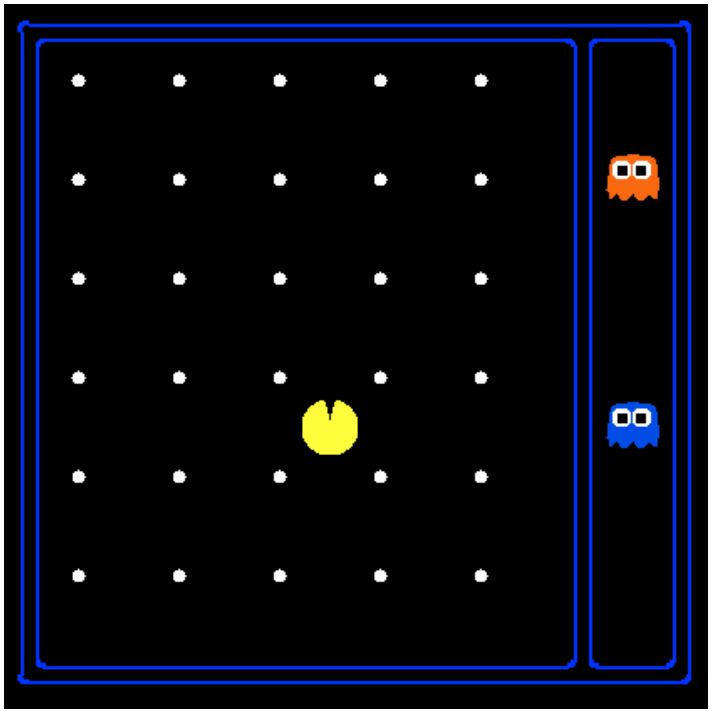
# Cardinal de l'espace d'états



- Espace d'état:
  - Positions de l'agent: $\mathbf{120}$
  - Position des points : $\mathbf{30}$
  - Position des fantômes $\mathbf{12}$
  - Action de l'agent : **NSEW**
- Cardinal espace d'état

$$120 * 2^{30} * 12^2 * 4$$

# Cardinal de l'espace d'états



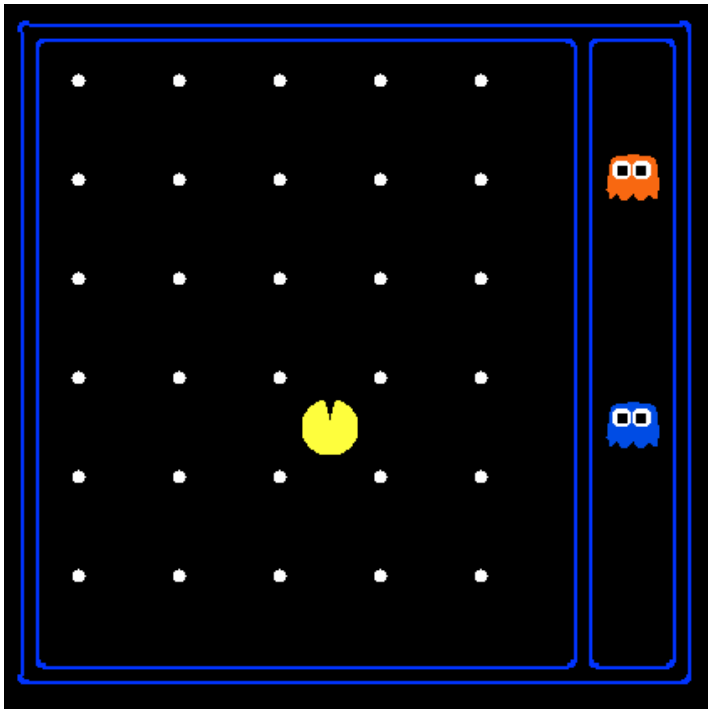
- Espace d'état:
  - Positions de l'agent: $\mathbf{120}$
  - Position des points : $\mathbf{30}$
  - Position des fantômes $\mathbf{12}$
  - Action de l'agent : **NSEW**
- Cardinal espace d'état

$$120 * 2^{30} * 12^2 * 4$$

- Cardinal (Chemin):

$$120$$

# Cardinal de l'espace d'états



- Espace d'état:
  - Positions de l'agent: $\mathbf{120}$
  - Position des points : $\mathbf{30}$
  - Position des fantômes $\mathbf{12}$
  - Action de l'agent : **NSEW**
- Cardinal espace d'état

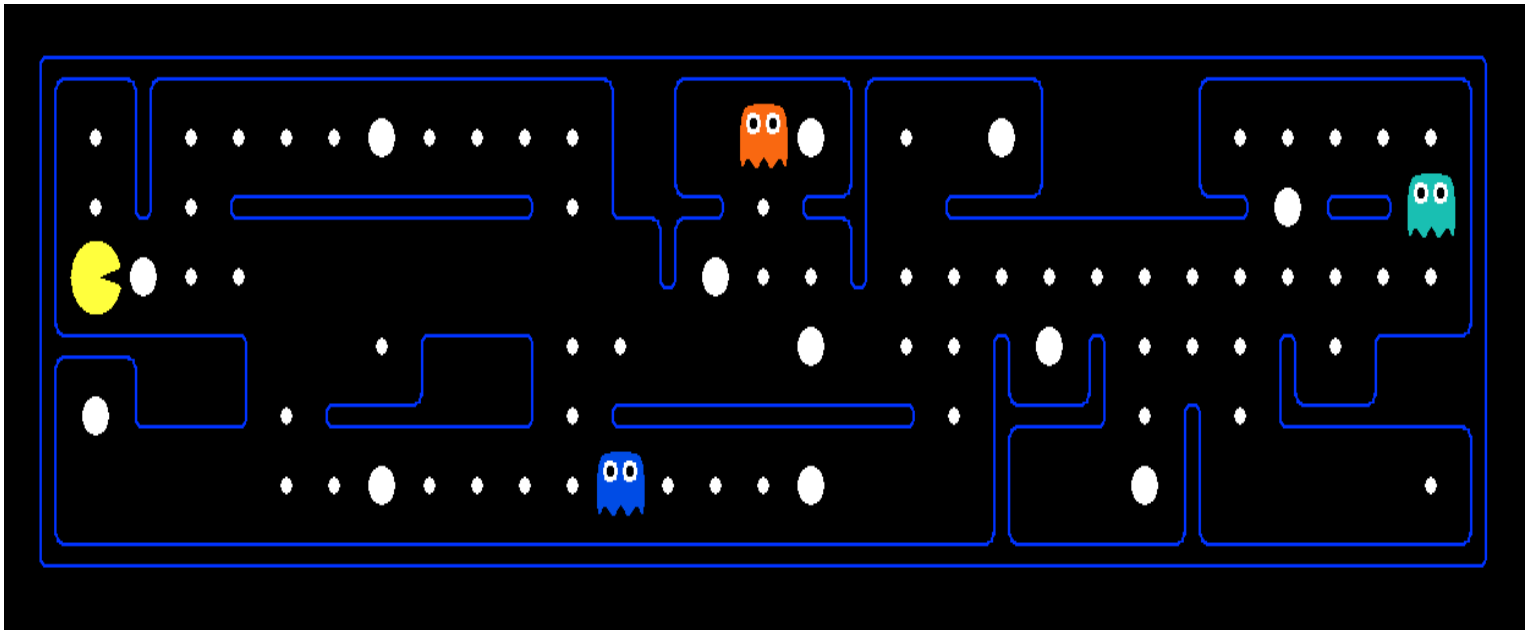
$$120 * 2^{30} * 12^2 * 4$$

- Cardinal (Chemin):

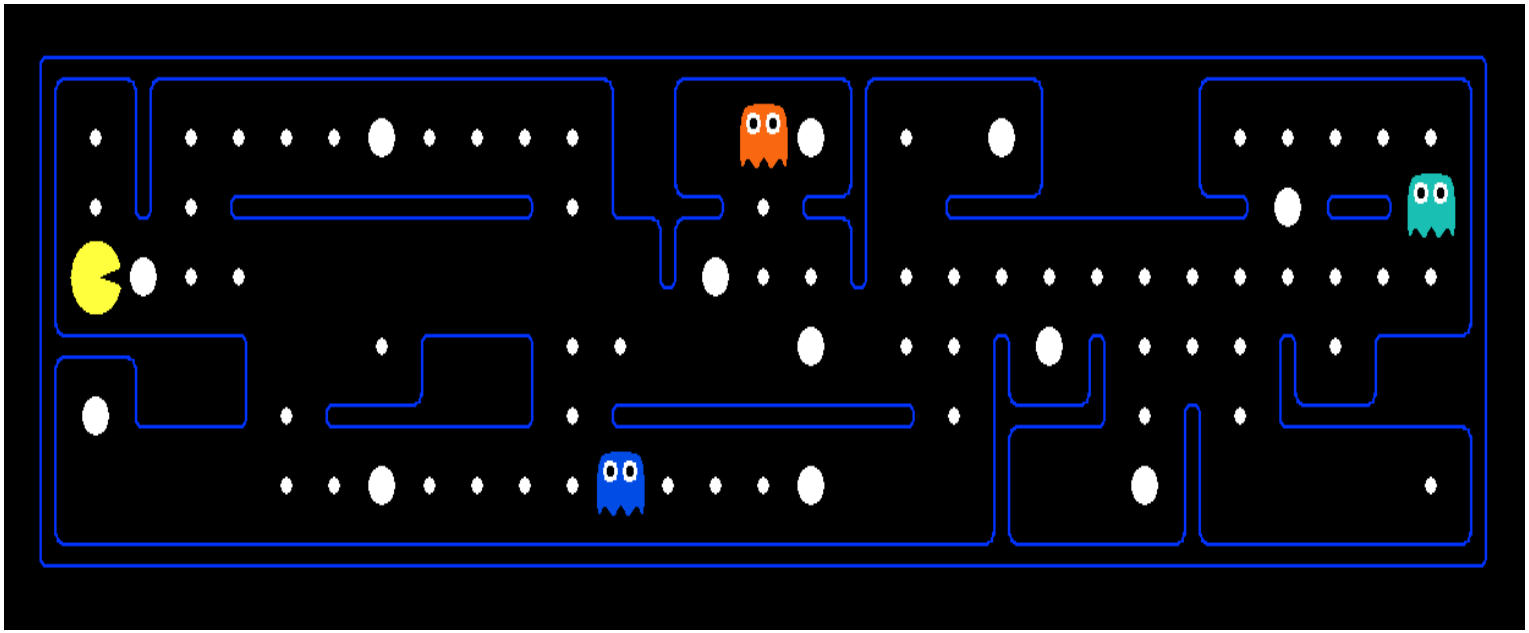
$$120$$

- Cardinal (Prendre tous les points)

$$120 * 2^{30}$$



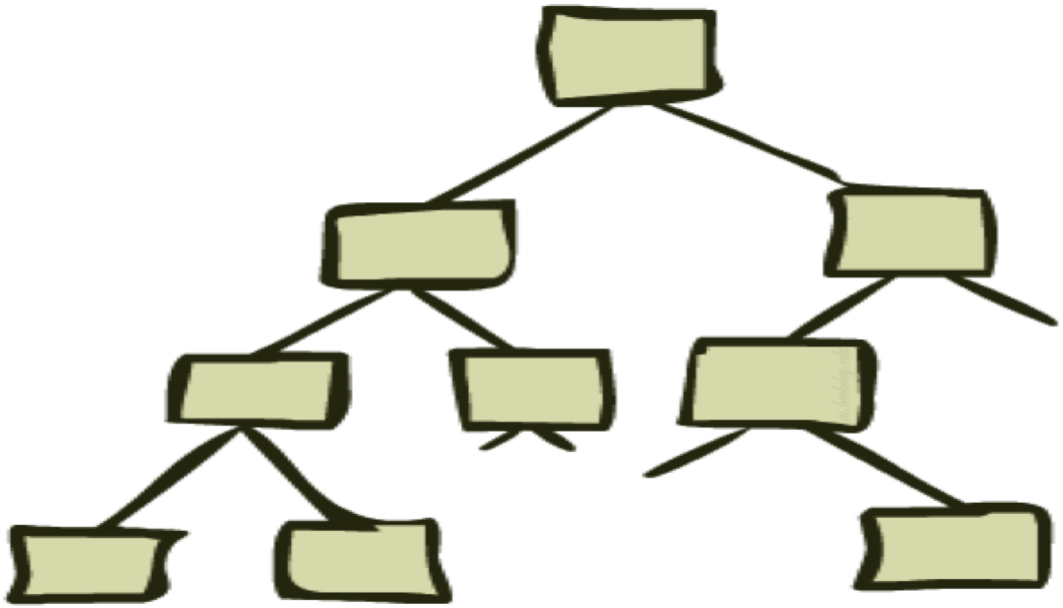
- Le problème est de prendre tous les points, en gardant les fantômes en peur constante.
- Quelle sont les **informations** à inclure dans votre espace d'exploration?



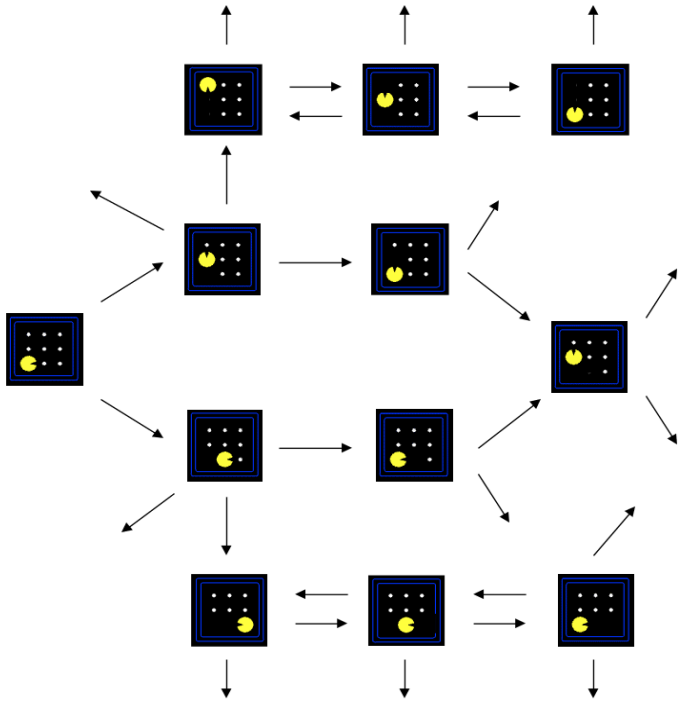
- Le problème est de prendre tous les points, en gardant les fantômes en peur constante.
- Quelle sont les **informations** à inclure dans votre espace d'exploration?

- Position Agent.
- Booléens des points
- Booléen des pullules de puissance.
- Temps restant de la cellule de puissance

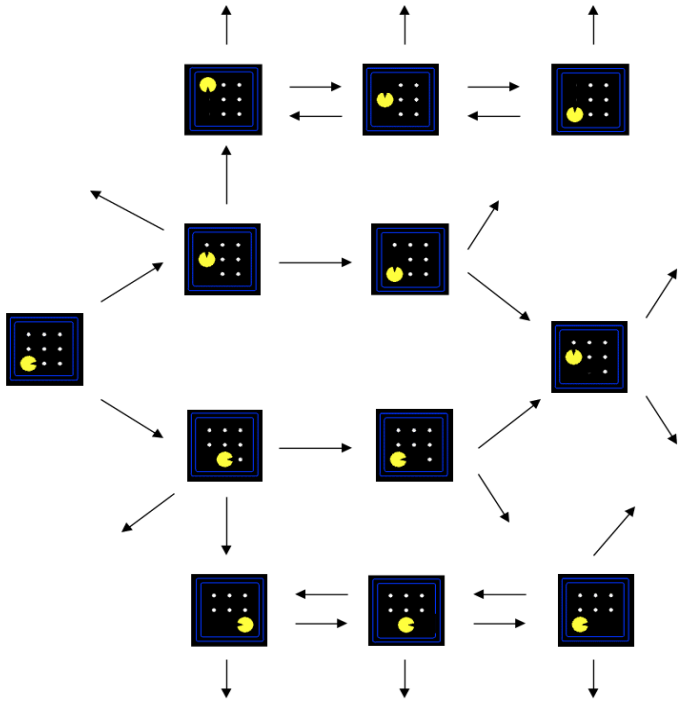
# Graphe d'espace d'états



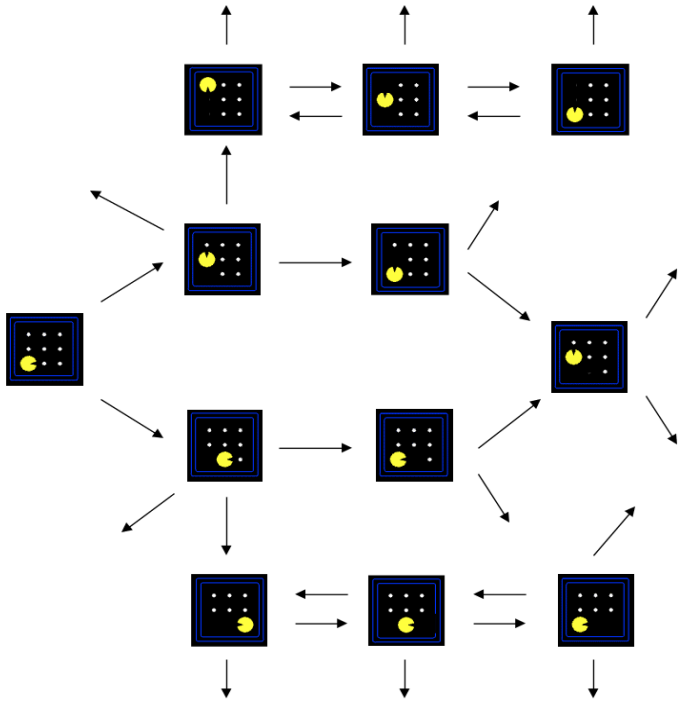
# Graphe d'espace d'états



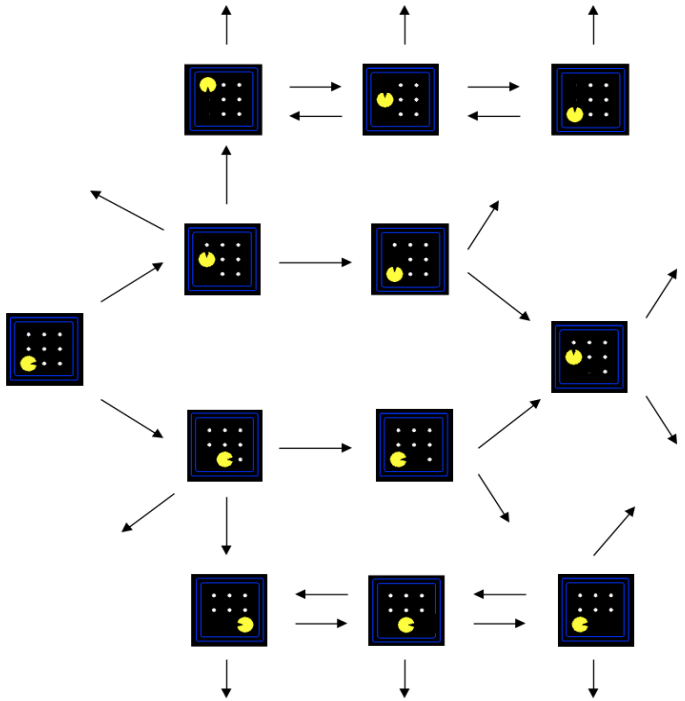
- **Représentation mathématique**



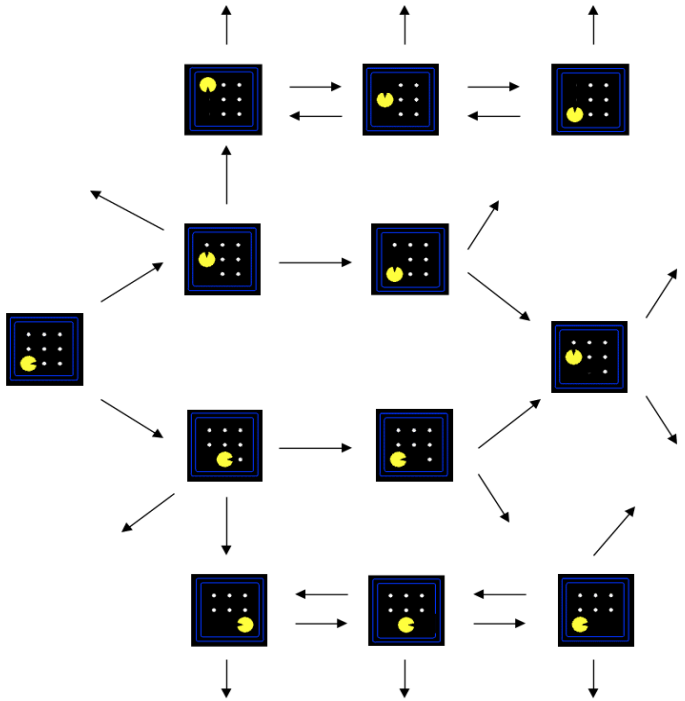
- **Représentation mathématique**
  - Les nœuds sont des **abstractions** des configurations possible.



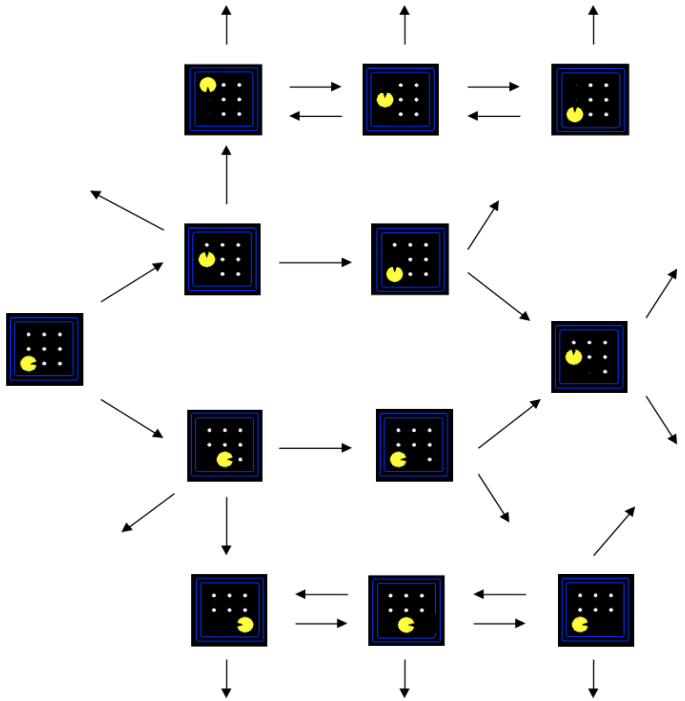
- **Représentation mathématique**
  - Les nœuds sont des **abstractions** des configurations possible.
  - Les arrêtes sont des **successions** selon une **action**



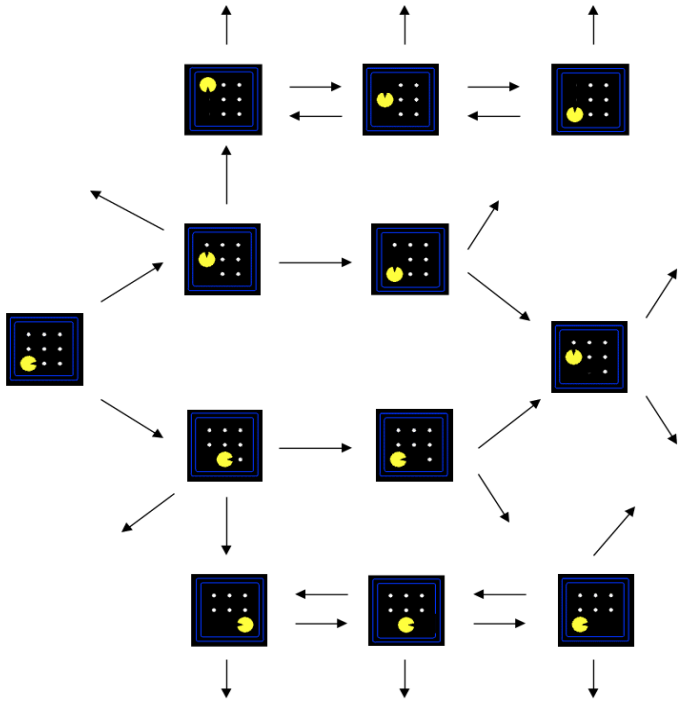
- **Représentation mathématique**
  - Les nœuds sont des **abstractions** des configurations possible.
  - Les arrêtes sont des **successions** selon une **action**
  - Le nœud objectif est un ensemble des nœuds du graphe.



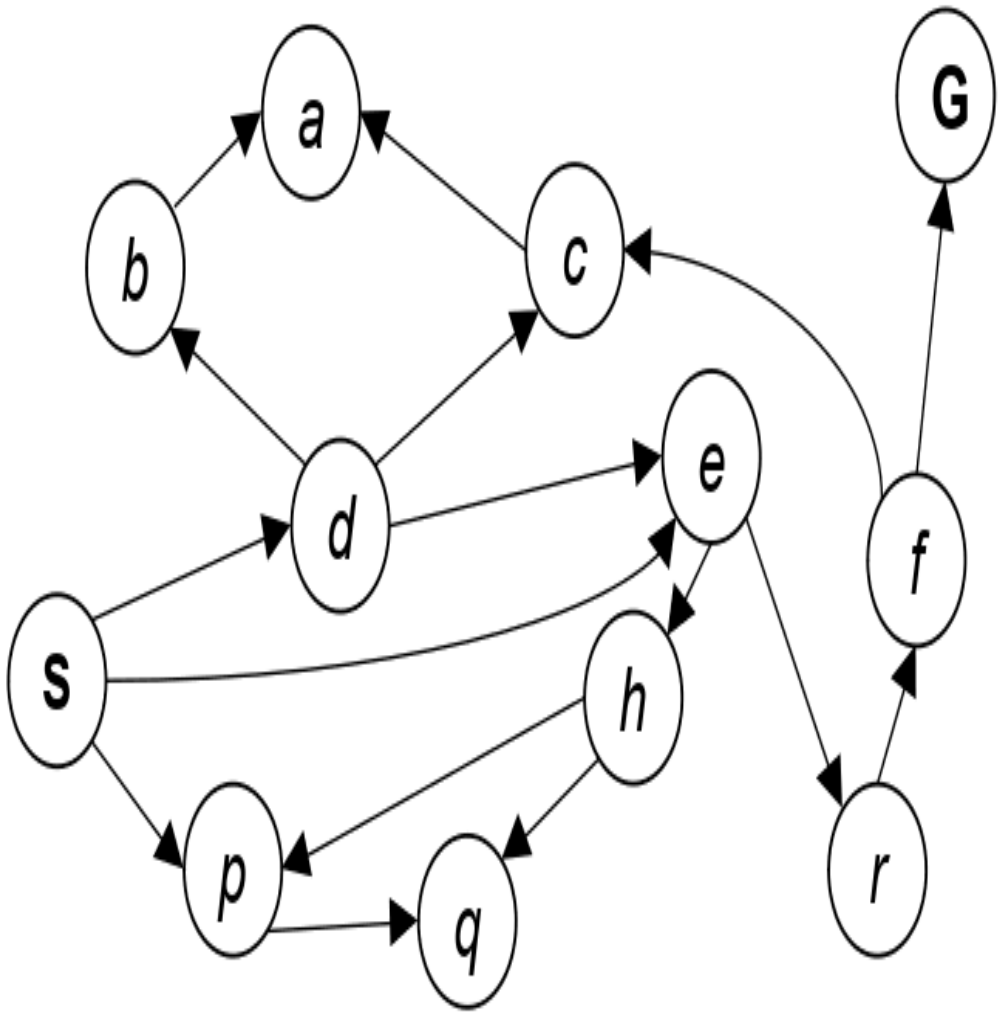
- **Représentation mathématique**
  - Les nœuds sont des **abstractions** des configurations possible.
  - Les arrêtes sont des **successions** selon une **action**
  - Le nœud objectif est un ensemble des nœuds du graphe.
- Chaque configuration est **unique**.

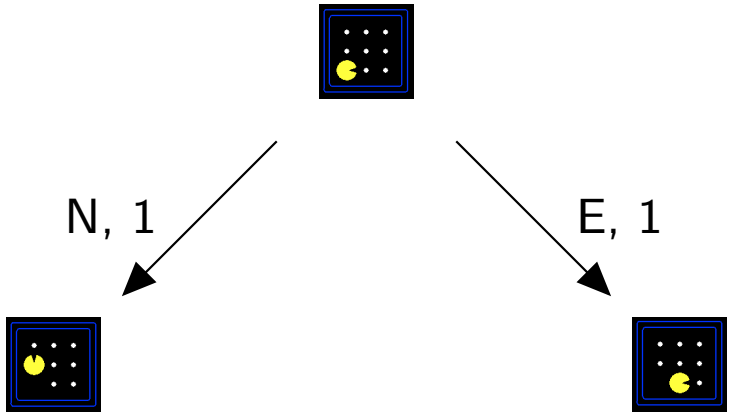


- **Représentation mathématique**
  - Les nœuds sont des **abstractions** des configurations possible.
  - Les arrêtes sont des **successions** selon une **action**
  - Le nœud objectif est un ensemble des nœuds du graphe.
- Chaque configuration est **unique**.
- On peut rarement construire un tel graphe. Mais il reste utile.

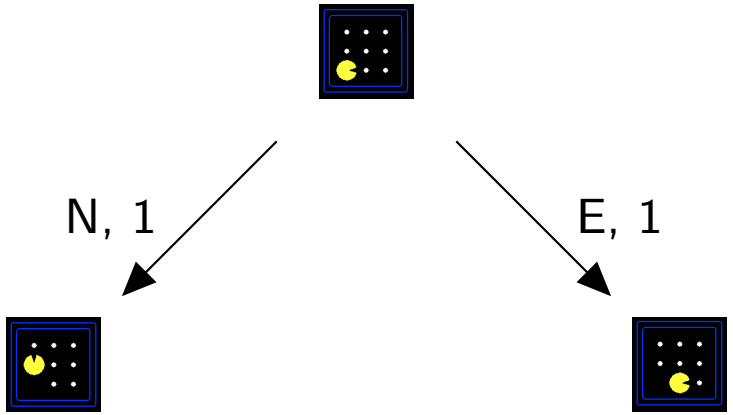


- **Représentation mathématique**
  - Les nœuds sont des **abstractions** des configurations possible.
  - Les arrêtes sont des **successions** selon une **action**
  - Le nœud objectif est un ensemble des nœuds du graphe.
- Chaque configuration est **unique**.
- On peut rarement construire un tel graphe. Mais il reste utile.



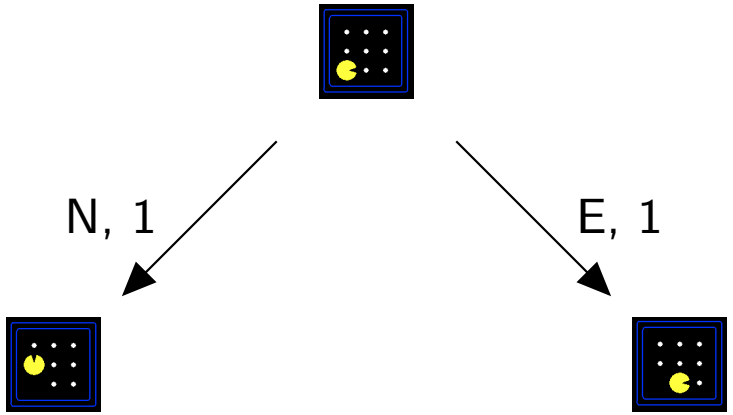



## Arbre d'exploration



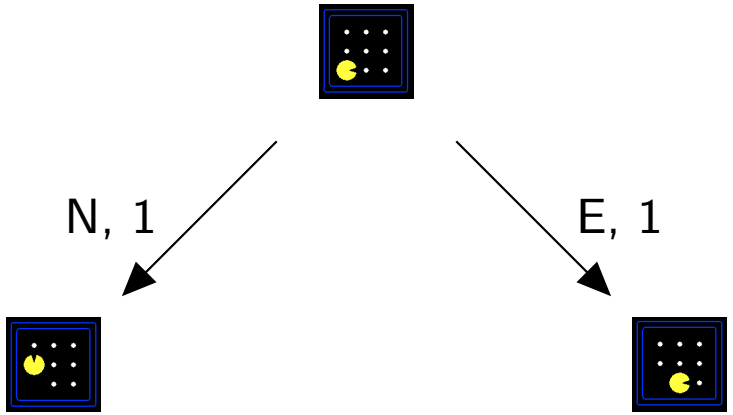

## Arbre d'exploration

- Liste des configurations possible suite à une action.



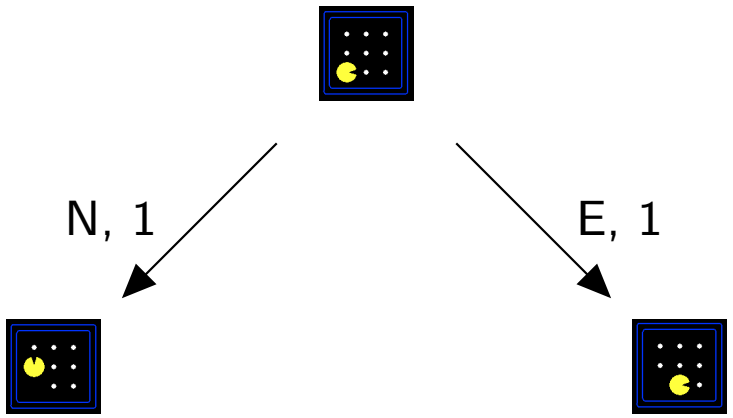

## Arbre d'exploration

- Liste des configurations possible suite à une action.
- La **racine** est l'état initial.



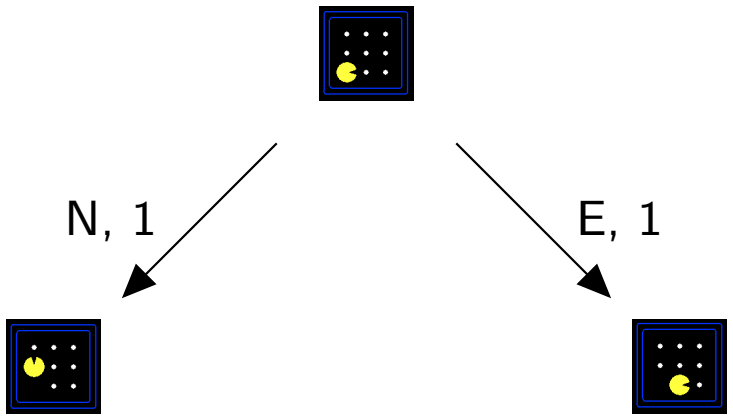

## Arbre d'exploration

- Liste des configurations possible suite à une action.
- La **racine** est l'état initial.
- Les **fils** sont les **successions**.




## Arbre d'exploration

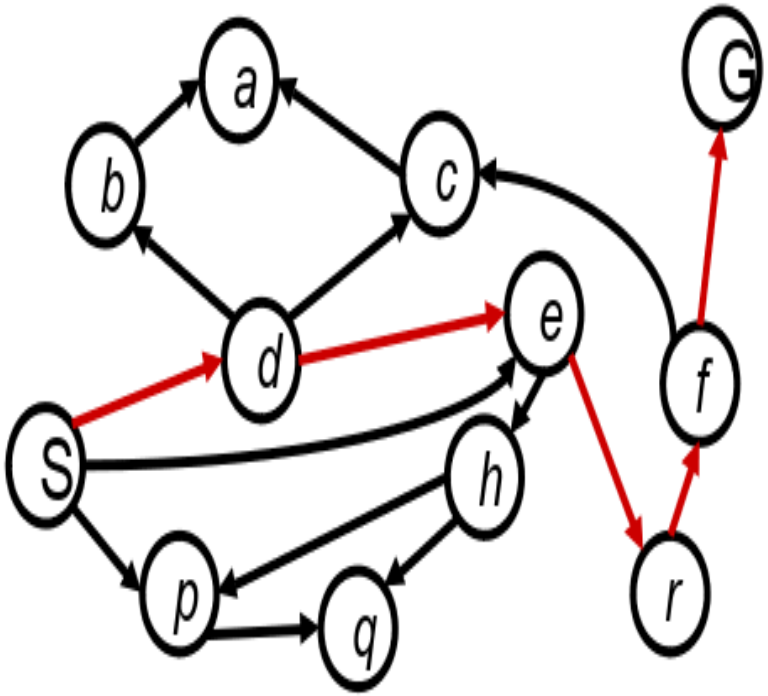
- Liste des configurations possible suite à une action.
- La **racine** est l'état initial.
- Les **fils** sont les **successions**.
- **Noeuds** sont des configurations avec un **plan**.(i.e meme configuration dans différents noeuds).

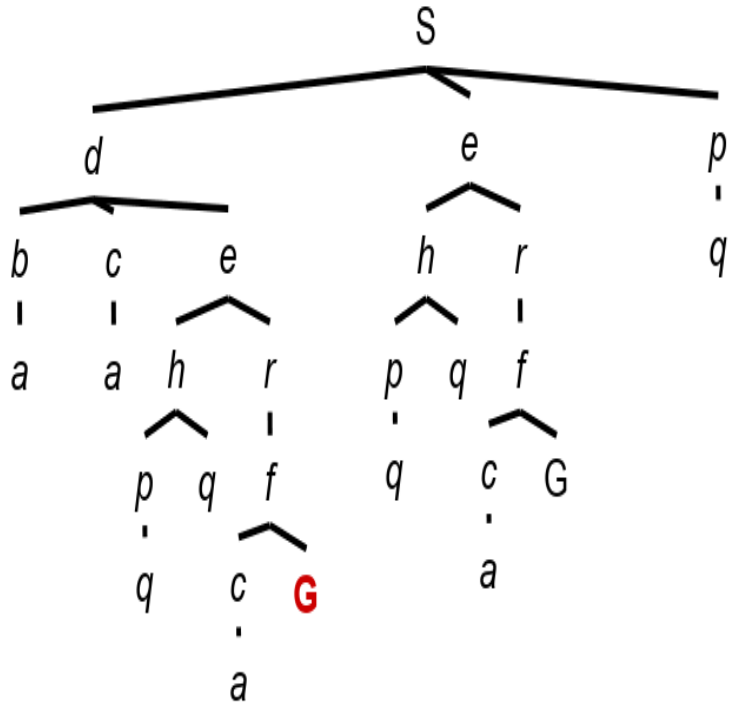



## Arbre d'exploration

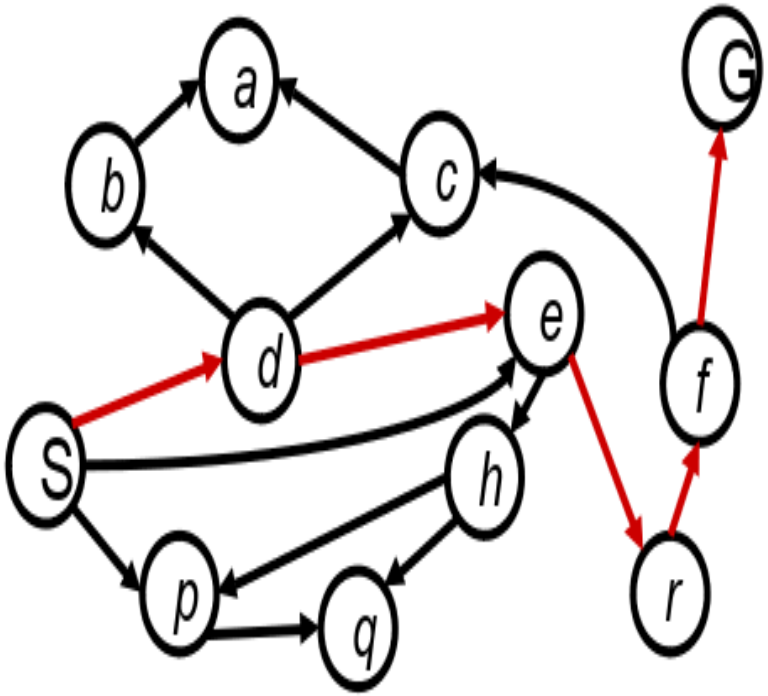
- Liste des configurations possible suite à une action.
- La **racine** est l'état initial.
- Les **fils** sont les **successions**.
- **Noeuds** sont des configurations avec un **plan**.(i.e meme configuration dans différents noeuds).
- **Pour la majorité des problèmes, impossible de construire toute l'arbre**.

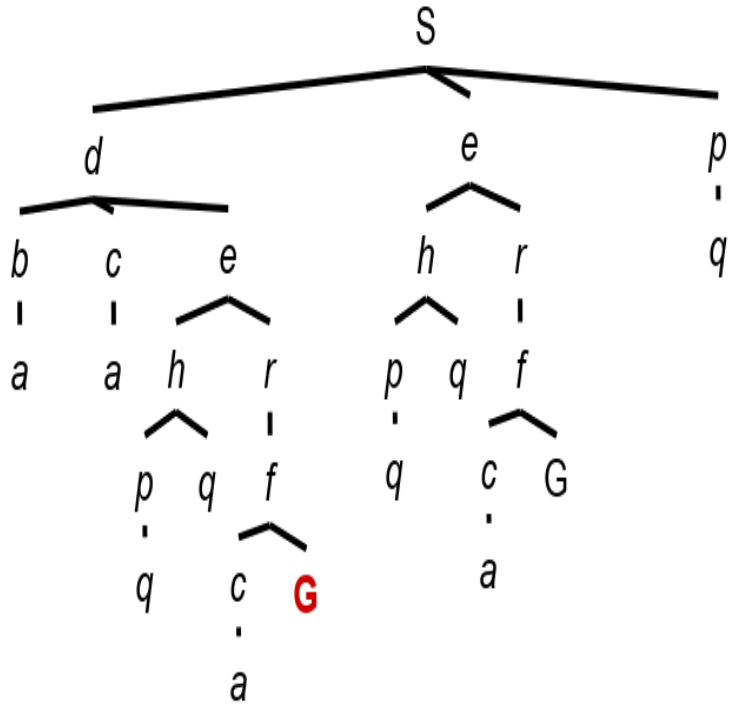
# Representation graph Vs Arbre



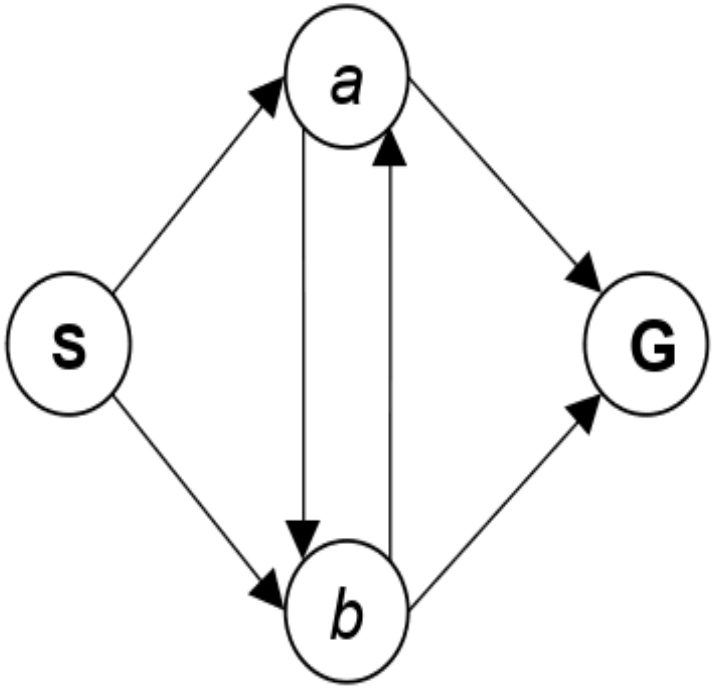
# Representation graph Vs Arbre




- Chaque noeud dans **l'arbre** est un chemin complet au node initial.
- L'arbre est construit **progressivement** par demande.

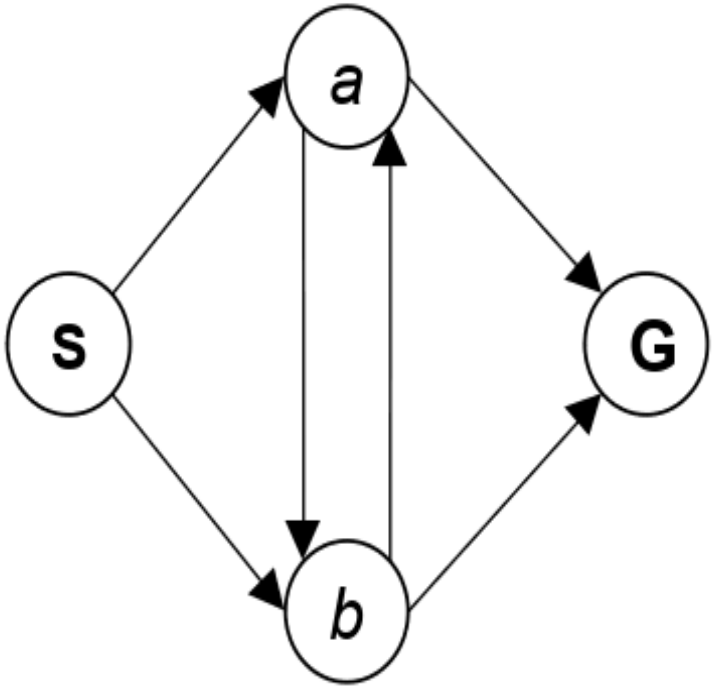
On considère le graphe d'états:




Quelle est la **profondeur** de l'arbre d'exploration?

# Quiz 3

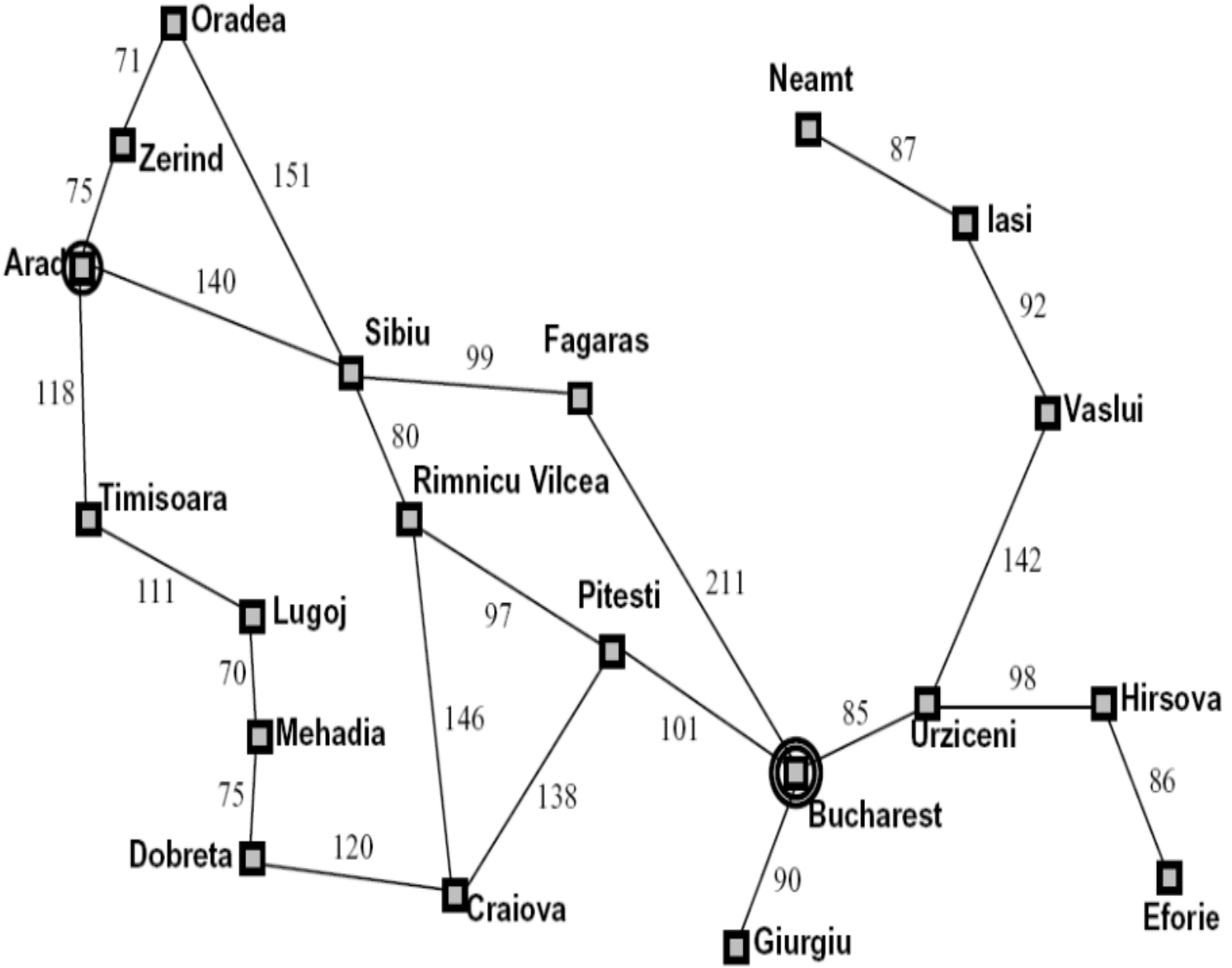
On considère le graphe d'états:



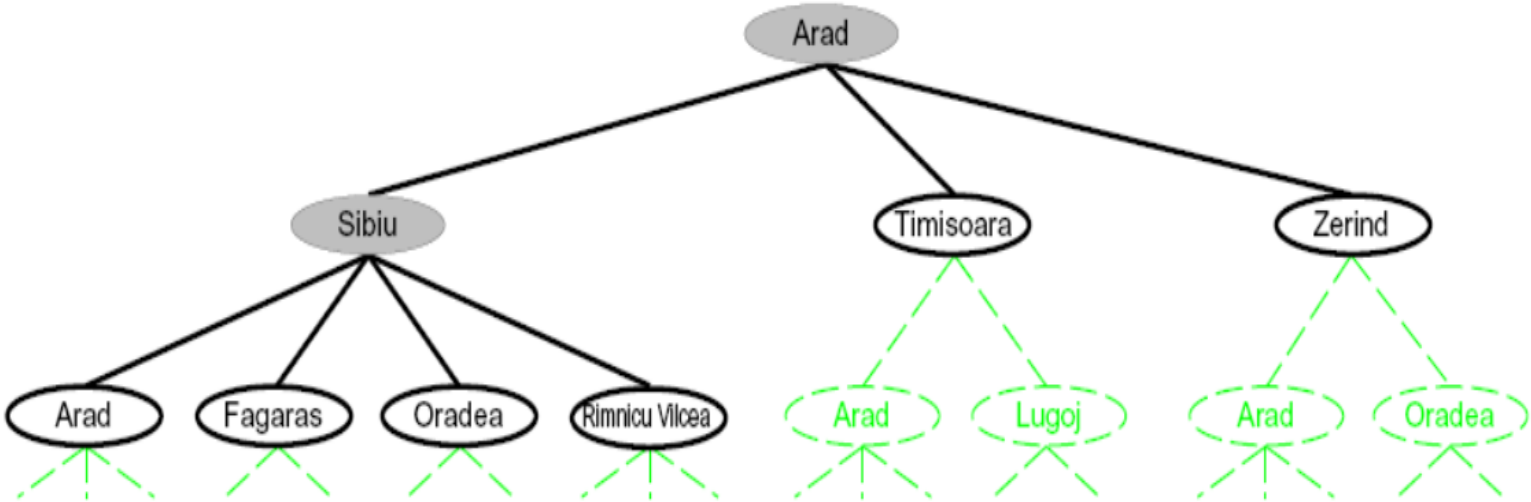

Quelle est la **profondeur** de l'arbre d'exploration?

$\infty$

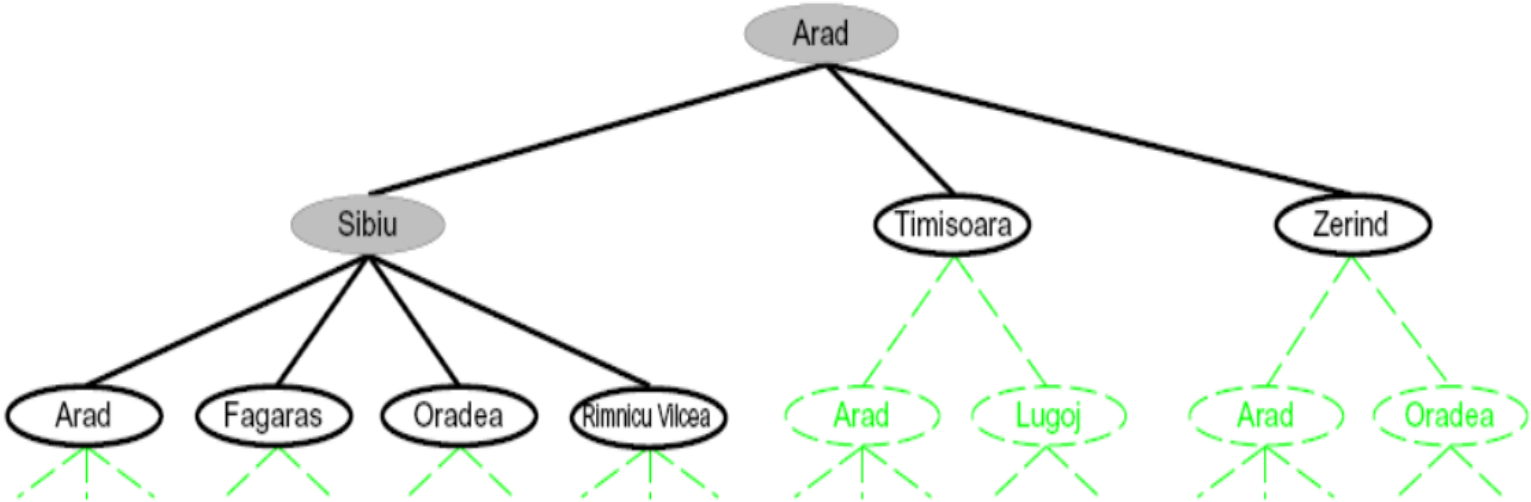
# Exemple Arbre d'exploration



# Arbre d'exploration



# Arbre d'exploration




## Exploration

- Explorer des plans *potentiels* (noeuds dans l'arbre).
- Maintenir une **frontière** des plans partiels qui doivent être **considérés**.
- Explorer le **minimum** de neouds possibles.

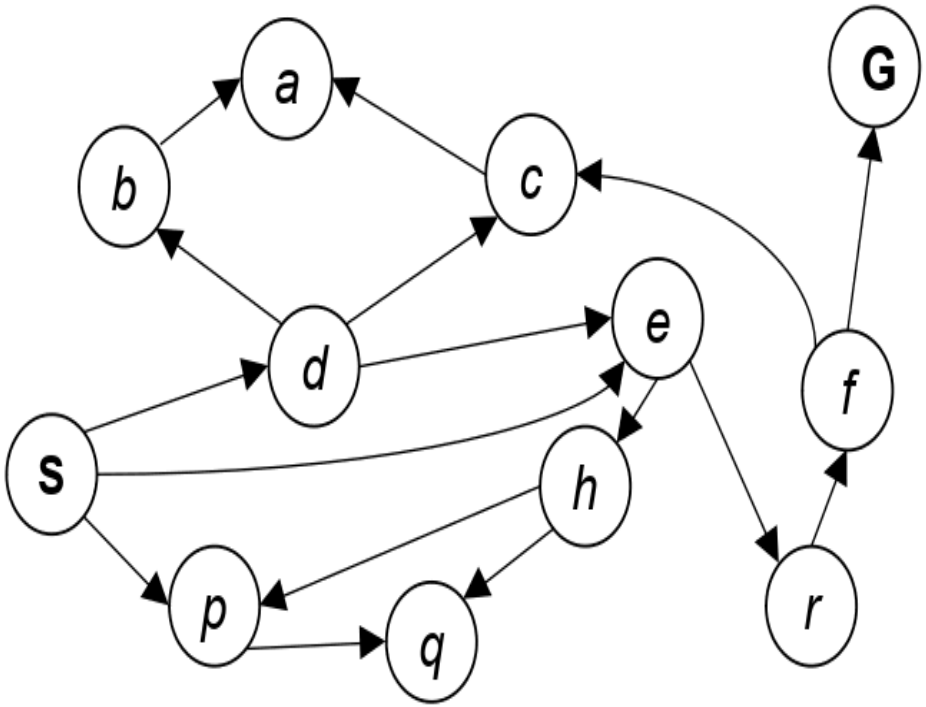
**Algorithm 1** Description informelle des algorithmes d'exploration en arbre

```
Function (EXPLORER-ARBRE)(problème, stratégie)
initaliser la frontière avec l'état initial.
loop
    if frontière vide then
        return échec                          #Aucun noeud à explorer
    end if
    Choisir une feuille de la frontière selon statégie
    if feuille contient Objectif then
        return Succès                         #Solution trouvée
    end if
    Développer le noeud en ajoutant ces fils à la frontière
end loop
EndProcedure
```

## Importants messages

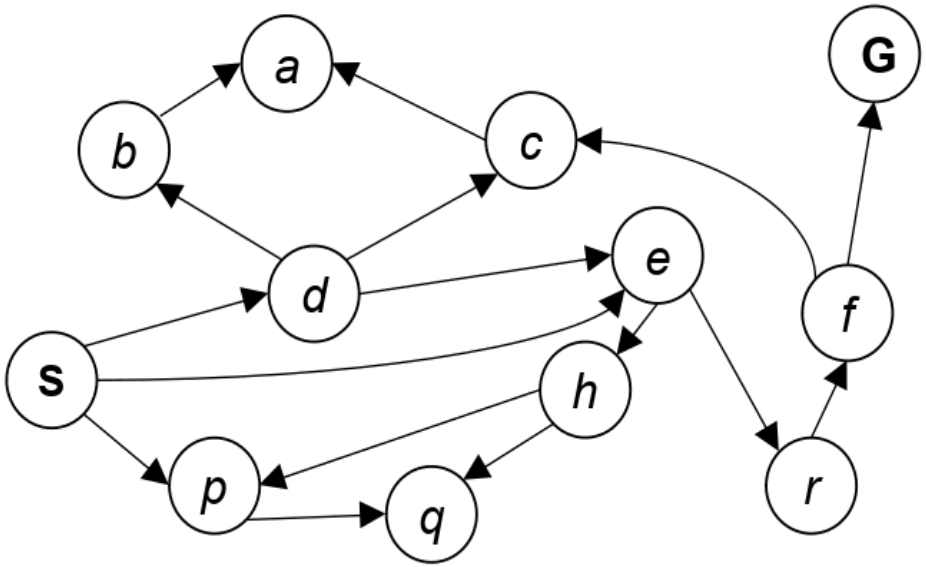
- Frontière
- Développement
- Stratégie d'exploration.

# Exploration détaillée

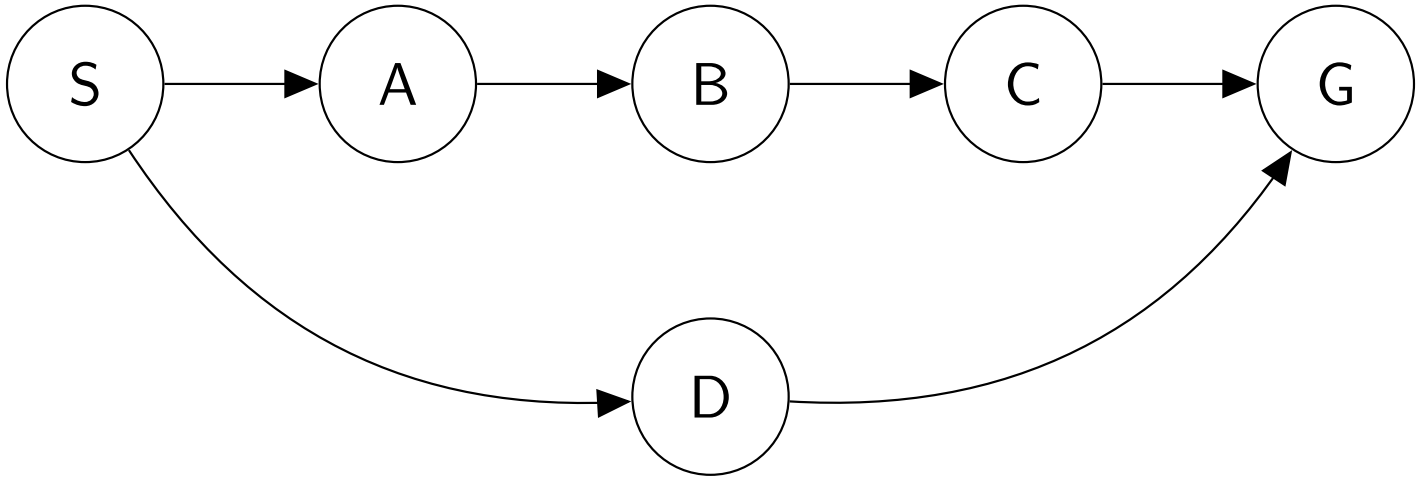


# Depth First Search (Recherche en profondeur)

# Depth First Search




- **Stratégie** : Développer le noeuds le plus **profond**.
- **Implémentation**: Frontière est une **PILE**(LIFO).



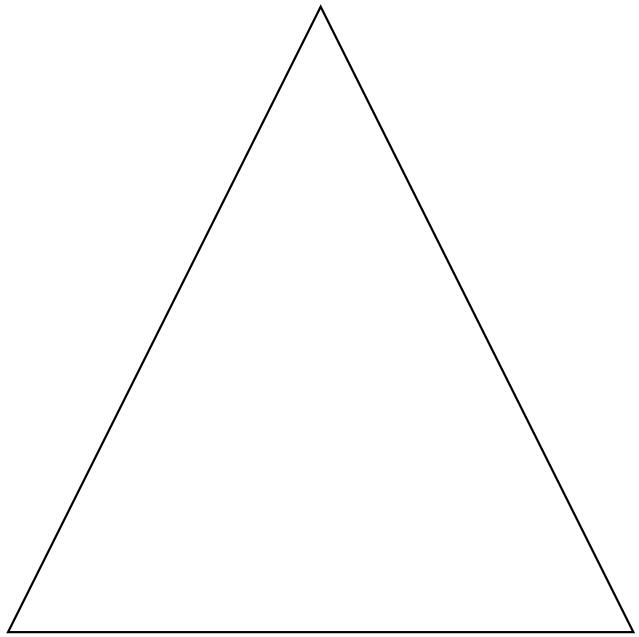

## Question

Donner le chemin calculé par **DPS** de son exploration du point **S** jusqu'au point **G**.

# Propriété d'un algorithme d'exploration

- **Complet**: Assure de trouver une solution s'il elle existe.

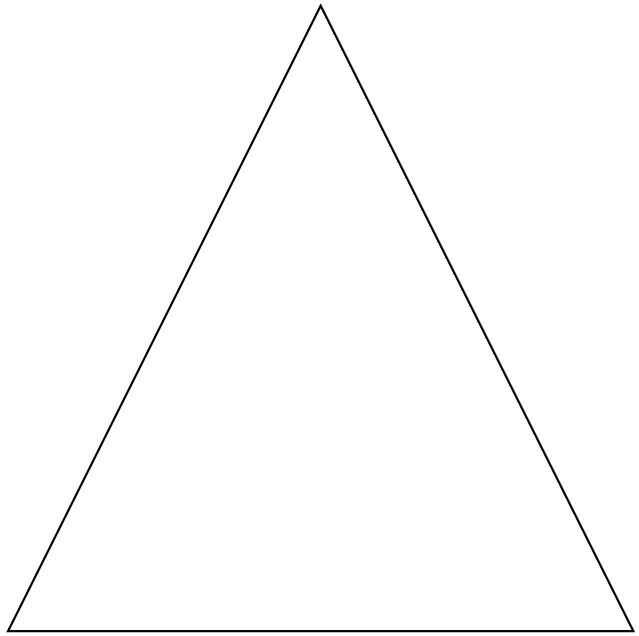
- **Arbre d'exploration**:



# Propriété d'un algorithme d'exploration

- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.

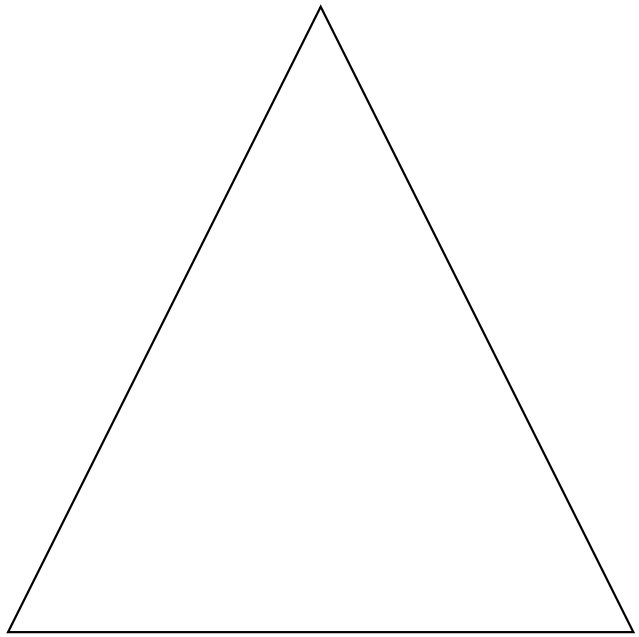
- **Arbre d'exploration**:



# Propriété d'un algorithme d'exploration

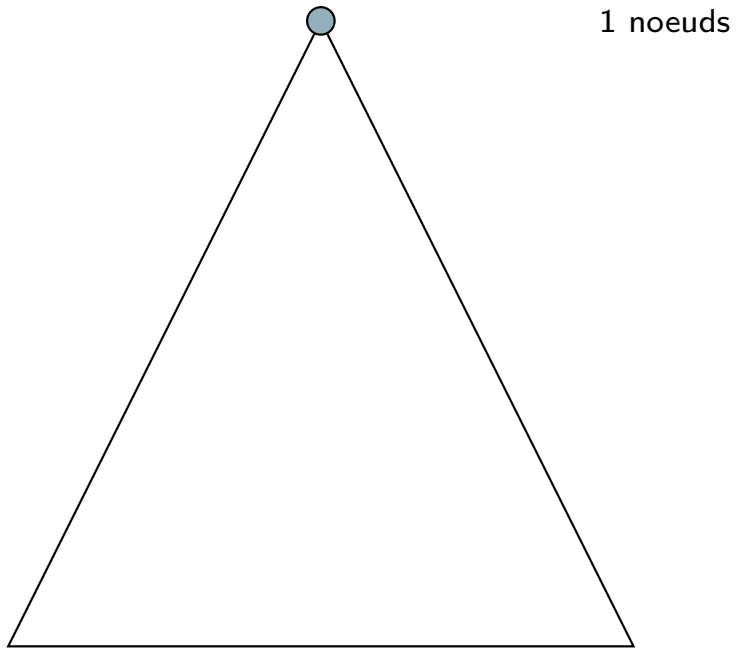
- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.
- **Complexité en temps**

- **Arbre d'exploration**:



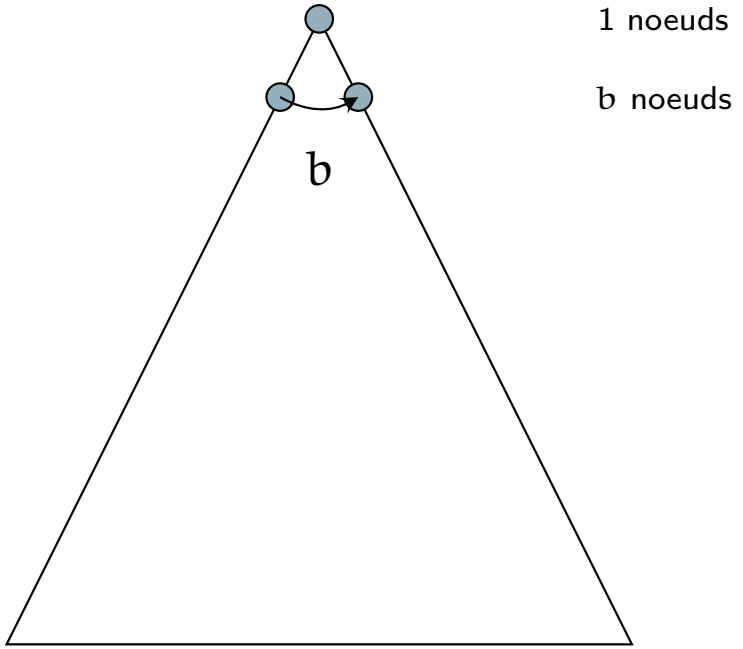
# Propriété d'un algorithme d'exploration

- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.
- **Complexité en temps**
- **Complexité en espace**
- **Arbre d'exploration**:



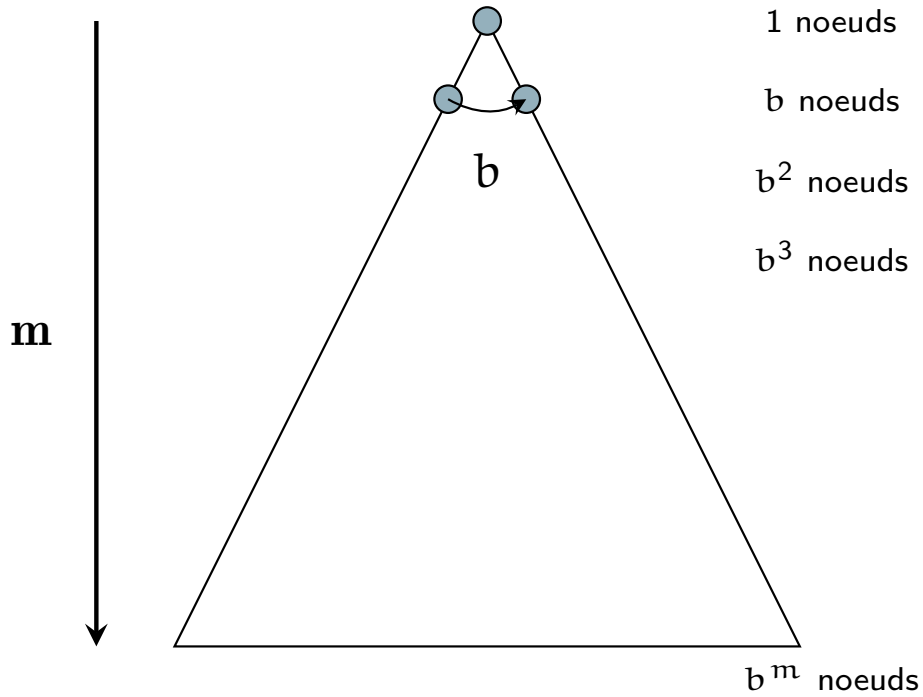
# Propriété d'un algorithme d'exploration

- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.
- **Complexité en temps**
- **Complexité en espace**
- **Arbre d'exploration**:
  - $\mathbf{b}$ : facteur de **branchement**



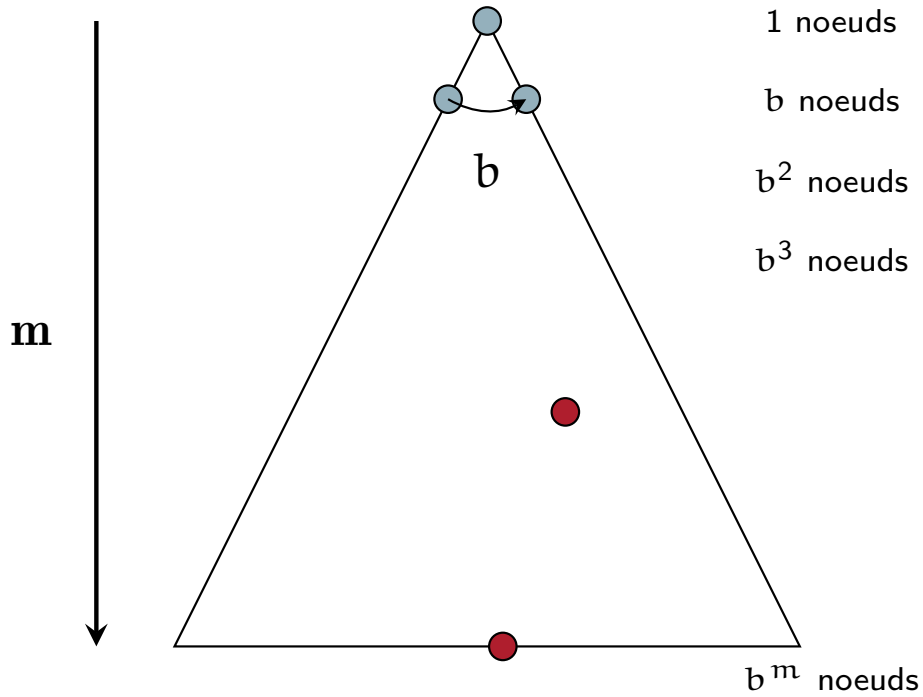
# Propriété d'un algorithme d'exploration

- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.
- **Complexité en temps**
- **Complexité en espace**
- **Arbre d'exploration**:
  - $\mathbf{b}$ : facteur de **branchement**
  - $\mathbf{m}$ : profondeur de l'arbre.



# Propriété d'un algorithme d'exploration

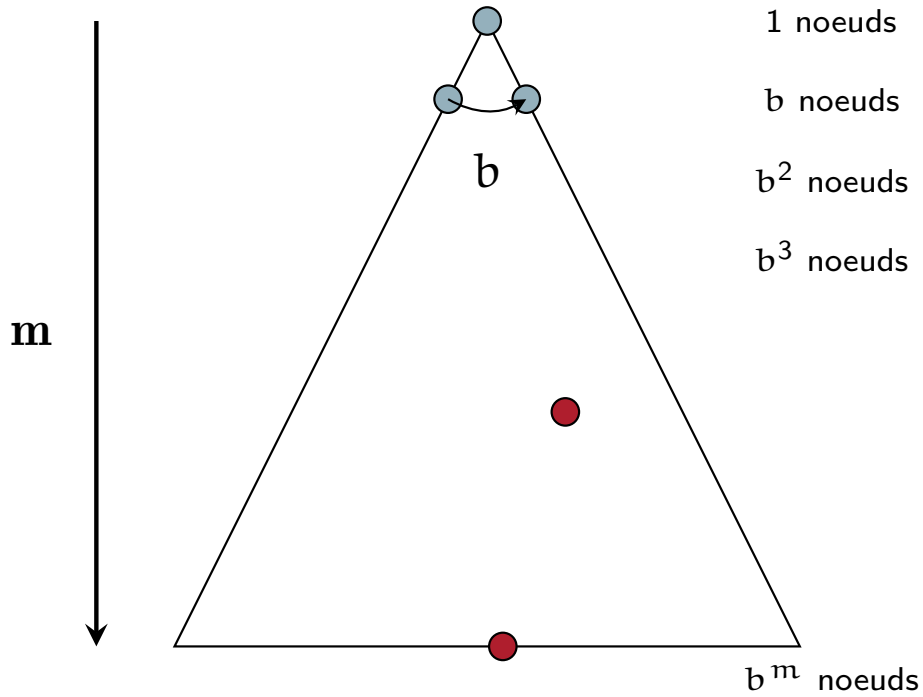
- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.
- **Complexité en temps**
- **Complexité en espace**
- **Arbre d'exploration**:
  - $\mathbf{b}$ : facteur de **branchement**
  - $\mathbf{m}$ : profondeur de l'arbre.
  - Solutions dans differents **profondeurs**



# Propriété d'un algorithme d'exploration

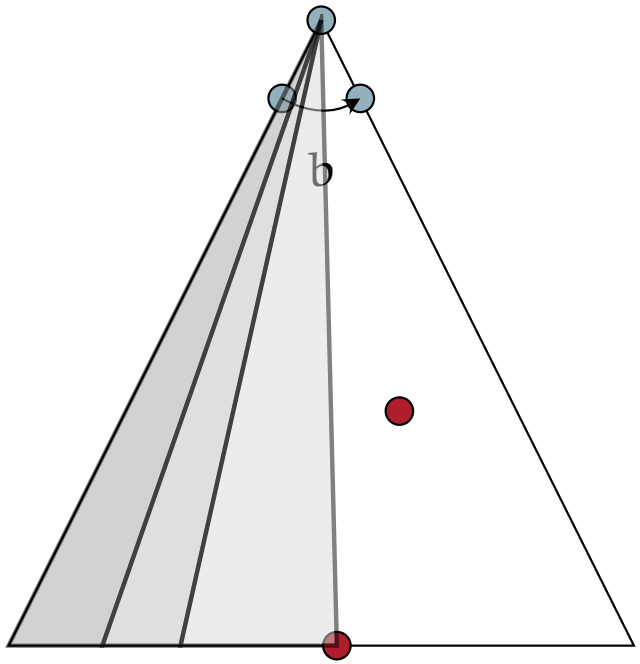
- **Complet**: Assure de trouver une solution s'il elle existe.
- **Optimal** : Trouve la solution avec un coût minimal.
- **Complexité en temps**
- **Complexité en espace**
- **Arbre d'exploration**:
  - $\mathbf{b}$ : facteur de **branchement**
  - $\mathbf{m}$ : profondeur de l'arbre.
  - Solutions dans differents **profondeurs**
- **Nombre de noeuds** :

$$1 + b^2 + b^3 + \ldots + b^m = \mathcal{O}(b^m)$$



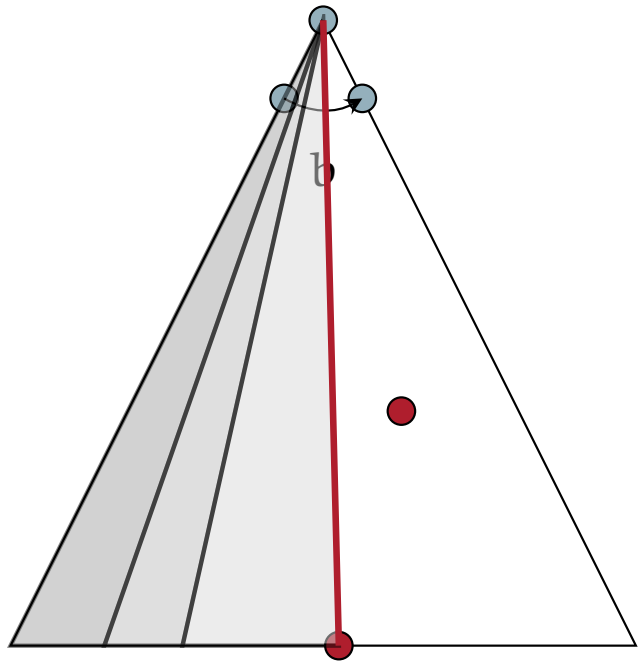
# Propriétés DFS

- **Noeuds visitées par DPS**:
  - Noeuds à **gauche**.
  - Peut visiter **toute** l'arbre
  - Si $m$ est fini, Temps est $\mathcal{O}(b^m)$
- **Taille de la frontière**:
- **Est il complet**?:
  - Au cas où on as pas de **cycle**.
- **Est il optimal**?
  - **No** trouve toujours le nouds à gauche (indépendemment du coût).

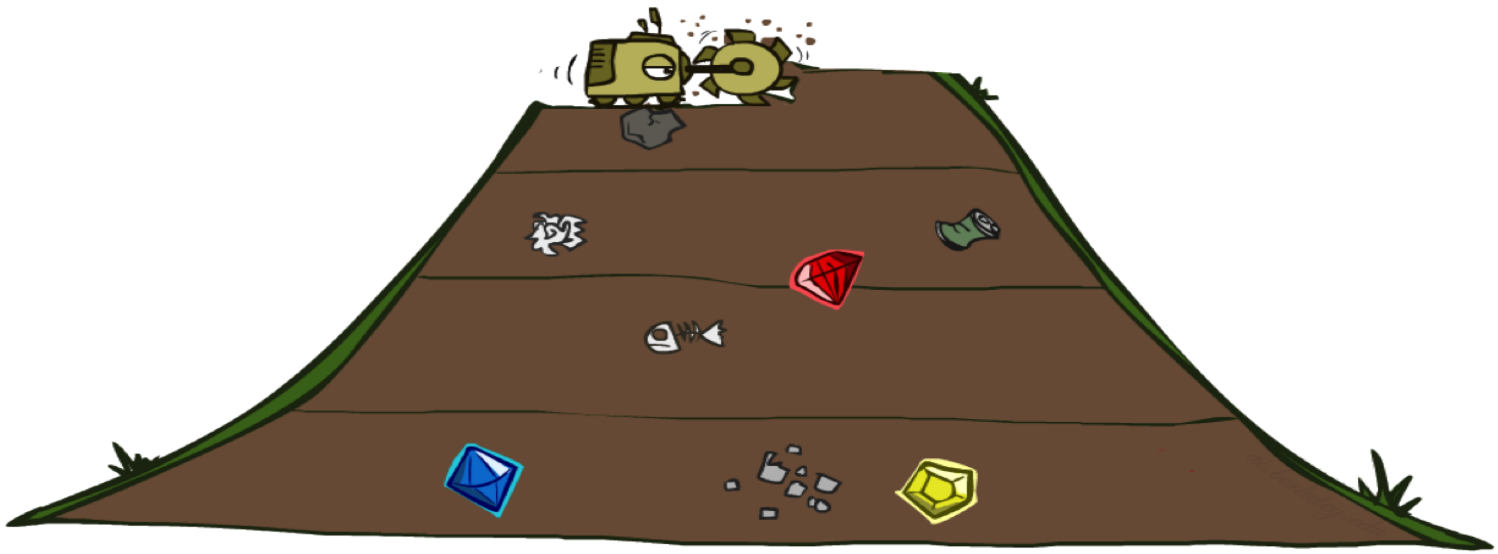


# Propriétés DFS

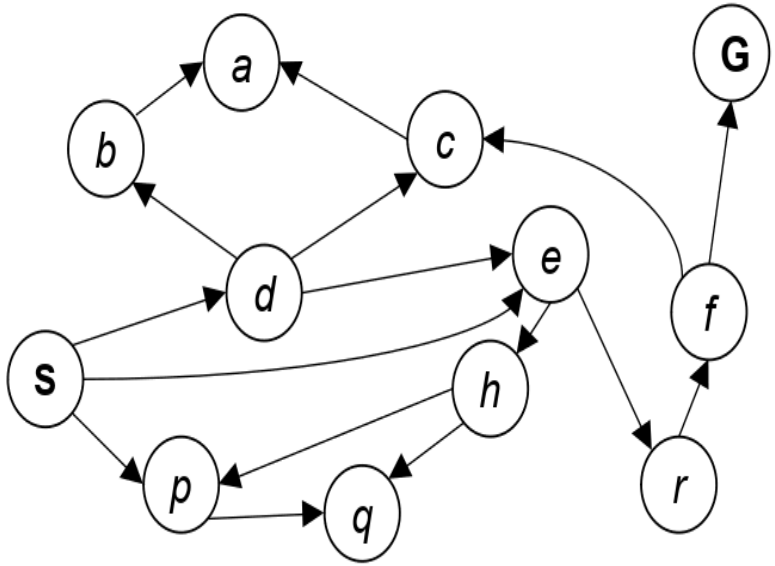
- **Noeuds visitées par DPS**:
  - Noeuds à **gauche**.
  - Peut visiter **toute** l'arbre
  - Si $m$ est fini, Temps est $\mathcal{O}(b^m)$
- **Taille de la frontière**:
  - $\mathcal{O}(bm)$
- **Est il complet**?:
  - Au cas où on as pas de **cycle**.
- **Est il optimal**?
  - **No** trouve toujours le nouds à gauche (indépendemment du coût).



# Breadth First Search (BFS)

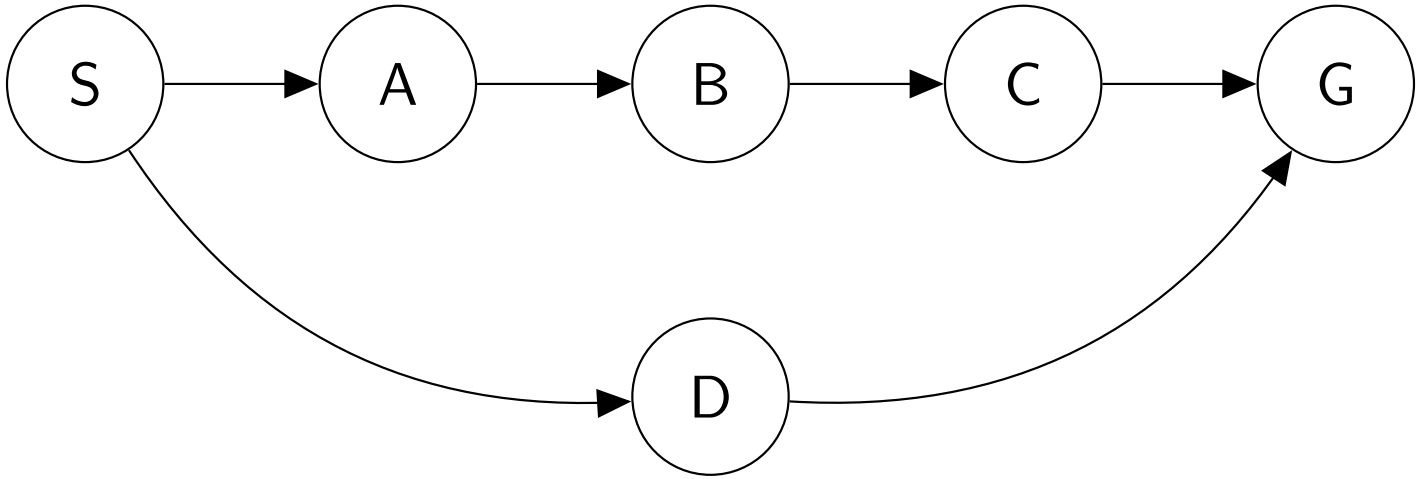


# BFS (Recherche en largeur)




## Propriétés

- **Stratégie**:Developper les noeuds superficiels
- **Frontière** : est une File (FIFO).

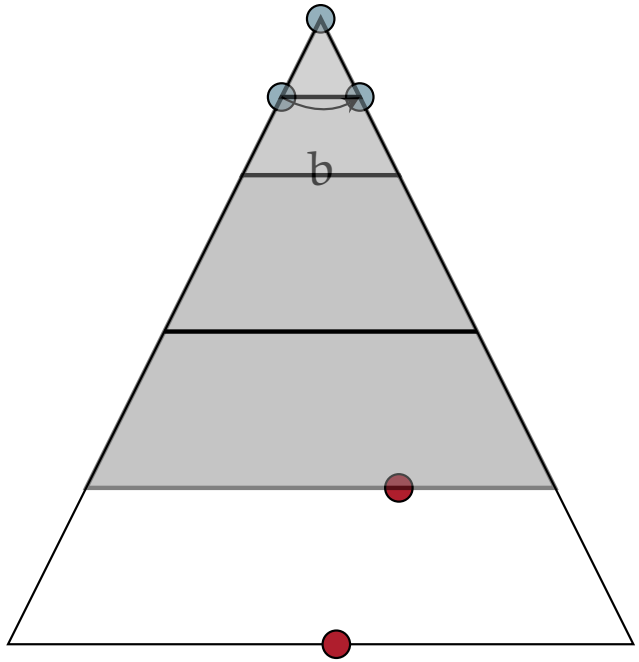



## Question

Donner le chemin calculé par **BPS** de son exploration du point **S** jusqu'au point **G**.

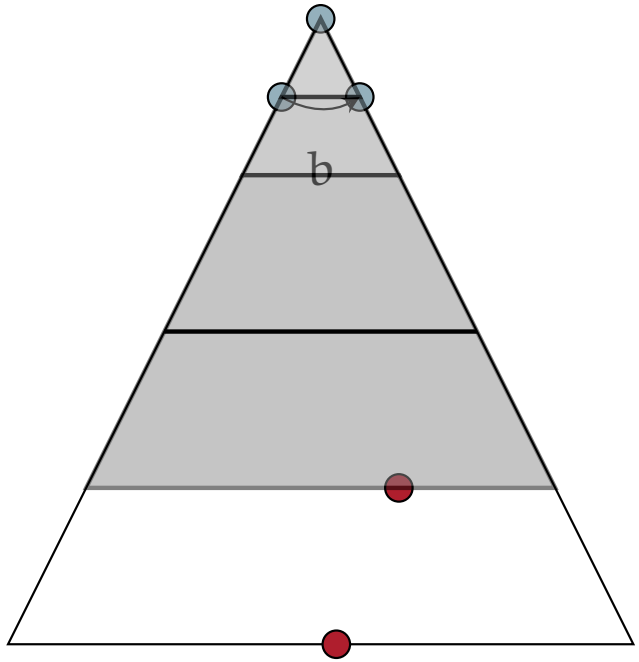
# Propriétés BFS

- **Noeuds visitées par BPS**:
  - Tous les noeuds d'une superficie.
  - Si $\mathfrak{m}$ est fini, Temps est $\mathcal{O}(\mathfrak{b}^s)$
  - $s$ est la pronfondeur de la solution.
- **Taille de la frontière**:
- **Est il complet**?:
  - Si une solution existe, $s$ est fini. Donc Oui.
- **Est il optimal**?
  - **OUI** Si tous les coûts sont $1$.



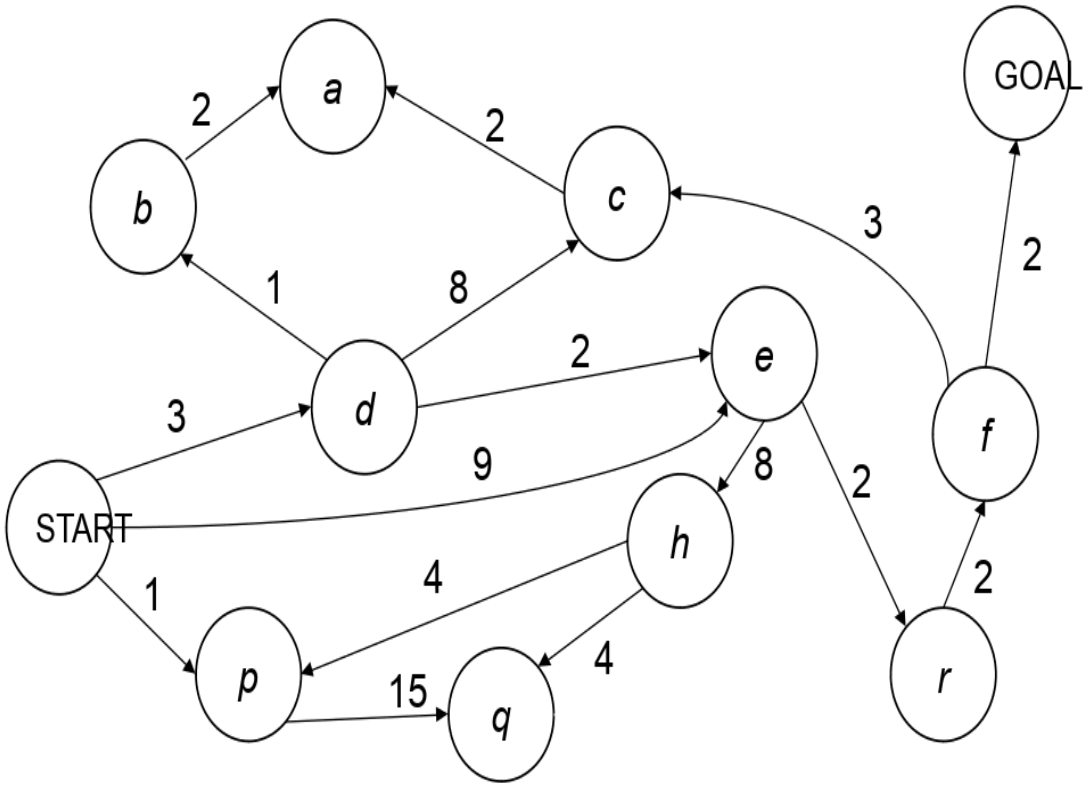
# Propriétés BFS

- **Noeuds visitées par BPS**:
  - Tous les noeuds d'une superficie.
  - Si $m$ est fini, Temps est $\mathcal{O}(b^s)$
  - $s$ est la pronfondeur de la solution.
- **Taille de la frontière**:
  - $\mathcal{O}(b^s)$
- **Est il complet**?:
  - Si une solution existe, $s$ est fini. Donc Oui.
- **Est il optimal**?
  - **OUI** Si tous les coûts sont $1$.



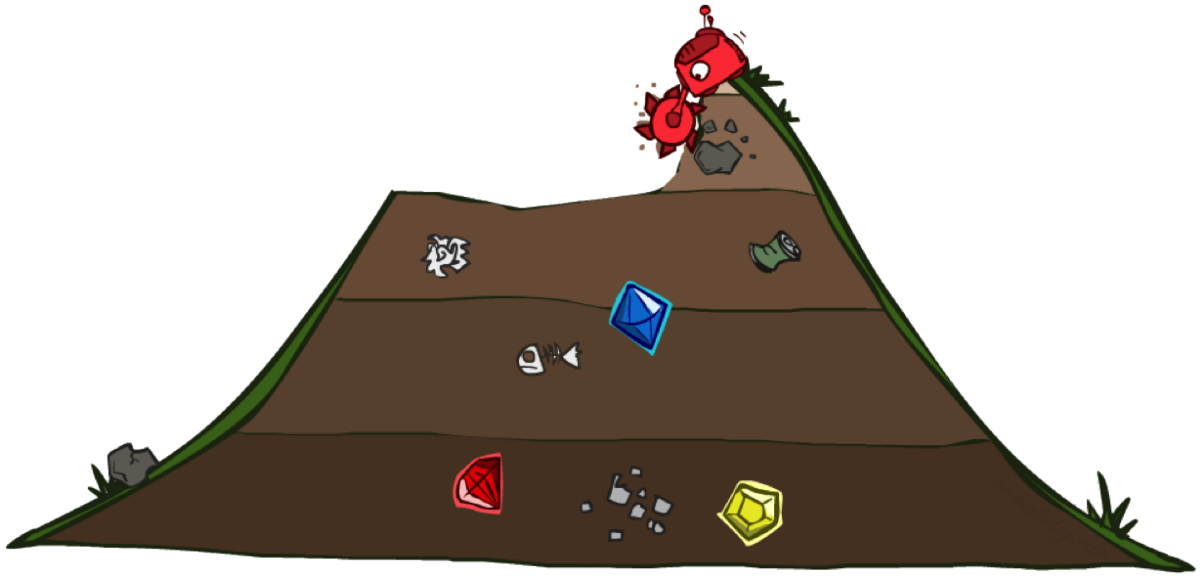
- Quand est ce qu'il est **préferable** d'utiliser **DFS** ?
- Quand est ce qu'il est **préferable** d'utiliser **BFS** ?

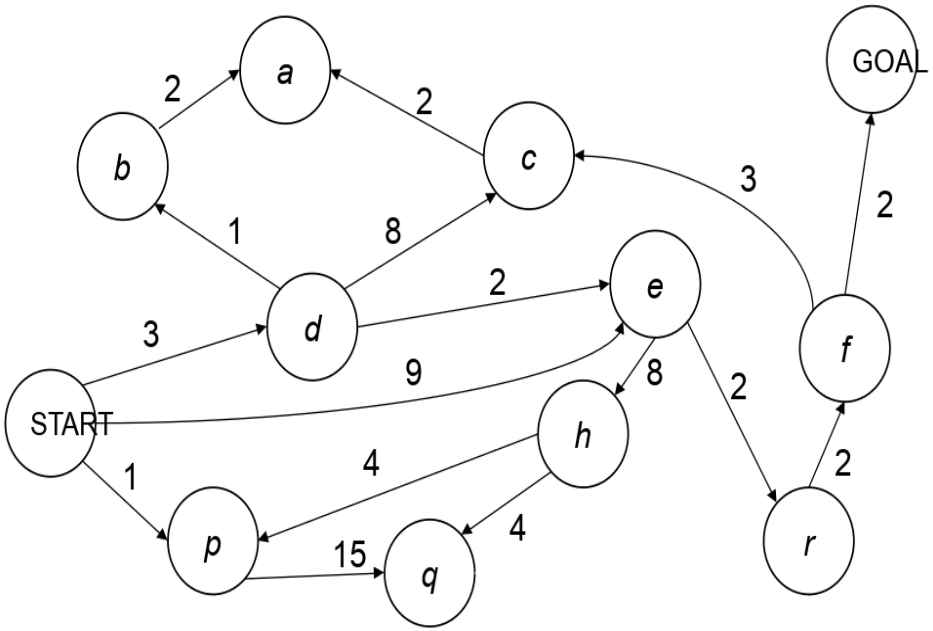
# Recherche avec un coût




BFS trouve le chemin optimal quand le **cout** de toutes les transitions est 1. Nous présentons alors son **analogue** quand on introduit des coûts **différents**.

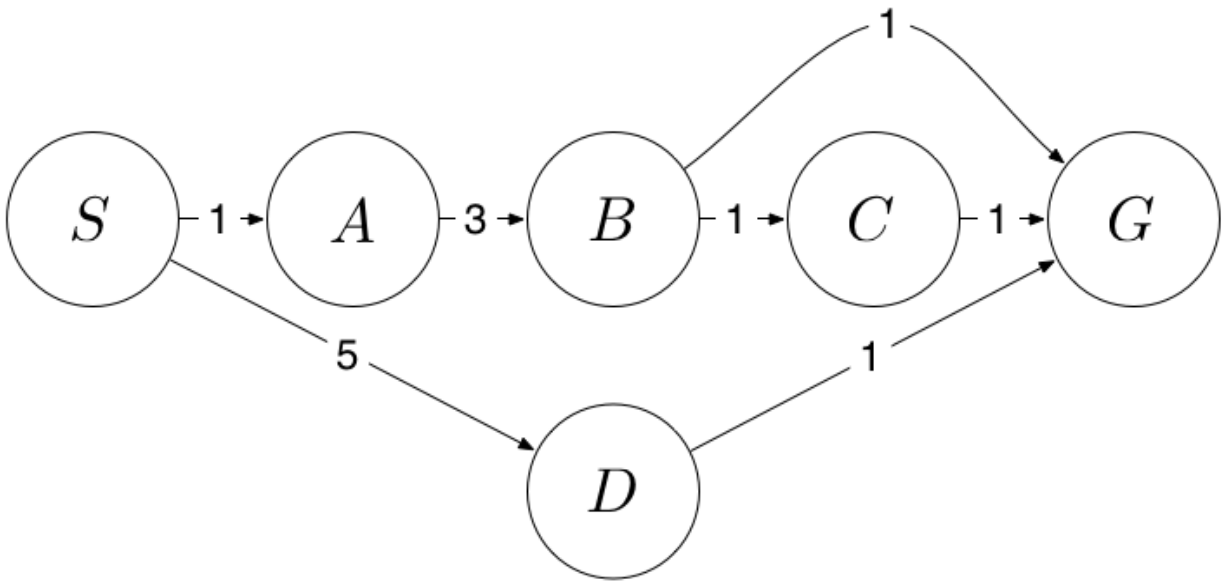
# Recherche uniforme en coût






## Idées

- Developper les noeuds les **Moins couteux**.
- La frontière est une **File d'attente** (priorité: coût **accumulé**).

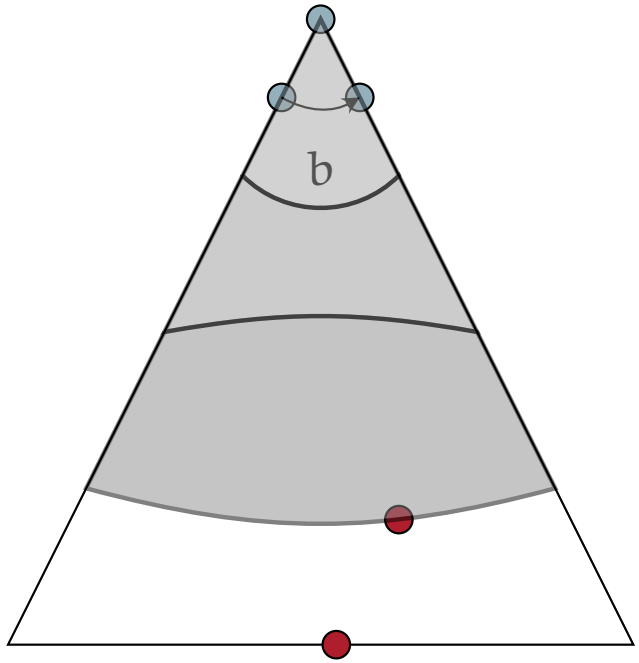



## Question

- Donner le chemin choisi par **BFS**.
- Donner le chemin choisi par **UCS**

# Propriétés UCS

- **Noeuds visitées par UCS**:
  - Tous les noeuds dont le coùt est **inférieur** à celui de la solution $\mathbf{C}^*$.
  - complexité temps est $\mathcal{O}(\mathrm{b}^{\frac{\mathrm{C}^*}{\epsilon}})$.
  - $\epsilon = \min_{\mathrm{c}} \mathrm{C}$.
- **Taille de la frontière**:
- **Est il complet**?:
  - Si une solution existe et tous les arrêtes sont **positifs** , Oui.
- **Est il optimal**?
  - **OUI** Preuve prochaine lecture ($\mathcal{A}^*$).



# Propriétés UCS

- **Noeuds visitées par UCS**:
  - Tous les noeuds dont le coùt est **inférieur** à celui de la solution $\mathbf{C}^*$.
  - complexité temps est $\mathcal{O}(\mathrm{b}^{\frac{\mathrm{C}^*}{\epsilon}})$.
  - $\epsilon = \min_{\mathrm{c}} \mathrm{C}$.
- **Taille de la frontière**:
  - $\mathcal{O}(\mathrm{b}^{\frac{\mathrm{C}^*}{\epsilon}})$
- **Est il complet**?:
  - Si une solution existe et tous les arrêtes sont **positifs** , Oui.
- **Est il optimal**?
  - **OUI** Preuve prochaine lecture $(\mathcal{A}^*)$.