Plus court chemin

Un *plus court chemin* entre deux sommets u et v est un chemin simple de longueur minimale.

Propriété

Un plus court chemin entre 2 sommets est élémentaire.

Cette propriété nous donne un premier algorithme pour trouver un plus court chemin.

Plus court chemin

Algo1:

1. Le nombre de chemins élémentaires entre s et t étant finis, nous pouvons tous les énumérer en calculant leur longueur et retenir le plus court.

En supposant que trouver un chemin et calculer sa longueur puissent se faire en une seule opération, sur un ordinateur pouvant effectuer 1 milliard d'opérations par seconde:

1 avec un graphe ayant 20 sommets il nous faudra patienter 77 ans pour obtenir le meilleur trajet;

2 si le graphe a 25 sommets, il faut plus de 490 millions d'années.

Principe de sous-optimalité

Si $p = (s, \ldots, t)$ est un plus court chemin entre s et t, alors pour tout sommet x sur le chemin p,

1. le sous-chemin de p jusqu'à x, $(s, \ldots, x)$, est un plus court chemin de s à x
2. le sous-chemin de p depuis x, $(x, \ldots, t)$, est un plus court chemin de x à t

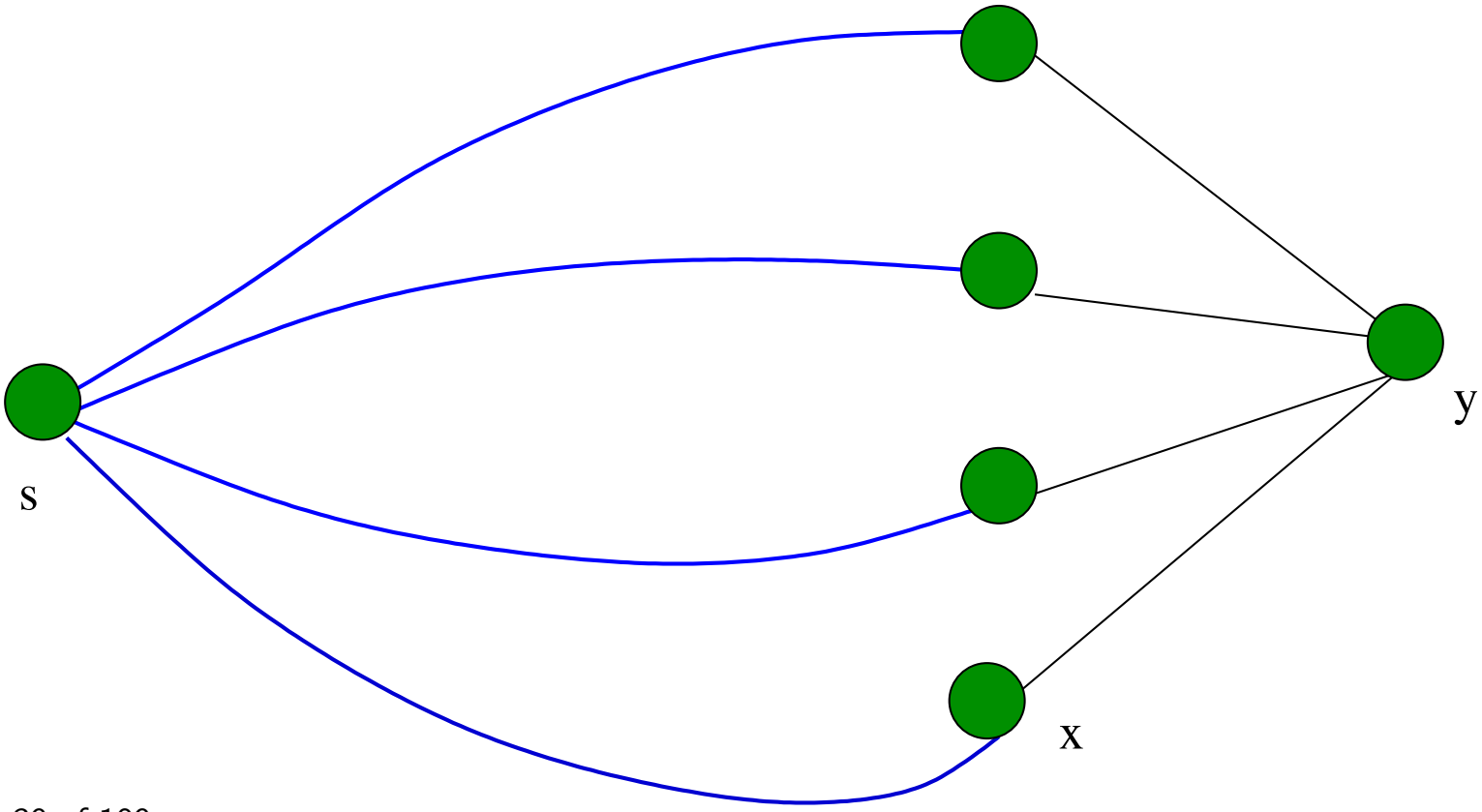
Le plus court chemin, p, entre s et un sommet y visite nécessairement, juste avant d'arriver à y, l'un de ses voisins, z : $p = (s, ..., z, y)$.

1. Si on note $D(y)$ la distance entre s et y alors on a $D(y) = D(z) + 1$.
2. A priori l'identification du voisins z est impossible. Mais on sait que $\forall x \in V(y)$, on a $D(x) + 1 \geqslant D(y)$.

Propriété

La longueur $D(y)$ du plus court chemin de s à y vérifie

1. $D(y) = 0$ si $y = s$
2. $D(y) = 1 +$ min $\{D(x) | x$ *voisin de* $y\}$ sinon



Propriété

Cette propriété semble nous donner une formule récursive pour calculer $D(x)$.

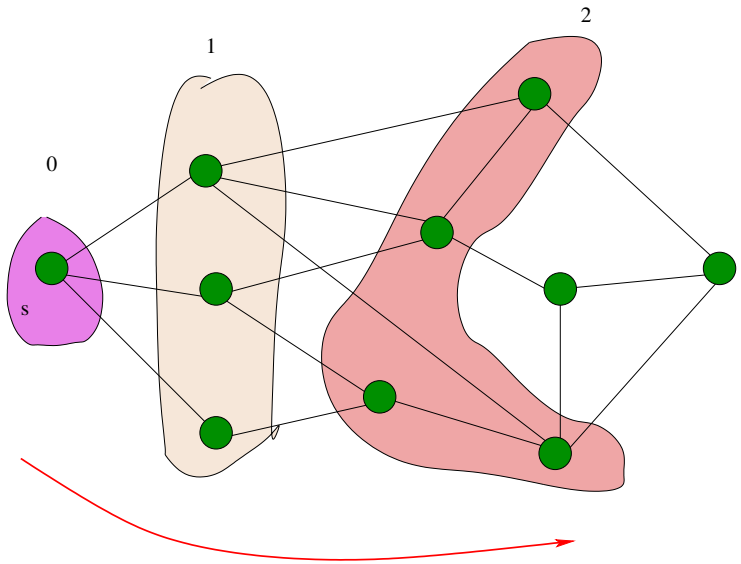
1. Malheureusement: le calcul de $D(x)$ demande le calcul de $D(y)$ qui dépend lui-même de $D(x)$...
2. Pour éviter de tourner éternellement en rond, une intuition consiste à visiter le graphe de proche en proche tout en calculant les distances $D(y)$ et;
3. en ne tenant compte que des voisins x déjà visités (leur distance $D(x)$ est donc déjà connue).

Principe

1. Partir de s (distance 0), et de visiter successivement ses voisins (distance 1),
2. puis les voisins de ses voisins non encore visités (distance 2),
3. les voisins des voisins des voisins non encore visités, et
4. ainsi de suite...

Principe

1. Partir de s (distance 0), et de visiter successivement ses voisins (distance 1),
2. puis les voisins de ses voisins non encore visités (distance 2),
3. les voisins des voisins des voisins non encore visités, et
4. ainsi de suite...



En d'autre termes

1. Au départ tous les sommets du graphe sont non marqués, à l'exception de notre sommet de départ, s.
2. A chaque étape on sélectionne un sommet non marqué, adjacent à un sommet déjà marqué, et nous le marquons à son tour.

En d'autre termes

1. Au départ tous les sommets du graphe sont non marqués, à l'exception de notre sommet de départ, s.
2. A chaque étape on sélectionne un sommet non marqué, adjacent à un sommet déjà marqué, et nous le marquons à son tour.

Le choix d'un sommet à marquer voisin d'un sommet déjà marqué assure que l'ensemble des sommets marqués est un sous-graphe connexe à chaque étape de l'algorithme.

Algorithme de recherche (Graphe $G = (V, E)$, Sommet s)

1. Initialiser tous les sommets à non marqué ; Marquer s
2. TantQue il existe un sommet non marqué adjacent à un sommet marqué
   2.1 Sélectionner un sommet y non marqué adjacent à un sommet marqué Marquer y
3. FinTantQue

Algorithme de recherche (Graphe $G = (V, E)$, Sommet s)

1. Initialiser tous les sommets à non marqué ; Marquer s
2. TantQue il existe un sommet non marqué adjacent à un sommet marqué
   2.1 Sélectionner un sommet y non marqué adjacent à un sommet marqué Marquer y
3. FinTantQue

Il existe deux grandes stratégies "opposées" pour sélectionner à chaque étape le sommet à marquer:

1. La recherche en profondeur DFS
2. La recherche en largeur BFS

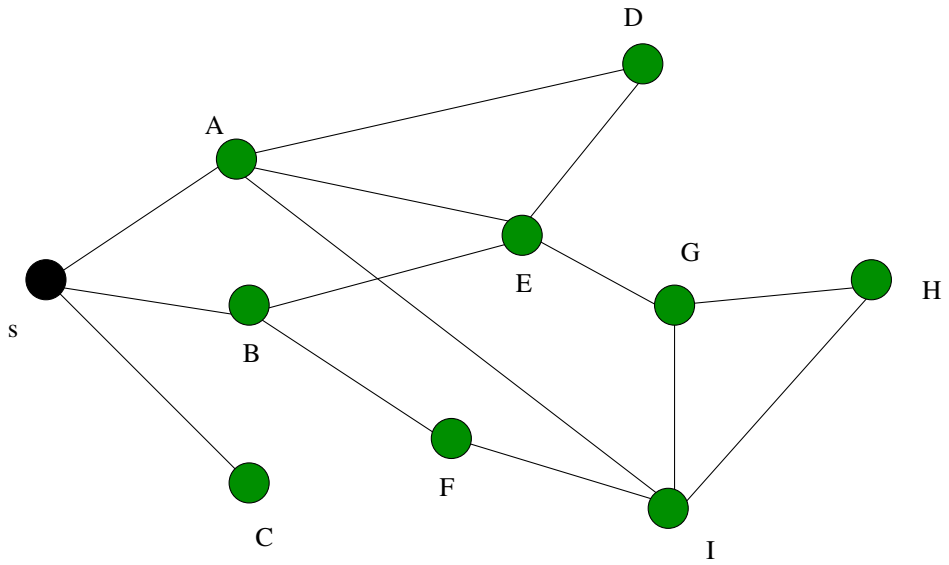
La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

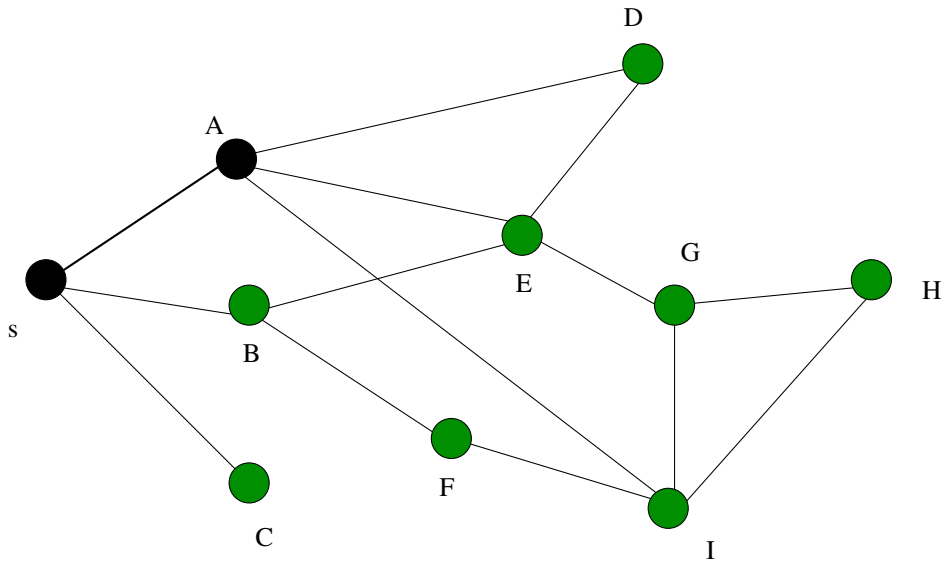
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

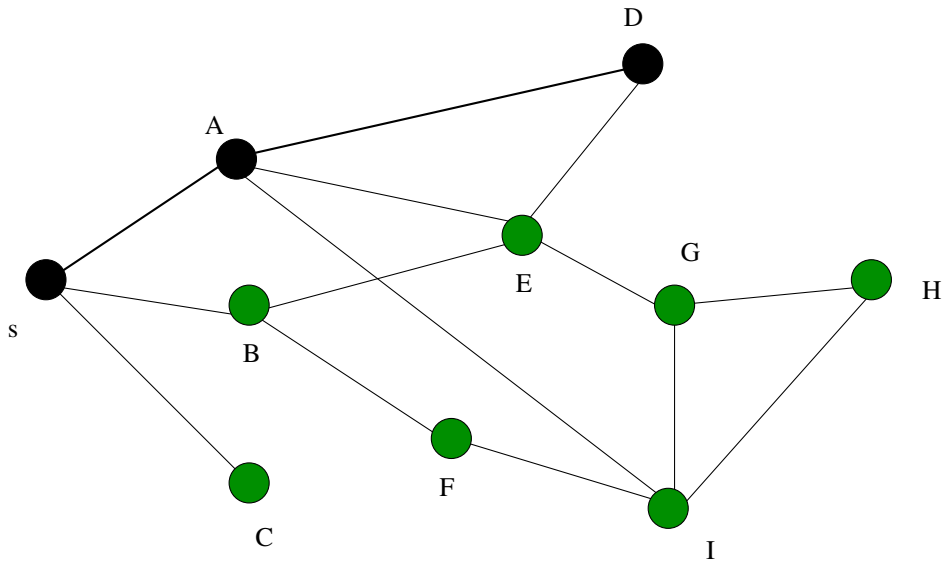
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

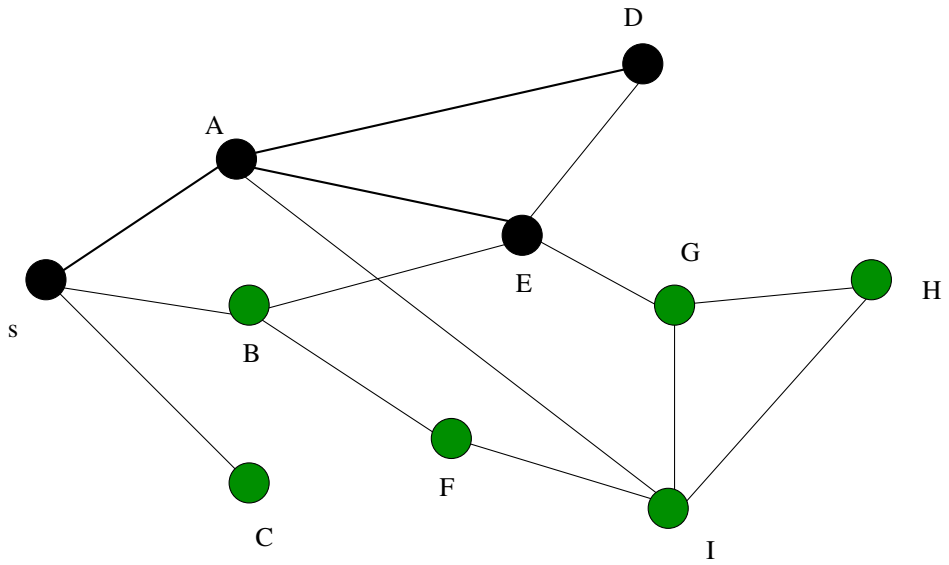
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

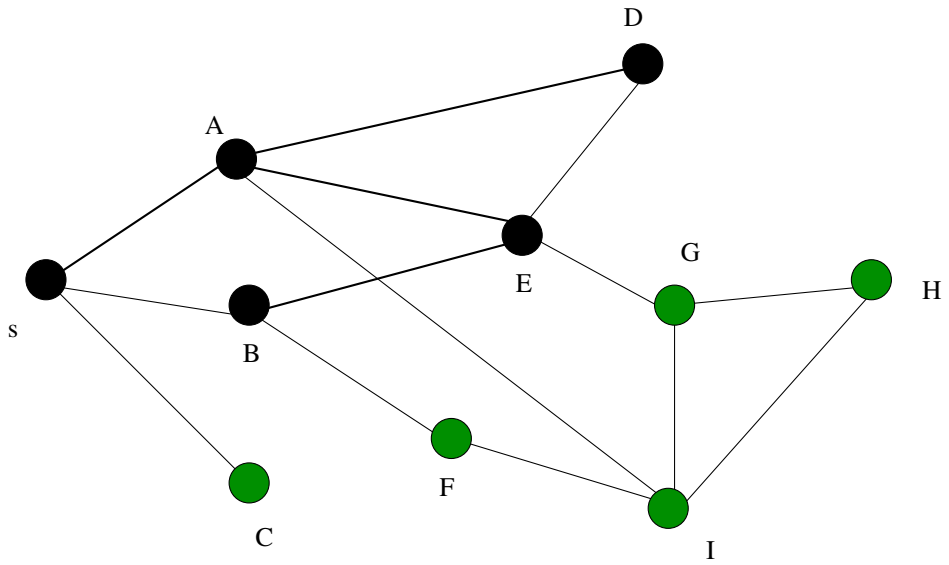
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

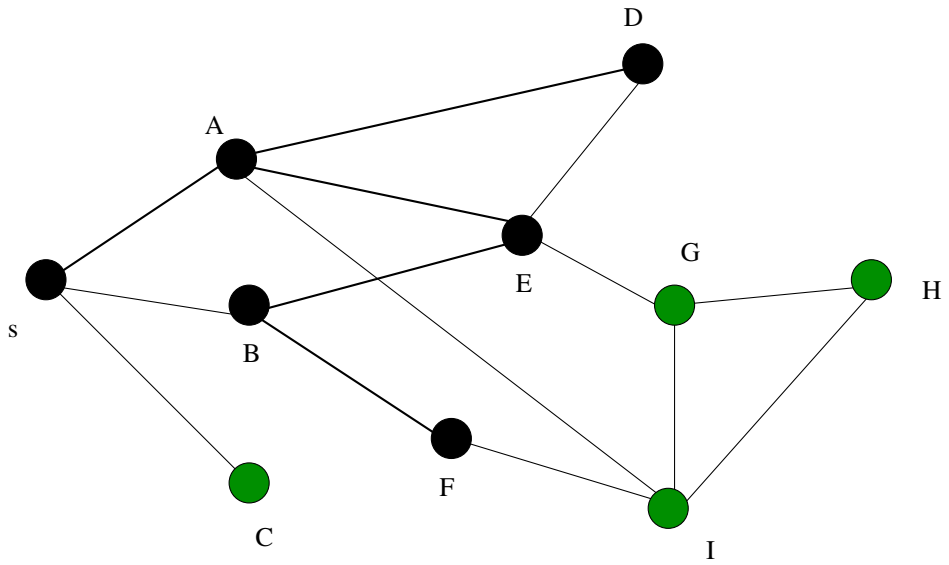
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

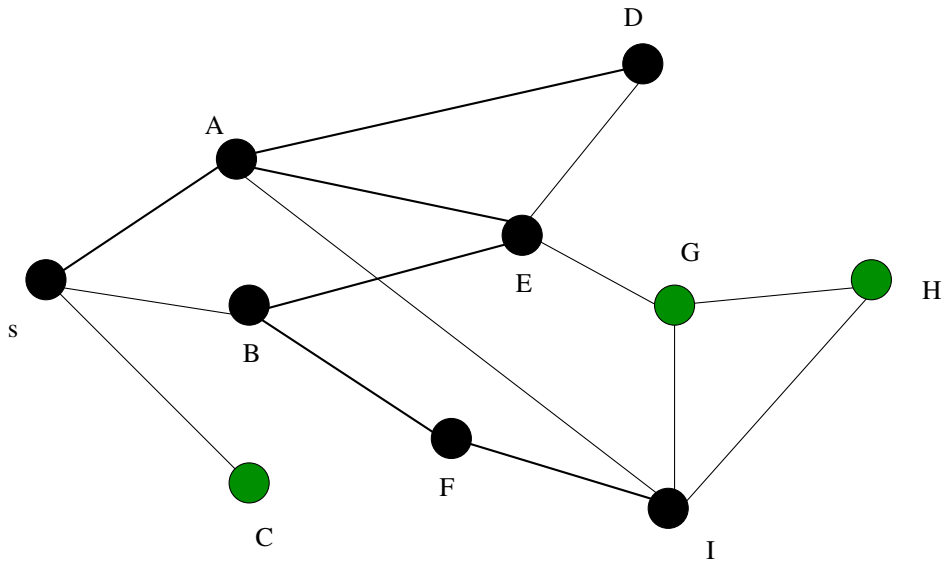
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

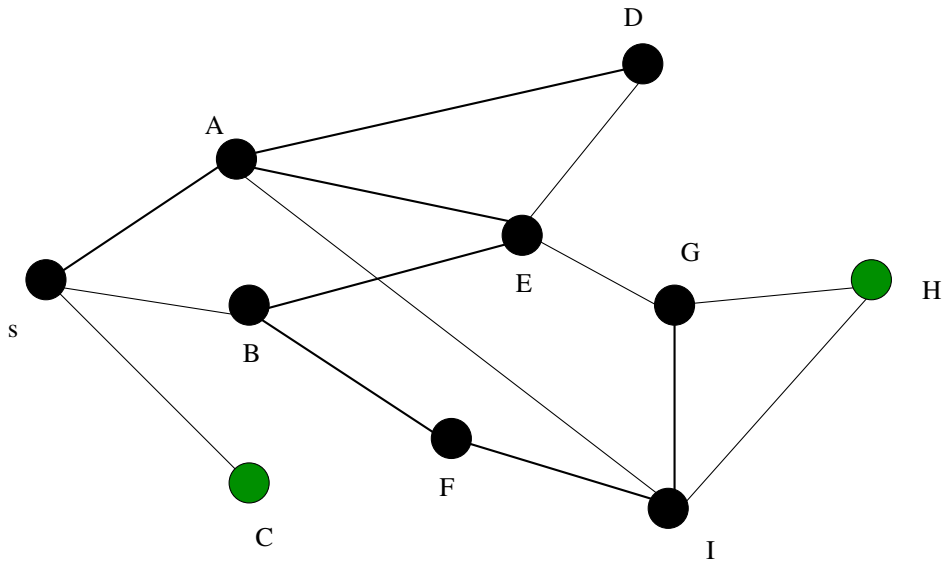
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

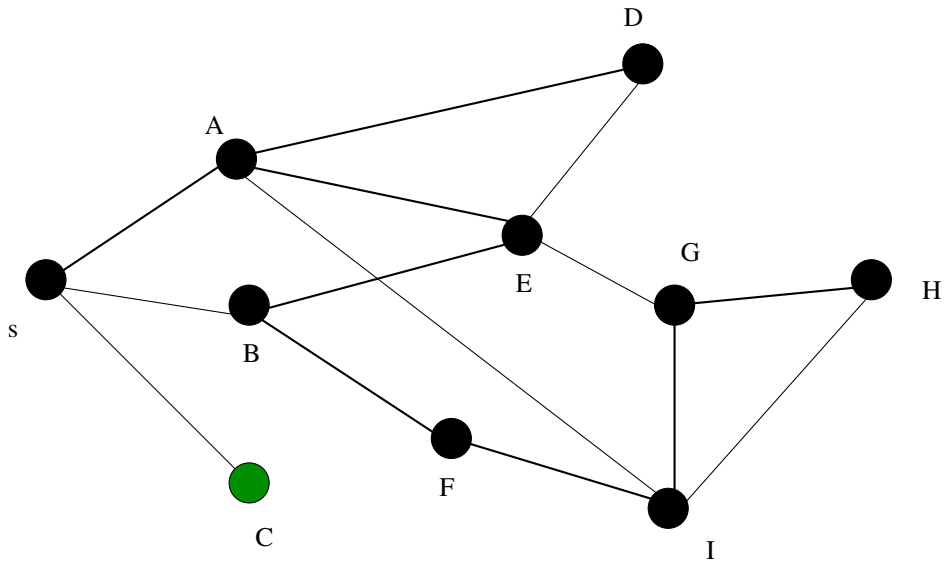
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

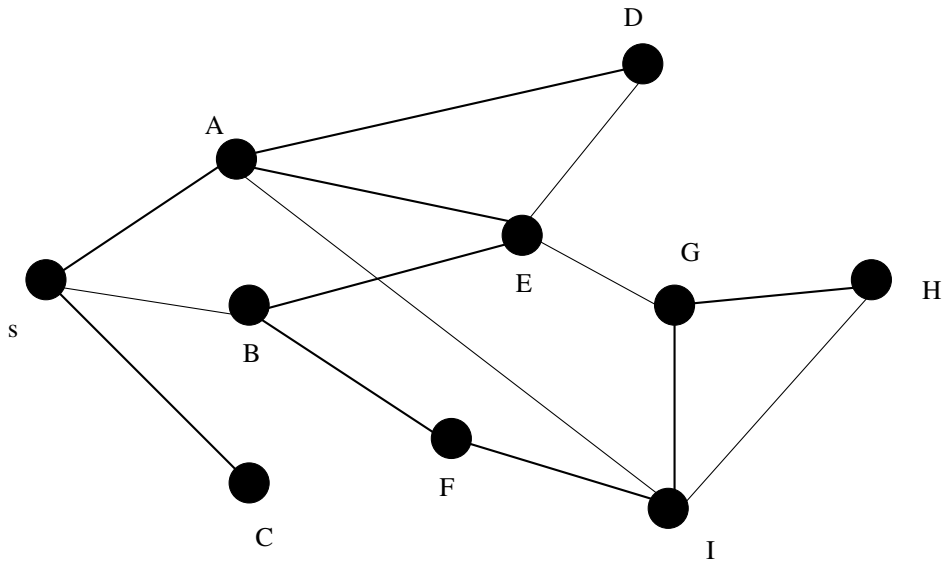
Exemple: DFS



La recherche en profondeur DFS (Depth First Search)

1. L'algorithme cherche à aller très vite profondément dans le graphe, en s'éloignant du sommet s de départ.
2. La recherche sélectionne à chaque étape (2.1) un sommet voisin du sommet marqué à l'étape précédente.

Exemple: DFS



La recherche en largeur BFS (Breath First Search)

1. Dans l'exploration l'algorithme cherche à épuiser la liste des sommets proches de s avant de poursuivre l'exploration du graphe.