Approximation de l'arborescence de Steiner

Master MPRO, Cours RORT

Dimitri Watel[1]

[1]CNAM, ENSIIE

Mardi 1er mars 2016

Table des matières

Table des matières

Un problème en Terre du Milieu

Un problème en Terre du Milieu

Un problème en Terre du Milieu

Un problème en Terre du Milieu

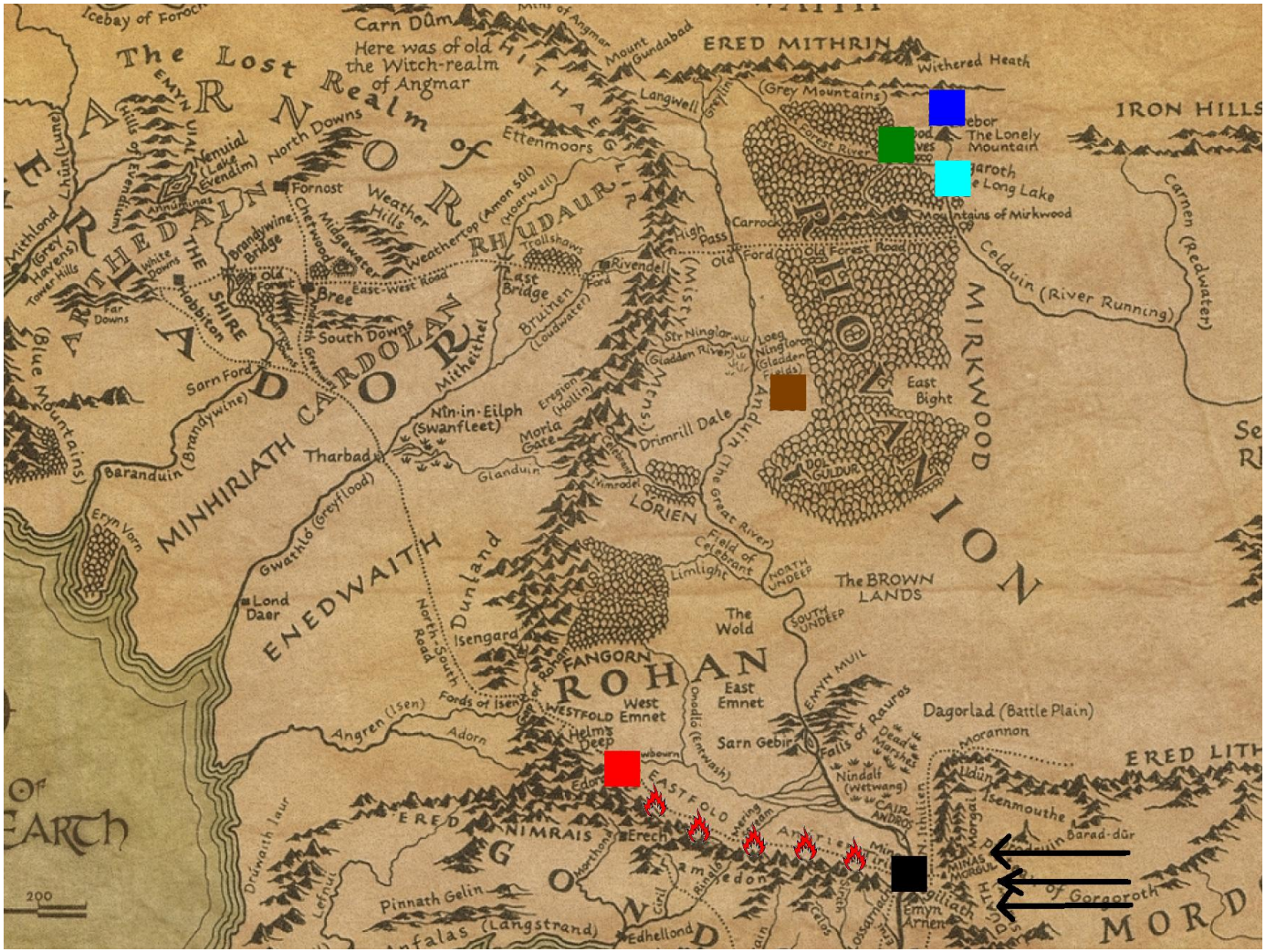
Un problème en Terre du Milieu

Un problème en Terre du Milieu

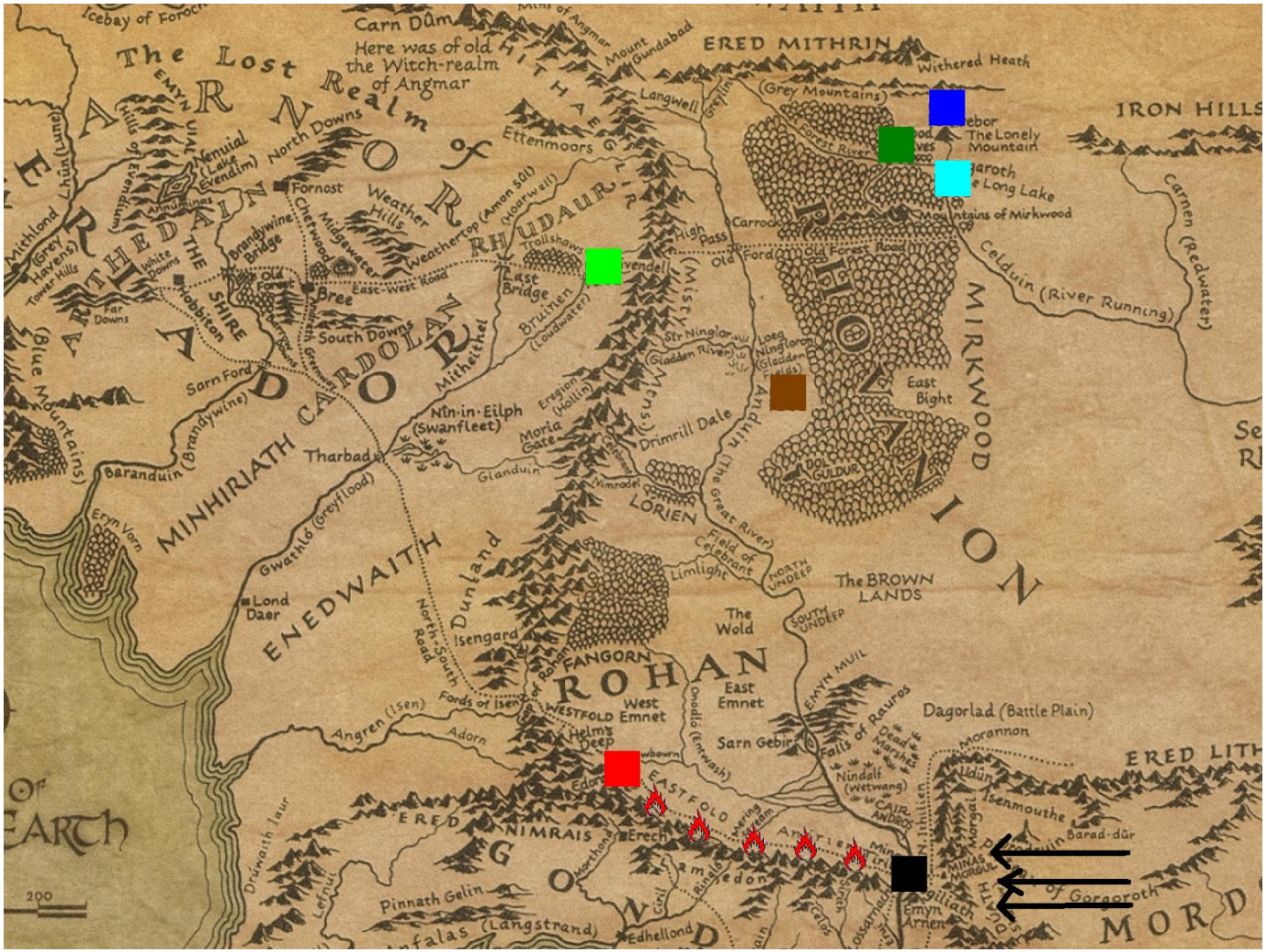
Un problème en Terre du Milieu

Un problème en Terre du Milieu

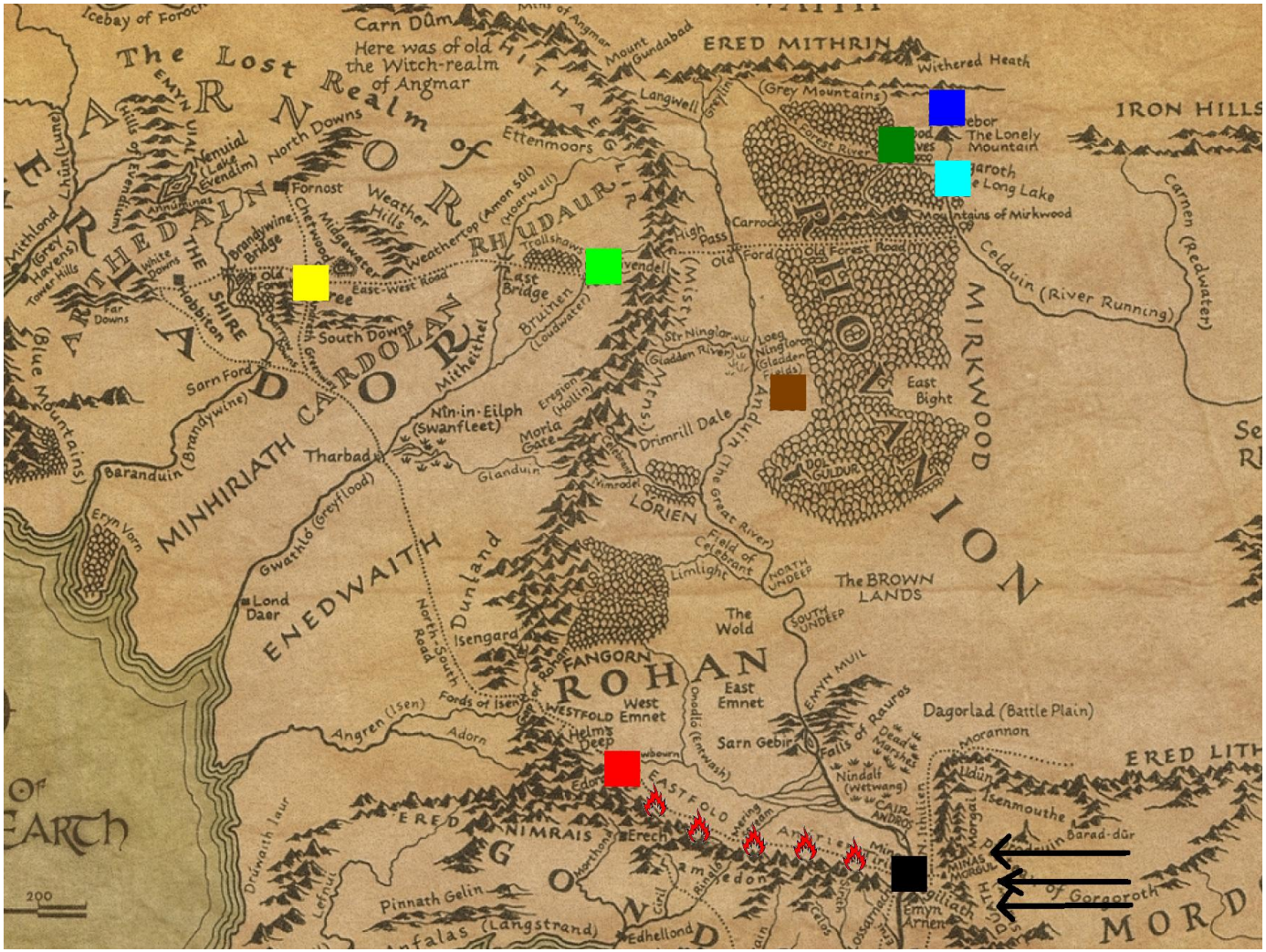
Un problème en Terre du Milieu



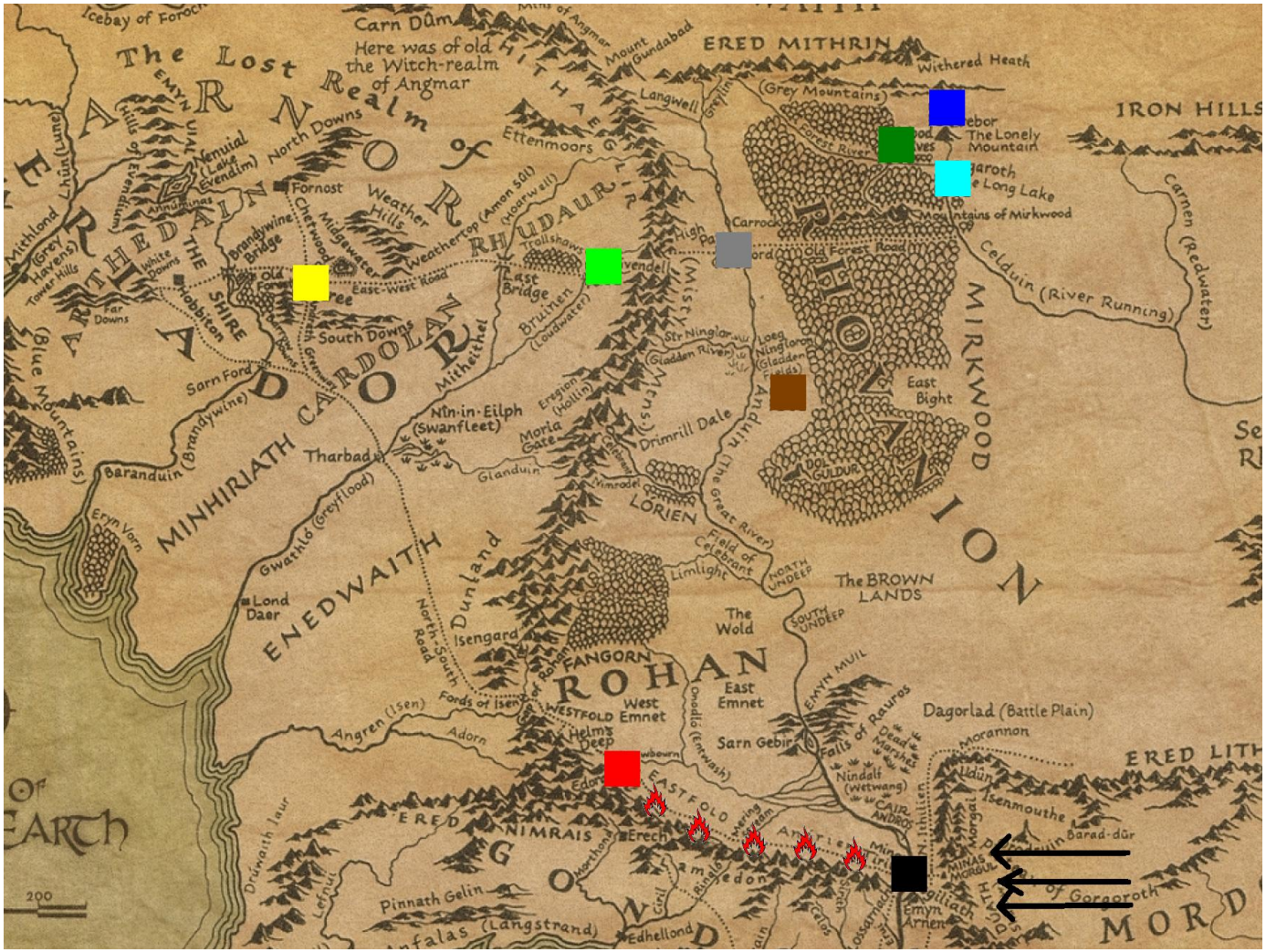
Un problème en Terre du Milieu



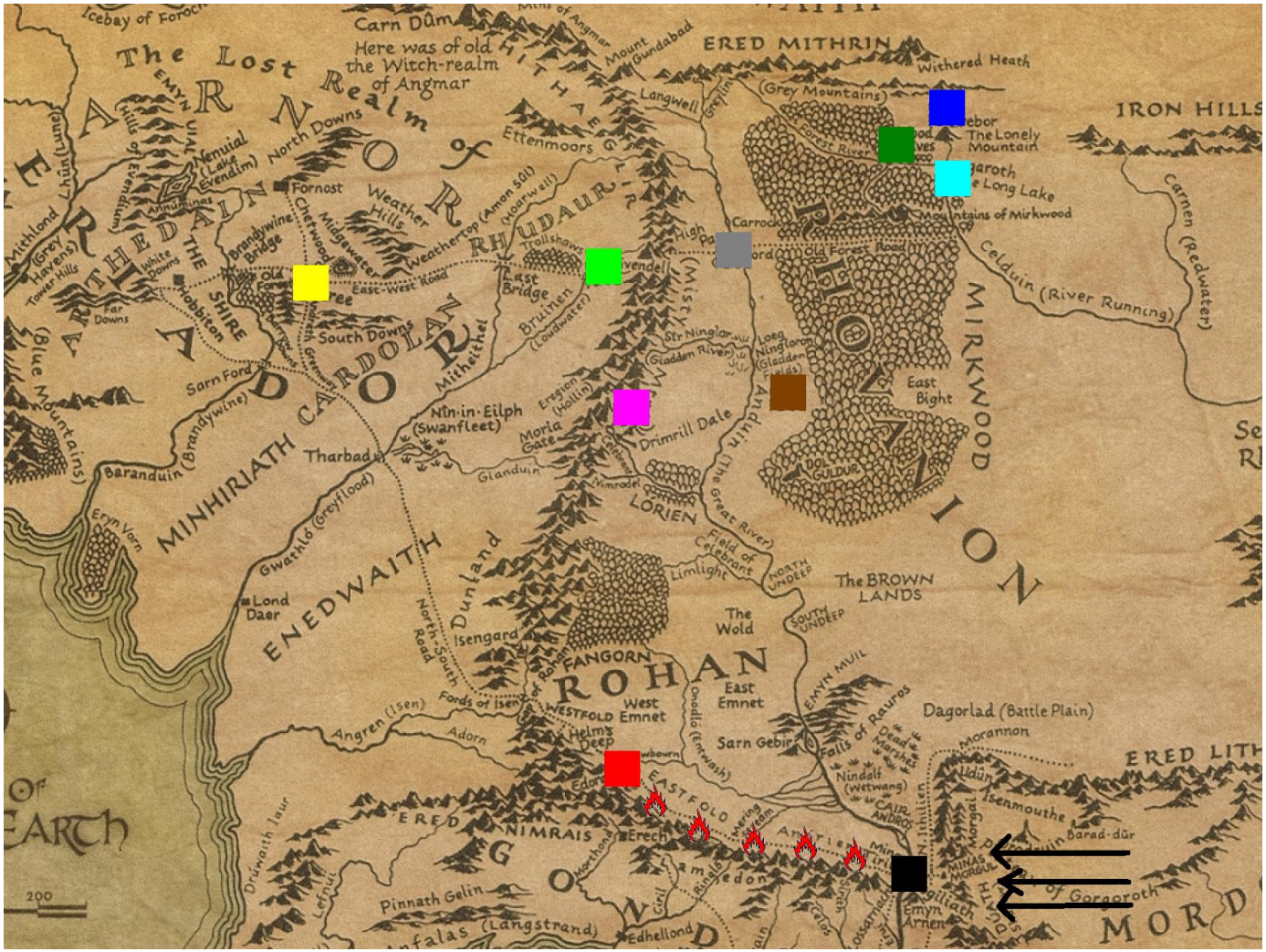
Un problème en Terre du Milieu



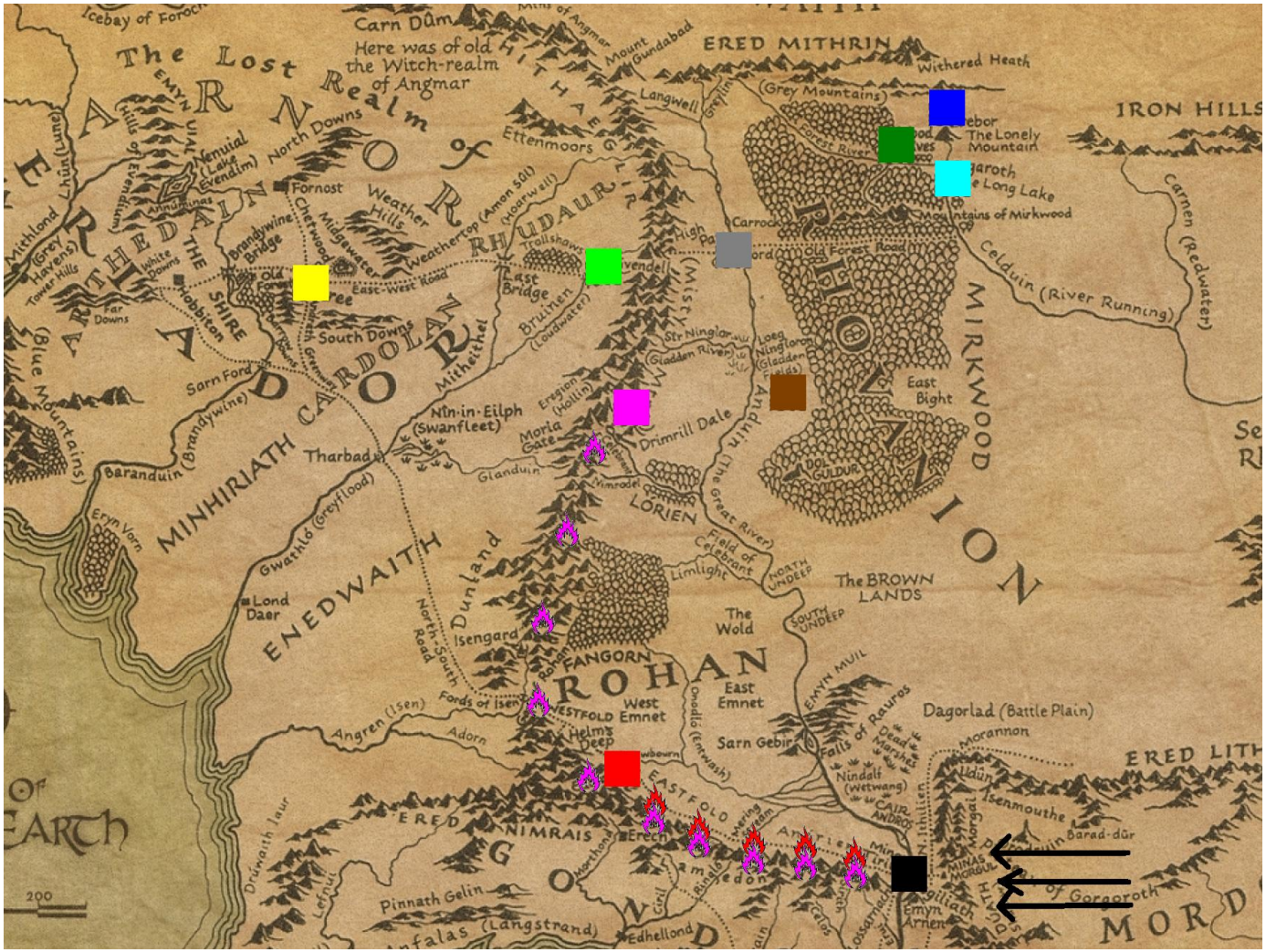
Un problème en Terre du Milieu



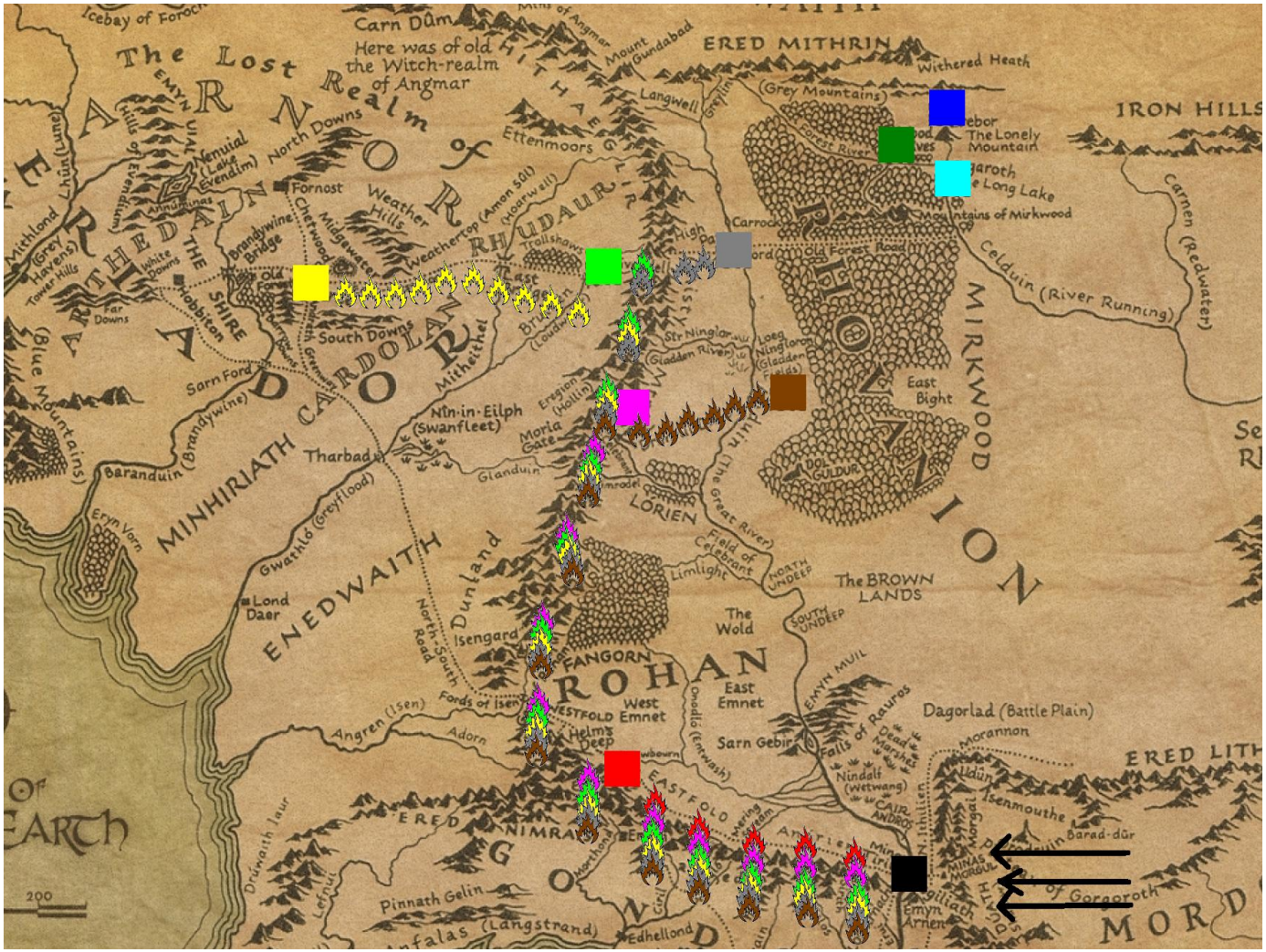
Un problème en Terre du Milieu



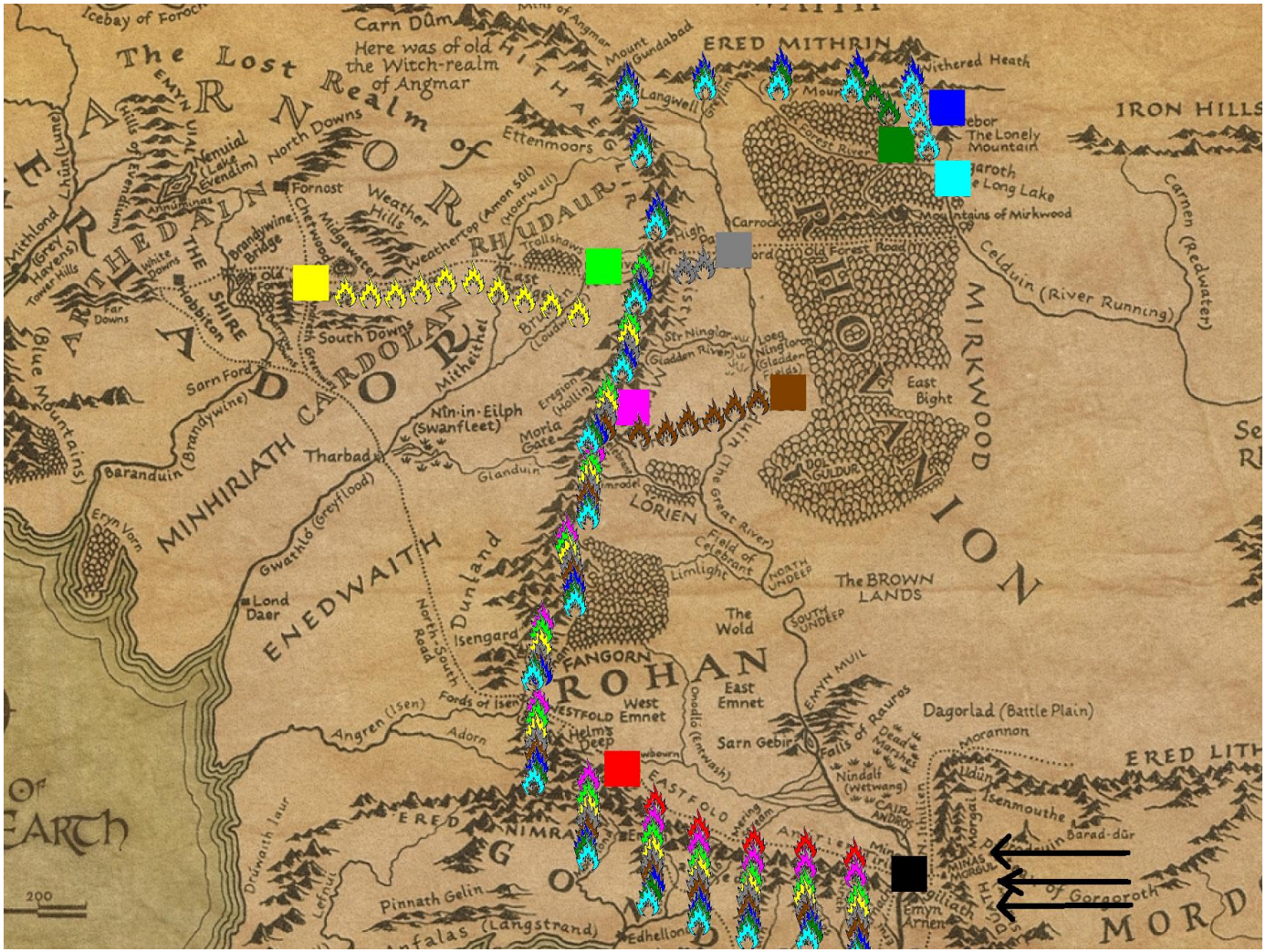
Un problème en Terre du Milieu



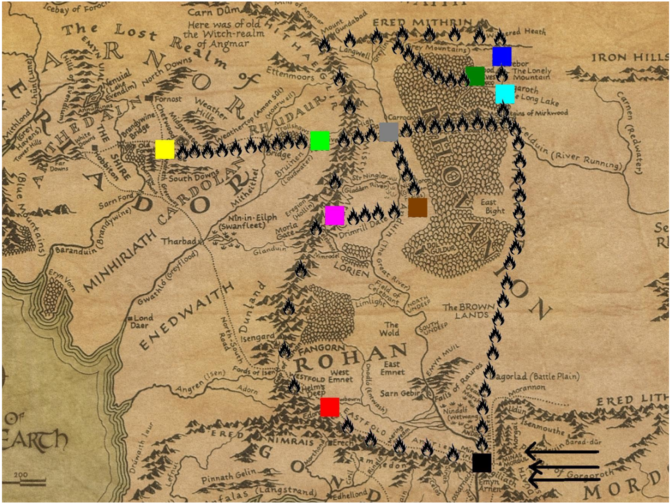
Un problème en Terre du Milieu



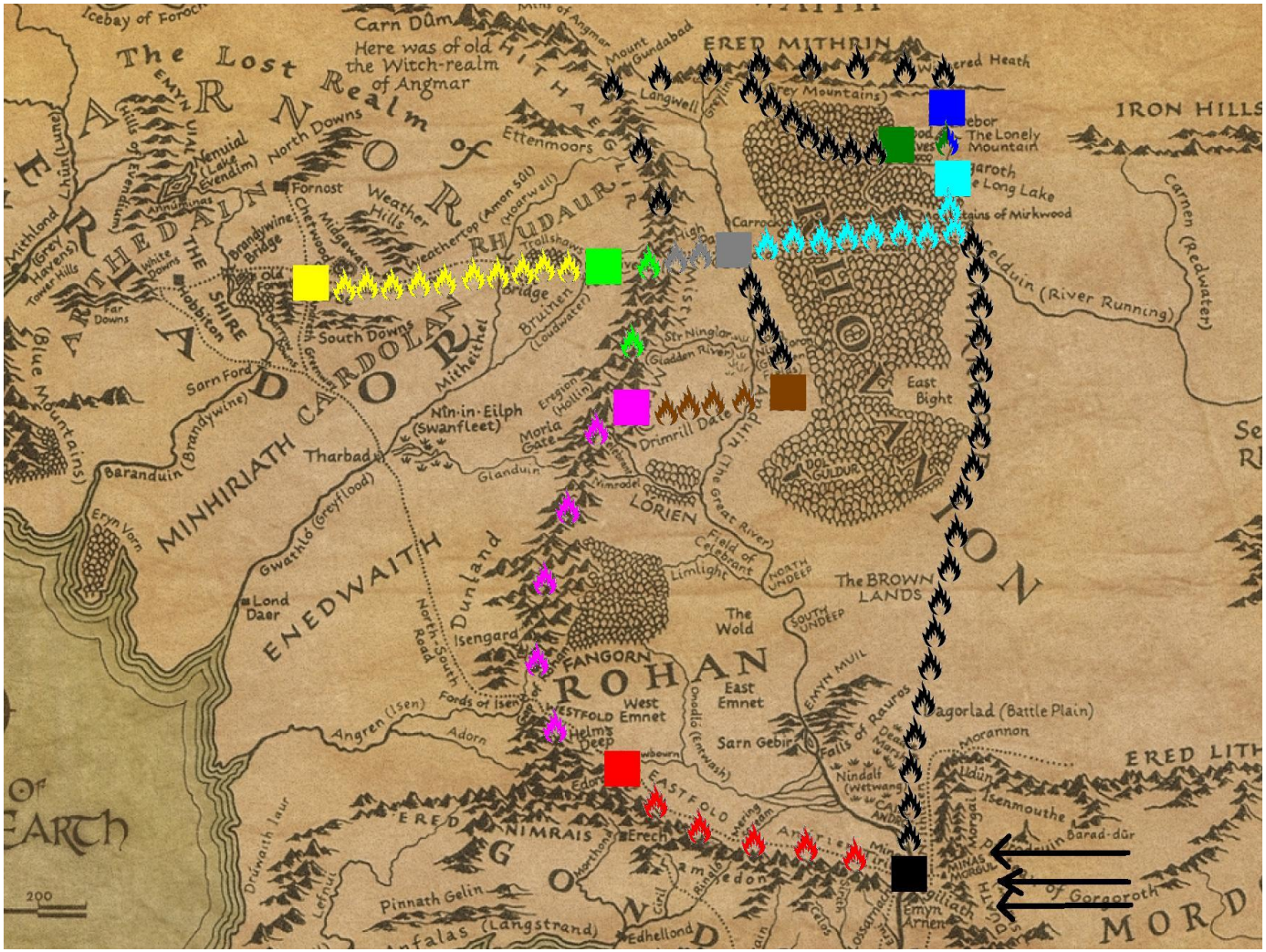
Un problème en Terre du Milieu



Un problème en Terre du Milieu



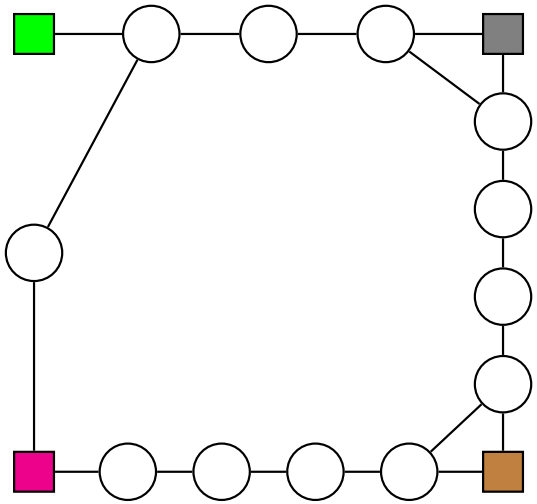
Un problème en Terre du Milieu



Problème de graphe associé

- Ville $\leftrightarrow$ Noeud
- Feux potentiel $\leftrightarrow$ Noeud
- Deux nœuds qui "se voient" $\leftrightarrow$ arête

On recherche dans ce graphe un arbre qui relie tous les nœuds de type "Ville".



Problème de graphe associé

- Ville $\leftrightarrow$ Noeud
- Feux potentiel $\leftrightarrow$ Noeud
- Deux nœuds qui "se voient" $\leftrightarrow$ arête

On recherche dans ce graphe un arbre qui relie tous les nœuds de type "Ville".

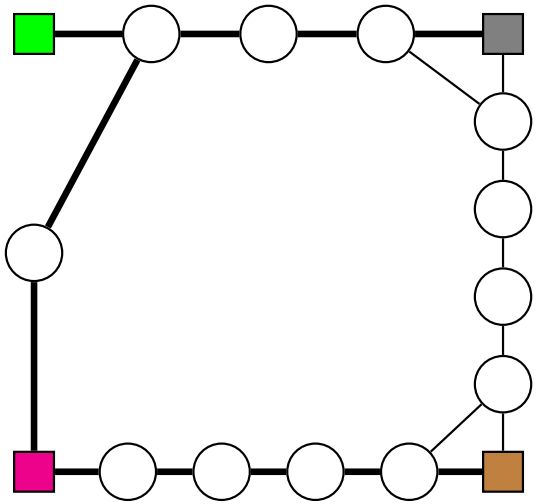
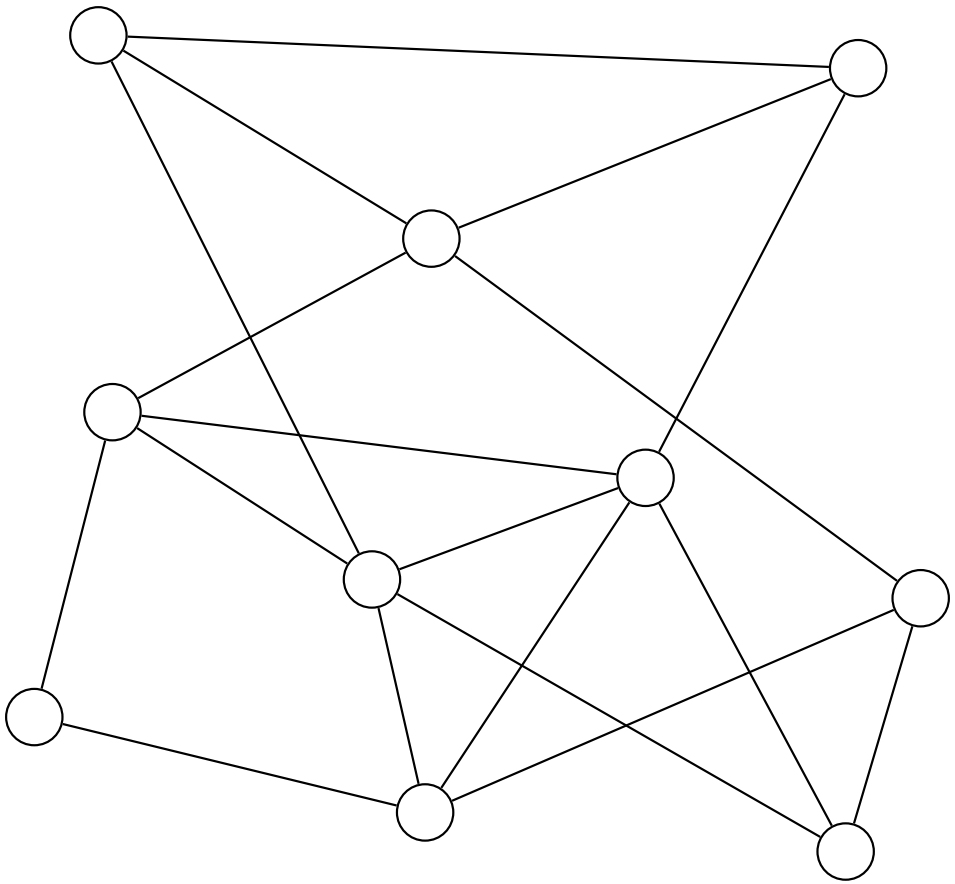


Table des matières

Problème de l'arbre de Steiner (UST)



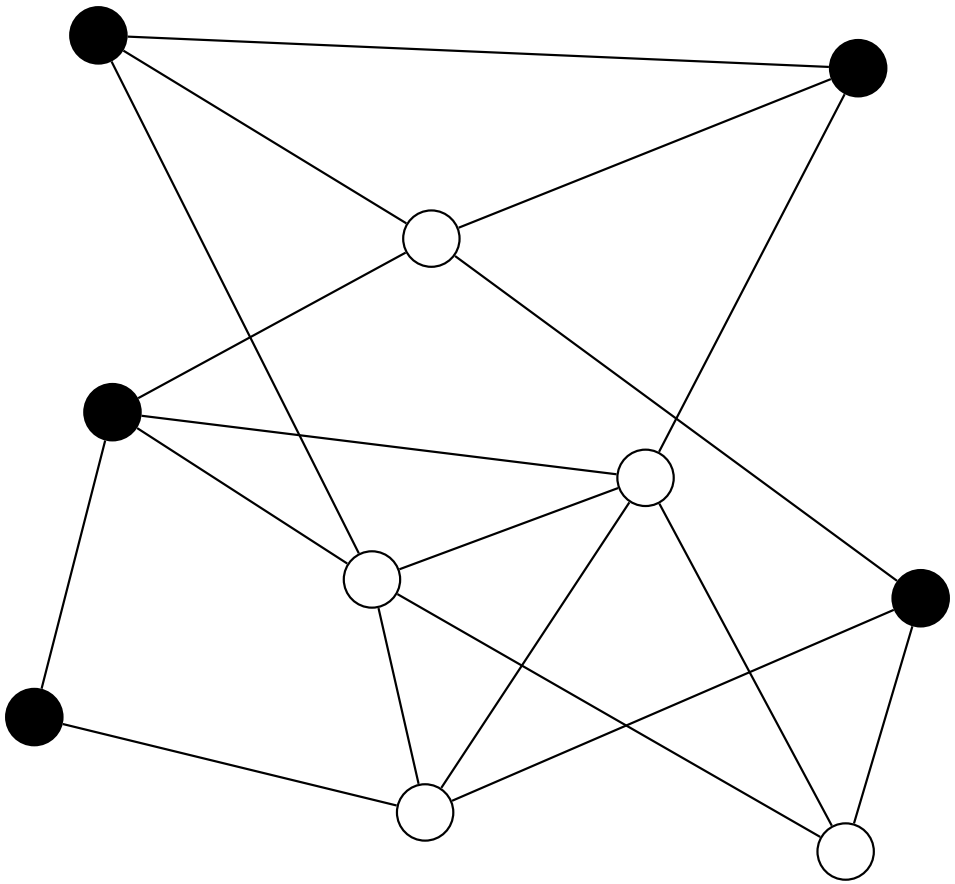
Instance de UST

- $G = (V, E)$;
- k terminaux $X \subset V$.

Solution optimale de UST

- Arbre T couvrant X ;
- Minimisant $|T|$.

Problème de l'arbre de Steiner (UST)



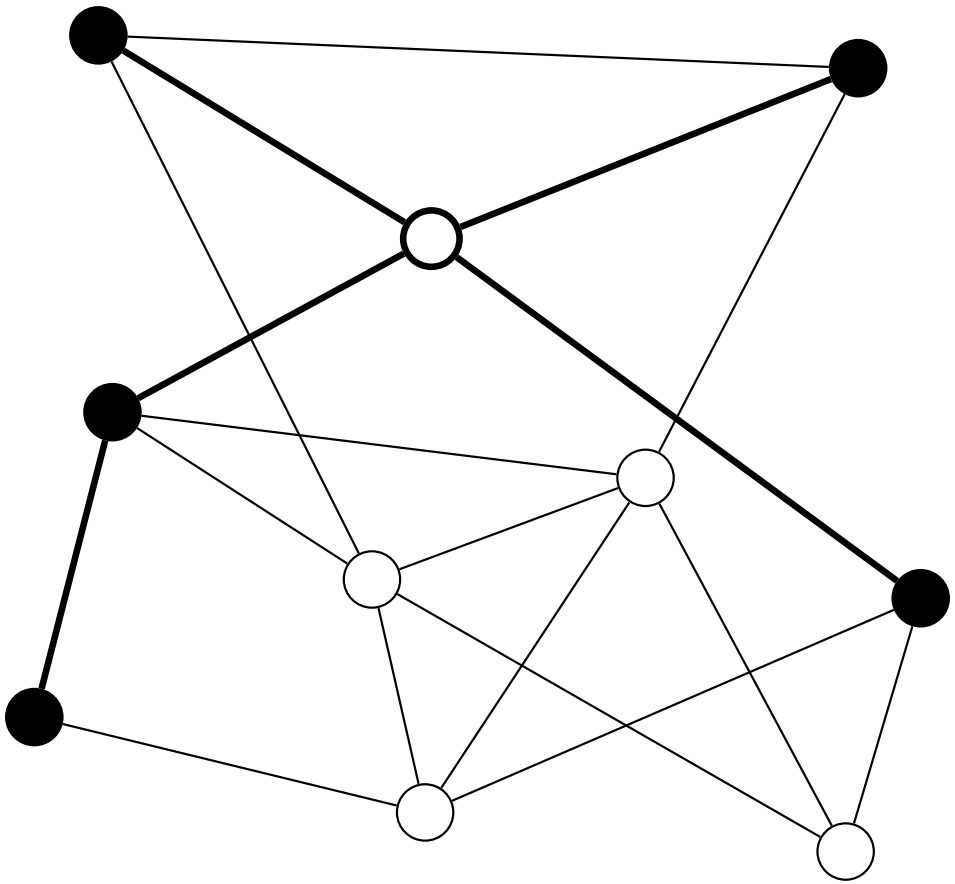
Instance de UST

- $G = (V, E)$;
- k terminaux $X \subset V$.

Solution optimale de UST

- Arbre T couvrant X ;
- Minimisant $|T|$.

Problème de l'arbre de Steiner (UST)



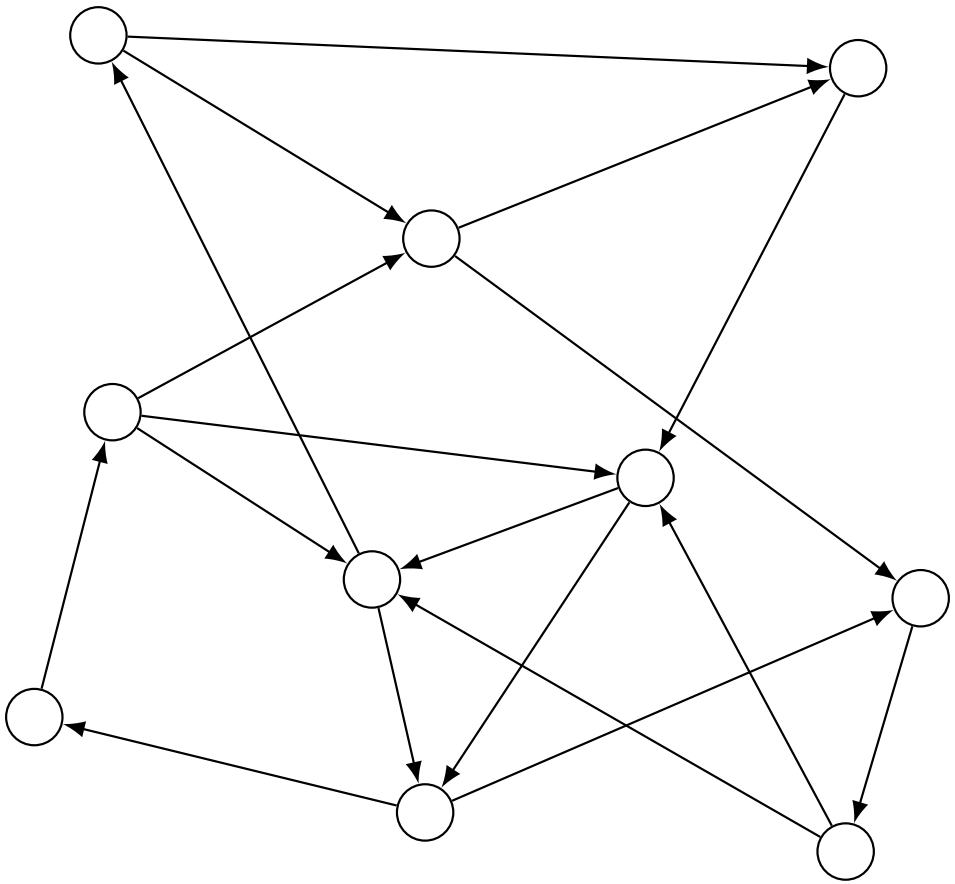
Instance de UST

- $G = (V, E)$;
- k terminaux $X \subset V$.

Solution optimale de UST

- Arbre T couvrant X ;
- Minimisant $|T|$.

Problème de l'arborescence de Steiner (DST)



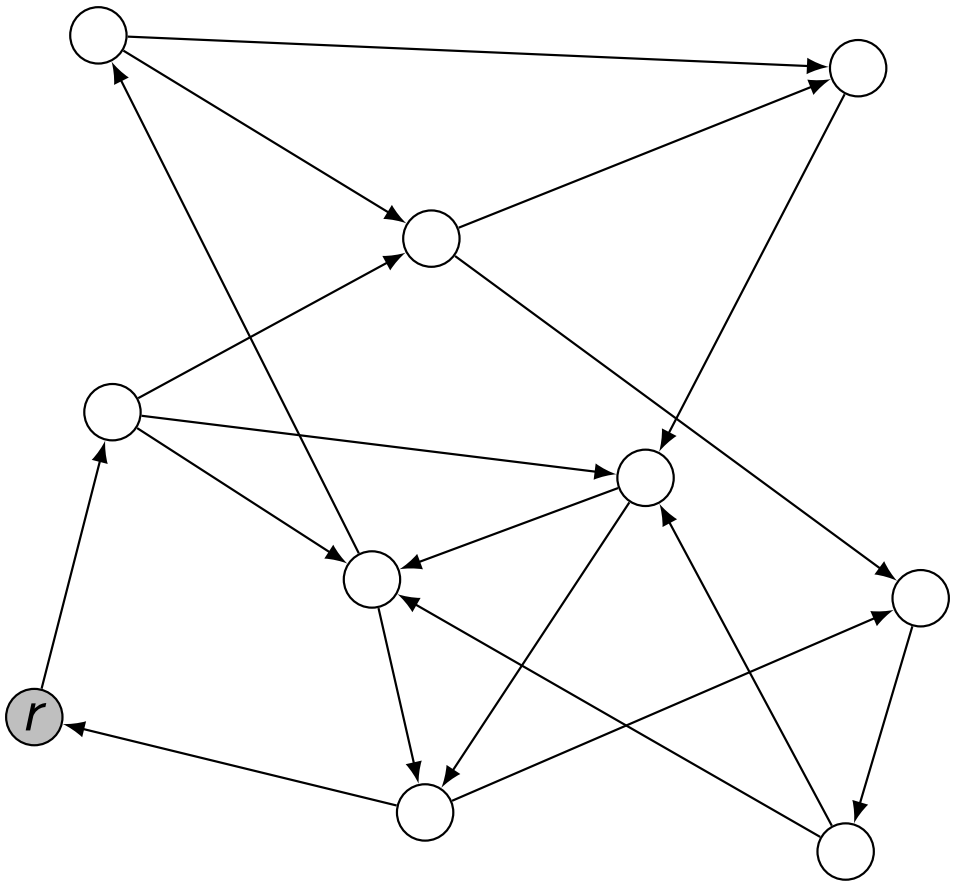
Instance de DST

- $G = (V, A)$;
- Racine $r \in V$;
- k terminaux $X \subset V$;

Solution optimale de DST

- Arborescence T couvrant X ;
- Enracinée en r ;
- Minimisant $|T|$.

Problème de l'arborescence de Steiner (DST)



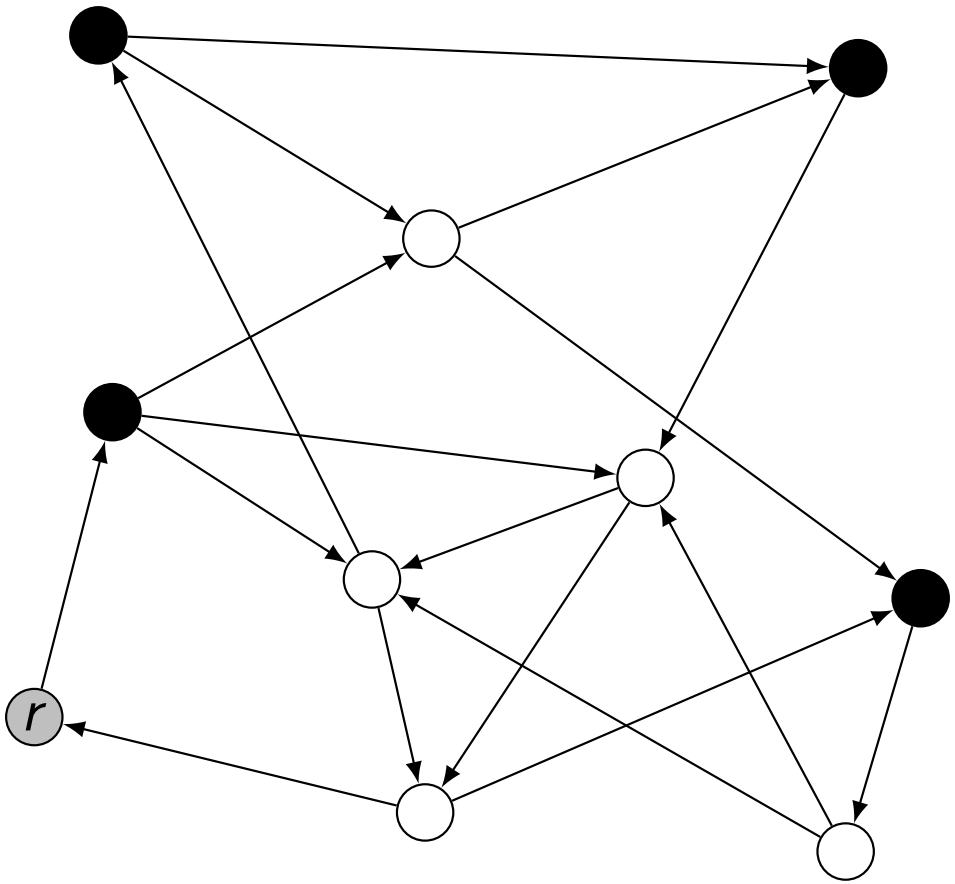
Instance de DST

- $G = (V, A)$;
- Racine $r \in V$;
- k terminaux $X \subset V$;

Solution optimale de DST

- Arborescence T couvrant X ;
- Enracinée en r ;
- Minimisant $|T|$.

Problème de l'arborescence de Steiner (DST)



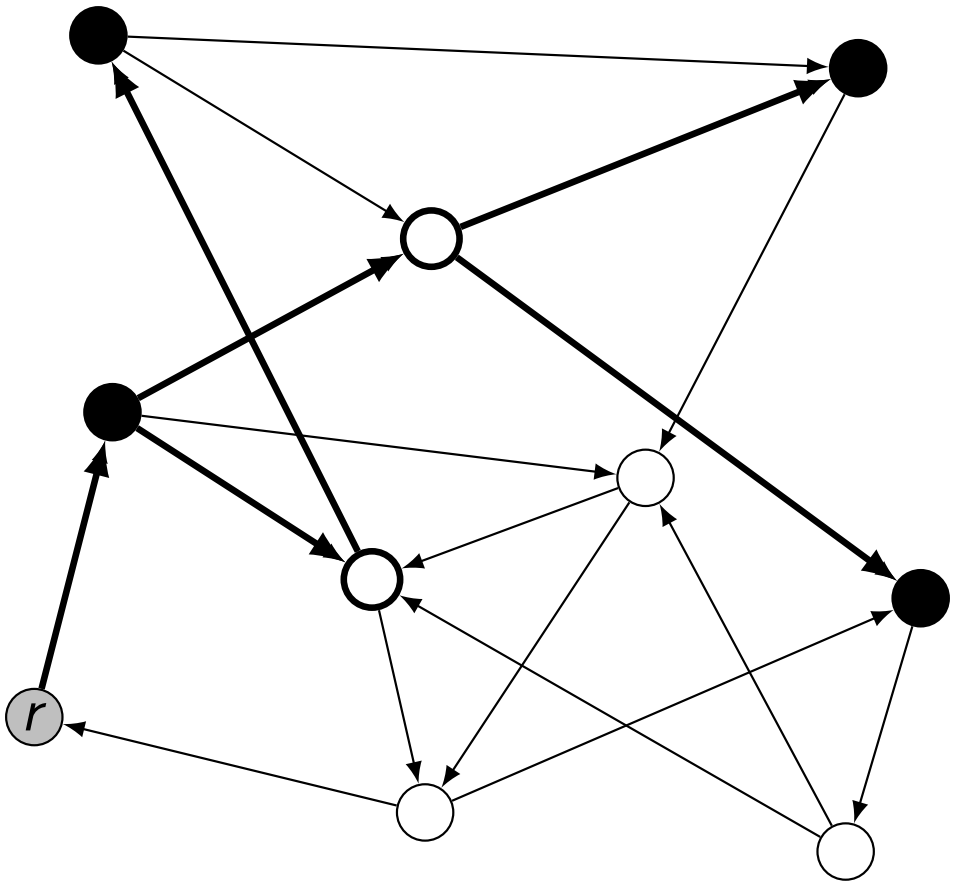
Instance de DST

- $G = (V, A)$;
- Racine $r \in V$;
- k terminaux $X \subset V$;

Solution optimale de DST

- Arborescence T couvrant X ;
- Enracinée en r ;
- Minimisant $|T|$.

Problème de l'arborescence de Steiner (DST)



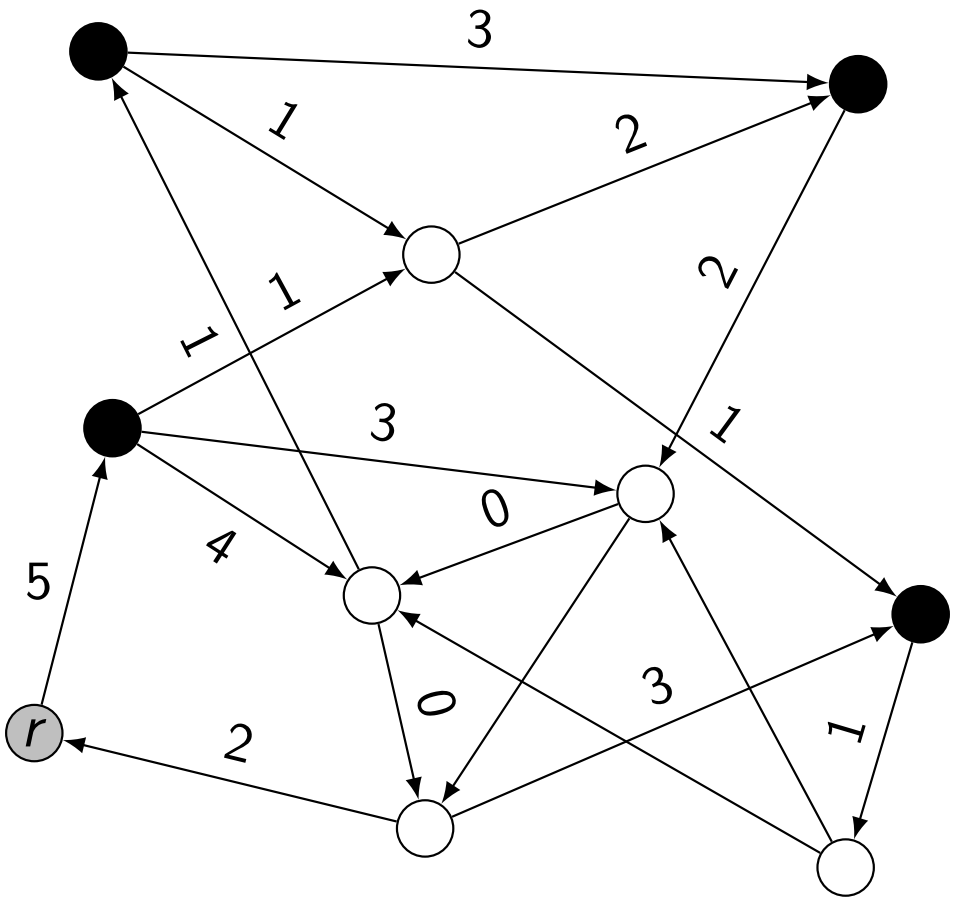
Instance de DST

- $G = (V, A)$;
- Racine $r \in V$;
- k terminaux $X \subset V$;

Solution optimale de DST

- Arborescence T couvrant X ;
- Enracinée en r ;
- Minimisant $|T|$.

Problème de l'arborescence de Steiner (DST)



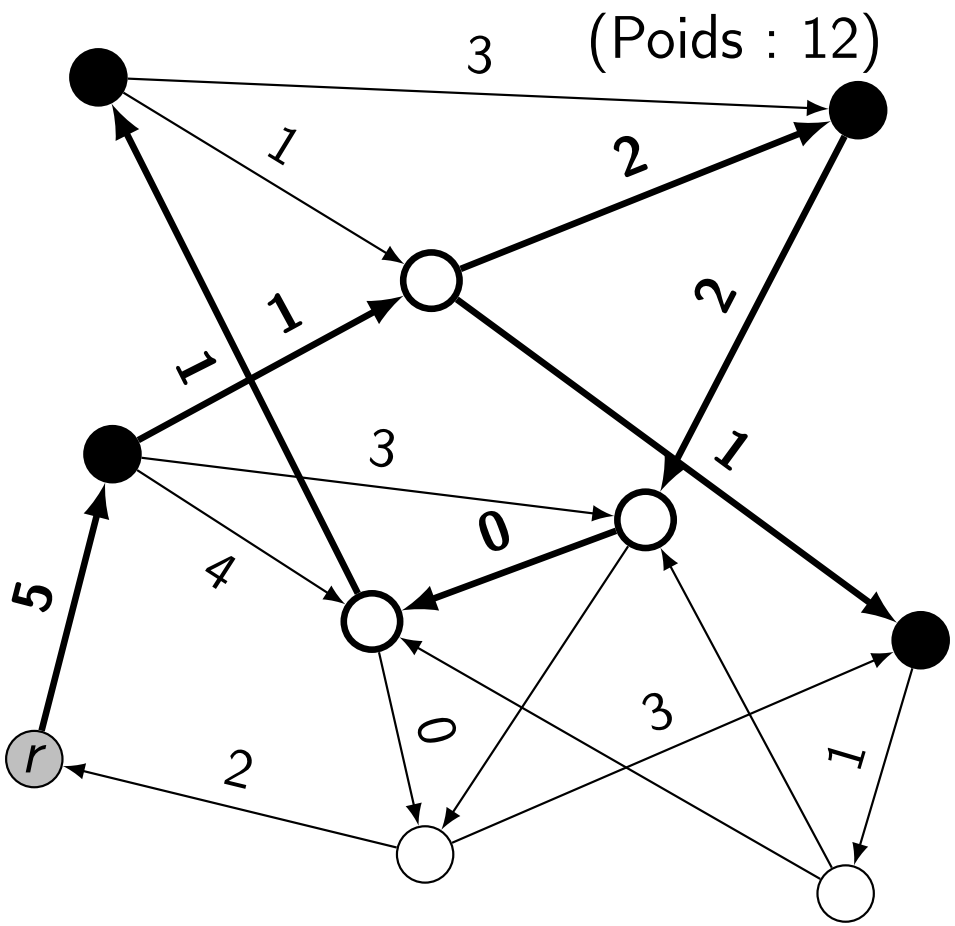
Instance de DST

- $G = (V, A)$;
- Racine $r \in V$;
- k terminaux $X \subset V$;
- **Poids** $\omega : A \to \mathbb{R}^+$.

Solution optimale de DST

- Arborescence T couvrant X ;
- Enracinée en r ;
- Minimisant $\sum_{a \in T} \omega(a)$.

Problème de l'arborescence de Steiner (DST)



Instance de DST

- $G = (V, A)$;
- Racine $r \in V$;
- k terminaux $X \subset V$;
- **Poids** $\omega : A \to \mathbb{R}^+$.

Solution optimale de DST

- Arborescence T couvrant X ;
- Enracinée en r ;
- Minimisant $\sum_{a \in T} \omega(a)$.

Remarques

Quand il n'y a pas de poids, on peut mettre un poids unitaire partout. On pourra donc toujours considérer qu'il y a des poids.

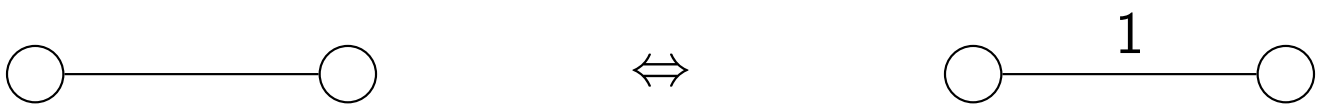
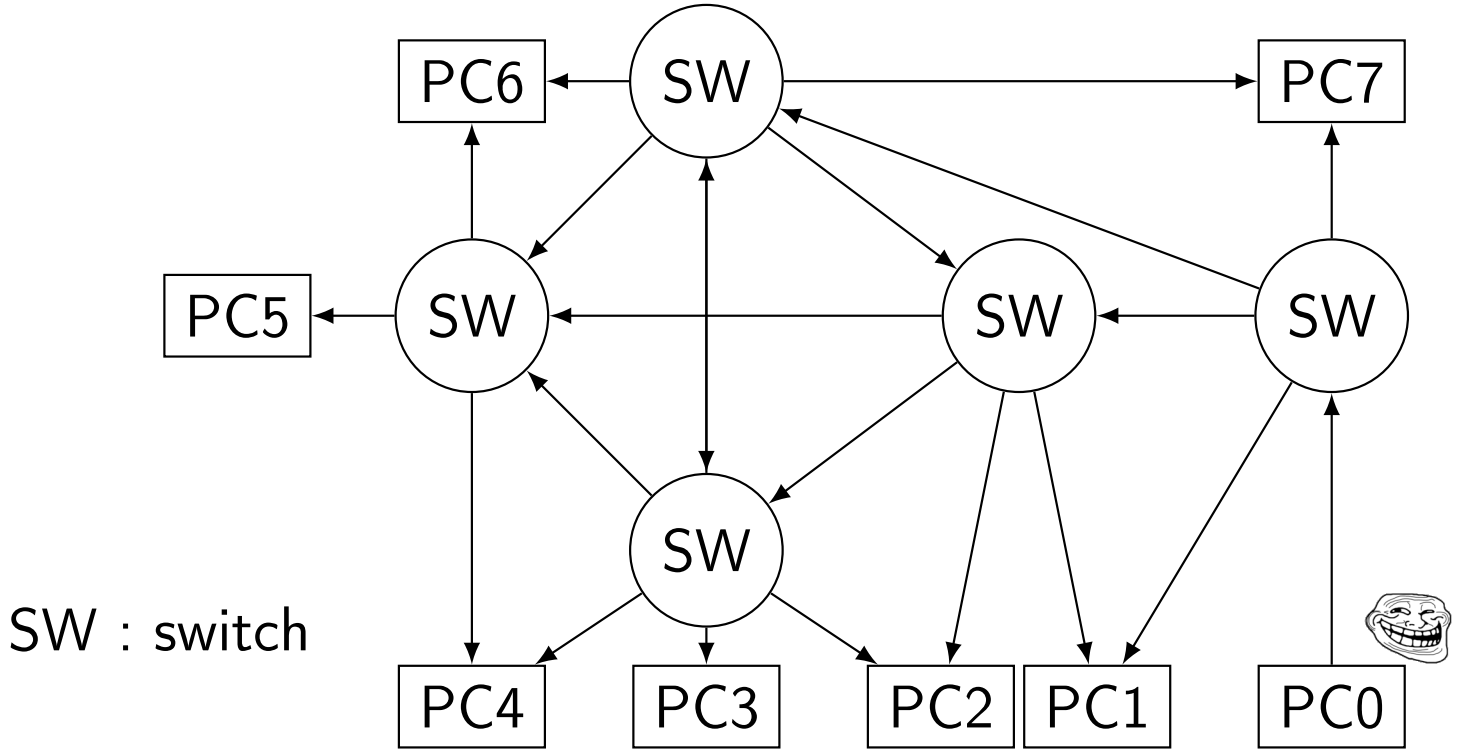


Table des matières

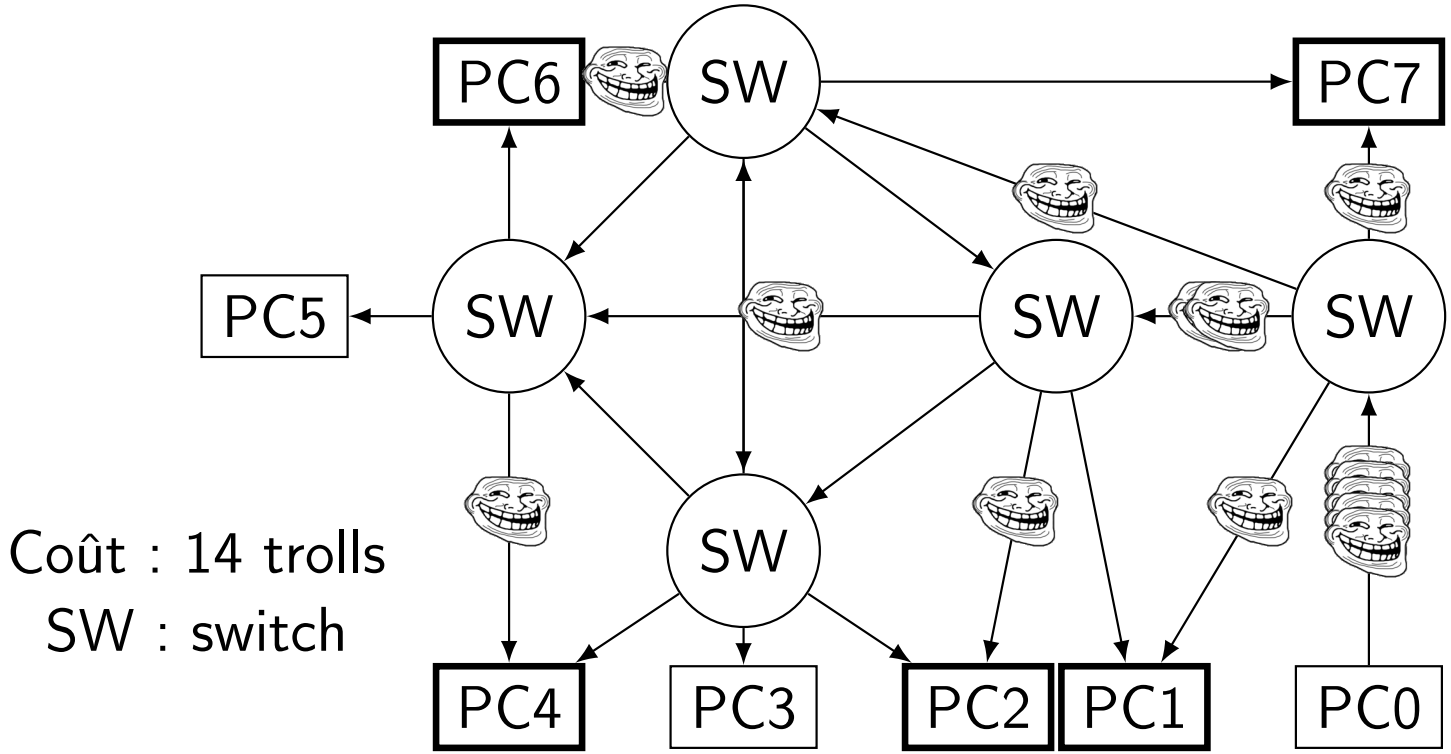
Exemple d'application : diffusion multicast dans un réseau



Problème

PC0 souhaite envoyer un message de 10Go à PC1, PC2, PC4, PC6 et PC7 à travers le réseau.

Exemple d'application : diffusion multicast dans un réseau

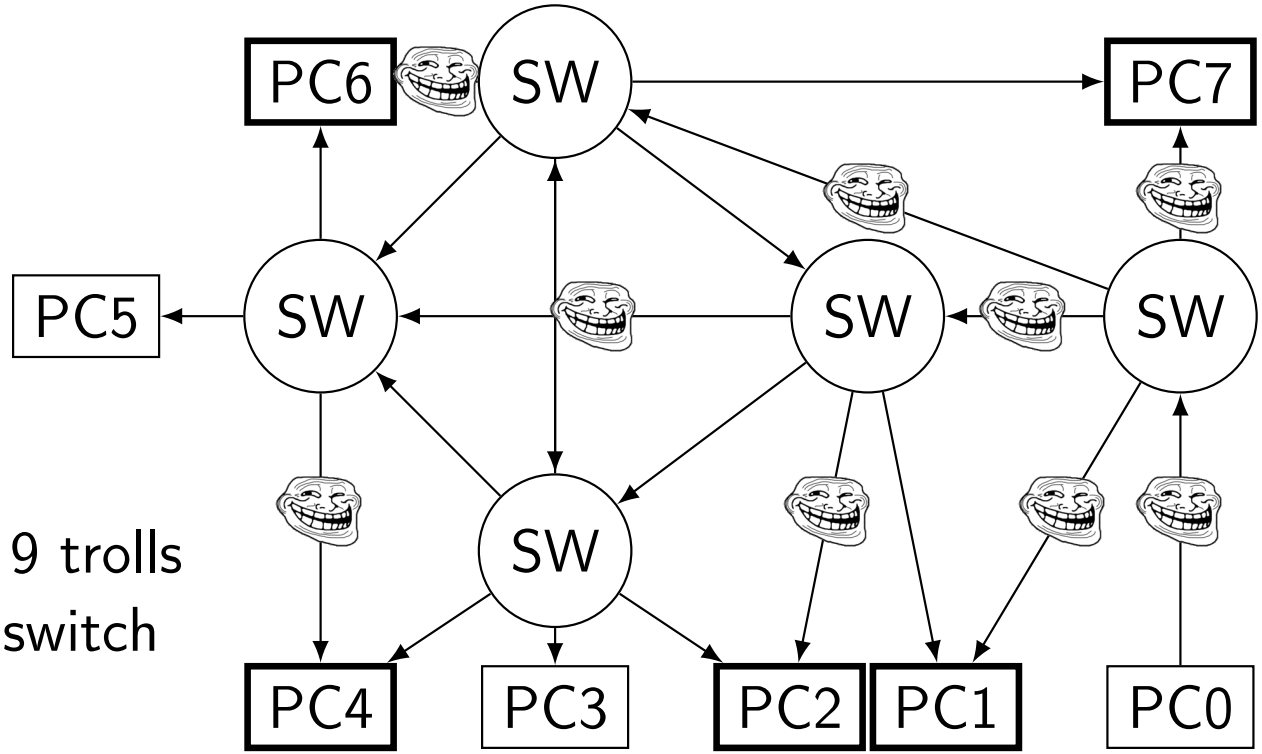


Coût : 14 trolls
SW : switch

Problème

Solution 1 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 séparément par les plus courts chemins.

Exemple d'application : diffusion multicast dans un réseau

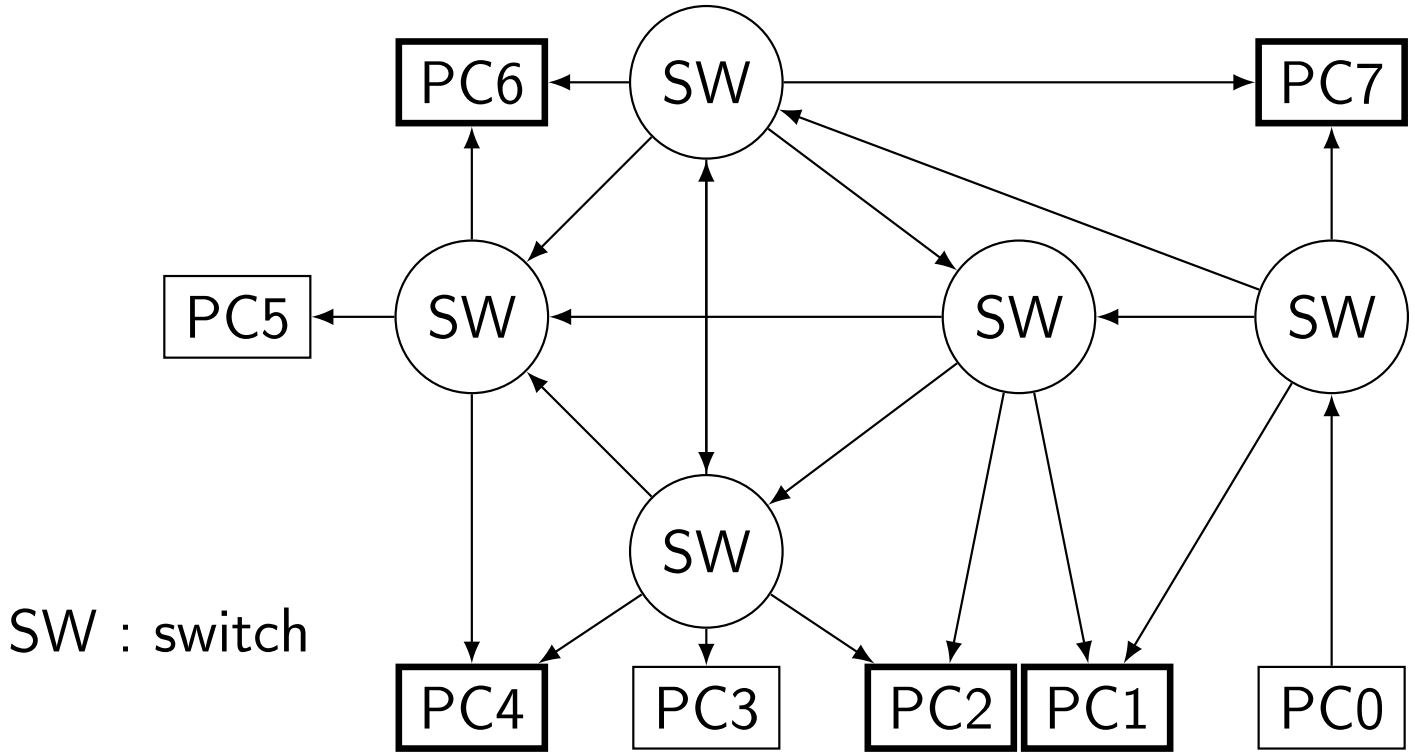


Coût : 9 trolls
SW : switch

Problème

Solution 2 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 par les plus courts chemins **sans faire de copie inutile**.

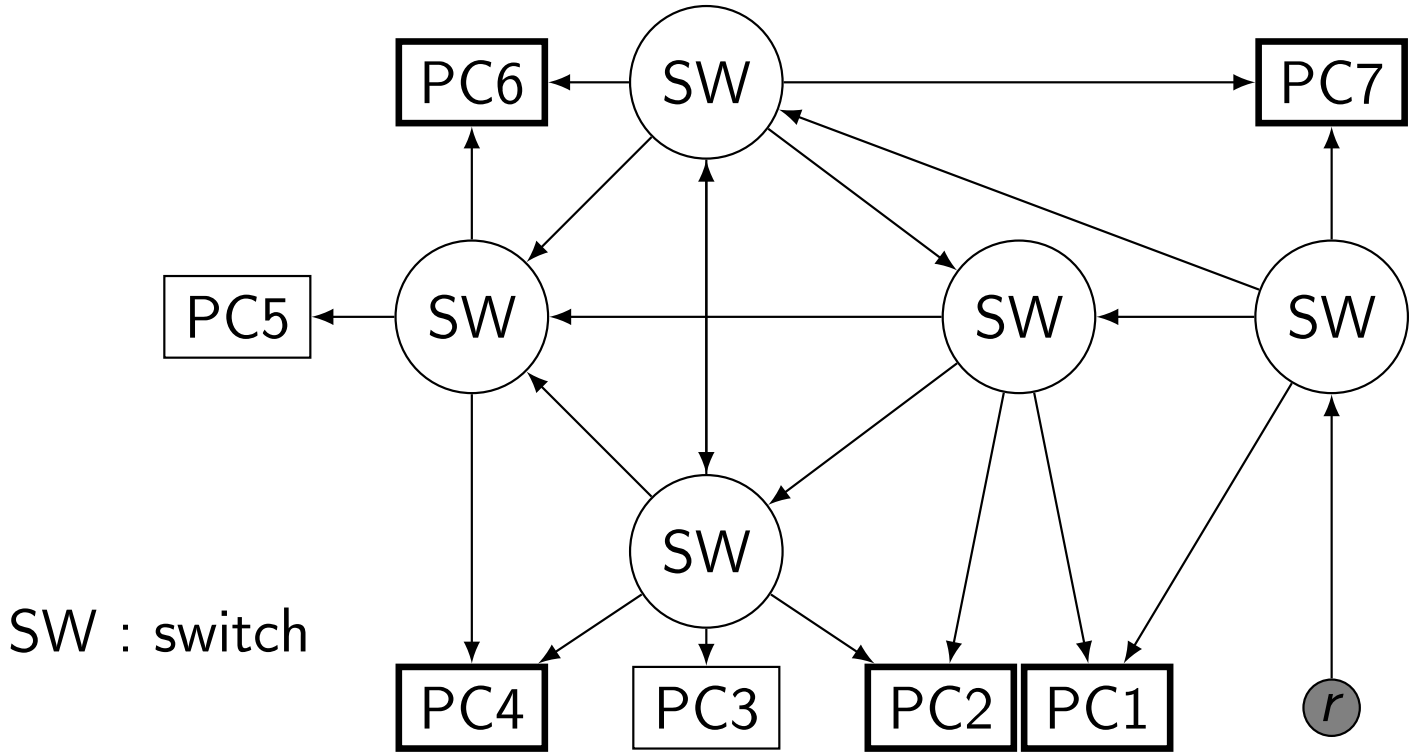
Exemple d'application : diffusion multicast dans un réseau



Problème

Solution 3 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 en utilisant une arborescence de Steiner.

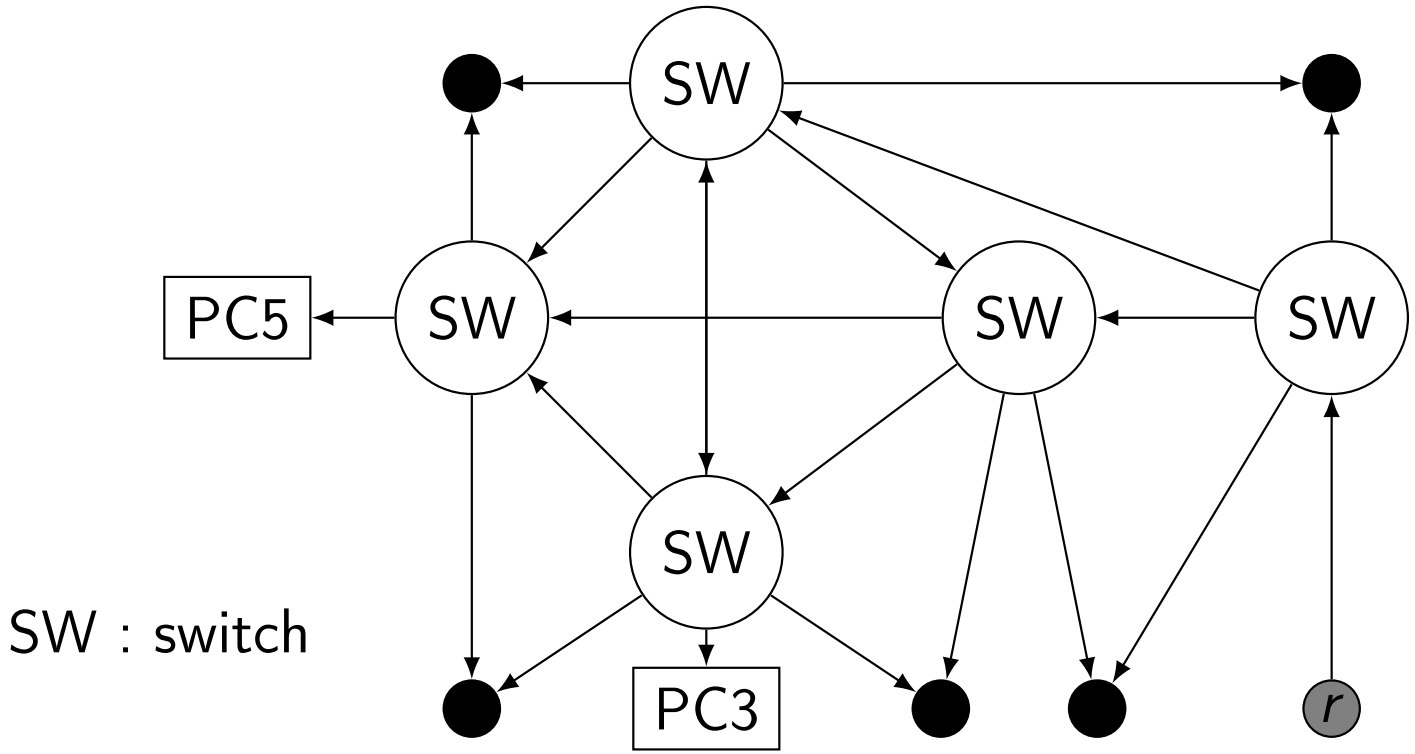
Exemple d'application : diffusion multicast dans un réseau



Problème

Solution 3 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 en utilisant une arborescence de Steiner.

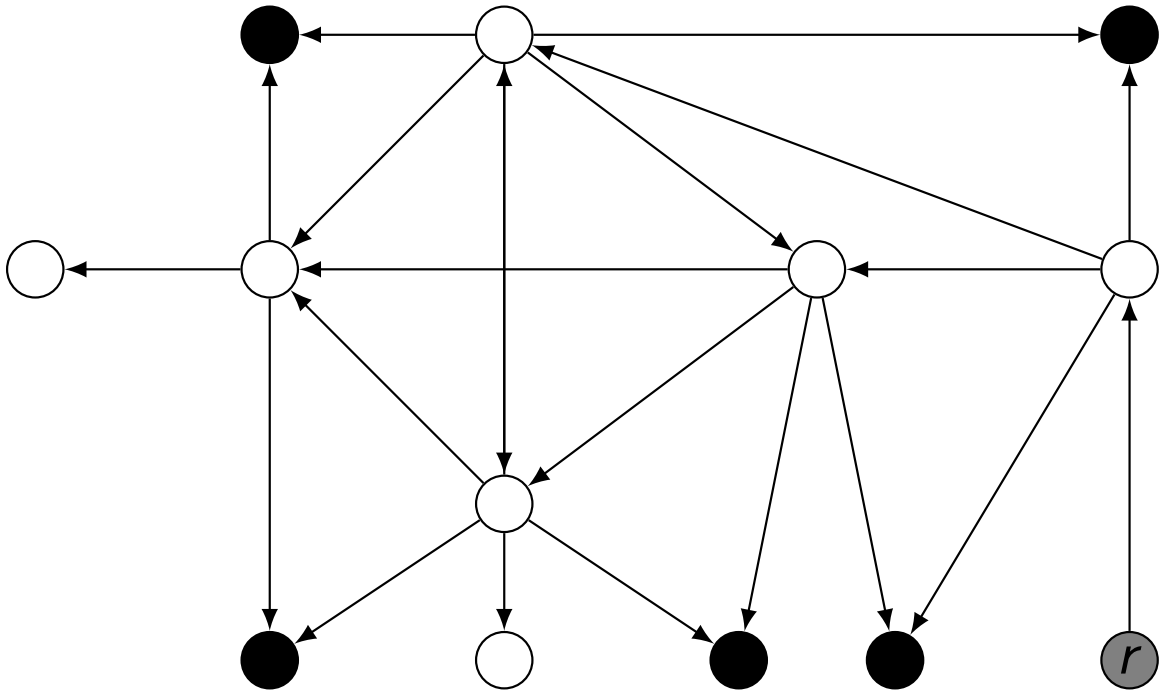
Exemple d'application : diffusion multicast dans un réseau



Problème

Solution 3 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 en utilisant une arborescence de Steiner.

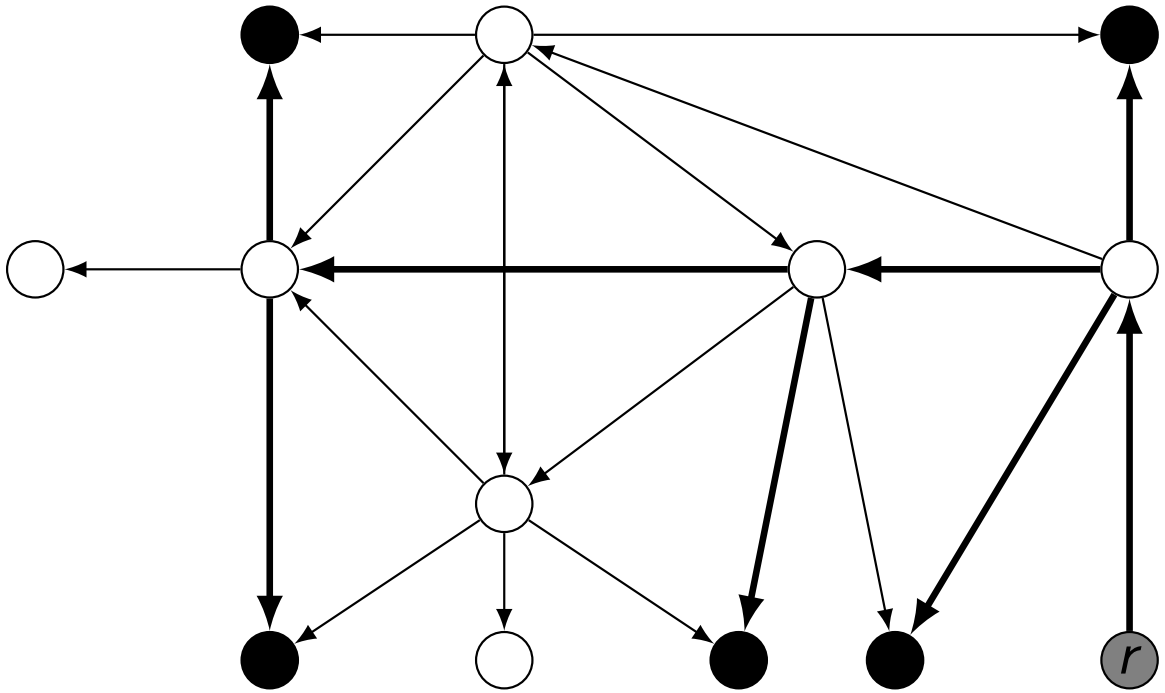
Exemple d'application : diffusion multicast dans un réseau



Problème

Solution 3 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 en utilisant une arborescence de Steiner.

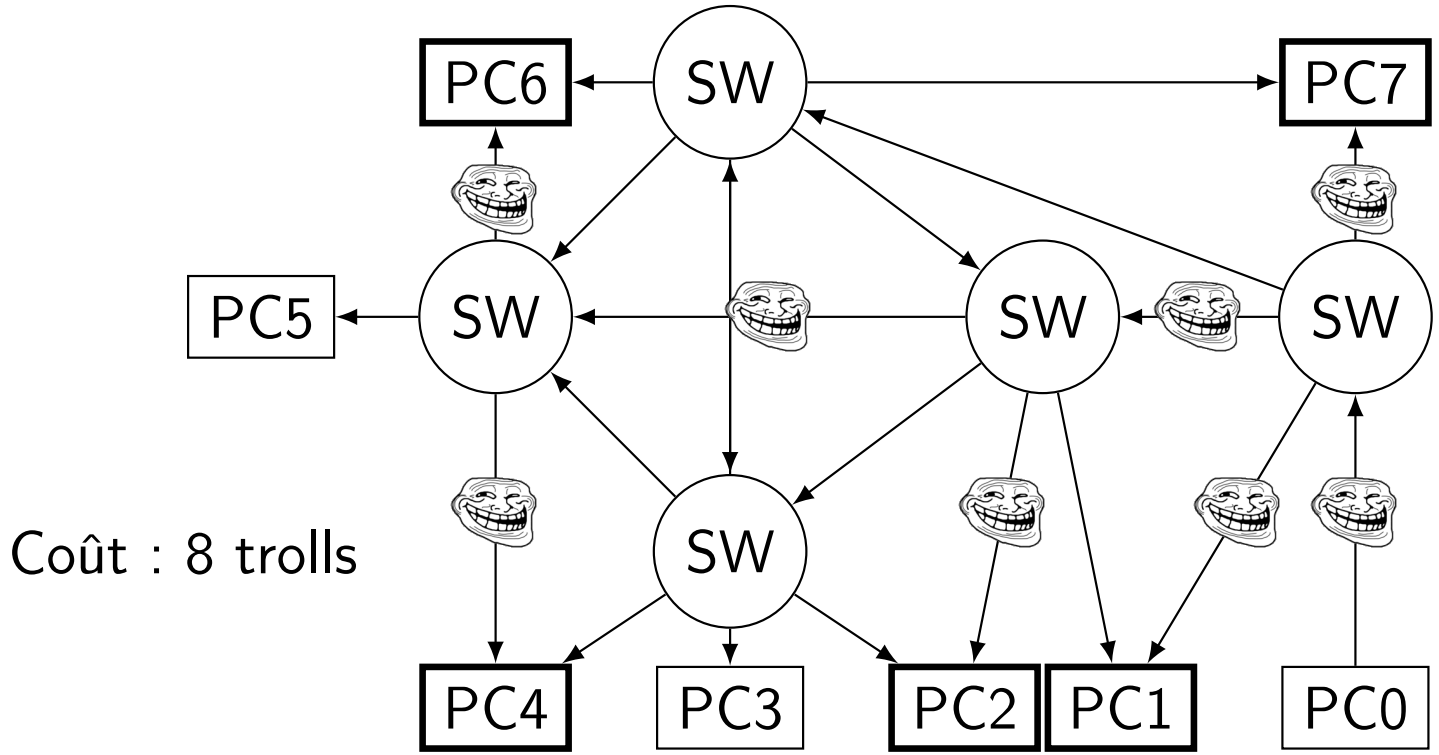
Exemple d'application : diffusion multicast dans un réseau



Problème

Solution 3 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 en utilisant une arborescence de Steiner.

Exemple d'application : diffusion multicast dans un réseau



Coût : 8 trolls

Problème

Solution 3 : PC0 l'envoie à PC1, PC2, PC4, PC6 et PC7 en utilisant une arborescence de Steiner.

Exemple d'application : diffusion multicast dans un réseau

Problèmes similaires

- Envoi d'un email.
- Plusieurs utilisateurs sur le réseau utilisent un serveur FTP décentralisé / un gestionnaire de versions : si l'un modifie le dossier, la modification est transmise aux autres.
- MMORPG
- ...

Table de routage

Attention

Le routage (multicast ou non) se fait rarement en temps réel. On suit des tables de routages préconstruites.

Définition

Une *table de routage* indique aux nœuds où envoyer les paquets en fonction des destinations.

On peut construire des tables de routages optimisées en utilisant le problème de Steiner.

Dimensionnement du réseau

Connu avant de créer le réseau : le besoin ; i.e. les flux probables de communications entre les gens.

Définition

Le *dimensionnement* consiste en la définition des différents paramètres du réseau et son coût : capacités des câbles, nombre de switchs, emplacement des switchs.

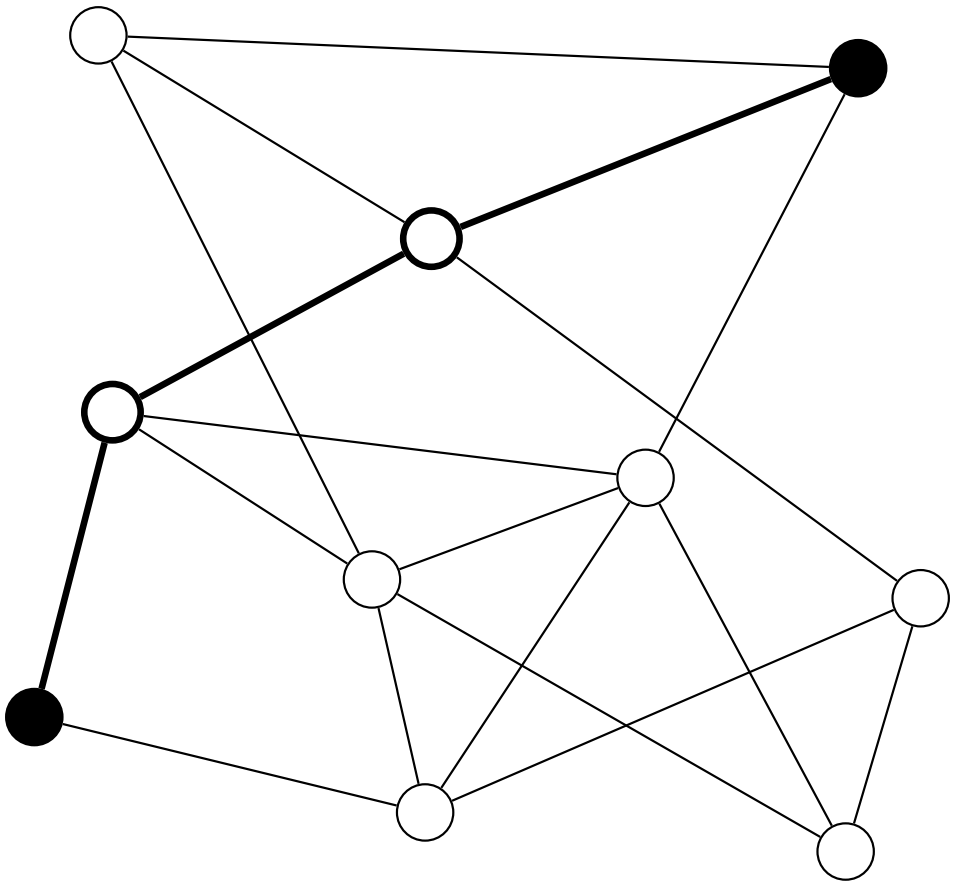
Exemple d'algorithme de dimensionnement

On commence par dimensionner de manière quelconque

- utiliser le problème de Steiner pour calculer des tables de routage optimale ;
- simuler le réseau, observer si des arcs sont saturés ou non ;
- adapter le dimensionnement et recommencer.

Table des matières

Problème de plus court chemin



Instance de SHP

- $G = (V, E)$;
- un nœud u
- un nœud v

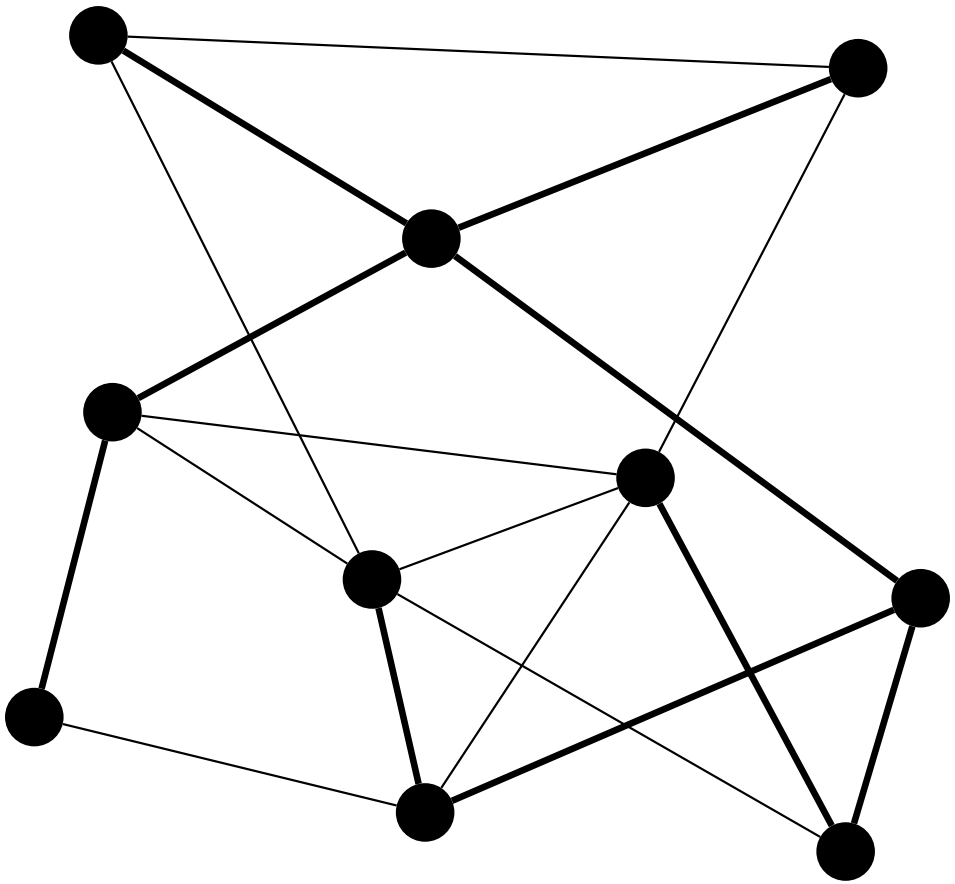
Solution optimale de SHP

- Chaîne P reliant u et v
- Minimisant $|P|$.

Variantes

- orientée ;
- avec des poids.

Arbre couvrant de poids minimum (USpT)



Instance de USpT

- $G = (V, E)$.

Solution optimale de USpT

- Arbre couvrant T de G
- Minimisant $|T|$.

Variantes

- orientée ;
- avec des poids.

Group Steiner Tree

Instance

- Un graphe $G = (V, E)$;
- des ensembles de nœuds $G_1, G_2, G_3, \ldots, G_k$.

Solution réalisable

Un arbre de Steiner T tel que

- T contient au moins 1 nœud de chaque ensemble G_i.

Forêt de Steiner

Instance

- Un graphe $G = (V, E)$;
- des ensembles de nœuds $X_1, X_2, X_3, \ldots, X_p$

Solution réalisable

Des arbres de Steiner $T_1, T_2, \ldots, T_p$ tels

- T_i relie les nœuds de X_i ;

minimisant $\sum_{i=1}^{p} |T_i|$.

Forêt de Steiner avec capacité

Instance

- Un graphe $G = (V, E)$;
- des ensembles de nœuds $X_1, X_2, X_3, \ldots, X_p$
- des capacités $c : E \to \mathcal{N}$ sur les arêtes.

Solution réalisable

Des arbres de Steiner $T_1, T_2, \ldots, T_p$ tels

- T_i relie les nœuds de X_i,
- le nombre d'arbres contenant l'arête e ne dépasse pas $c(i)$;

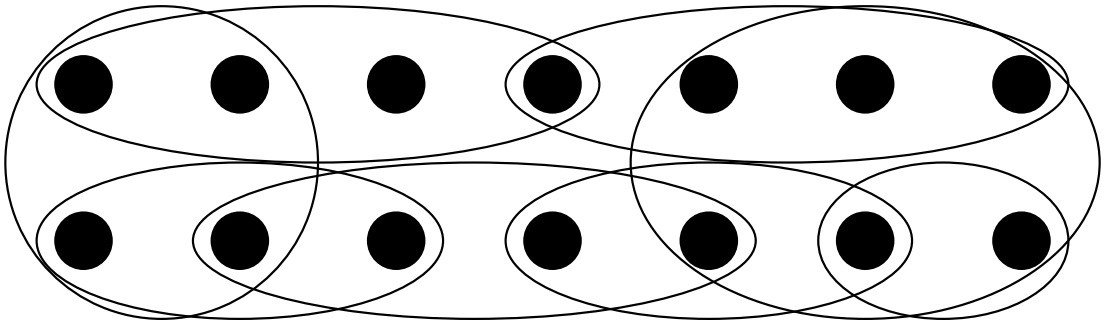
minimisant $\sum_{i=1}^{p} |T_i|$.

Table des matières

Problème de Couverture par ensembles

Définition du problème

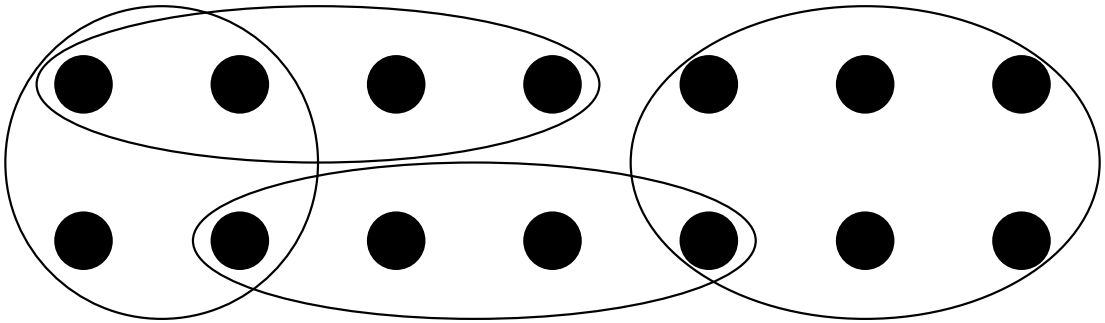
Connaissant un ensemble $\mathcal{X}$ et un sous-ensemble $\mathcal{S}$ de parties de $\mathcal{X}$, trouver un sous-ensemble C de $\mathcal{S}$ couvrant $\mathcal{X}$ et minimisant $|C|$.



Problème de Couverture par ensembles

Définition du problème

Connaissant un ensemble $\mathcal{X}$ et un sous-ensemble $\mathcal{S}$ de parties de $\mathcal{X}$, trouver un sous-ensemble C de $\mathcal{S}$ couvrant $\mathcal{X}$ et minimisant $|C|$.



Réduction polynomiale

Théorème

DST est NP-Difficile.

Preuve : Réduction de Couverture par ensembles vers DST. (cf Tableau)

Réduction polynomiale

Théorème

UST est NP-Difficile.

Preuve : Réduction de Couverture par ensembles vers UST. (cf Tableau)

NP-Complétude

Théorème

Les versions de décision de DST et UST sont dans NP. Ces problèmes sont donc NP-Complets.

Preuve : Tableau !

Table des matières

Algorithme naïf

Exercice !

Décrire, en français, un algorithme simple pour résoudre DST ou UST.

Algorithme naïf

Exercice !

Décrire, en français, un algorithme simple pour résoudre DST ou UST.

Solution

- Enumérer tous les sous-ensembles d'arcs/arêtes.
- Pour chacun vérifier qu'il s'agit d'une solution réalisable.
- Ne conserver que la meilleur solution lors du parcours.

Complexité en temps : voir au tableaux !

Algorithm de Dreyfus Wagner

Exercice !

Pour résoudre le problème de plus courts chemins, l'algorithme de Dijkstra utilise le principe suivant :

- un sous-chemin d'un plus court chemin est aussi un plus court chemin.

Énoncer une propriété similaire pour les arborescences de Steiner.

Algorithm de Dreyfus Wagner

Principe

- $\omega(u, Y)$: poids **présumé** d'un arbre de Steiner optimal enraciné en $u \in V$ et couvrant $Y \subset X$. Ce poids évolue au fur et à mesure que l'algorithme progresse
- L : liste de tous les sous-ensembles (u, Y) **triée** par $\omega(u, Y)$
- **Initialisation** : $\omega[x, \{x\}] = 0$, les autres sont infinis.
- **Itération** :
  - retirer (u, Y), le premier élément de L
  - si $u = r$ et $Y = X$, **fin !**
  - *Étendre* $(u, Y) \Rightarrow$ cf Tableau

Complexité en temps : voir au tableaux !

Algorithm de Dreyfus Wagner

Principe

- $\omega(u, Y)$: poids **présumé** d'un arbre de Steiner optimal enraciné en $u \in V$ et couvrant $Y \subset X$. Ce poids évolue au fur et à mesure que l'algorithme progresse
- L : liste de tous les sous-ensembles (u, Y) **triée** par $\omega(u, Y)$
- **Initialisation** : $\omega[x, \{x\}] = 0$, les autres sont infinis.
- **Itération** :
  - retirer (u, Y), le premier élément de L
  - si $u = r$ et $Y = X$, **fin !**
  - *Étendre* $(u, Y) \Rightarrow$ cf Tableau

Complexité en temps : voir au tableaux !

Algorithm de Dreyfus Wagner

Principe

- $\omega(u, Y)$: poids **présumé** d'un arbre de Steiner optimal enraciné en $u \in V$ et couvrant $Y \subset X$. Ce poids évolue au fur et à mesure que l'algorithme progresse
- L : liste de tous les sous-ensembles (u, Y) **triée** par $\omega(u, Y)$
- **Initialisation** : $\omega[x, \{x\}] = 0$, les autres sont infinis.
- Itération :
  - retirer (u, Y), le premier élément de L
  - si $u = r$ et $Y = X$, **fin !**
  - *Étendre* $(u, Y) \Rightarrow$ cf Tableau

Complexité en temps : voir au tableaux !

Algorithm de Dreyfus Wagner

Principe

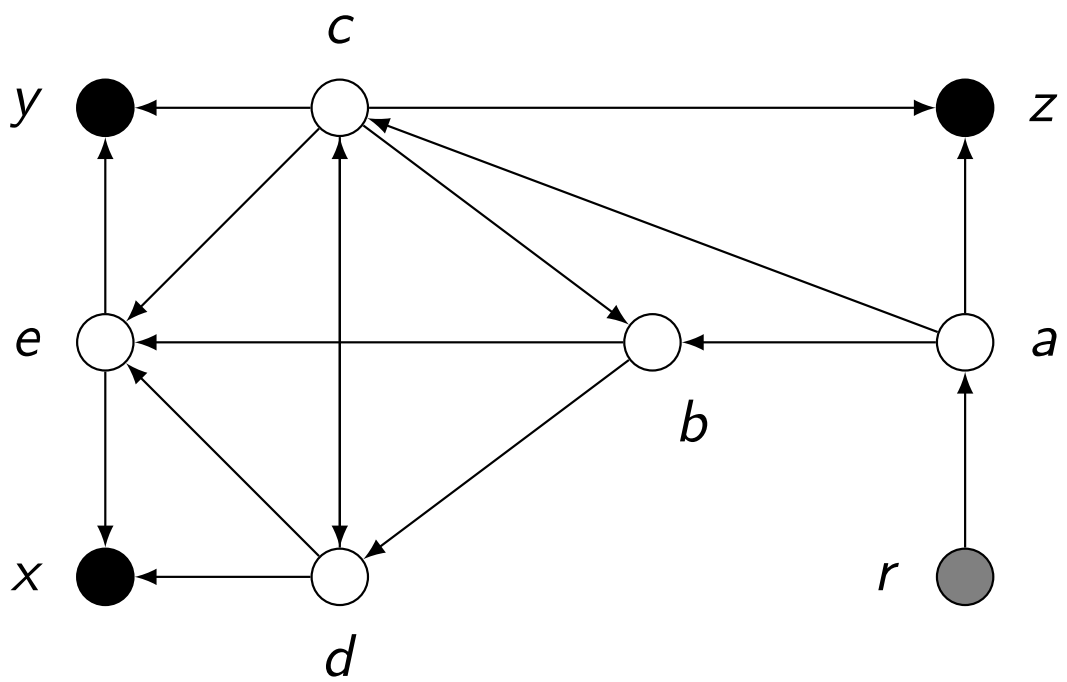
- $\omega(u, Y)$: poids **présumé** d'un arbre de Steiner optimal enraciné en $u \in V$ et couvrant $Y \subset X$. Ce poids évolue au fur et à mesure que l'algorithme progresse
- L : liste de tous les sous-ensembles (u, Y) **triée** par $\omega(u, Y)$
- **Initialisation** : $\omega[x, \{x\}] = 0$, les autres sont infinis.
- **Itération** :
  - retirer (u, Y), le premier élément de L
  - si $u = r$ et $Y = X$, **fin !**
  - *Étendre* $(u, Y) \Rightarrow$ cf Tableau

Complexité en temps : voir au tableaux !

Algorithm de Dreyfus Wagner

Exercice !

Appliquer l'algorithme sur le graphe suivant.



Et si on fixait le nombre de terminaux ?

Question !

Que devient le problème de Steiner avec 1 seul terminal ? Le problème est-il polynomial ?

Et si on fixait le nombre de terminaux ?

Exercice !

Décrire un algorithme (simple) en temps polynomial pour résoudre le problème avec 2 terminaux.

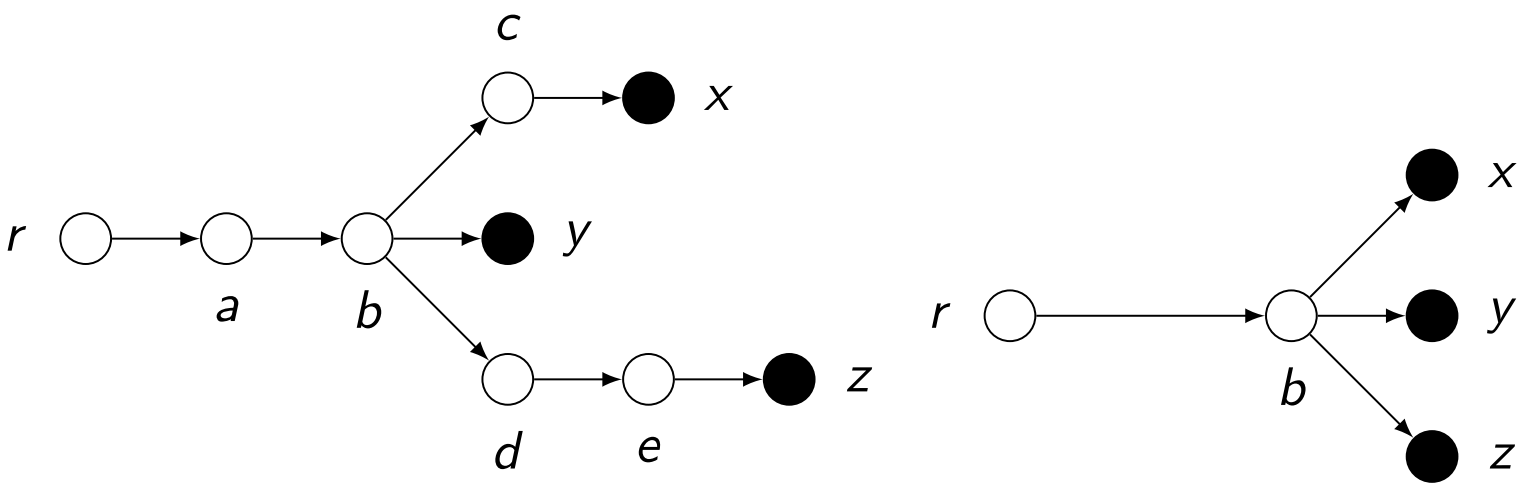
Et si on fixait le nombre de terminaux ?

Exercice !

Décrire un algorithme (simple) en temps polynomial pour résoudre le problème avec 3 terminaux.

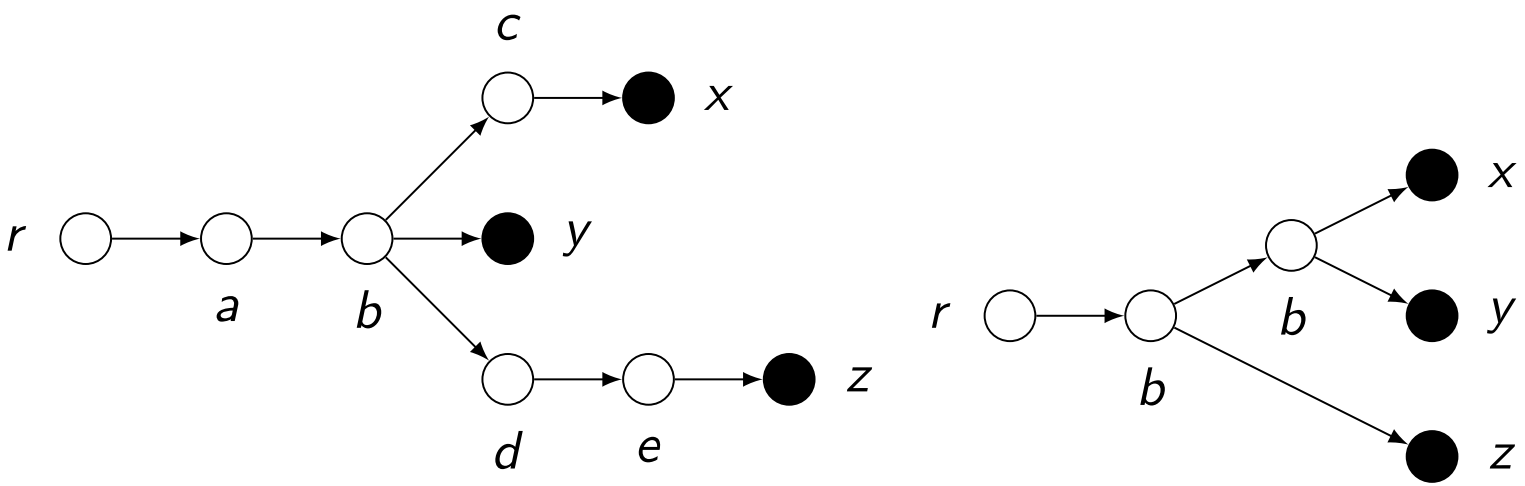
Patron

Un patron d'une arborescence T est une version **binaire** *raccourcie* de T.



Patron

Un patron d'une arborescence T est une version **binaire** *raccourcie* de T.



Énumération des patrons

Question !

Combien y a-t-il de forme d'arborescences binaires complètes (2 fils non ordonnés ou 0 fils) avec

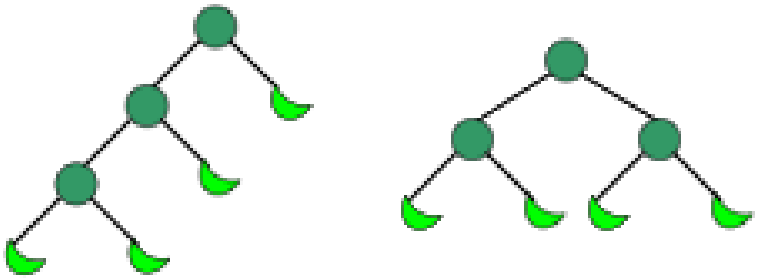
- 1 feuille
- 2 feuilles
- 3 feuilles
- k feuilles

Énumération des patrons

Question !

Combien y a-t-il de forme d'arborescences binaires complètes (2 fils non ordonnés ou 0 fils) avec

- 1 feuille
- 2 feuilles
- 3 feuilles
- k feuilles



Énumération des patrons

Question !

Combien y a-t-il de forme d'arborescences binaires complètes (2 fils non ordonnés ou 0 fils) avec

- 1 feuille
- 2 feuilles
- 3 feuilles
- k feuilles

0, 1, 1, 1, 2, 3, 6, 11, 23, 46, 98, 207, 451, 983, 2179, 4850, 10905, 24631, 56011, 127912, 293547, ...(`https://oeis.org/A001190`)

Énumération des patrons

Question !

Combien y a-t-il de forme d'arborescences binaires complètes (2 fils non ordonnés ou 0 fils) avec

- 1 feuille étiquetée
- 2 feuilles étiquetées
- 3 feuilles étiquetées
- k feuilles étiquetées

sachant que les nœuds internes ne sont pas étiquetés.

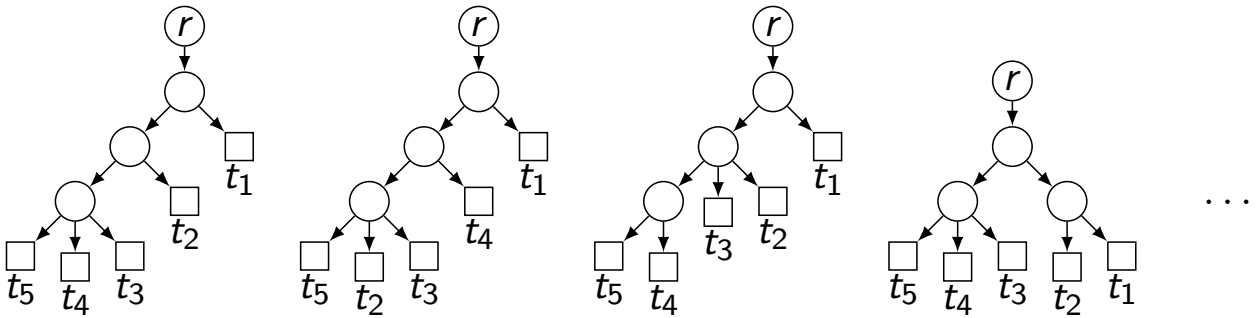
Énumération des patrons

Question !

Combien y a-t-il de forme d'arborescences binaires complètes (2 fils non ordonnés ou 0 fils) avec

- 1 feuille étiquetée
- 2 feuilles étiquetées
- 3 feuilles étiquetées
- k feuilles étiquetées

sachant que les nœuds internes ne sont pas étiquetés.



Énumération des patrons

Question !

Combien y a-t-il de forme d'arborescences binaires complètes (2 fils non ordonnés ou 0 fils) avec

- 1 feuille étiquetée
- 2 feuilles étiquetées
- 3 feuilles étiquetées
- k feuilles étiquetées

sachant que les nœuds internes ne sont pas étiquetés.

$k!! = 1 \cdot 3 \cdot 5 \cdots 2k-3 = 1, 1, 3, 15, 105, 945, 10395, 135135, 2027025, 34459425, 654729075, 13749310575, 316234143225,$ (`https://oeis.org/A001147`)

Énumération des patrons

Problème

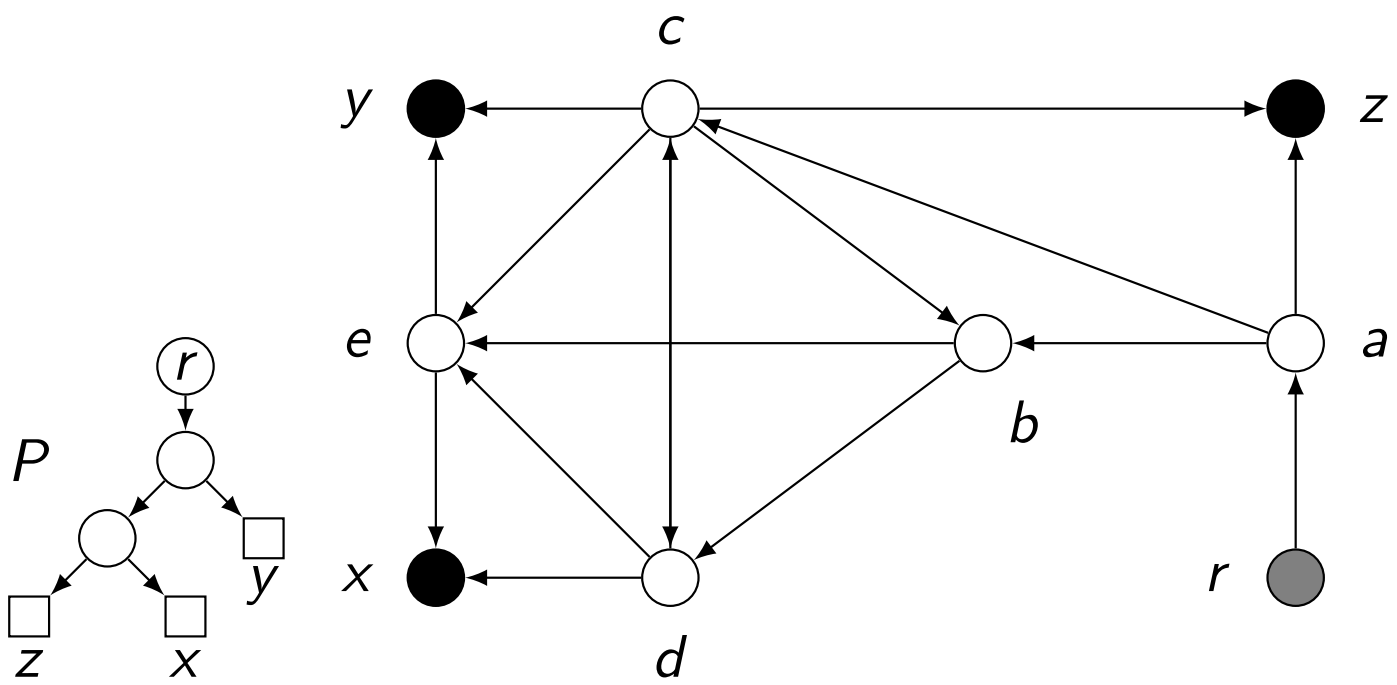
Soit $\mathcal{I}$ une instance de DST, et P un patron avec k terminaux, trouver une arborescence T de $\mathcal{I}$ dont P est le patron, de poids minimum.

Il existe un algorithme en temps polynomial.

Énumération des patrons

Exercice !

Par exemple, trouver l'arborescence de poids minimum dont P est le patron dans le graphe suivant.



Énumération des patrons

Algorithme complet

- Énumérer tous les patrons, un par un.
- Pour chaque patron P, trouver une arborescence de poids minimum dont P est le patron.
- Conserver et renvoyer la meilleure des solutions.

Complexité en temps : $n^3 \cdot (1 \cdot 3 \cdots 2k - 3)$.

Et si on fixait le nombre de terminaux ?

- Complexité de Dreyfus Wagner $O(3^k \cdot poly)$
- Complexité des patrons $O((1 \cdot 3 \cdots 2k-3) \cdot poly)$

Question !

Que se passe-t-il si on applique ces algos uniquement sur des instances avec un nombre fixé de terminaux ?

Table des matières

Programme linéaire (flots)

Propriété

Une arborescence de Steiner contient un chemin de la racine vers chaque terminal.

Programme linéaire (flots)

Modélisation d'un chemin de r vers x :

- une variable binaire f_a par arcs a : $f_a = 1$ ssi a appartient au chemin ;
- un arc sort de r : $\sum_{v \in V} f_{(r,v)} = 1$;
- un arc entre en x : $\sum_{v \in V} f_{(v,x)} = 1$;
- un arc entre en w ssi un arc sort de w si $w \neq r, x$: $\sum_{v \in V} f_{(v,w)} = \sum_{v \in V} f_{(w,v)}$.

$\Rightarrow$ il s'agit d'un flot entier de valeur 1 de r à x.

Programme linéaire (flots)

Modélisation d'un arbre de r vers tous les terminaux :

- une variable binaire x_a par arc a : $x_a = 1$ ssi a appartient à l'arbre de Steiner ;
- une variable binaire f_a^x par arcs a et par terminal x ;
- modéliser un chemin de r à x avec les variables f_a^x ;
- $\forall x, x_a \geq f_a^x$: $x_a = 1$ ssi $\exists x \backslash f_a^x = 1$.

Objectif : minimiser $\sum_{a \in A} \omega(a) \cdot x_a$.

Programme linéaire (flots)

On peut relâcher les variables f_a^x

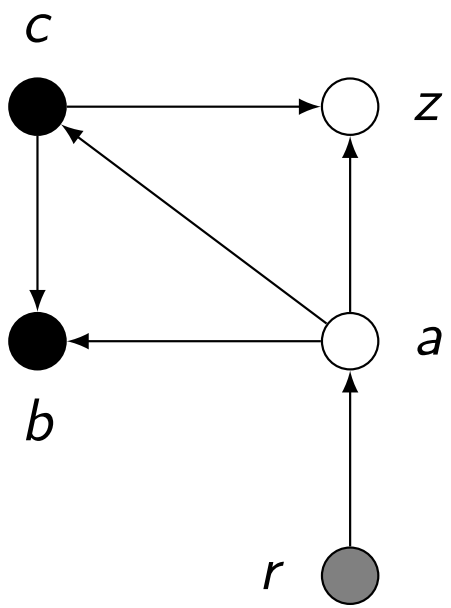
- une variable binaire x_a par arc a : $x_a = 1$ ssi a appartient à l'arbre de Steiner ;
- une variable $f_a^x \in [0; 1]$ par arcs a et par terminal x ;
- modéliser un **flot** de valeur 1 de r à x avec les variables f_a^x ;
- $\forall x, x_a \geq f_a^x$: $x_a = 1$ ssi $\exists x \backslash f_a^x > 0$.

Objectif : minimiser $\sum_{a \in A} \omega(a) \cdot x_a$.

Programme linéaire (flots)

Exercice !

Décrire un programme linéaire pour DST (version flot).



Programme linéaire (coupes)

Propriété

Une arborescence de Steiner contient un chemin de la racine vers chaque terminal.

Programme linéaire (coupes)

Modélisation d'un chemin de r vers x :

- une variable binaire x_a par arc a : $x_a = 1$ ssi a appartient au chemin ;
- soit S tel que $r \in S$ et $x \in V \backslash S$;
- au moins 1 arc du chemin relie S à $V \backslash S$: $\sum_{\substack{(v,w) \in A \\ v \in S \\ w \in V \backslash S}} x_{(v,w)} \geq 1$

Programme linéaire (coupes)

Modélisation d'un arbre de r vers tous les terminaux :

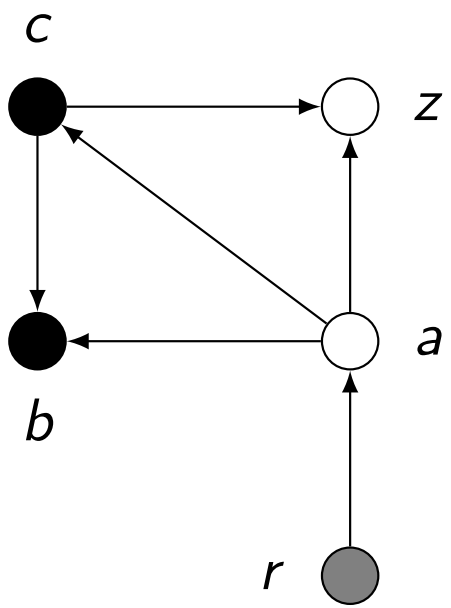
- une variable binaire x_a par arc a : $x_a = 1$ ssi a appartient à l'arbre ;
- soit S tel que $r \in S$ et $X \cap V \backslash S \neq \emptyset$;
- au moins 1 arc de l'arbre relie S à $V \backslash S$: $\sum_{\substack{(v,w) \in A \\ v \in S \\ w \in V \backslash S}} x_{(v,w)} \geq 1$

Objectif : minimiser $\sum_{a \in A} \omega(a) \cdot x_a$.

Programme linéaire (coupes)

Exercice !

Décrire un programme linéaire pour DST (version coupe).



Programmation linéaire

Propriété

Si on relâche les variables x_a, la solution des deux programmes linéaires est la même.

Or, puisqu'ils renvoie une arbre de Steiner optimal, la solution entière des deux programmes est la même aussi.

Théorème

Les deux programmes linéaires ont même saut intégral.

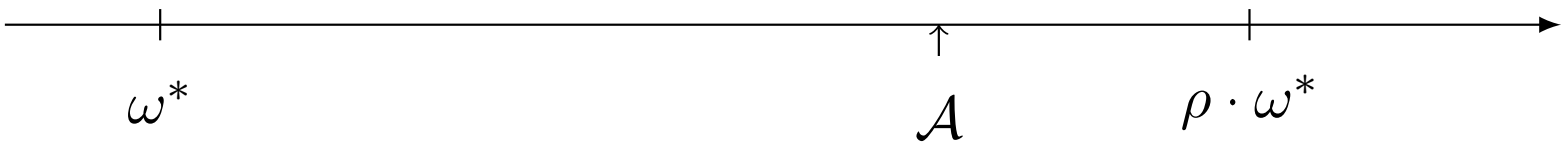
Table des matières

Rappels

Soit un problème de **minimisation** Π et un algorithme heuristique $\mathcal{A}$ pour résoudre Π.

Définition

$\mathcal{A}$ est une ρ-approximation si et seulement si pour toute instance de Π de valeur optimale ω^*, $\mathcal{A}$ renvoie une solution de valeur au plus $\rho \cdot \omega^*$.



Rappels

Remarques

- $\mathcal{A}$ n'est pas nécessairement un algorithme polynomial
- ρ n'est pas nécessairement une constante
- un algorithme exact est une 1-approximation

Remarque

Un problème Π est dit ρ-approximable s'il est approximable en temps polynomial avec un facteur ρ.

Classes d'approximabilité

On peut classer les problèmes d'optimisation selon leur approximabilité.

- APX : ρ-approximable en temps polynomial, ρ constant.
- PTAS : $(1+\varepsilon)$-approximable en temps polynomial, pour tout $\varepsilon > 1$.
- FPTAS : $(1+\varepsilon)$-approximable en temps polynomial en n et $\frac{1}{\varepsilon}$, pour tout $\varepsilon > 1$.
- $f(n)-$APX : $f(n)$-approximable en temps polynomial.

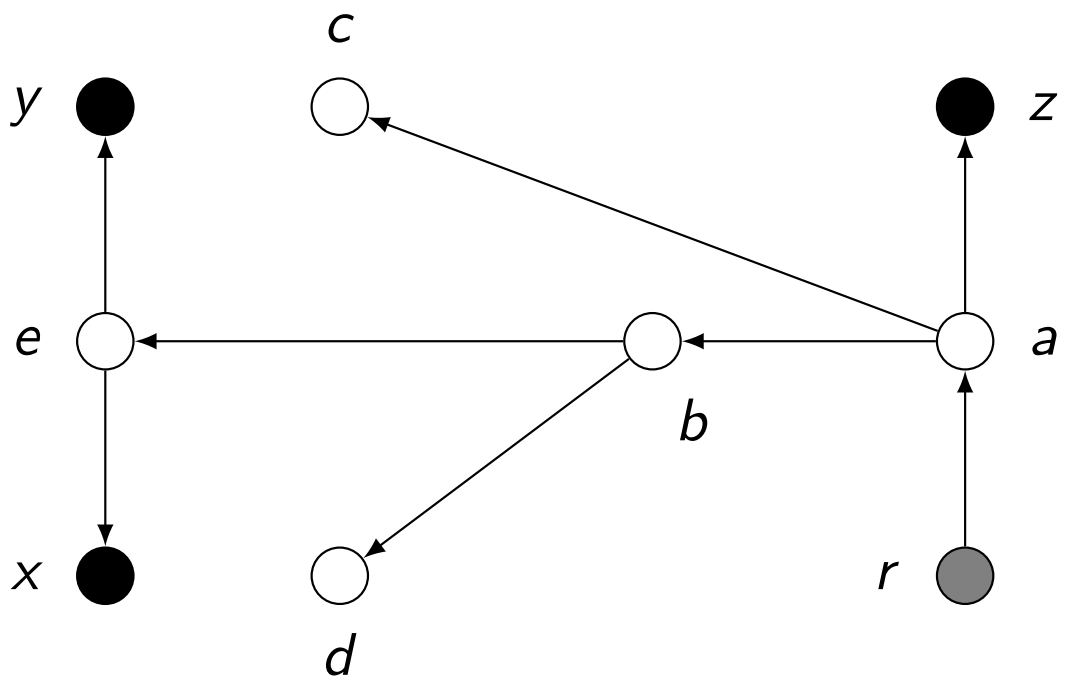
- $\mathcal{A}$: ρ-approximation pour Π.
- $\mathcal{B}$: transforme une solution T de Π en solution T' de valeur plus petite.

Propriété

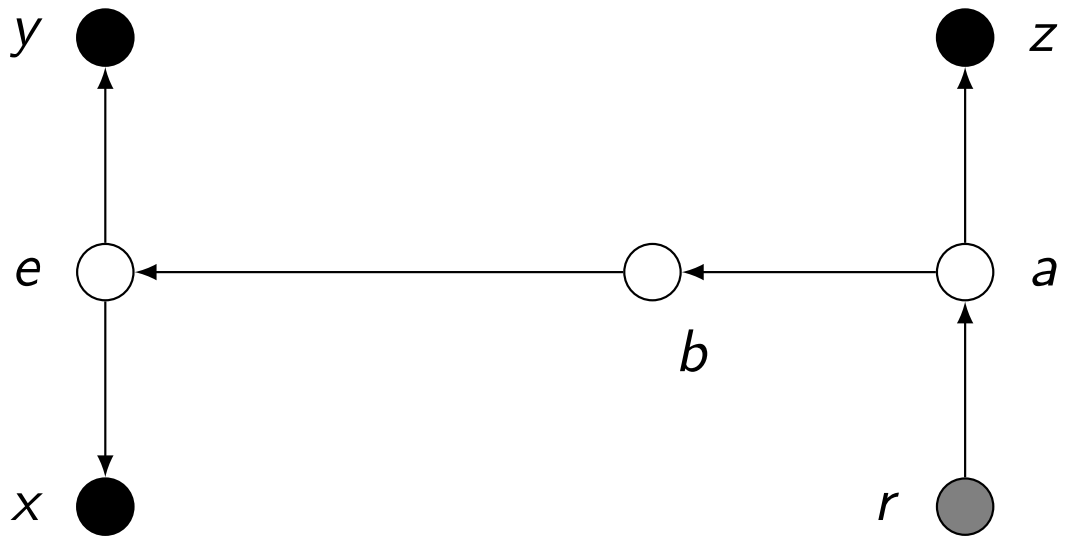
$\mathcal{C} = \mathcal{B} \circ \mathcal{A}$ est une ρ-approximation pour Π.

Table des matières

Élagage



Élagage



Version *faible* des DST et UST

Définition

On considère que tout sous-graphe reliant les terminaux entre eux est une solution réalisable de UST : il n'est pas nécessaire de renvoyer un arbre.

Transformer en temps polynomial une telle solution en arbre de Steiner moins cher :

Version *faible* des DST et UST

Définition

On considère que tout sous-graphe reliant les terminaux entre eux est une solution réalisable de UST : il n'est pas nécessaire de renvoyer un arbre.

Transformer en temps polynomial une telle solution en arbre de Steiner moins cher :

- En construire un arbre couvrant
- Elaguer l'arbre

Propriété

Si la version faible de UST (resp. DST) est ρ-approximable alors UST (resp. DST) l'est aussi.

Table des matières

Approximabilité de DST et UST

Exercice !

Décrire une heuristique simple pour trouver une solution de UST ou DST en temps **constant**. Quelle est son rapport d'approximation ?

Approximabilité de DST et UST

Exercice !

Décrire une heuristique simple pour trouver une solution de UST ou DST en temps **constant**. Quelle est son rapport d'approximation ?

Solution : Renvoyer le graphe tout entier (temps constant si on a un pointeur sur le graphe).
Rapport d'approximation : au tableau.

Approximabilité de UST

Exercice !

Décrire une heuristique simple pour trouver une solution de UST.

Approximabilité de UST

Théorème

UST est 2-approximable : approximation des plus courts chemins.

- Trouver deux terminaux (x_1, x_2) les plus proches.
- Ajouter leur plus court chemin à la solution en cours.
- Contracter le chemin et recommencer.

Rapport d'approximation : au tableau !

Approximabilité de UST

Théorème

UST est $\frac{11}{6}$-approximable : extensions à des triplets.

- Trouver trois terminaux (x_1, x_2, x_3) *minimisant une critère*.
- Critère : gain de relier (x_1, x_2, x_3) entre eux plutôt que d'utiliser la solution proposée par la 2-approximation.
- Ajouter leur arbre couvrant de poids min à la solution en cours.
- Contracter l'arbre et recommencer.

On peut généraliser à des p-upplets, p constant => 1.598-approximation.

Meilleure approximabilité de UST connue.

Théorème

UST est 1.39-approximable.

(en réalité : $\log(4) + \varepsilon$-approximable, pour tout ε)

Approximabilité de DST

Exercice !

Décrire une heuristique simple pour trouver une solution de DST.

Approximabilité de DST

Exercice !

Quel est le rapport d'approximation de cet algorithme ?

Soit p une constante

- Relier la racine aux p terminaux les plus proches avec un arbre de Steiner optimal.
- Recommencer tant qu'il reste des terminaux.

Approximabilité de DST : restriction des instances

Exercice !

Montrer que l'approximation du problème de Steiner est équivalente dans le cas général et dans le cas des graphes orientés complets respectant l'inégalité triangulaire.

Approximabilité de DST : restriction des instances

Exercice !

Montrer que l'approximation du problème de Steiner est équivalente dans le cas général et dans le cas des graphes orientés complets respectant l'inégalité triangulaire.

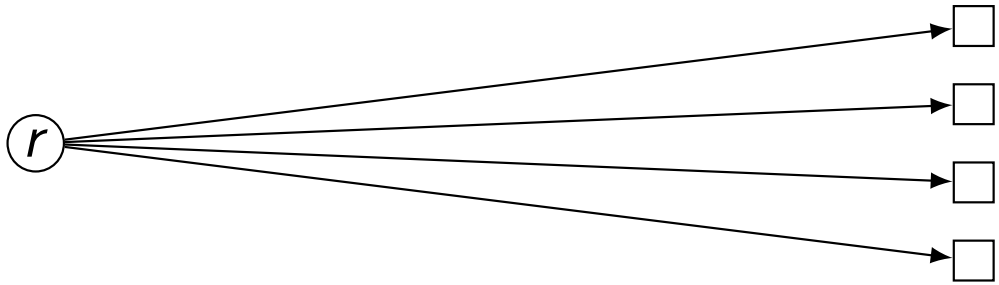
On supposera maintenant que l'on est toujours dans ce cas.

Approximabilité de DST

Théorème

DST est k-approximable : approximation des plus courts chemins.

- Relier r à X avec des plus courts chemins.

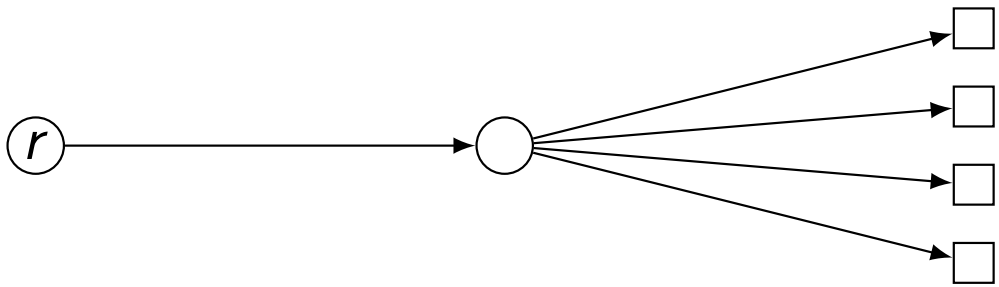


Approximabilité de DST : approximation de Charikar *et al.*

Théorème

DST est $O(\sqrt{k})$-approximable.

- Relier r à X avec deux plus courts chemins *pas trop chers*.

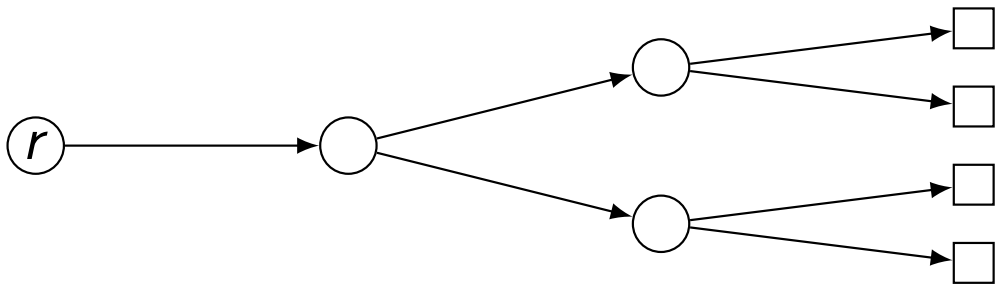


Approximabilité de DST : approximation de Charikar *et al.*

Théorème

DST est $O(\sqrt[3]{k})$-approximable.

- Relier r à X avec trois plus courts chemins *pas trop chers*.

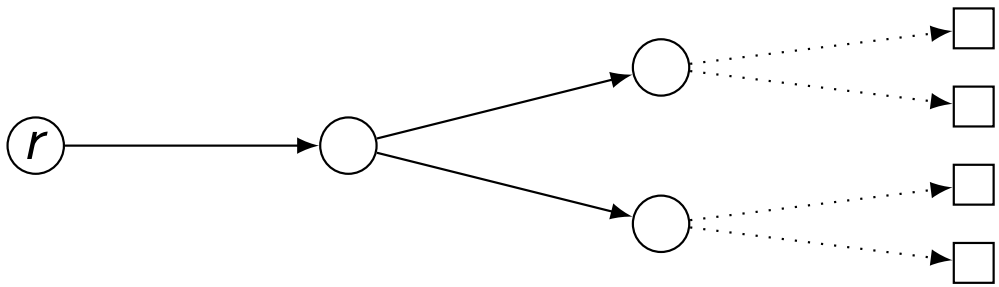


Approximabilité de DST : approximation de Charikar *et al.*

Théorème

DST est $O(\sqrt[p]{k})$-approximable, pour tout p.

- Relier r à X avec p plus courts chemins *pas trop chers*.



Approximation de Charikar *et al.* : des arbres pas trop chers...

Lemme

Si T_i^* est l'arbre de hauteur i de poids minimum, alors $\omega(T_i^*) \leq O(\sqrt[i]{k}) \cdot \omega^*$.

Approximation de Charikar *et al.* : des arbres pas trop chers...

Lemme

Si T_i^* est l'arbre de hauteur i de poids minimum, alors $\omega(T_i^*) \leq O(\sqrt[i]{k}) \cdot \omega^*$.

MAIS :

Théorème

Trouver T_i^* est un problème NP-Complet, quel que soit i constant supérieur à 2 ! ! ! !

Il faut donc approcher T_i^*.

Approximation de Charikar *et al.* : des arbres pas trop chers...

Définition

Une solution est dite partielle si elle relie la racine à au moins un terminal, mais pas nécessairement tous.

Définition

La densité $d(T)$ d'une solution partielle est son coût sur le nombre de terminaux couverts : $d(T) = \frac{\omega(T)}{k(T)}$.

Exemple : au tableau.

Approximation de Charikar *et al.* : des arbres pas trop chers...

Plus la densité des solutions partielles est faible, meilleure est l'approximation.

Définition

Une $\rho(k)$-approximation partielle est un algorithme qui renvoie une solution partielle dont la densité est au plus $\rho(k)\frac{\omega^*}{k}$.

Remarque :

- ω^* : poids d'une solution optimale
- $\frac{\omega^*}{k}$: densité d'une solution optimale

Approximation de Charikar *et al.* : des arbres pas trop chers...

Exercice !

Donnez une borne supérieure du rapport d'approximation de l'algorithme ci-dessous. On suppose que $\frac{\rho(k)}{k}$ décroissante.

Soit $\mathcal{A}$ une $\rho(k)$-approximation partielle.

- Utiliser $\mathcal{A}$ pour couvrir une partie des terminaux.
- Supprimer les terminaux couverts.
- Recommencer tant qu'il reste des terminaux.

Approximation de Charikar *et al.* : des arbres pas trop chers...

Réponse :

Lemme

Si $\mathcal{A}$ est une $\rho(k)$-approximation partielle et si $\frac{\rho(k)}{k}$ décroit, utiliser $\mathcal{A}$ pour couvrir de manière gloutonne des terminaux est une $\int_{x=0}^{k} \frac{\rho(x)}{x} dx$-approximation.

Approximation de Charikar *et al.*

Théorème

DST est $O(\sqrt[p]{k})$-approximable.

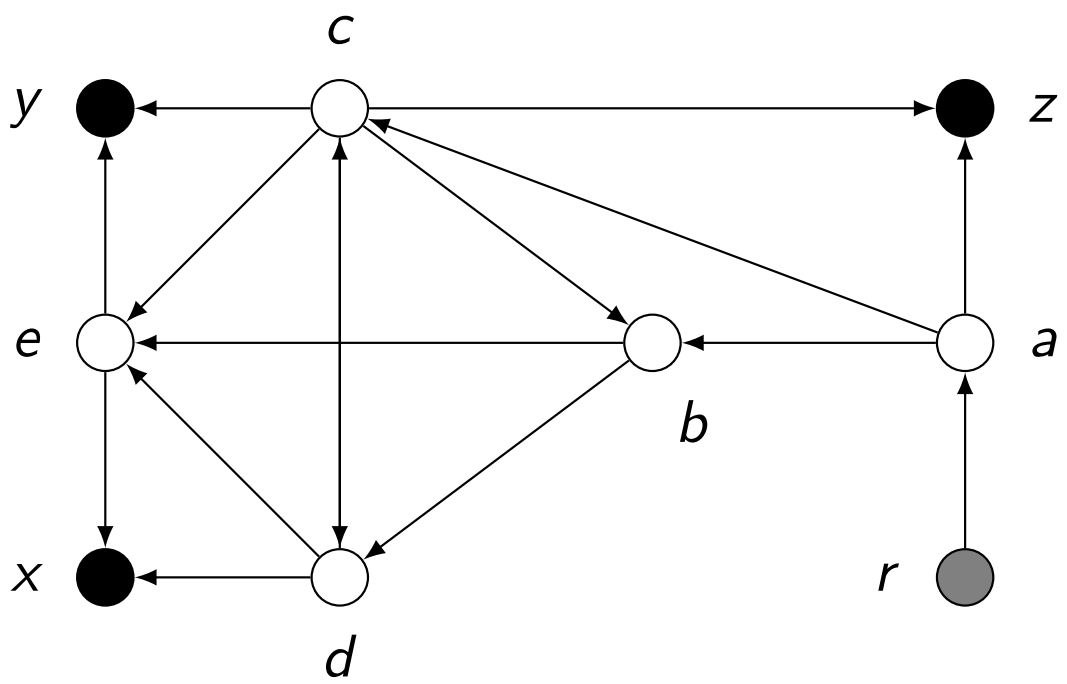
Approximation partielle récursive $\mathcal{A}_i(k, r, X))$:

- si $i = 1$, relier r aux k plus proches terminaux qui sont dans X ;
- sinon
  - $T \leftarrow \emptyset$
  - Tant que $X \neq \emptyset$
    - $T' \leftarrow \arg\min(d(\{r, v\} \cup A_{i-1}(k', v, X)), v \in V, 1 \leq k' \leq k)$
    - $T \leftarrow T \cup T'$
    - $X \leftarrow X \backslash X(T')$
    - $k \leftarrow k - k(T')$
  - Renvoyer T.

Algorithme de Charikar *et al.*

Exercice !

Tester $A_2(3, r, \{x, y, z\})$ sur l'exemple suivant (attention !, transformer d'abord ce graphe en graphe complet avec inégalité triangulaire).



Algorithme de Charikar *et al.* : $O(\sqrt{k})$-approximation (1/2)

Exercice !

Montrer que $\arg\min(d(\{r, v\} \cup A_{i-1}(k', v, X)), v \in V, 1 \leq k' \leq k)$ renvoie l'arbre de hauteur 2 de densité minimum.

Algorithme de Charikar *et al.* : $O(\sqrt{k})$-approximation (1/2)

Lemme

Si T_2^* est l'arbre de hauteur 2 de poids minimum, alors $\omega(T_2^*) \leq O(\sqrt{k}) \cdot \omega^*$.

Lemme

Si $\mathcal{A}$ est une $\rho(k)$-approximation partielle et si $\frac{\rho(k)}{k}$ décroit, utiliser $\mathcal{A}$ pour couvrir de manière gloutonne des terminaux est une $\int_{x=0}^{k} \frac{\rho(x)}{x} dx$-approximation.

Exercice !

À l'aide de l'exo précédent, et des deux lemmes précédents : montrer que $A_2(k, r, X)$ est une $O(\sqrt{k})$-approximation.

Table des matières

Rappels sur l'inapproximabilité

3 méthodes usuelles :

- L-Réductions
- Réductions isofacteurs
- GAP

L-Réduction

Définition

Il existe une L-réduction de Ξ vers Π s'il existe deux réels α et β et une réduction polynomiale des instances ξ de Ξ en instances π de Π tels que :

- $OPT(\pi) \leq \alpha OPT(\xi)$
- pour toute solution réalisable P de π, il existe une solution réalisable X de ξ telle que $(\omega(X) - OPT(\xi)) \leq \beta(\omega(P) - OPT(\pi))$.

Exercice !

Montrer que si Π est ρ-approximable alors Ξ est $1 + \alpha\beta(\rho - 1)$-approximable.

Réduction isofacteur

Définition

Il existe une réduction isofacteur de Ξ vers Π s'il existe deux réels α et β et une réduction polynomiale des instances ξ de Ξ en instances π de Π tels que :

- $OPT(\pi) \leq \alpha OPT(\xi)$
- pour toute solution réalisable P de π, il existe une solution réalisable X de ξ telle que $\omega(X) \leq \beta\omega(P)$.

Exercice !

Montrer que si Π est ρ-approximable alors Ξ est $1 + \alpha\beta\rho$-approximable.

Inapproximabilité par GAP

Définition

Il existe une GAP-Réduction du problème de **décision** Ξ NP-Complet vers le problème d'**optimisation** Π s'il existe une constante α, une réduction polynomiale des instances ξ de Ξ en instances π de Π et une valeur G telles que :

- si ξ est positive, $OPT(\pi) \leq G$;
- si ξ est négative, pour toute solution réalisable P de Π, $\omega(P) \geq \alpha G$

Exercice !

Montrer que Π ne peut être approché avec un rapport α si P $\neq$ NP.

Inapproximabilité de UST

Théorème

UST est APX-Complet (au sens de la L-réduction), même si tous les poids sont unitaires.

Corollaire

UST n'est pas dans PTAS, même s'il n'y a que des arêtes de poids 1 et 2 sauf si P = NP.

Démonstration : L-Réduction depuis le problème de Vertex-Cover-B, au tableau.

Inapproximabilité de UST

Théorème

UST ne peut être approché avec un rapport meilleur que $\frac{96}{95}$ sauf si $P = NP$.

Inapproximabilité de DST

Théorème

Il existe une constante $c > 0$ telle que DST ne peut être approché avec un rapport meilleur que $c \log(k)$, même si tous les poids sont unitaires sauf si P = NP.

Démonstration : réduction isofacteur depuis le problème de couverture par ensembles.

Inapproximabilité de DST

Théorème

Pour tout $\varepsilon > 0$, DST ne peut être approché avec un rapport meilleur que $\log^{2-\varepsilon}(k)$ sauf si NP $\subseteq$ ZTIME($n^{\text{polylog}(n)}$).

Arborescence de Steiner avec capacités

Instance

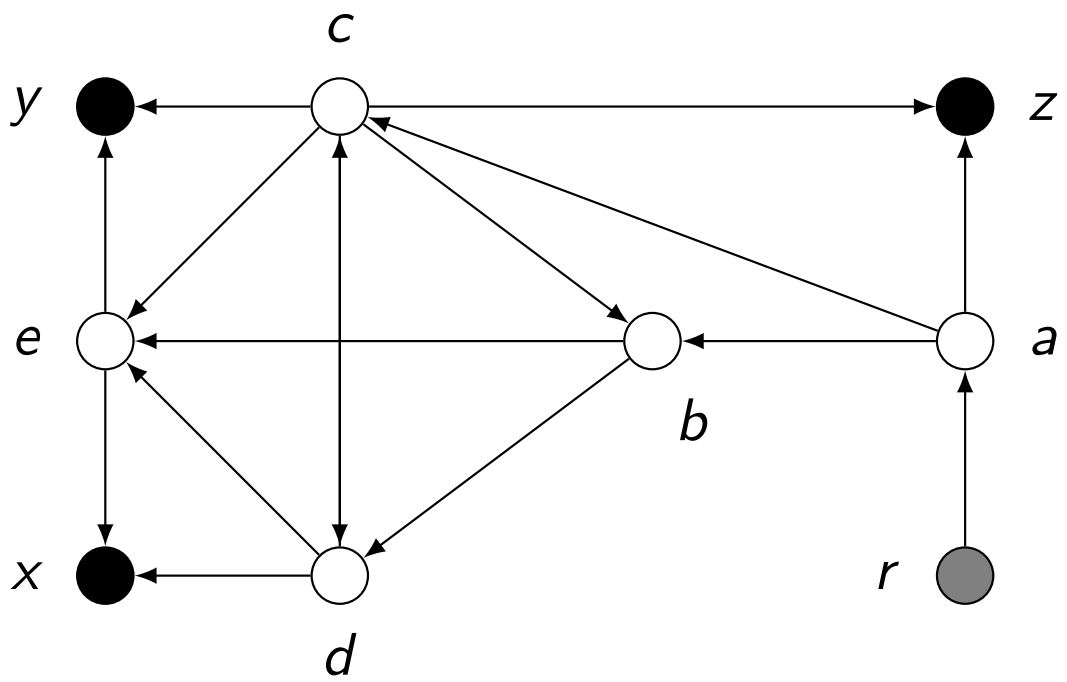
- Un graphe $G = (V, A)$ orienté ;
- une racine r ;
- des terminaux X ;
- des capacités $c : A \to \mathbb{N}$ sur les arcs ;
- des poids $\omega : A \to \mathbb{N}$ sur les arcs ;

Solution réalisable

Une arborescence de Steiner enracinée en r, couvrant X telle que, pour tout arc (u, v) de r, le sous-arbre de T enraciné en v ne contient pas plus de $c(u, v)$ terminaux.

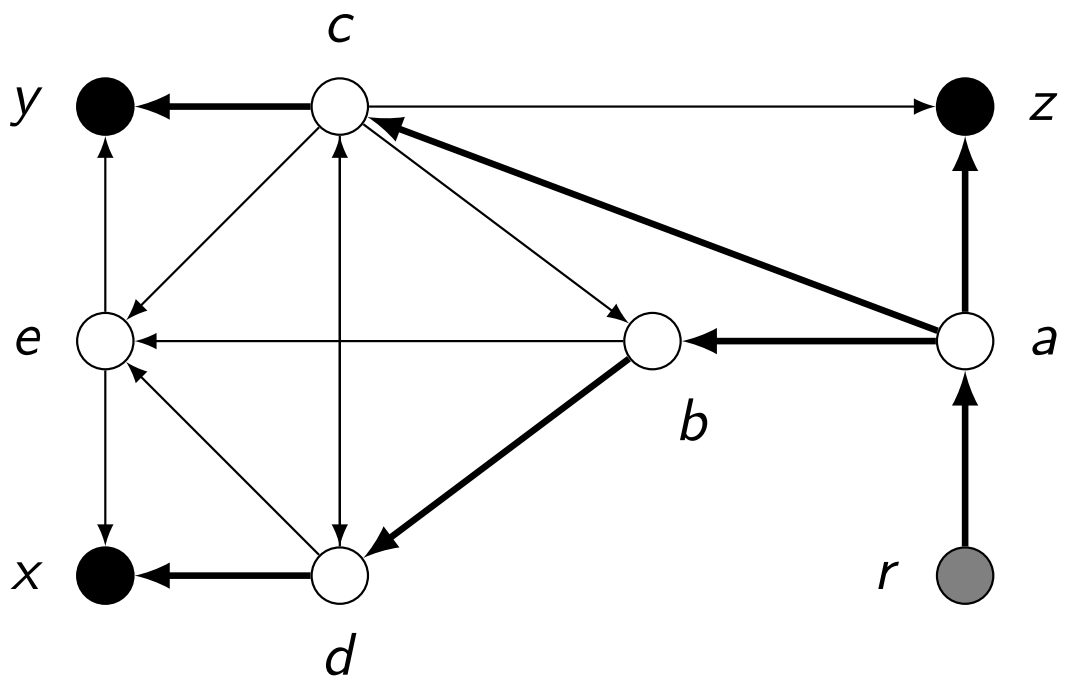
Arborescence de Steiner avec capacités

Exemple : capacité 1 partout, sauf $c(r, a) = 3$:



Arborescence de Steiner avec capacités

Exemple : capacité 1 partout, sauf $c(r, a) = 3$:



3-Partition

Instance

- des entiers $a_1, a_2, \ldots, a_{3m}$

Solution réalisable

Une partition de ces entiers en m ensembles $S_1, S_2, \ldots, S_m$ de taille 3 tels la somme des éléments de chaque ensemble est la même.

Exemple : 1 1 2 3 3 3 3 5 6 $\Rightarrow$ (1,3,5) ; (3,3,3) et (1,2,6).

Inapproximabilité de DST avec capacité

Théorème

Le problème DST avec capacité est inapproximable, même si tous les nœuds sont terminaux, même si toutes les capacités sont les mêmes.

Démonstration : GAP réduction depuis le problème 3-Partition.