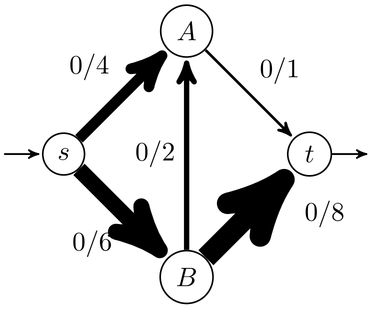
ALGO1 – Flots

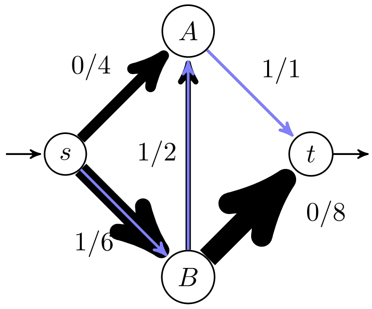
François Schwarzentruber

17 novembre 2020

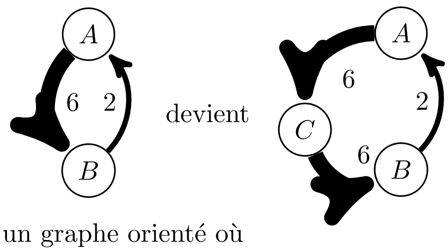
1 Définitions



Un réseau de flot est un graphe orienté, pondéré positivement par des entiers. On distingue deux sommets : une source s et un puits t. On y fait circuler de l'eau de la source s au puits t.



Pour simplifier notre discours, on suppose sans perte de généralité qu'il n'y a pas d'arcs *anti-parallèles* dans le graphe, c'est-à-dire il n'y a jamais d'arcs de A vers B et un de B vers A. On peut toujours supprimer les arcs anti-parallèles.



Définition 1 (réseau de flot) Un réseau de flot $G = (S, A, c, s, t)$ est un graphe orienté où
- (S, A) est un graphe orienté irréflexif, sans arcs anti-parallèles ;
- $c : A \to \mathbb{N}^*$;
- s est un sommet appelé source, sans arcs entrants ;
- t est un sommet appelé puits, sans arcs sortants.

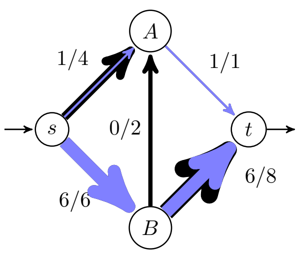
On note $c(u, v)$ la capacité de l'arête (u, v). Par convention, si l'arête (u, v) n'existe pas, $c(u, v) = 0$.

Définition 2 (flot) Un flot de $G = (S, A)$ est une fonction $f : S^2 \to \mathbb{R}^+$ telle que :
1. Pour tous sommets u, v dans S, on a $0 \leqslant f(u, v) \leqslant c(u, v)$.
2. Pour tout sommet $\underline{u}$ de S différent de s et de t, $\sum_{v \in S} f(v, \underline{u}) = \sum_{w \in S} f(\underline{u}, w)$. (loi de Kirchhoff)

Définition 3 (valeur du flot) La valeur du flot f est $|f| = \sum_{v \in S} f(s, v)$.

Définition 4 (flot maximal) Un flot f est maximal de G si la valeur $|f|$ est maximale, i.e. pour tout flot g de G, $|g| \leq |f|$.

Exemple 5 Voici un flot maximum de valeur 7 :

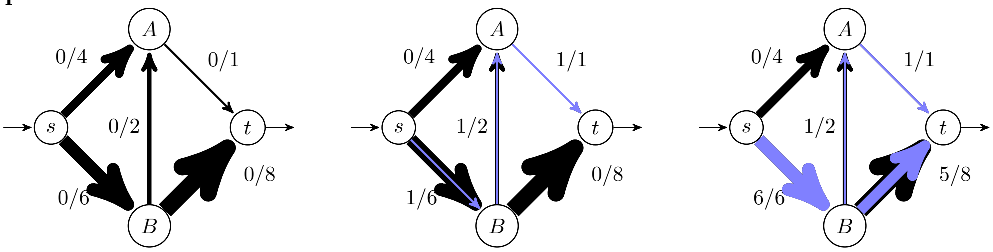


Définition 6 (problème du flot maximum)
- entrée : un réseau G ;
- sortie : un flot maximal de G.

2 Échec d'une méthode gloutonne

pre : Un réseau G
post : l'algorithme ne garantit pas d'avoir un flot f maximal
fonction algoMalFichu(G)
    $f :=$ flot nul
    tant que il existe un chemin simple de s à t qui améliore le flot **faire**
        augmenter f à l'aide de ce chemin
    retourner f

Exemple 7



On obtient un flot de 6 et là on est bloqué, jamais on ne trouvera un flot de 7.

3 Réseaux résiduels

Comme le montre l'exemple précédent, on voudrait annuler les mauvais choix.

Exemple 8 On voudrait annuler l'eau passant par l'arrête $B \to A$ et la rediriger de B à t.

On introduit le graphe résiduel qui indique les modifications possibles du flot courant :
- Ajout d'eau ;
- Annulation d'eau.

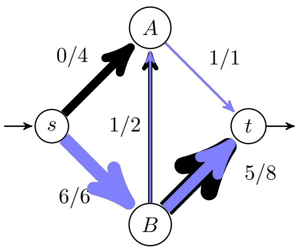
Définition 9 (graphe résiduel) Le graphe résiduel d'un réseau $G = (S, A, c, s, t)$ et d'un flot f est le graphe pondéré $R_f = (S, A_f, r_f)$ où les sommets sont S, la fonction de poids r_f est :

$$r_f(u,v) = \begin{cases} c(u,v) - f(u,v) \text{ si } (u,v) \in A \\ f(v,u) \text{ si } (v,u) \in A \\ 0 \text{ sinon} \end{cases}$$

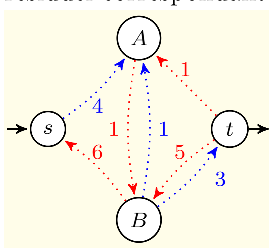
et $A_f = \{(u,v) \in S \times S \mid r_f(u,v) > 0\}$.

Exemple 10

Flot dans le réseau



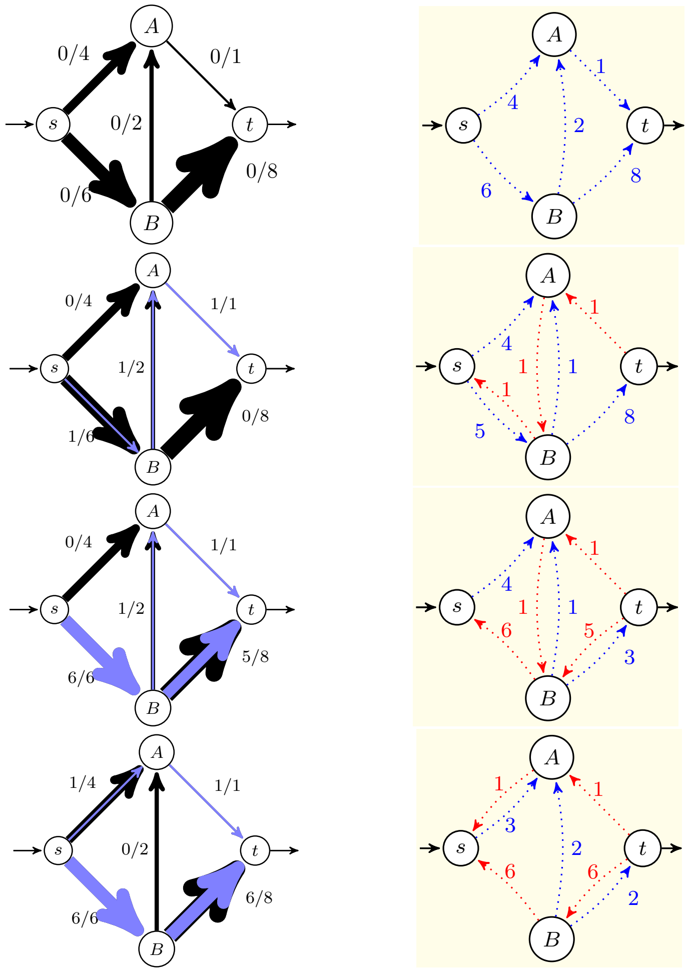
Graphe résiduel correspondant



4 Algorithme de Ford-Fulkerson

pre : Un réseau G
post : Un flot f maximal
fonction fordFulkerson(G)
  $f :=$ flot nul
  tant que il y a un chemin simple γ de s à t dans le graphe résiduel R_f **faire**
    $\Delta = \min\{r_f(u,v) \mid \text{l'arc } u \to v \text{ dans le chemin } \gamma\}$
    pour tout arc $u \to v$ dans γ **faire**
      si $u \to v \in A$ **alors**
        $f(u,v) := f(u,v) + \Delta$
      sinon
        $f(v,u) := f(v,u) - \Delta$
  retourner f

Exemple 11



Théorème 12 On a l'invariant : « f est un flot ».

Théorème 13 La valeur du flot $|f|$ augmente strictement à chaque itération.

Démonstration.
$f \xrightarrow{\text{itération}} f'$. Soit x le sucesseur de s dans le chemin γ. On a :

$$|f'| = \sum_{v \in S} f'(s,v) = \sum_{v \in S, v \neq x} f(s,v) + f'(s,x) = \sum_{v \in S, v \neq x} f(s,v) + f(s,x) + \Delta = |f| + \Delta.$$

■

Définition 14 (flot à valeurs entières) Un flot f est à valeurs entières si $f(u, v)$ est un entier pour tout u, v.

Définition 15 (problème du flot maximum à valeurs entières)
- entrée : un réseau G où les capacités sont des entiers ;
- sortie : un flot maximal de G à valeurs entières.

Théorème 16 Si les capacités sont des entiers, l'algorithme termine.

DÉMONSTRATION. Par l'absurde, supposons que l'algorithme ne termine pas. Les flots calculés par l'algorithme sont tous à valeurs entières. Considérons la suite des valeurs prises par $|f|$.
- Elle est strictement croissante ;
- Elle est majorée par $|f^*|$ où f^* est un flot maximal ;
- Elle est à valeur entière.

C'est une suite infinie d'entiers strictement croissante et majorée. Contradiction. ■

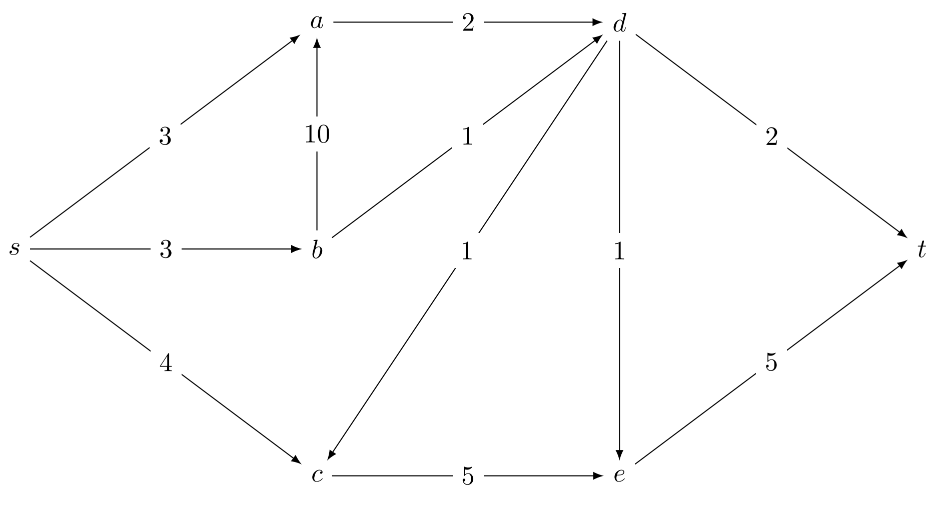
Proposition 17 Si les capacités sont à valeurs entières, alors l'algorithme retourne un flot à valeurs entières.

Proposition 18 L'algorithme de Ford-Fulkerson est en $O(|f^*|(S + A))$ où f^* est un flot maximal.

Définition 19 (algorithme de Edmonds-Karp) L'algorithme de EDMONDS-KARP est le même algorithme mais en choisissant à chaque fois un plus court chemin en nombre d'arcs dans G_f (parcours en largeur).

Théorème 20 (cf. TD) L'algorithme de Edmonds-Karp résout le problème du flot maximum en $O(SA^2)$.

Exercice 1 Appliquer l'algorithme de Ford-Fulkerson sur le réseau suivant.



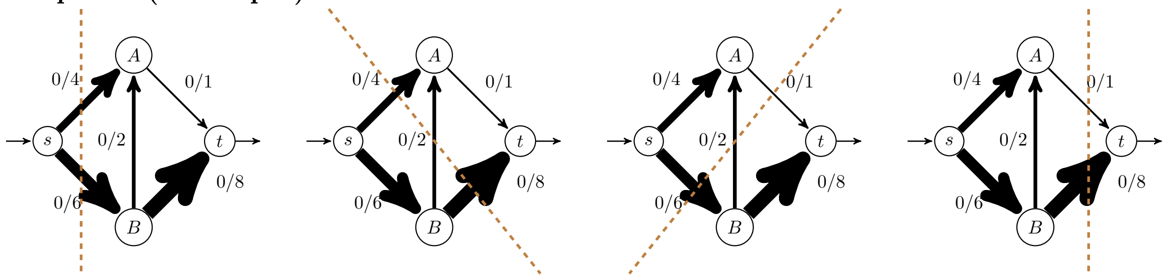
5 Correction de Ford-Fulkerson

Pour démontrer la correction de Ford-Fulkerson, on introduit un certificat d'optimalité : une coupe minimale.

5.1 Coupes

Définition 21 (s-t-coupe) Une s-d-coupe (E, T) d'un réseau est une partition de S avec $s \in E$ et $t \in T$.

Exemple 22 (s-t-coupes)



Définition 23 (capacité d'une coupe) On note

$$c(E, T) = \sum_{(u,v) \in E \times T} c(u, v).$$

Exemple 24 Les capacité des coupes ci-dessus sont

4+6=10 4+2+8=14 6+1=7 1+8=9

Théorème 25 (principe de dualité) Valeur d'un flot maximal = capacité d'une coupe minimale.

DÉMONSTRATION. $\leq$ par le corollaire 28, et $\geq$ par le théorème 29. ∎

5.2 Flot maximal $\leq$ Coupe minimale

Définition 26 Soit f un flot. Soit A et B des ensembles de sommets. On note[1]

$$f(A, B) = \sum_{(u,v) \in A \times B} f(u, v).$$

Lemme 27 Soit f un flot. Soit (E, T) une coupe. On a $|f| = f(E, T) - f(T, E)$.

DÉMONSTRATION.

$$\begin{aligned}
|f| &= f(s, S) \\
&= f(s, S) - \underbrace{f(S, s)}_{\text{0 car } s \text{ pas d'arcs entrants}} + \textstyle\sum_{u \in E, u \neq s} \underbrace{f(u, S) - f(S, u)}_{\text{0 par Kirchhoff}} \\
&= \textstyle\sum_{u \in E} f(u, S) - f(S, u) \\
&= f(E, S) - f(S, E) \\
&= f(E, T) + f(E, E) - f(E, E) - f(T, E) \\
&= f(E, T) - f(T, E).
\end{aligned}$$

∎

Corollaire 28 Pour tout flot f, pour toute coupe (E, T), on a $|f| \leq c(E, T)$.

DÉMONSTRATION. Le lemme 27 donne $|f| = f(E, T) - f(T, E)$. Donc $|f| \leq f(E, T) \leq c(E, T)$. ∎

1. Attention, la définition dans Cormen $f(E, T) = \sum_{(u,v) \in E \times T} f(u, v) - \sum_{(u,v) \in E \times T} f(v, u)$ n'est pas symétrique, alors qu'il est plus naturel de définir $f(E, T) = \sum_{(u,v) \in E \times T} f(u, v)$... Dans le livre de Kleinberg et Tardos, c'est le cas, même si c'est lourd avec les f^{out}, dû à leur notation $f(e)$ où e est un arc, où il est difficile de regarder les couples (u, v).

5.3 Flot maximal $\geq$ Coupe minimale

Théorème 29 Soit G un réseau et f un flot. On a équivalence entre :

1. Le flot f est maximal
2. Il n'existe pas de chemin de s à t dans le graphe résiduel R_f.
3. Il existe une coupe (E, T) tel que $|f| = c(E, T)$.

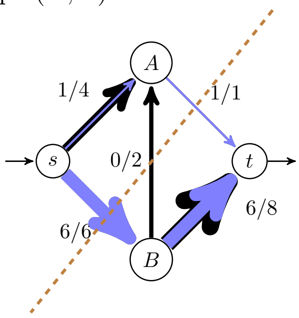
Démonstration.
$\boxed{1 \Rightarrow 2}$ S'il y avait un tel chemin, une itération de Ford-Fulkerson améliorerait le flot.

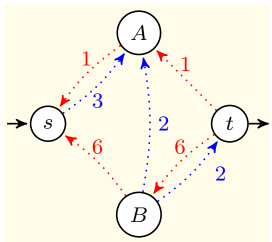
$\boxed{3 \Rightarrow 1}$ Cela signifie que l'on a égalité dans le corollaire 28 et donc f maximal.

$\boxed{2 \Rightarrow 3}$ Supposons qu'il n'y a pas de chemin améliorant. Définissons la coupe (E, T) candidate suivante avec

$$E = \{u \in S, \text{ il existe un chemin de } s \text{ à } u \text{ dans } R_f\}$$

et $T = S \setminus E$. Comme il n'existe pas de chemin améliorant, par définition de E, $t \notin E$ donc $t \in T$. Nous avons bien une coupe (E, T).





Montrons que $|f| = c(E, T)$. Le tableau suivant explique le calcul de $|f|$ en s'appuyant sur la formule $|f| = f(E, T) - f(T, E)$ donnée dans le lemme 27. On considère $u \in E$ et $v \in T$. Par définition de E, on a $u \to v \notin R_f$. Sinon, on aurait un chemin de s à v en passant par u dans R_f et donc $v \in E$. Contradiction.

	$f(u,v)$	$f(v,u)$
u v — $u \to v \notin G$ et $v \to u \notin G$	$0 = c(u,v)$	0
$u \to v$ — $u \to v \in G$ et $v \to u \notin G$	$c(u,v)$ car capacité $u \to v$ pleine	0
$u \leftarrow v$ — $v \to u \in G$ et $u \to v \notin G$	$0 = c(u,v)$	0 car rien à annuler pour $v \to u$

Donc $|f| = f(E, T) - f(T, E) = \sum_{(u,v) \in E \times T} c(u, v) \ - \ 0 = c(E, T)$.

■

Corollaire 30 Ford-Fulkerson est correct.

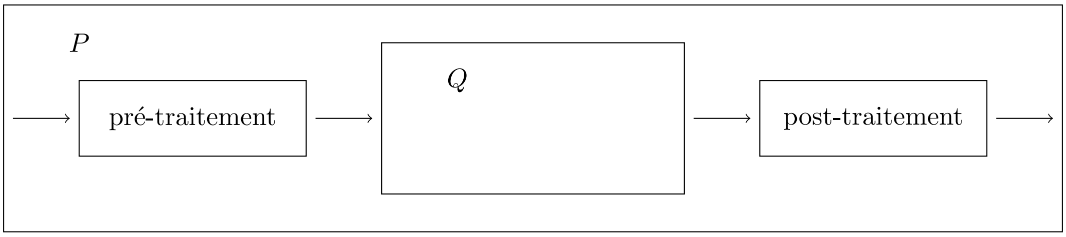
Démonstration. Quand Ford-Fulkerson s'arrête, il n'existe plus de chemin de s à t dans le graphe résiduel. D'après le théorème précédent, f est donc un flot maximal dans G. ■

6 Réduction du problème du couplage maximum biparti vers flots

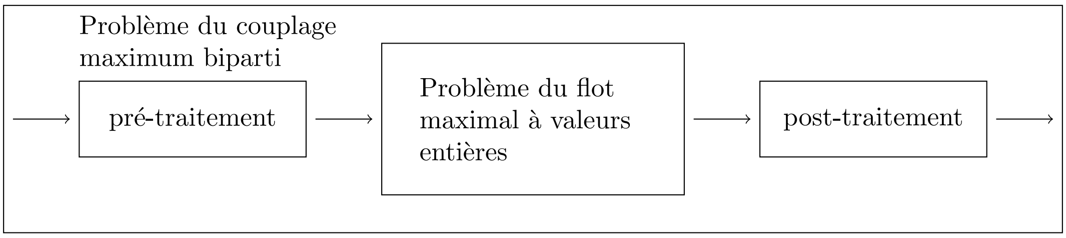
Définition 31 (réduction avec pré- et post-traitements) Soit P, Q deux problèmes algorithmiques. Une réduction de P à Q est la donnée :

— d'un algorithme de pré-traitement qui transforme une instance de P en une instance Q ;
— d'un algorithme de post-traitement qui transforme une solution de Q en une solution de P

telle que la boîte suivante donne un algorithme correct pour P :

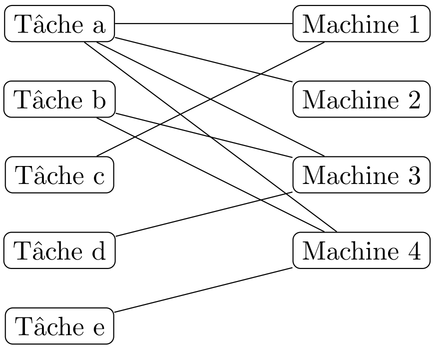


Dans cette section, nous allons réduire le problème du couplage maximum biparti dans le problème du flot maximal à valeurs entières.



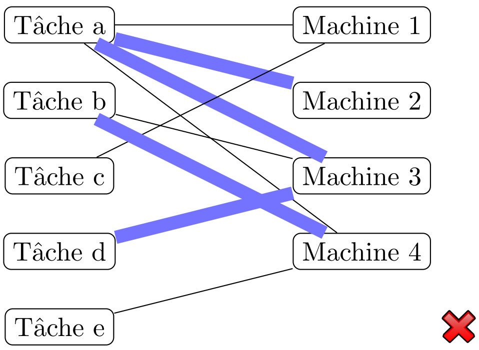
Définition 32 Un graphe non orienté $G = (S, A)$ est biparti s'il existe une partition $S = S_1 \sqcup S_2$ telle que $A \subseteq S_1 \times S_2$.

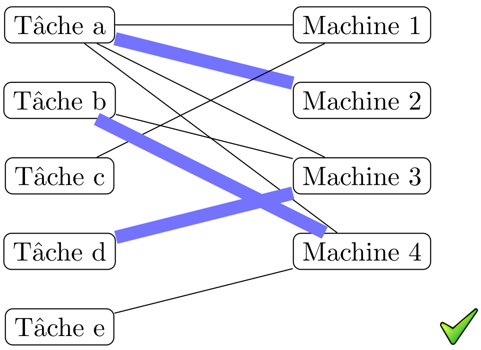
Exemple 33



Définition 34 (couplage) Un couplage dans un graphe $G = (S, A)$ est un ensemble M tel que tout sommet apparaît au plus dans un couple de M.

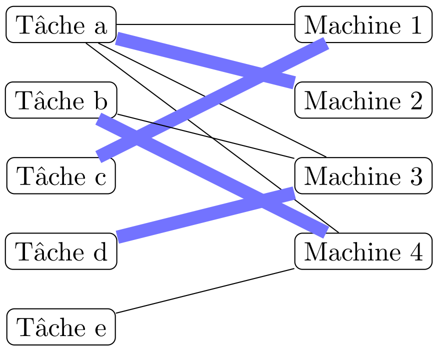
Exemple 35





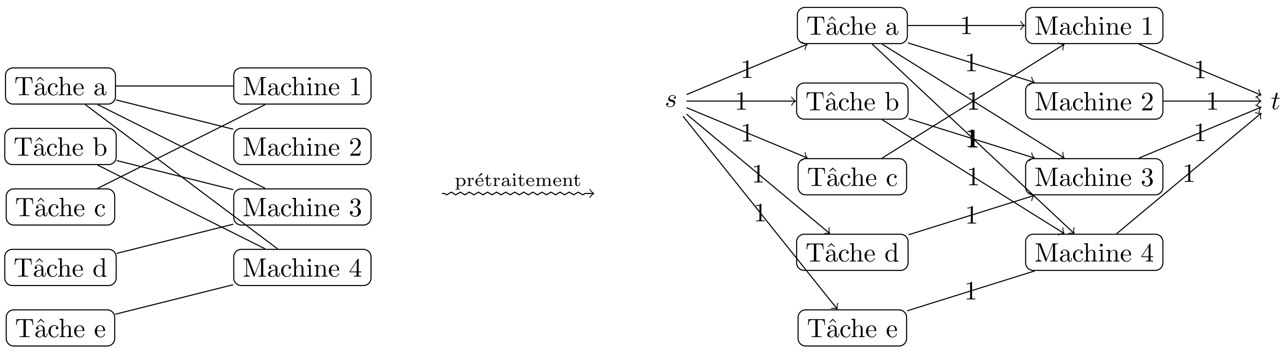
Définition 36 (couplage maximum) Un couplage M est maximum dans G si $|M|$ est maximal, i.e. pour tout couplage M' dans G on a $|M'| \leq |M|$.

Exemple 37 (couplage maximum) Un ensemble M d'arêtes qui répond au problème est représenté en gras :

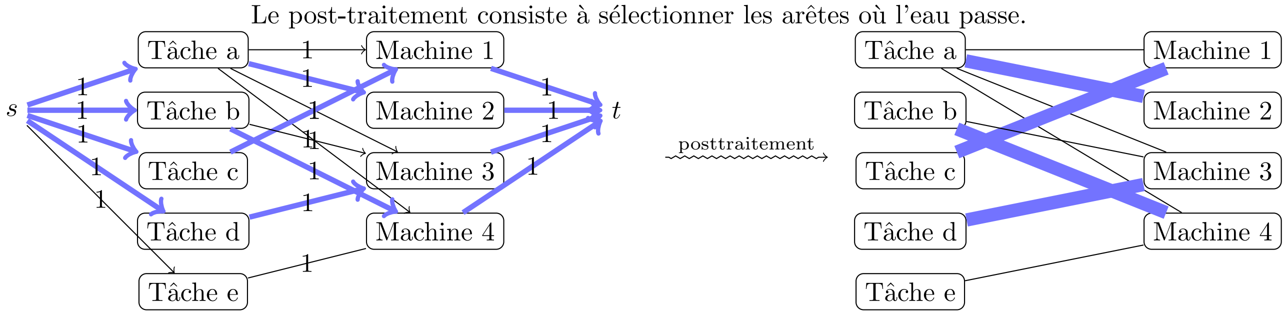


Définition 38 (problème du couplage maximum dans un graphe biparti)
- Entrée : un graphe biparti G ;
- Sortie : un couplage maximum M de G.

Réduction du problème du couplage maximal dans un graphe biparti au problème du flot maximum à valeurs entières
Le prétraitement consiste à ajouter une source s que l'on connecte à tous les sommets de S_1, et à ajouter une destination t à laquelle on connecte tous les sommets de S_2.



Le post-traitement consiste à sélectionner les arêtes où l'eau passe.



Définition 39 (prétraitement dans la réduction) La fonction de prétraitement *pretr* est définie par $pretr((S_1 \sqcup S_2, A)) = (S', A', c, s, t)$ avec :
- $S' = S_1 \sqcup S_2 \sqcup \{s, t\}$;
- $A' = A \cup \{(s, v) \mid v \in S_1\} \cup \{(u, t) \mid u \in S_2\}$;
- $c(u, v) = 1$ pour tout $(u, v) \in A'$.

Définition 40 (posttraitement dans la réduction) La fonction de prétraitement *posttr* est définie par

$$posttr(f) = \{(u, v) \in S_1 \times S_2 \mid f(u, v) = 1\}.$$

Proposition 41 Étant donné un flot f maximal à valeurs entières dans $pretr(G)$. L'ensemble d'arêtes $posttr(f)$ est un couplage maximum dans G.

Notes bibliographiques

Ce cours est inspiré de [CLRS09] et de [KT06]. Il y a une littérature énorme sur les flots, même un livre dédié au sujet : [AMO93]. Ford et Fulkerson est à l'origine du problèmes du flot maximum et du couplage biparti [FF62]. De manière indépendante, Edmonds et Karp [EK72], mais aussi Dinic [Din70], ont montré que l'algorithme est polynomial si on utilise le parcours en largeur pour trouver un chemin améliorant dans le graphe résiduel. Il existe des algorithmes plus performants, basés sur d'autres idées comme les flots bloquants et la méthode pousser-réétiqueter (voir [CLRS09]).

Le calcul du flot maximum est utilisé en pratique en bioinformatique [PPA+15] pour de la reconstruction de données à partir de brins d'ARN.

Références

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows - theory, algorithms and applications*. Prentice Hall, 1993.

[CLRS09] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[Din70] Efim A Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. In *Soviet Math. Doklady*, volume 11, pages 1277–1280, 1970.

[EK72] Jack R. Edmonds and Richard M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2) :248–264, 1972.

[FF62] DR Fulkerson and LR Ford. *Flows in networks*. Princeton University Press, 1962.

[KT06] Jon Kleinberg and Eva Tardos. *Algorithm design*. Pearson Education India, 2006.

[PPA+15] Mihaela Pertea, Geo M Pertea, Corina M Antonescu, Tsung-Cheng Chang, Joshua T Mendell, and Steven L Salzberg. Stringtie enables improved reconstruction of a transcriptome from rna-seq reads. *Nature biotechnology*, 33(3) :290–295, 2015.