Réseaux de flot

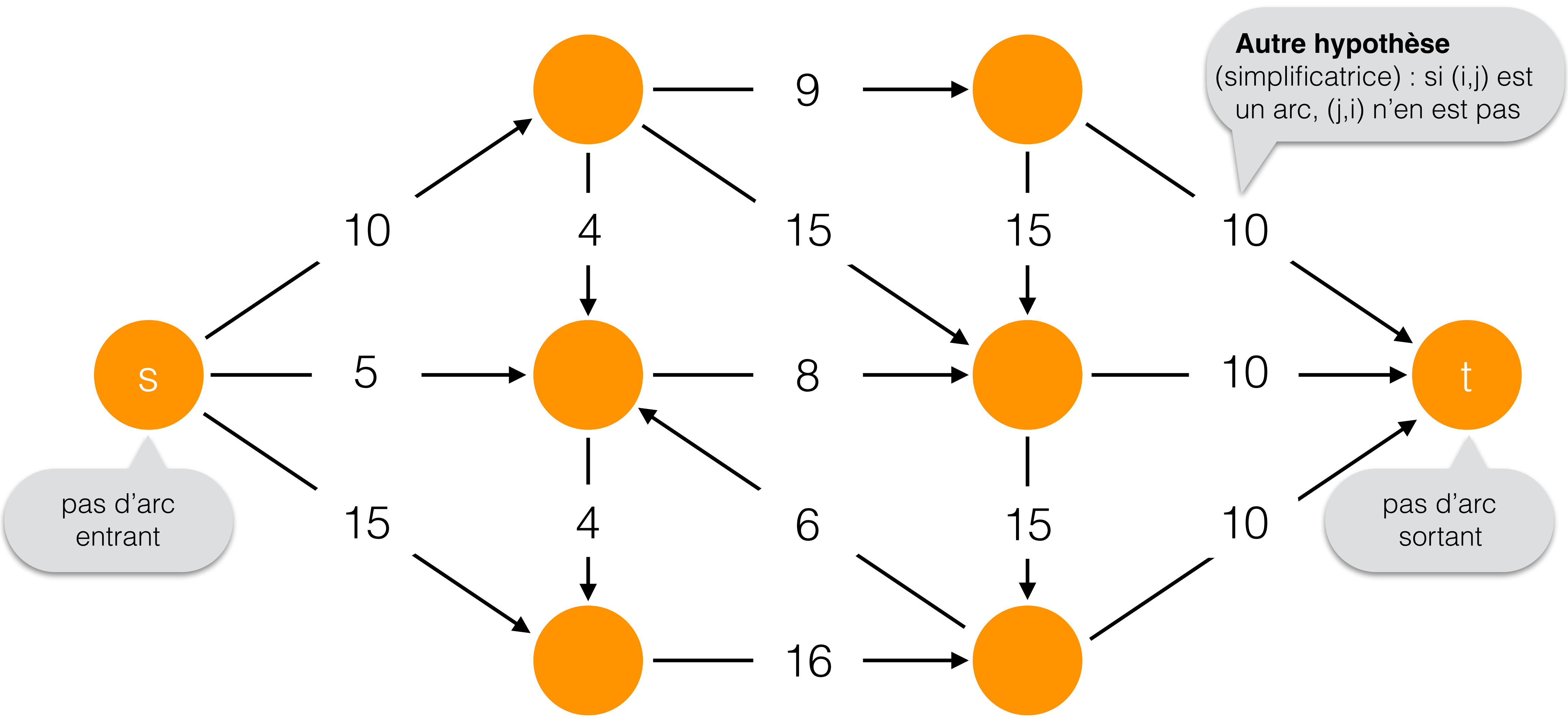
8 novembre 2016

David Pichardie

Le problème du flot maximum

Entrée

- un graphe pondéré (par une *capacité* notée *c*)
- poids strictement positifs
- un sommet *source* s et un sommet *cible* t

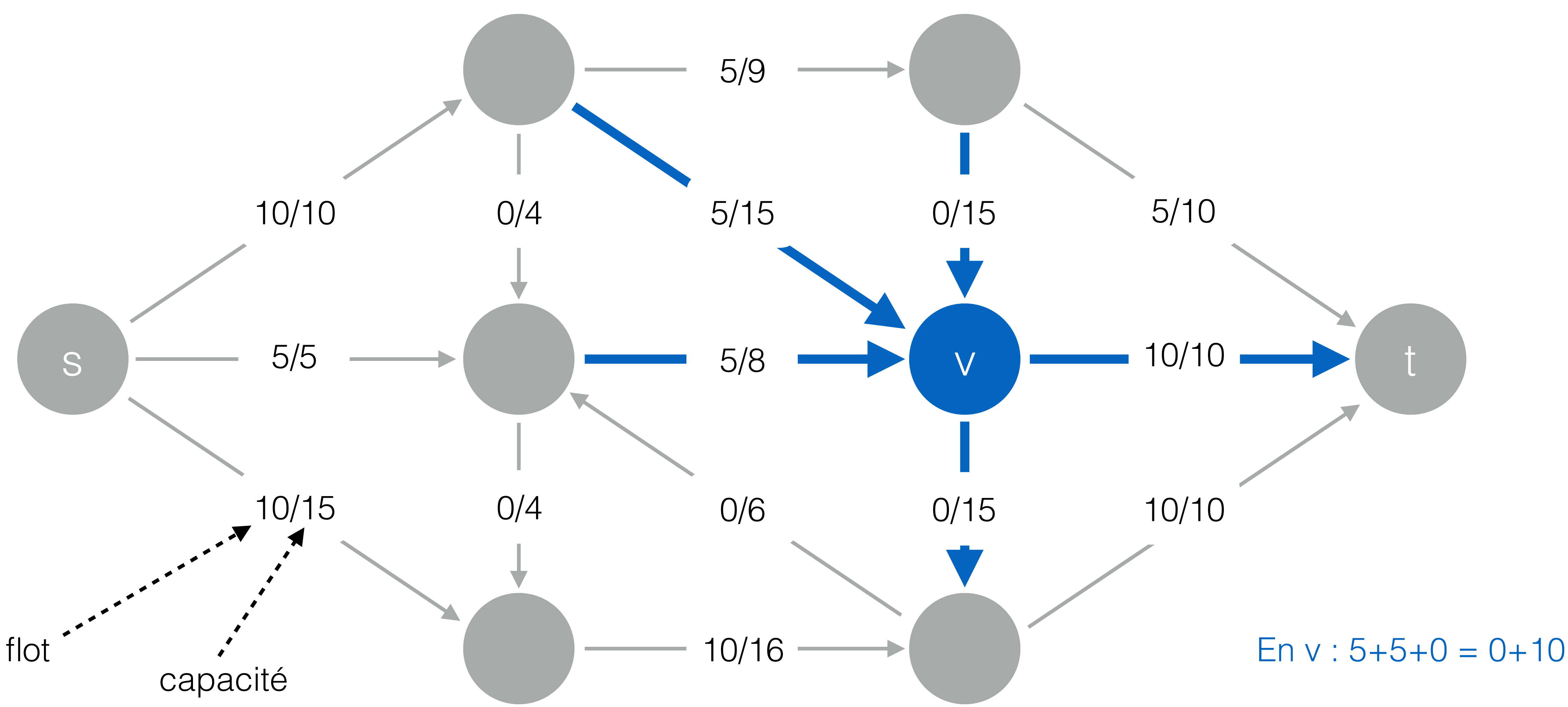


Le problème du flot maximum

Définition. Un *flot* est une pondération (notée *f*) des arcs telle que

- pour chaque arc *e*, $0 \leq f(e) \leq c(e)$
- en chaque sommet,
  somme des poids des arcs entrants = somme des poids des arcs sortant

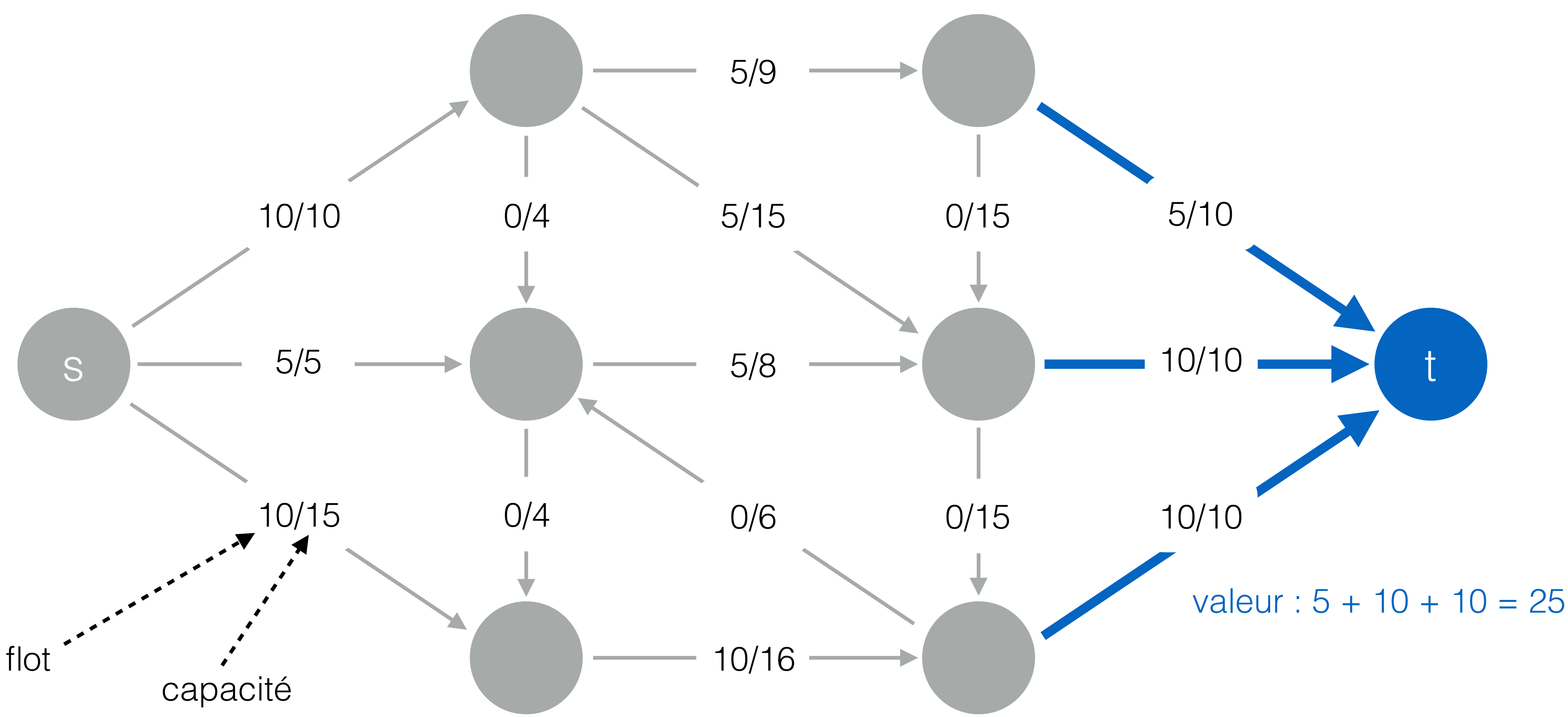
équilibre local



Le problème du flot maximum

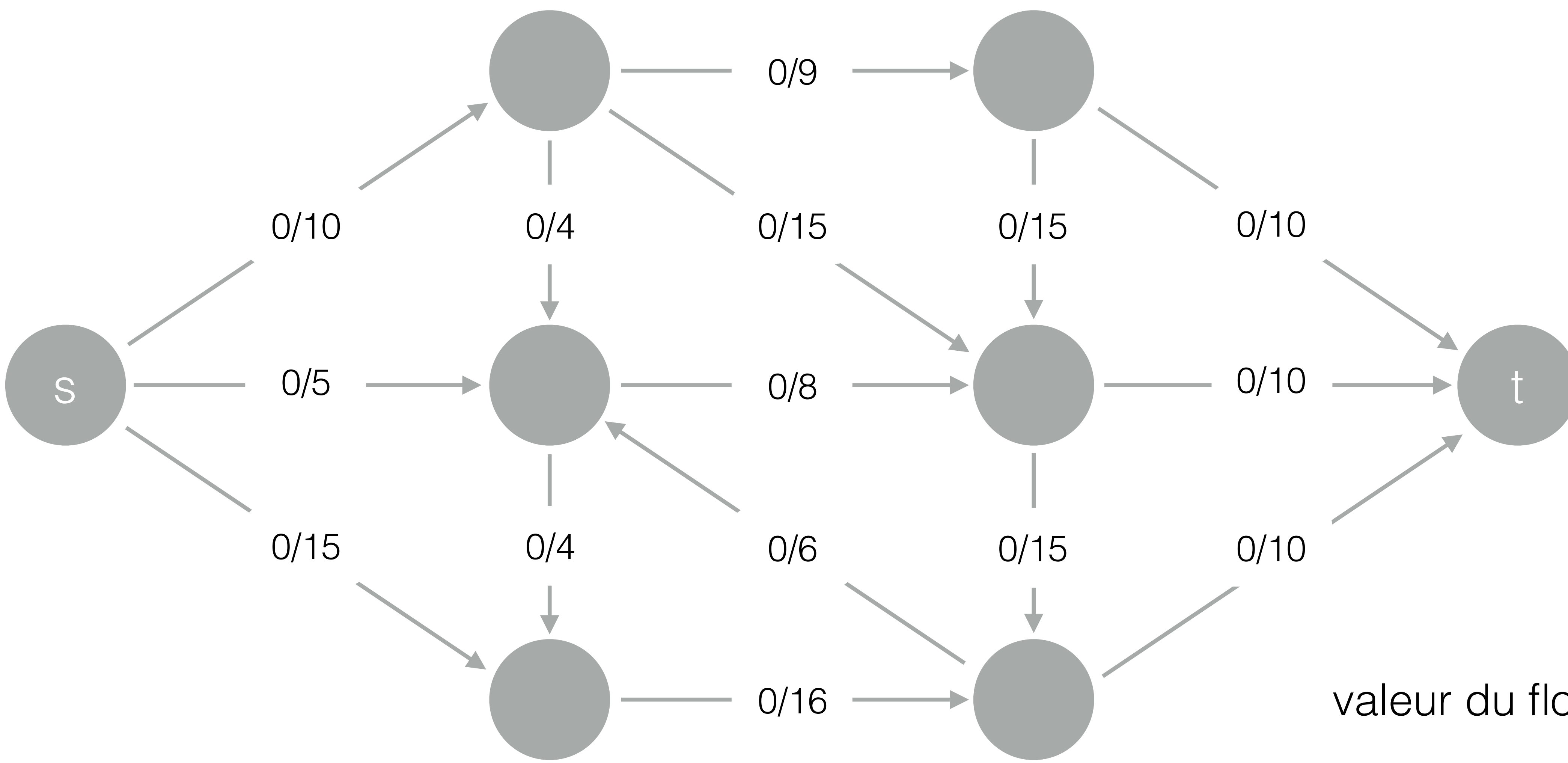
Définition. La *valeur* du flot est la somme des flots entrants dans le sommet cible

Problème du flot maximum : trouver un flot de valeur maximum



Algorithme de Ford-Fulkerson

Initialisation. Au départ, un flot nul.



valeur du flot = 0

Algorithme de Ford-Fulkerson

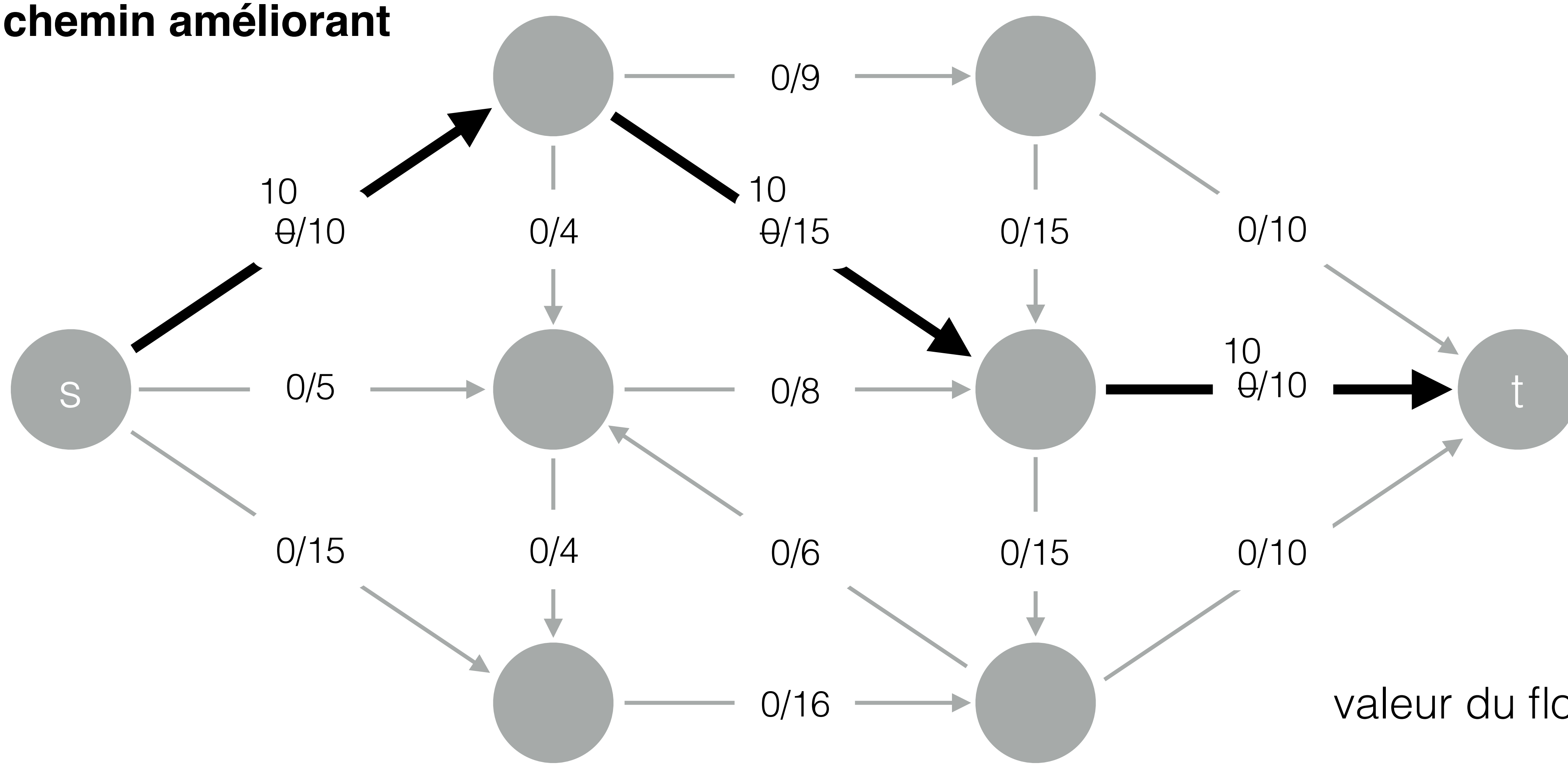
Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

sans dépasser la capacité de l'arc

sans rendre le flot négatif sur cet arc

1er chemin améliorant



valeur du flot = 10

Algorithme de Ford-Fulkerson

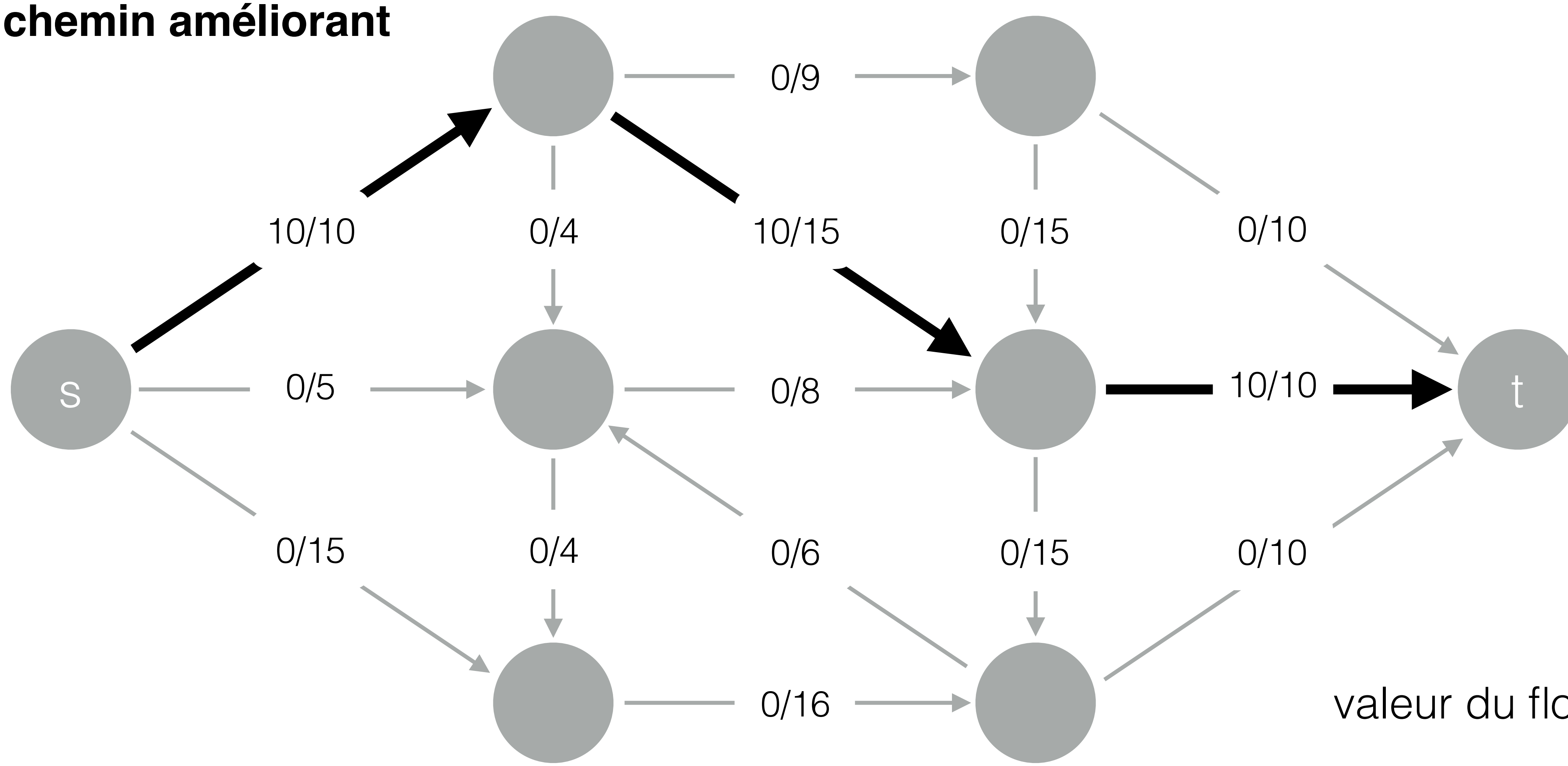
Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

sans dépasser la capacité de l'arc

sans rendre le flot négatif sur cet arc

1er chemin améliorant



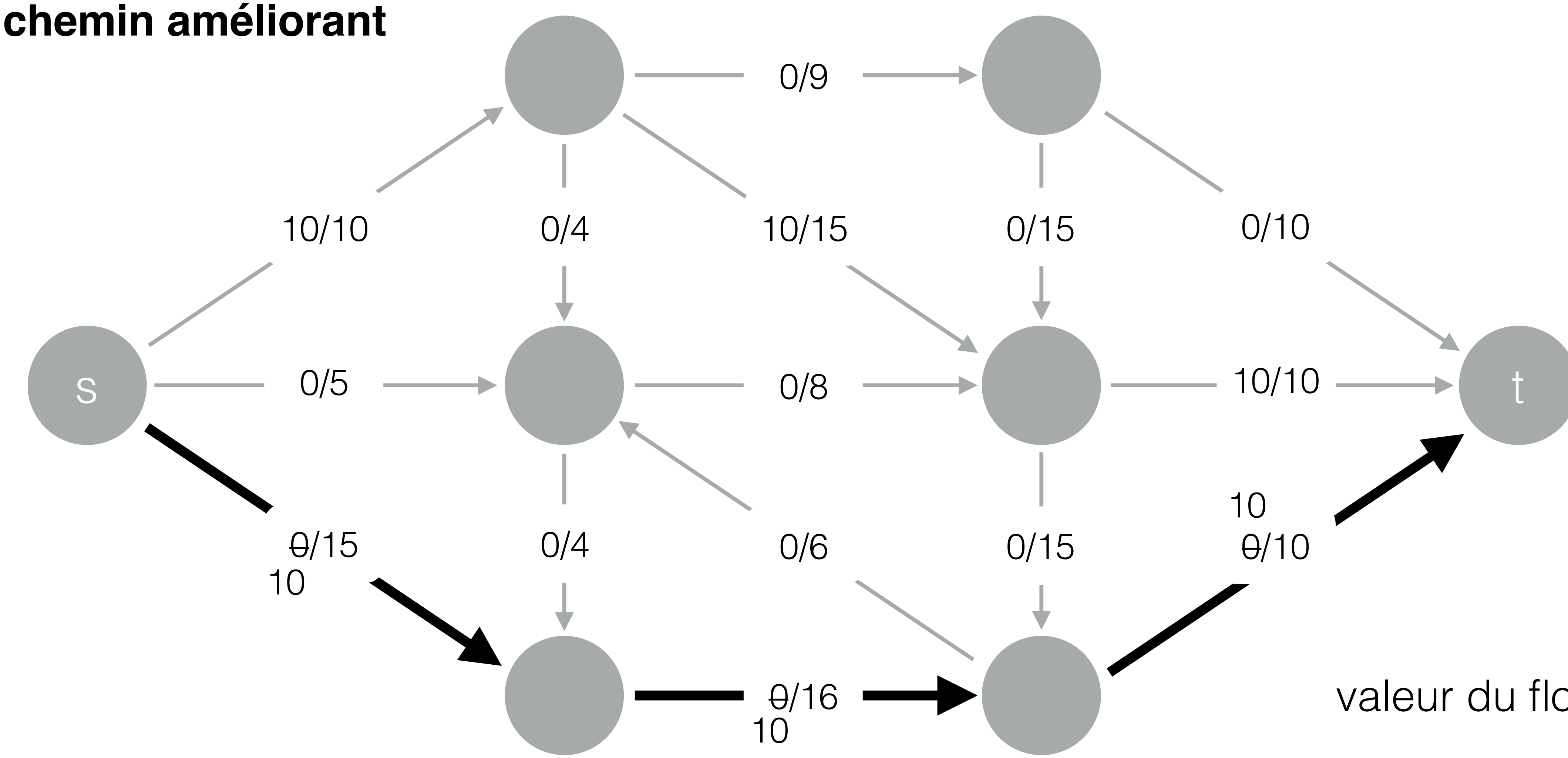
valeur du flot = 10

Algorithme de Ford-Fulkerson

Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

2e chemin améliorant



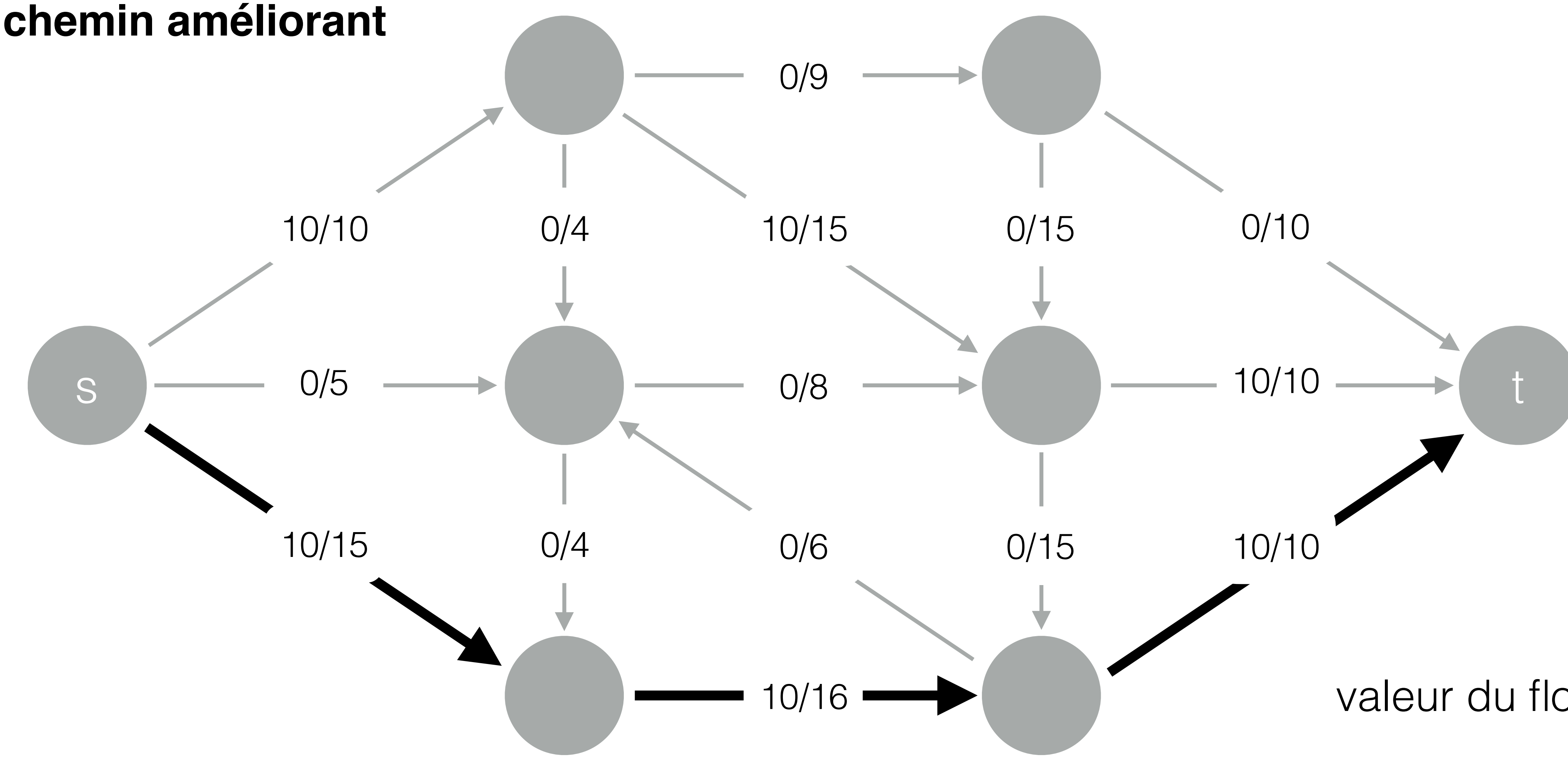
valeur du flot = 10+10

Algorithme de Ford-Fulkerson

Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

2e chemin améliorant



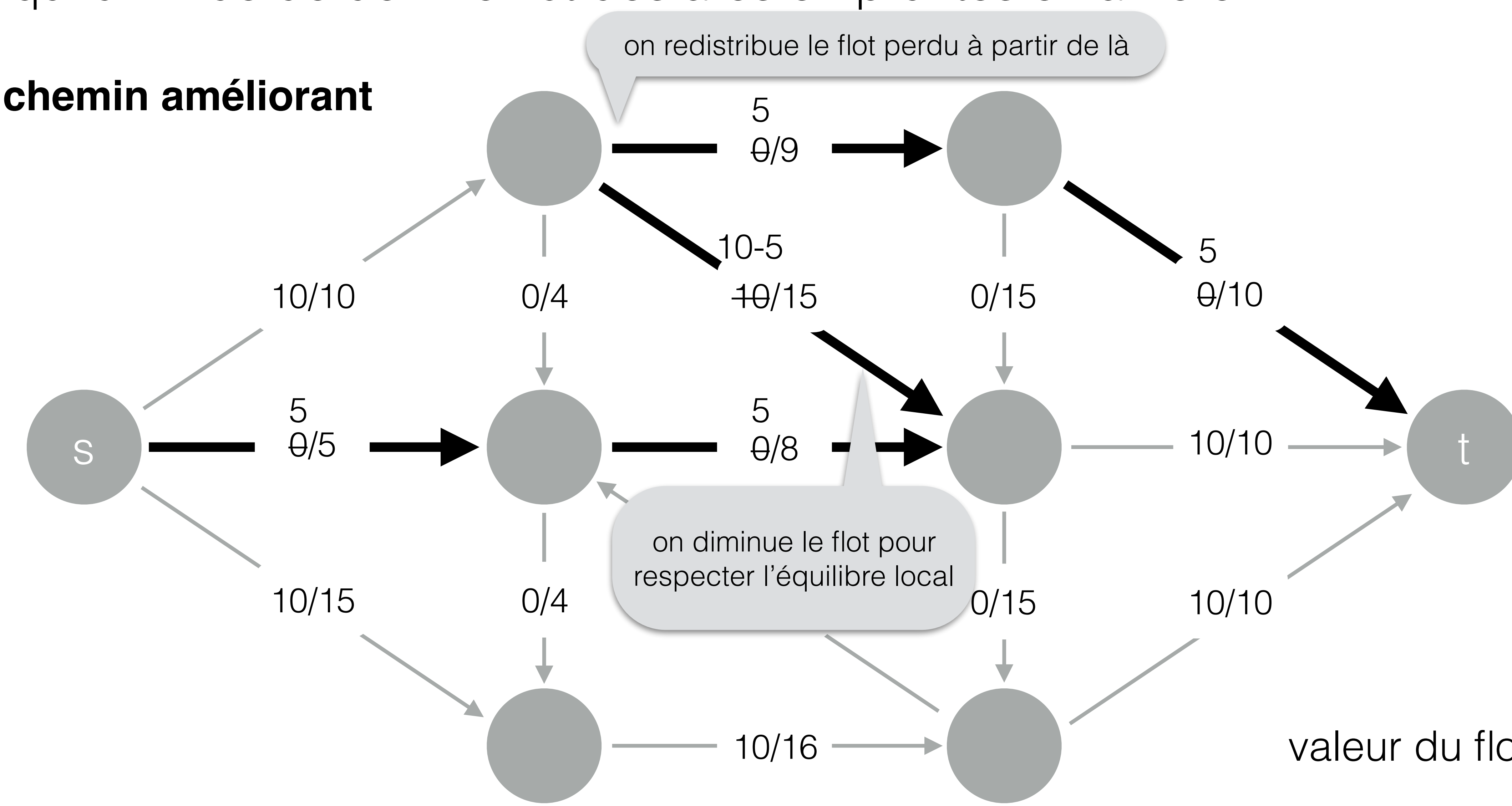
valeur du flot = 10+10

Algorithme de Ford-Fulkerson

Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

3e chemin améliorant



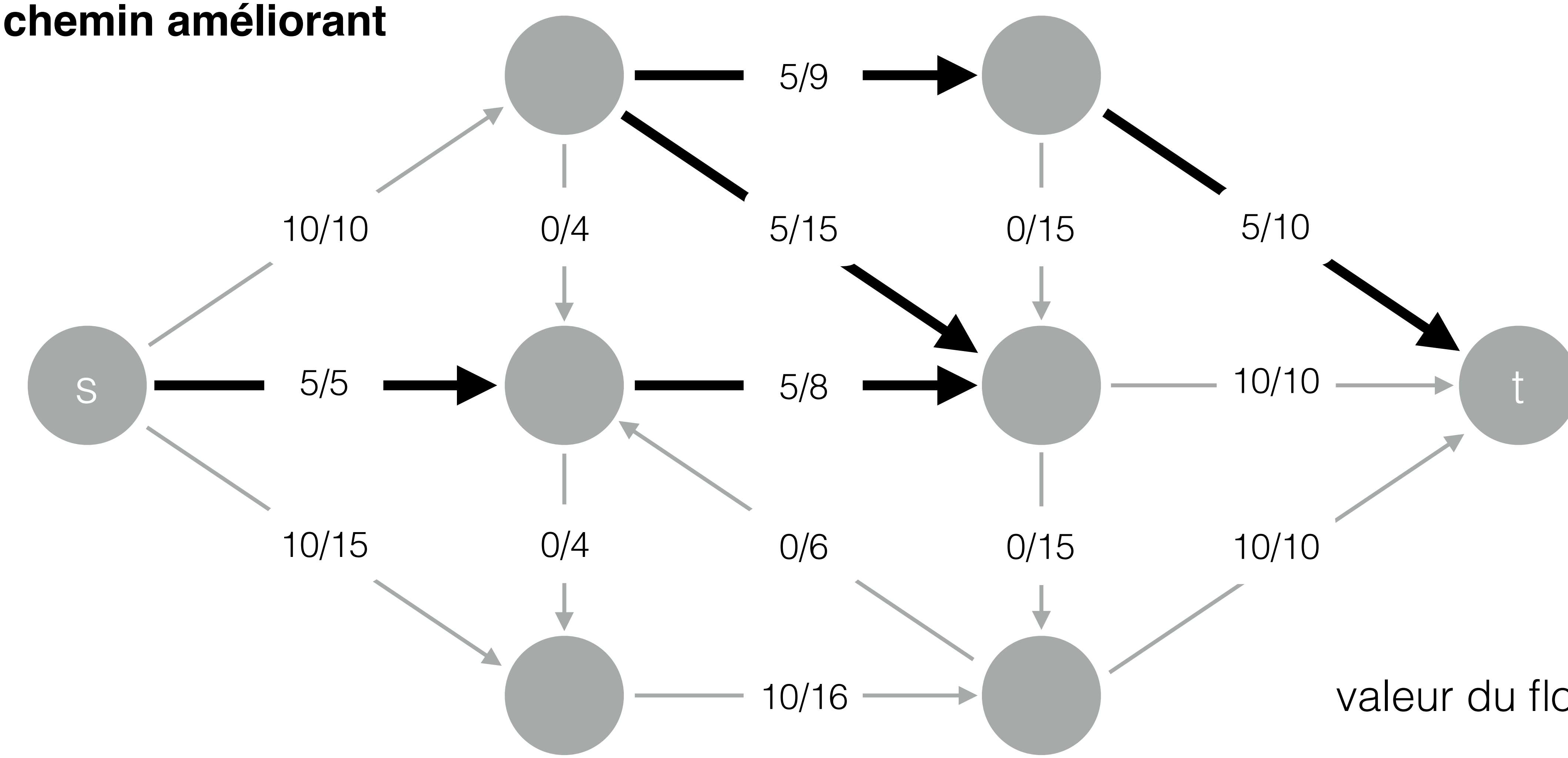
valeur du flot = 20+5

Algorithme de Ford-Fulkerson

Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

3e chemin améliorant



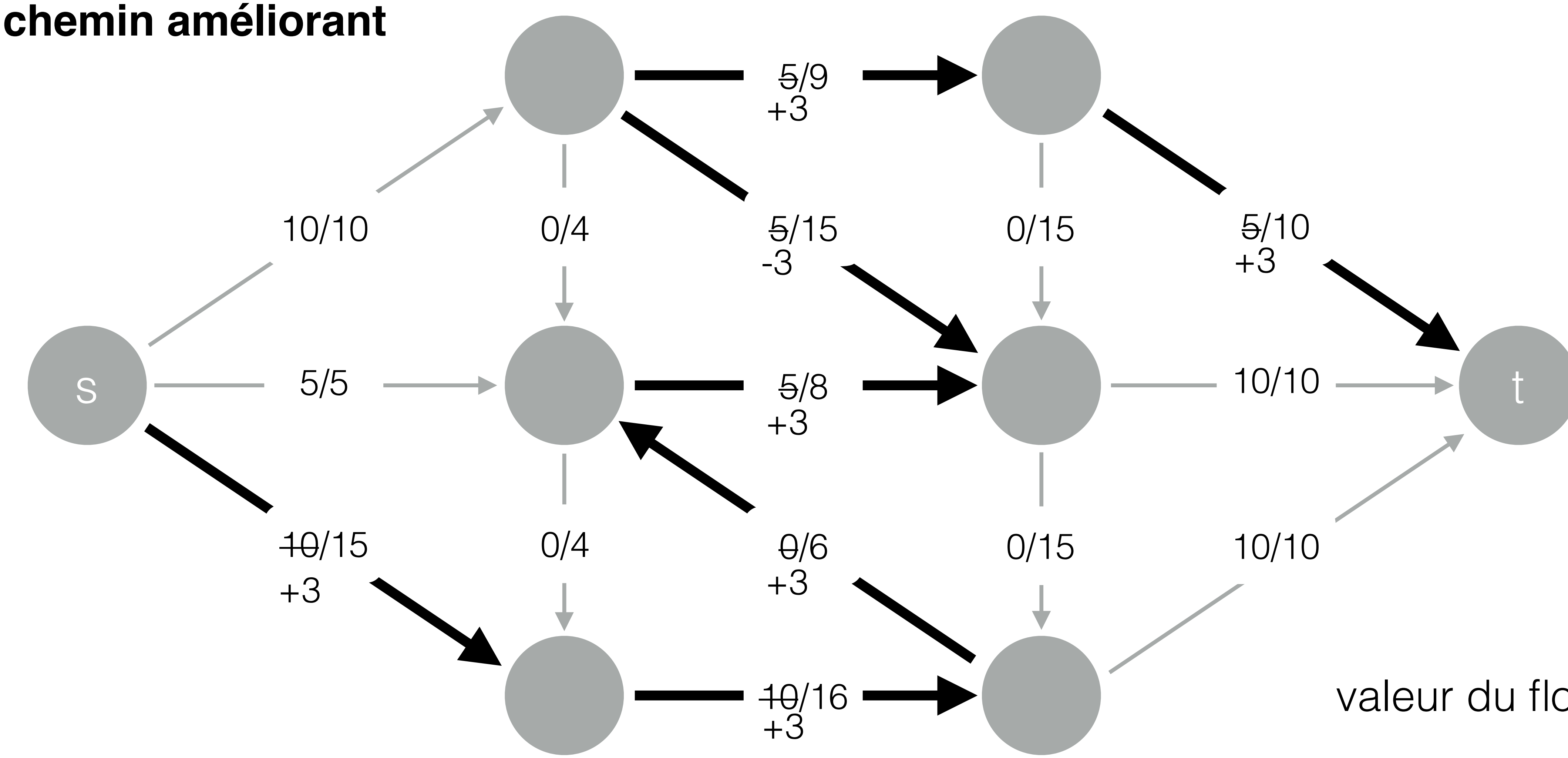
valeur du flot = 20+5

Algorithme de Ford-Fulkerson

Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

4e chemin améliorant



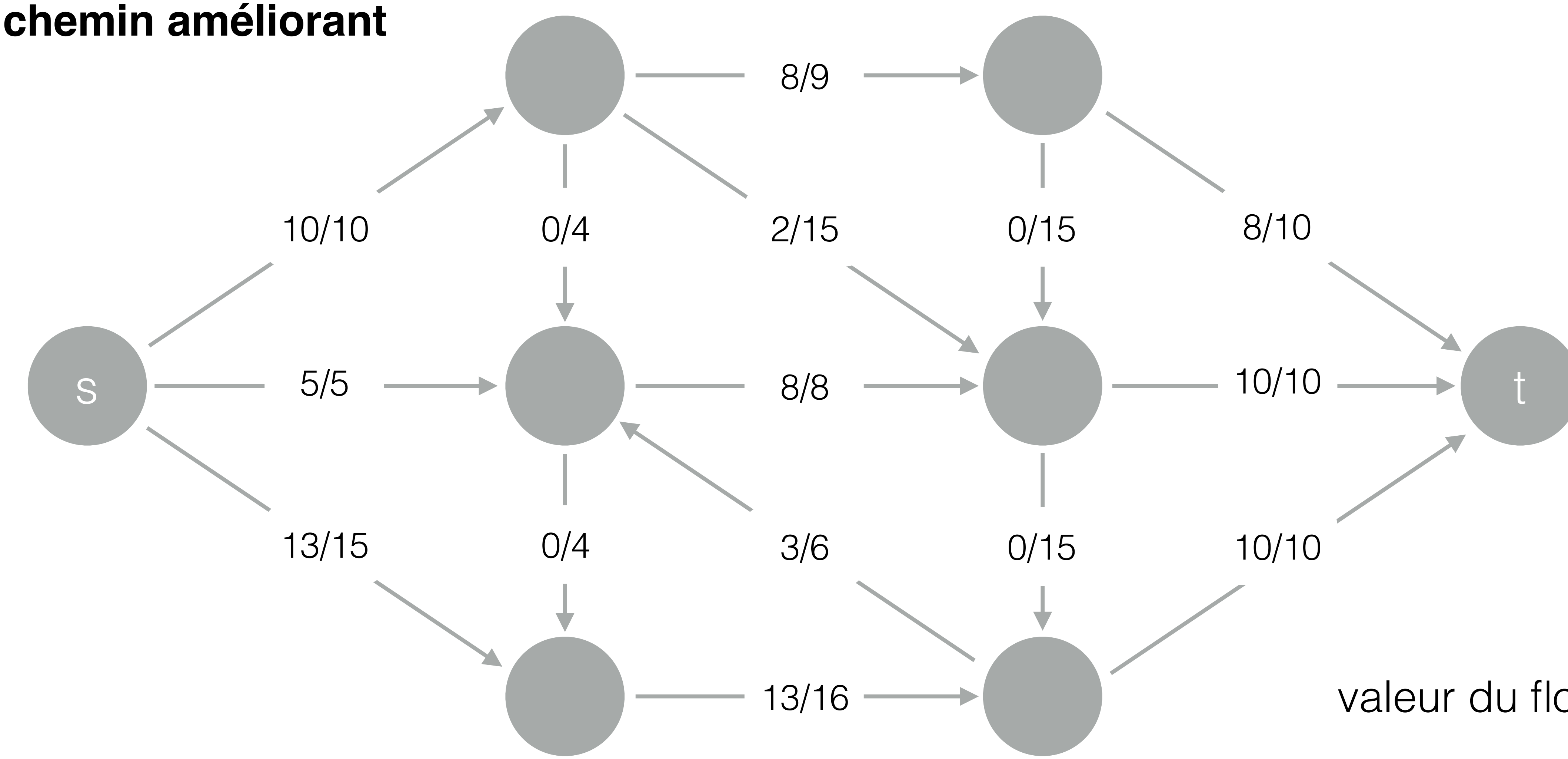
valeur du flot = 25+3

Algorithme de Ford-Fulkerson

Chemin améliorant : on cherche un chemin (dans le graphe non orienté associé) de *s* à *t*

- qui augmente de *w* le flot des arcs empruntés en avant
- qui diminue de de *w* le flot des arcs empruntés en arrière

4e chemin améliorant



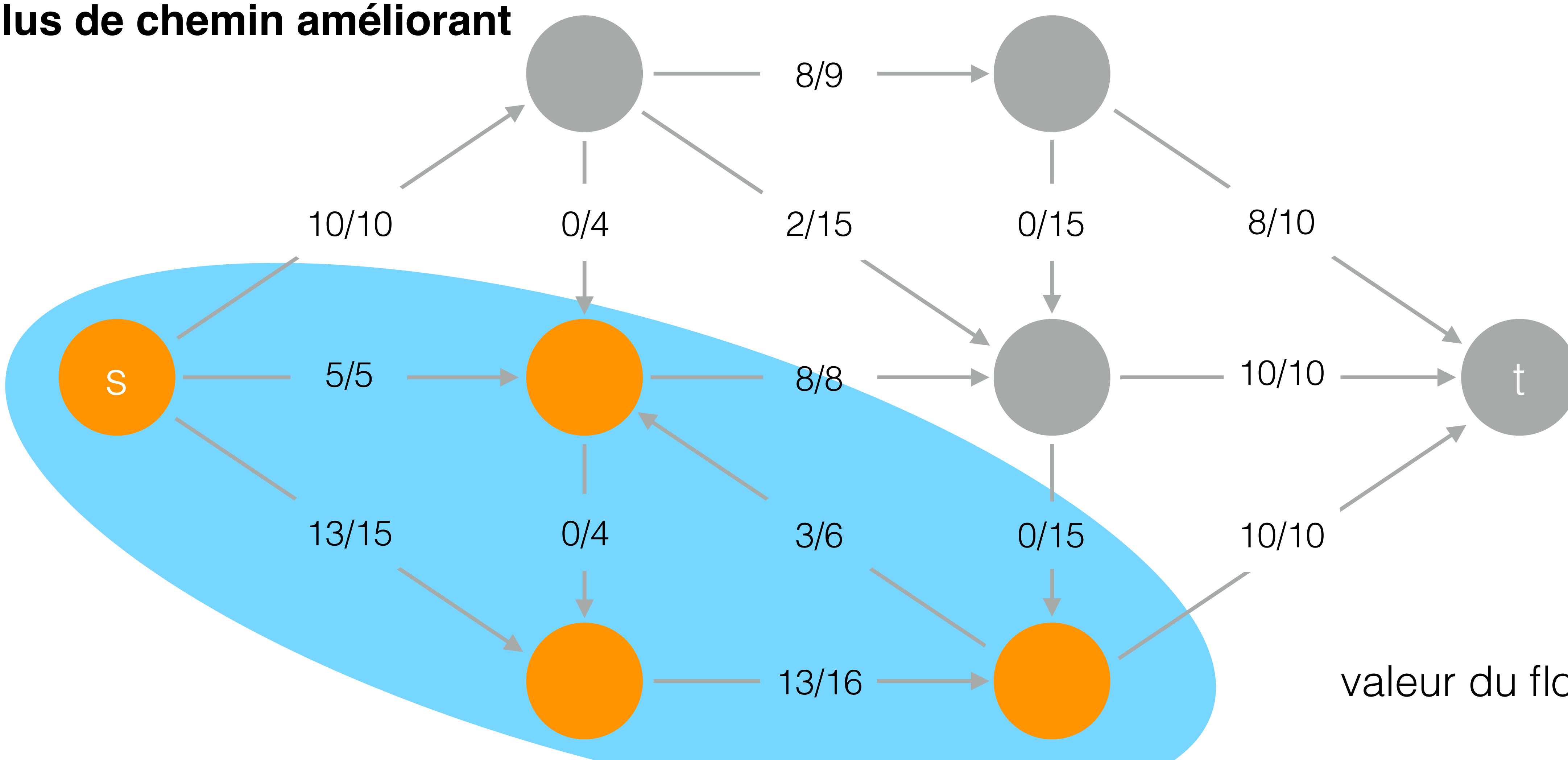
valeur du flot = 25+3

Algorithme de Ford-Fulkerson

Terminaison : tous les chemins de *s* à *t* (dans le graphe non orienté associé) sont bloqués

- soit sur un arc *e* emprunté en avant tel que *f(e)=c(e)*
- soit sur une arc *e* emprunté en arrière tel que *f(e)=0*

plus de chemin améliorant

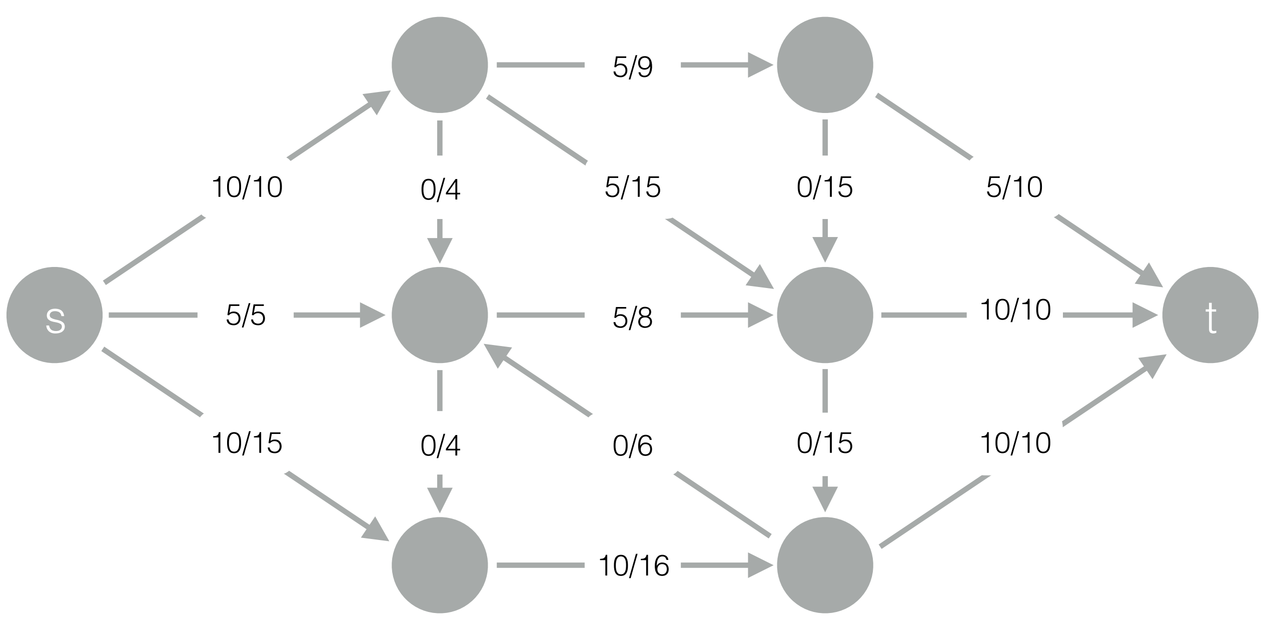


valeur du flot = 28

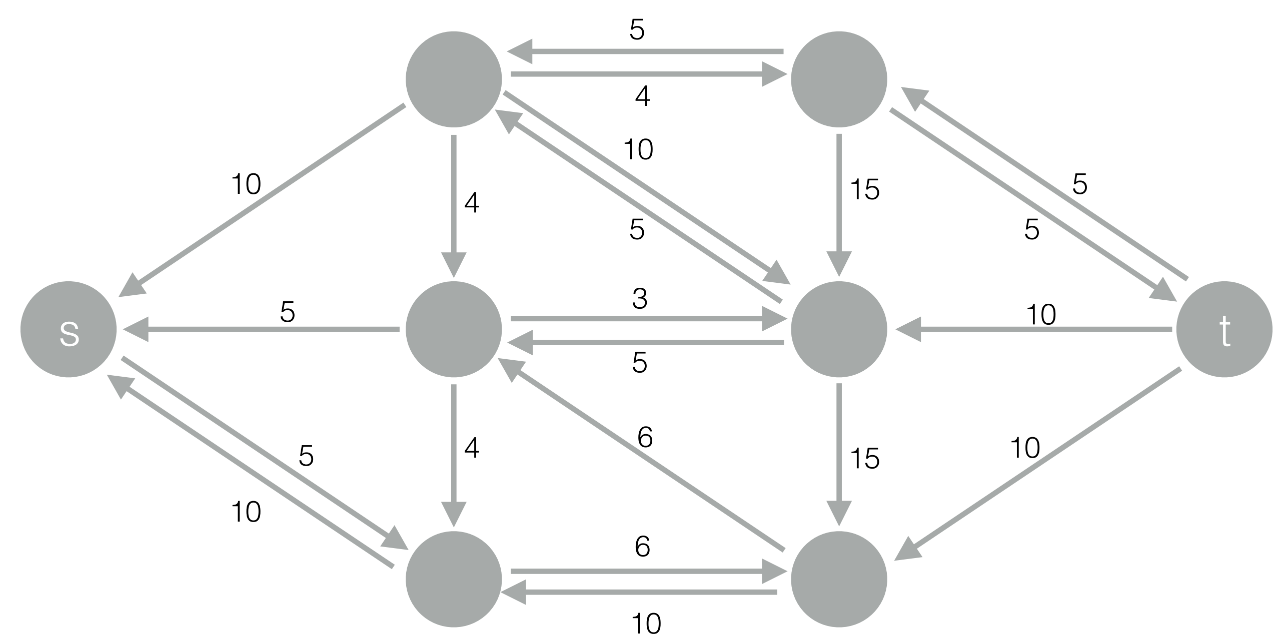
Algorithme de Ford-Fulkerson

Algorithme

1. Commence avec un flot nul
2. Tant qu'il existe un chemin améliorant
    1. Trouve un chemin améliorant (parcours du graphe orienté résiduel)
    2. Calcule le poid maximum *w* le long de ce chemin
    3. Augmente le flot avec *w* le long de ce chemin



Graphe de flot

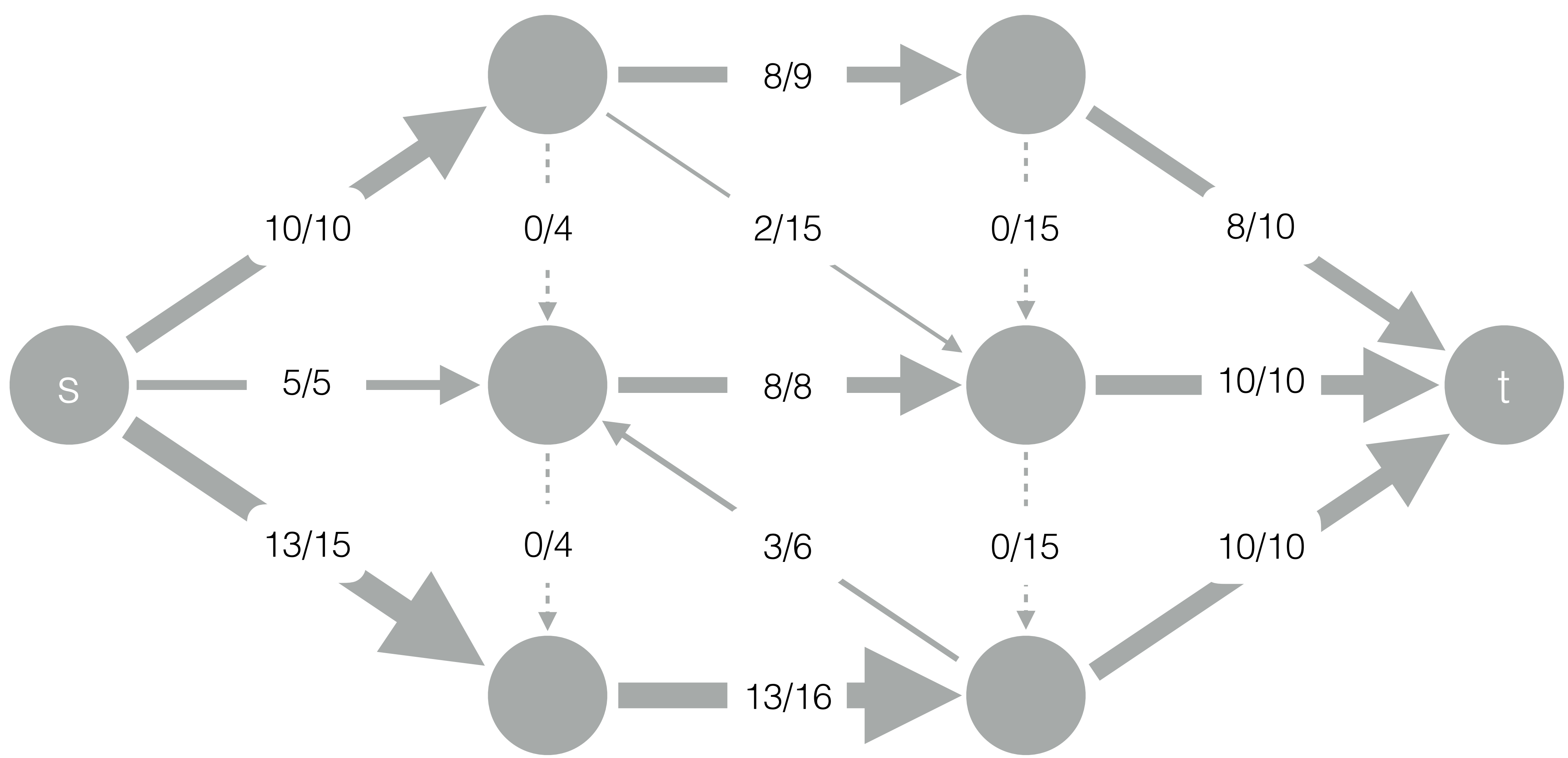


Graphe résiduel

Algorithme de Ford-Fulkerson

Questions

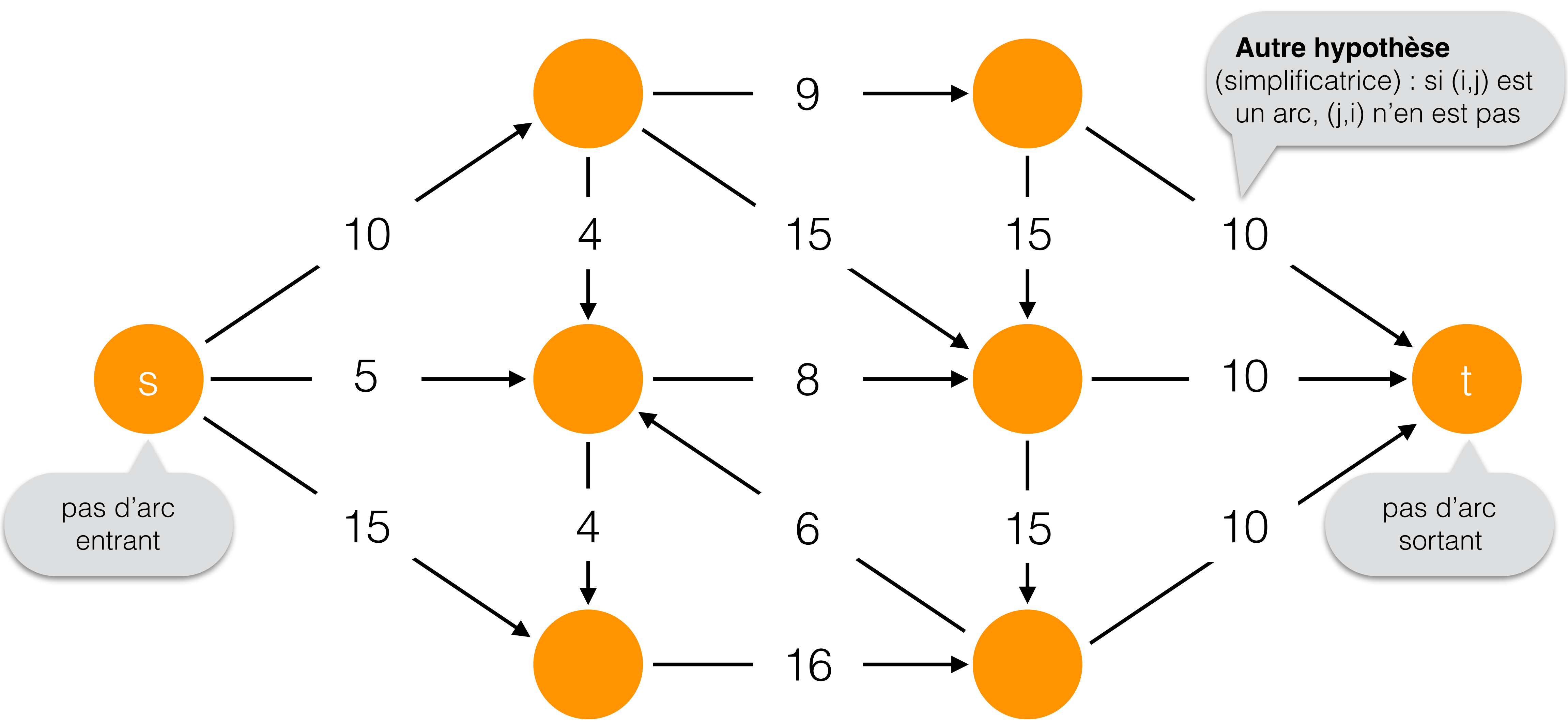
- Quand l'algorithme termine, est-ce qu'il calcule bien un flot maximum ?
- Est-ce que l'algorithme termine toujours ?
- Si oui, après combien de recherche de chemins augmentants ?



Le problème de la coupe minimum

Entrée

- un graphe pondéré (par une *capacité* notée *c*)
- poids strictement positifs
- un sommet *source* s et un sommet *cible* t

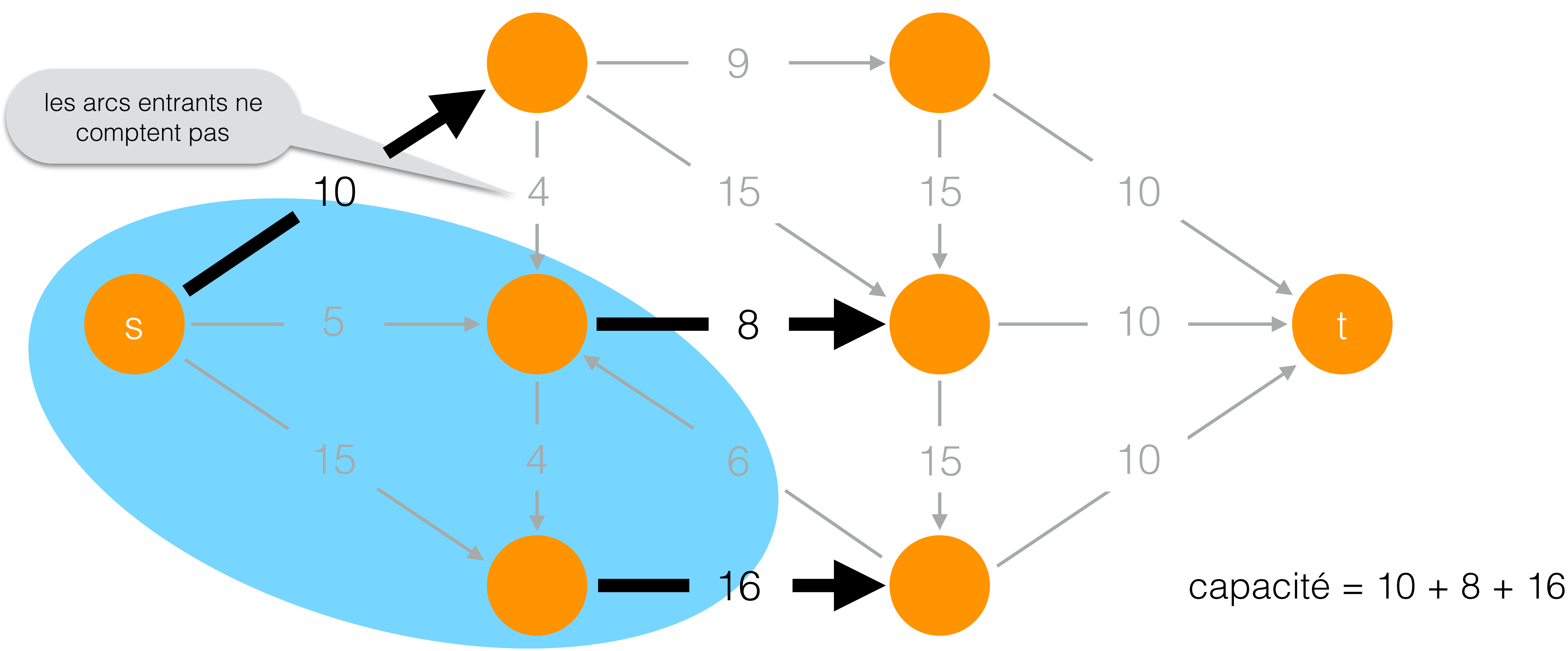


Le problème de la coupe minimum

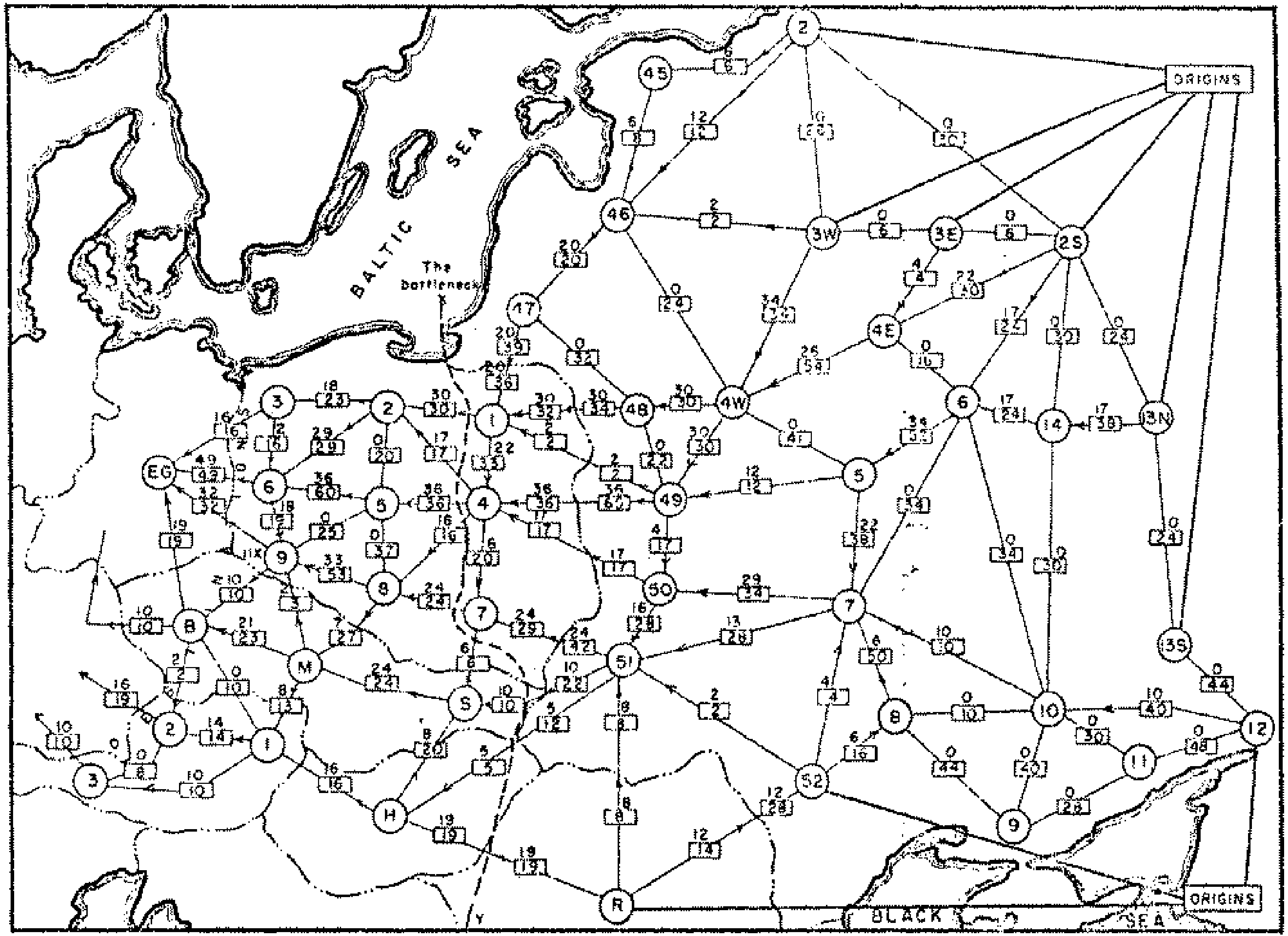
Définition. Une *coupure* est une une partition *(A,B)* de l'ensemble des sommets du graphe telle que *s* appartient à *A et t* appartient à *B.*

Définition. La capacité d'une coupure est la somme des capacités des arcs allant de *A* vers *B*.

Problème de la coupure minimum : trouver une coupure de capacité minimum.



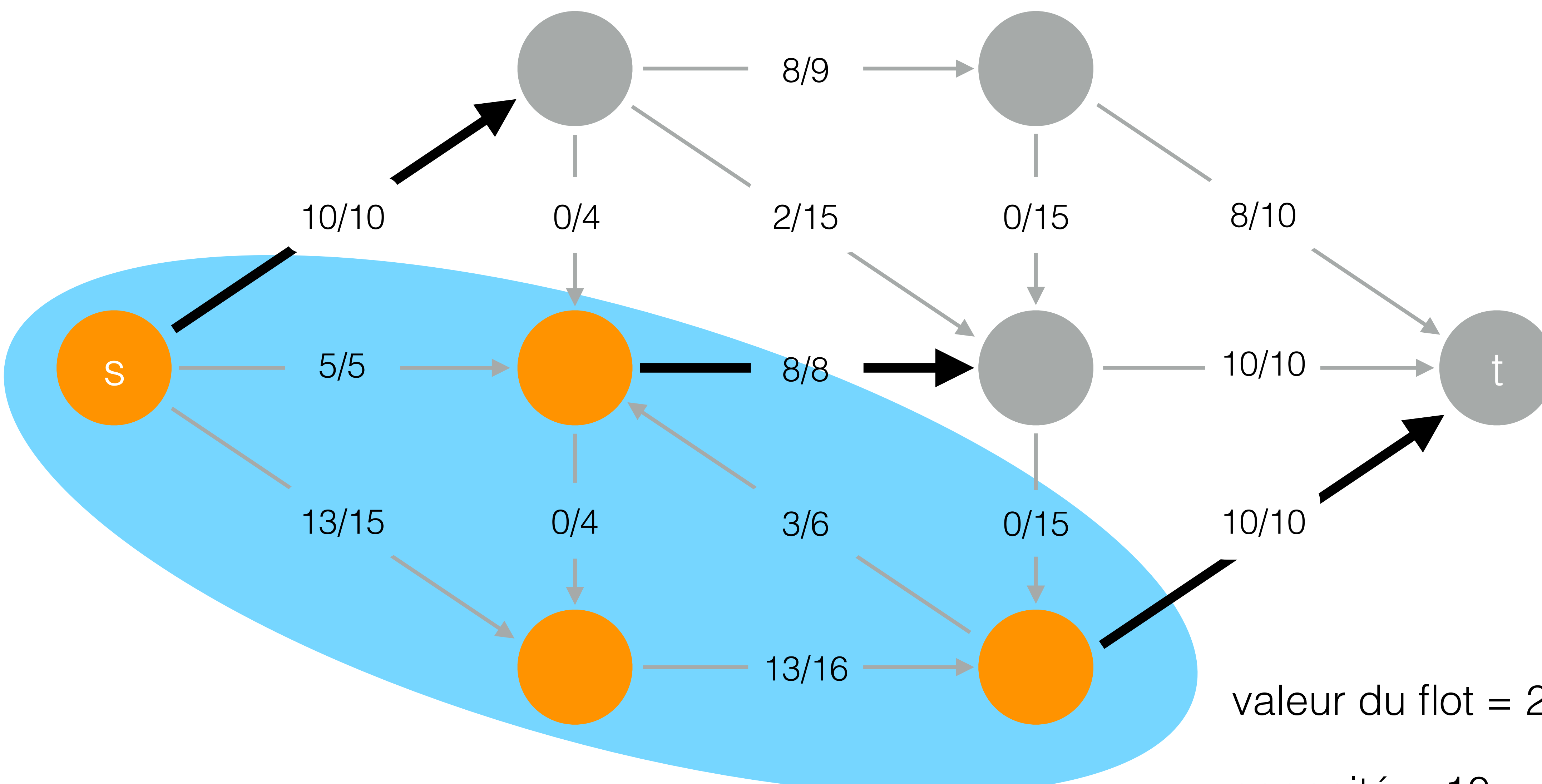
Deux problèmes historiques



On the history of the transportation and the maximum flow problem. Alexander Schrijver. http://homepages.cwi.nl/~lex/files/histtrpclean.pdf

Le théorème du *maxflow-mincut*

Théorème. La capacité de la coupe minimum est égale à la valeur du flot maximum.



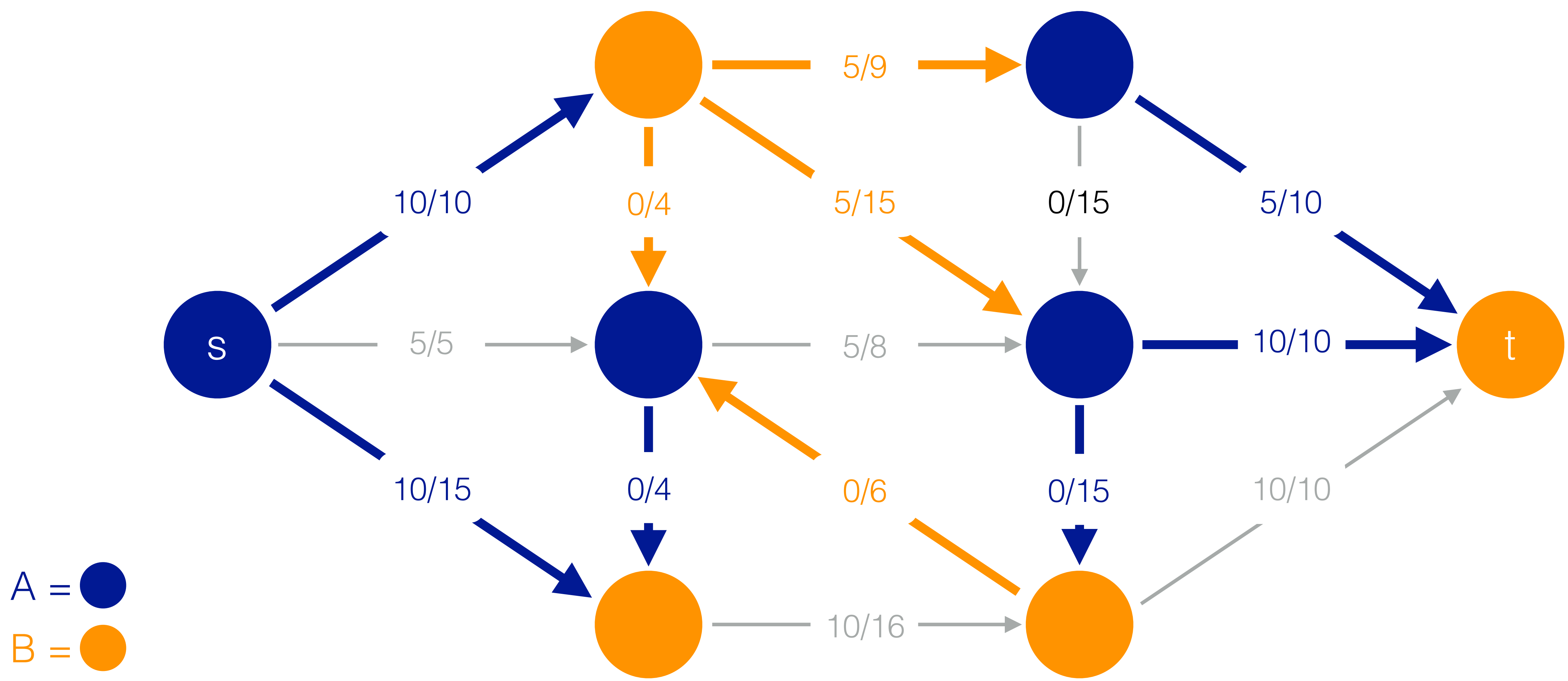
valeur du flot = 28

capacité = 10 + 8 + 10 = 28

Preuve du théorème du *maxflow-mincut*

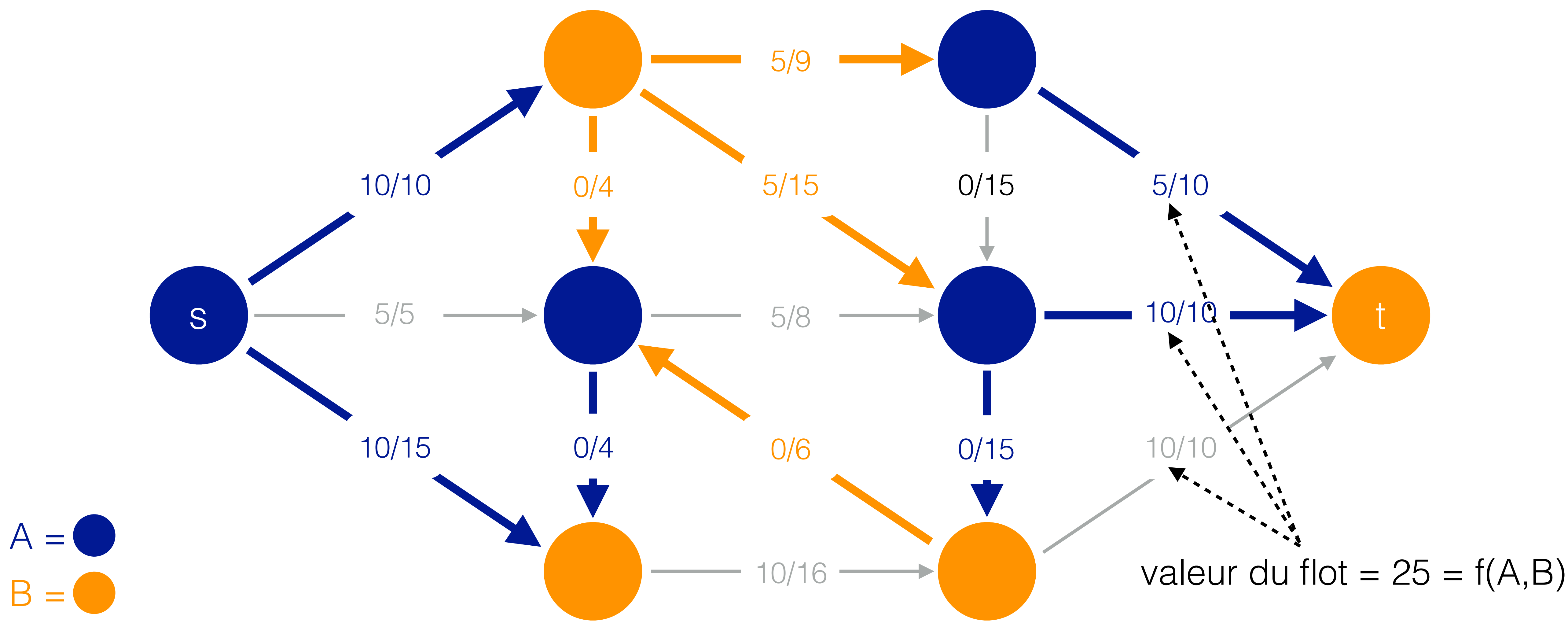
Définition. Soit *f* un flot et *(A,B)* une coupure. Le *flot net f(A,B)* est définie comme la somme des flots des arcs de *A* vers *B*, moins la somme des flots des arcs de *B* vers *A*.

Ici : (10 + 10 + 0 + 5 + 10) - (0 + 5 + 5 + 0) = 25



Preuve du théorème du *maxflow-mincut*

Lemme. Le flot net *f(A,B)* est égal à la valeur du flot *f.*



Preuve du théorème du *maxflow-mincut*

Lemme. Le flot net *f(A,B)* est égal à la valeur du flot *f.*

Preuve. Par récurrence sur la taille de *B*.

- Cas de base : *B = {t}.*
- Hérédité : supposons le résultat vrai pour *(A,B)* *[f(A,B) = |f|],* fixons *y* appartenant à A et montrons le résultat pour la coupe (A-{y}, B+{y}).

$$f(A\backslash\{y\}, B \cup \{y\}) = \sum_{u \in A\backslash\{y\}, v \in B \cup \{y\}} f(u,v) \quad - \sum_{u \in B \cup \{y\}, v \in A\backslash\{y\}} f(u,v)$$

si un arc *e* n'existe pas, nous prenons la convention que *f(e)=0*

$$f(A\backslash\{y\}, B \cup \{y\}) = \sum_{u \in A\backslash\{y\}, v \in B \cup \{y\}} f(u,v) - \sum_{u \in B \cup \{y\}, v \in A\backslash\{y\}} f(u,v)$$

Or

$$\begin{aligned}
\sum_{u \in A\backslash\{y\}, v \in B\cup\{y\}} f(u,v) &= \sum_{u \in A, v \in B \cup \{y\}} f(u,v) - \sum_{v \in B \cup \{y\}} f(y,v) \\
&= \sum_{u \in A, v \in B} f(u,v) + \sum_{u \in A} f(u,y) - \sum_{v \in B} f(y,v) - f(y,y) \\
&= \sum_{u \in A, v \in B} f(u,v) + \sum_{u \in A} f(u,y) - \sum_{v \in B} f(y,v)
\end{aligned}$$

Donc

$$\begin{aligned}
f(A\backslash\{y\}, B \cup \{y\}) &= \left(\sum_{u \in A, v \in B} f(u,v) + \sum_{u \in A} f(u,y) - \sum_{v \in B} f(y,v) \right) \\
&\quad - \left(\sum_{u \in B, v \in A} f(u,v) + \sum_{v \in A} f(y,v) - \sum_{u \in B} f(u,y) \right) \\
&= f(A,B) + \sum_{u \in A \cup B} f(u,y) - \sum_{v \in A \cup B} f(y,v) \\
&= f(A,B) + 0 \\
&= |f|
\end{aligned}$$

équilibre local en *y*

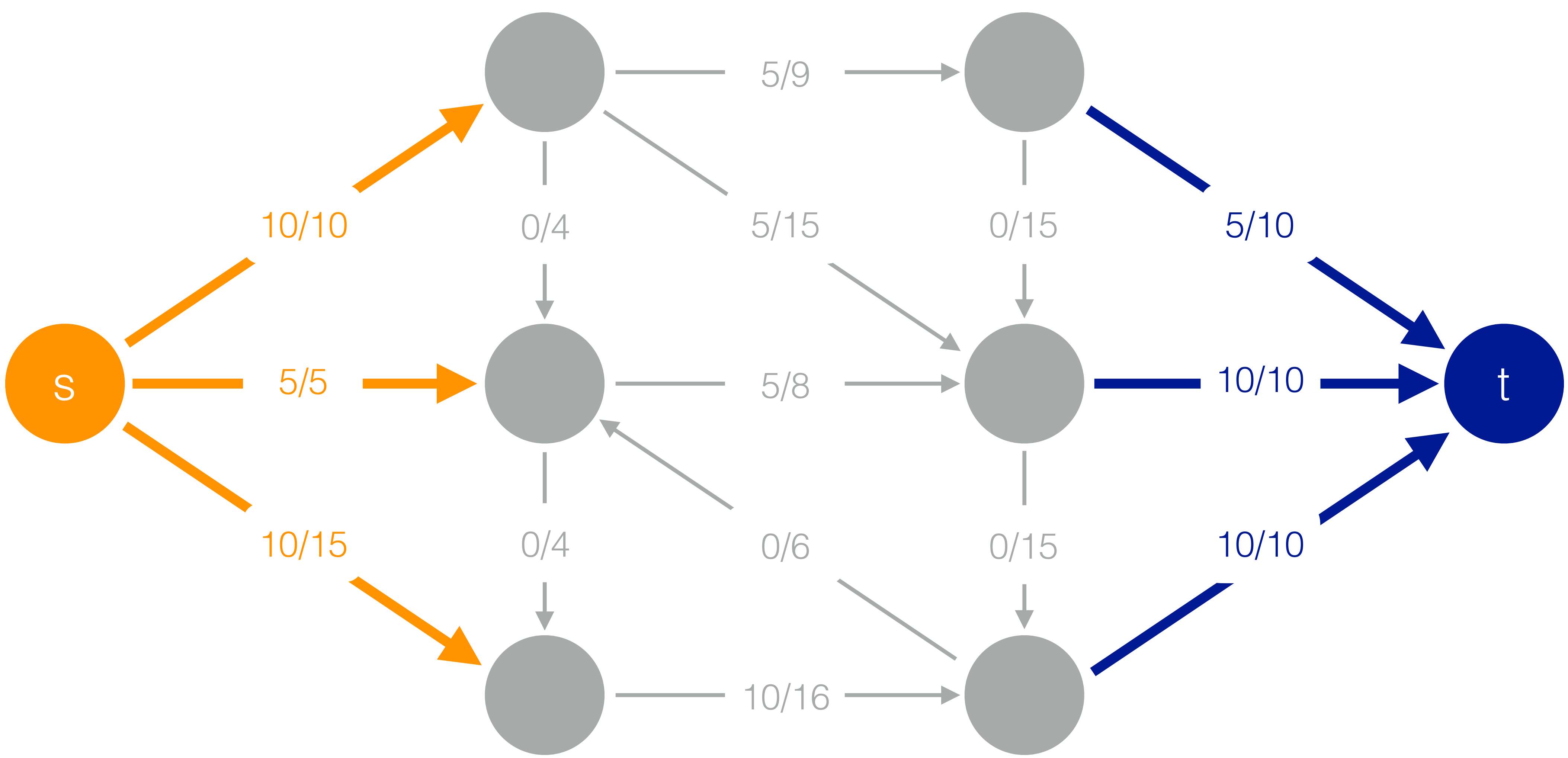
par hypothèse de récurrence

Preuve du théorème du *maxflow-mincut*

Lemme. Le flot net *f(A,B)* est égal à la valeur du flot *f.*

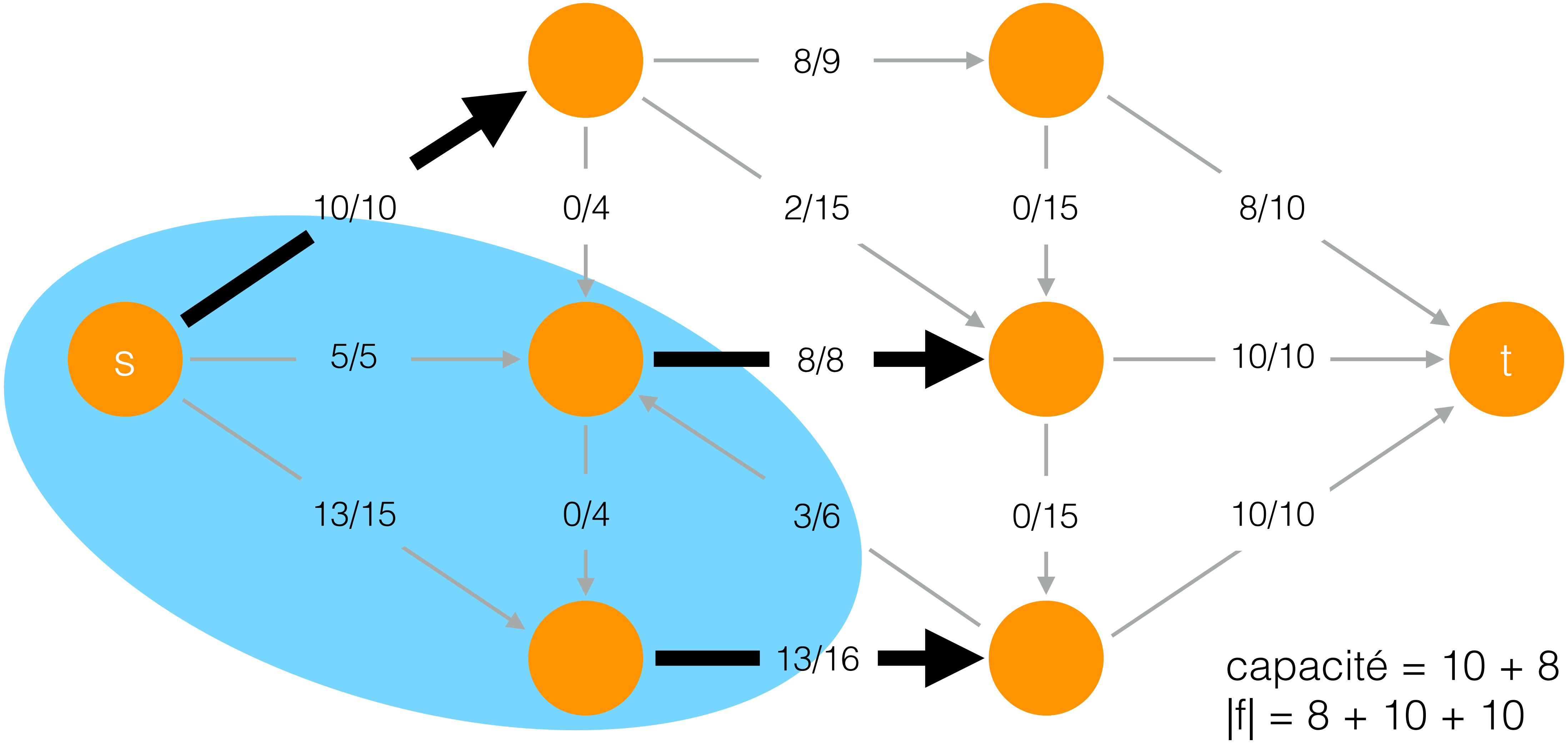
Corollaire. La somme des flots sortants du sommet *s* est égale à la somme des flots entrants dans *t.*

$$10 + 5 + 10 = 5 + 10 + 10$$



Preuve du théorème du *maxflow-mincut*

Lemme. Pour tout flot *f* et toute coupure *(A,B)*, la valeur du flot *f* est inférieure ou égal à la capacité de la coupure



capacité = 10 + 8 + 16
|f| = 8 + 10 + 10
$24 \leq 28$

Preuve du théorème du *maxflow-mincut*

Lemme. Pour tout flot *f* et toute coupure *(A,B)*, la valeur du flot *f* est inférieure ou égal à la capacité de la coupure.

Preuve.

$$|f| = \underbrace{\sum_{u \in A, v \in B} f(u,v) - \sum_{u \in B, v \in A} f(u,v)}_{\text{le flot net est égal à la valeur du flot}} \underbrace{\leq}_{\forall e \in A, f(e) \geq 0} \sum_{u \in A, v \in B} f(u,v) \underbrace{\leq}_{\forall e \in A, f(e) \leq c(e)} \sum_{u \in A, v \in B} c(u,v) \underbrace{=}_{\text{capacité de la coupure}} cap(A,B)$$

le flot net est égal à la valeur du flot

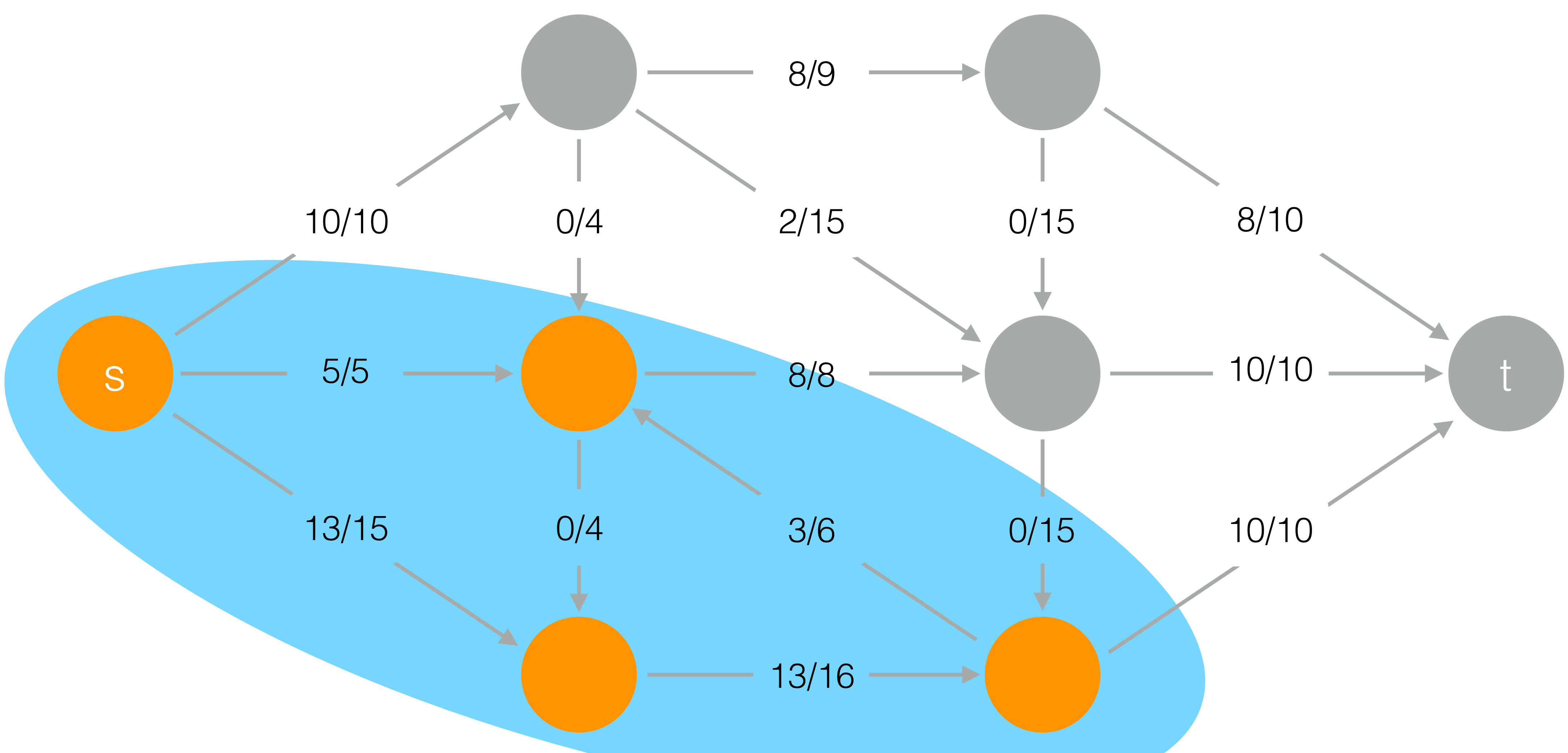
$\forall e \in A, f(e) \geq 0$

$\forall e \in A, f(e) \leq c(e)$

capacité de la coupure

Preuve du théorème du *maxflow-mincut*

Lemme. Soit *f* un flot. Les trois propriétés suivantes sont équivalentes.
i) Il existe une coupure dont la capacité est égal à la valeur du flot.
ii) Le flot *f* est maximal.
iii) Il n'existe pas de chemin améliorant par rapport à *f.*



Preuve du théorème du *maxflow-mincut*

Lemme. Soit f un flot. Les trois propriétés suivantes sont équivalentes.

i) Il existe une coupure dont la capacité est égal à la valeur du flot.
ii) Le flot f est maximal.
iii) Il n'existe pas de chemin améliorant par rapport à f.

Preuve. [i) => ii)]

Soit (A,B) une coupure telle que $cap(A,B)=|f|$. Pour tout flot f', sa valeur est inférieure à la capacité cap(A,B). Donc $|f'|$ est inférieure à $|f|$.

Donc f est maximal.

Preuve du théorème du *maxflow-mincut*

Lemme. Soit *f* un flot. Les trois propriétés suivantes sont équivalentes.

i) Il existe une coupure dont la capacité est égal à la valeur du flot.
ii) Le flot *f* est maximal.
iii) Il n’existe pas de chemin améliorant par rapport à *f*.

Preuve. [ii) => iii)]

Contraposée : si il existe un chemin améliorant, alors *f* n’est pas maximal puisque le chemin permet d’augmenter la valeur du flot.

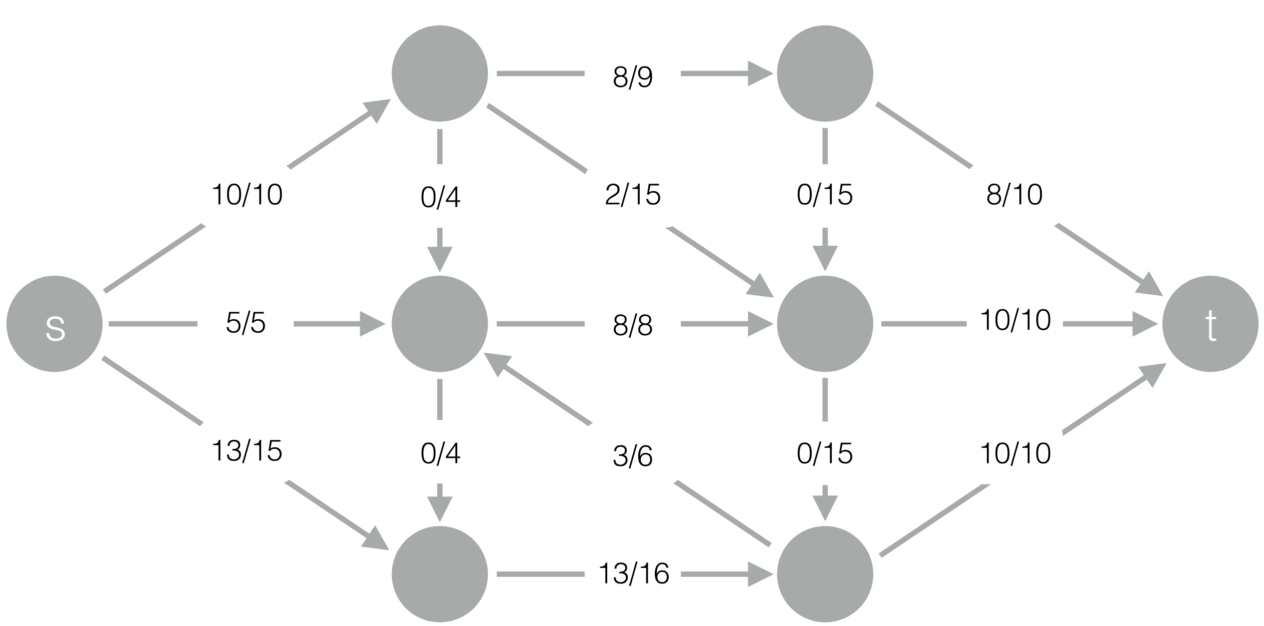
Preuve du théorème du *maxflow-mincut*

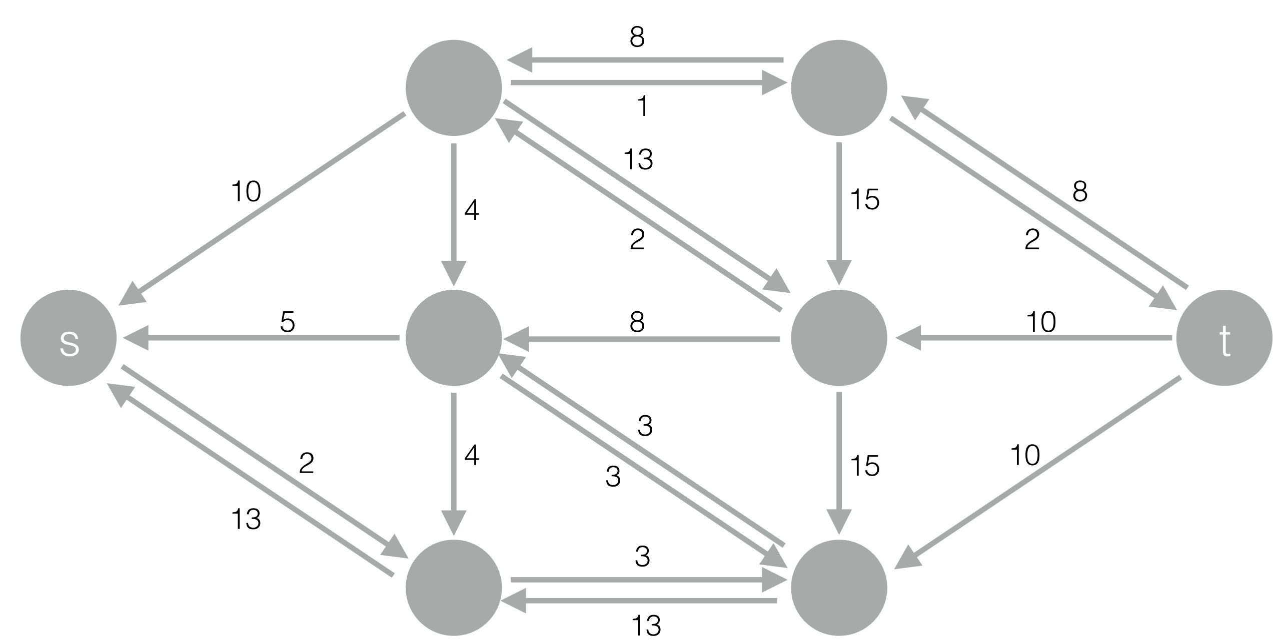
Lemme. Soit *f* un flot. Les trois propriétés suivantes sont équivalentes.

i) Il existe une coupure dont la capacité est égal à la valeur du flot.
ii) Le flot *f* est maximal.
iii) Il n'existe pas de chemin améliorant par rapport à *f.*

Preuve. [iii) => i)]

Supposons l'absence de chemins améliorants et exhibons une coupure dont la capacité soit égal à |f|.





Preuve du théorème du *maxflow-mincut*

Preuve. [iii) => i)]

Supposons l'absence de chemins améliorants et exhibons une coupure dont la capacité soit égal à |f|.

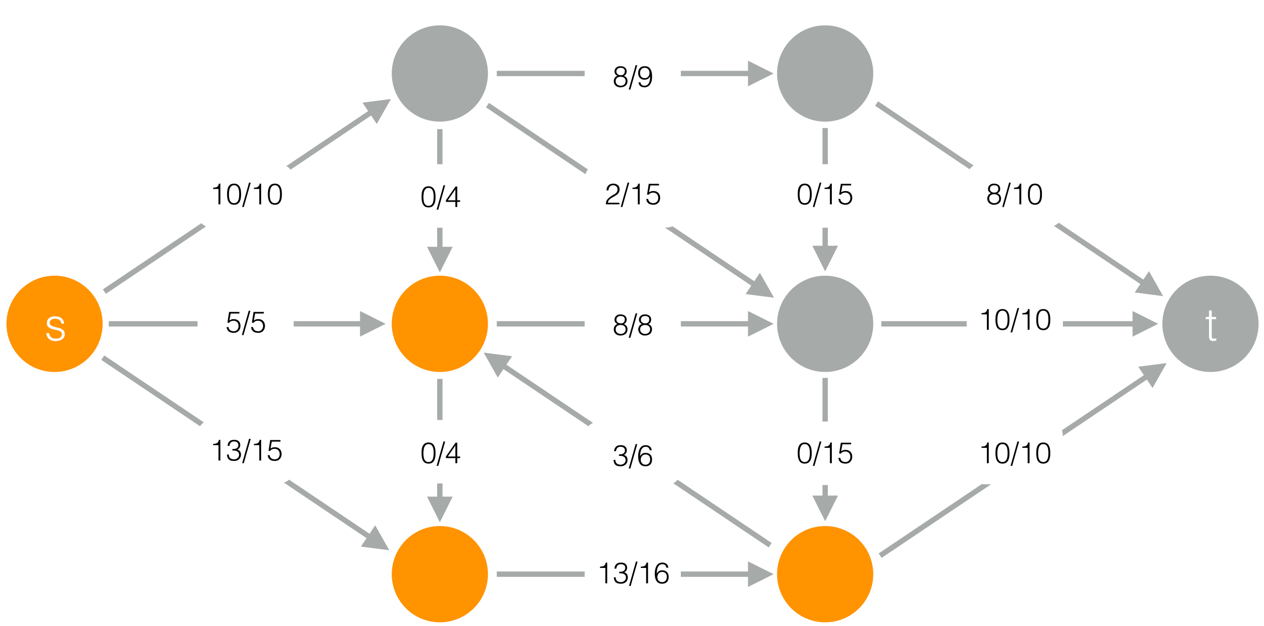
Soit *A* l'ensemble des sommets *u* accessibles depuis *s* dans le graphe résiduel (vis à vis de *f*). Soit *B* le complémentaire de *A*.

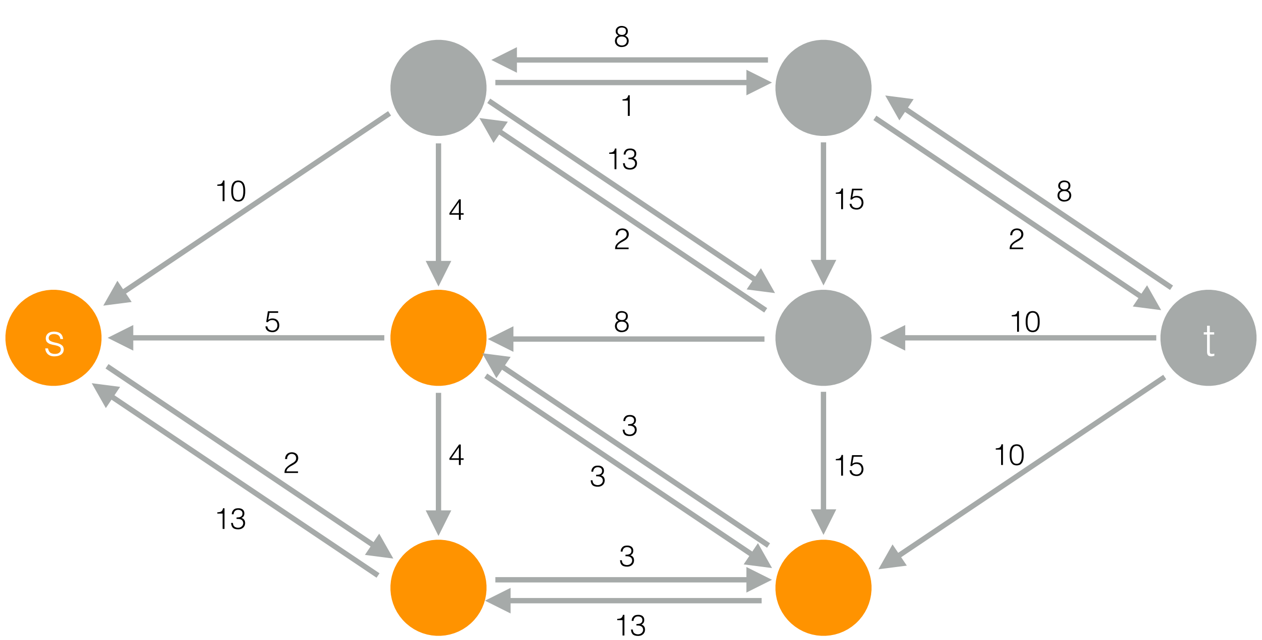
Par définition, *s* appartient à *A*.

Comme il n'y pas de chemin améliorant, *t* appartient à *B*.

Tous les arc allant de B à A ont un flot nul, donc

$$cap(A,B) = f(A,B) = |f|$$





Preuve du théorème du *maxflow-mincut*

Lemme. Soit *f* un flot. Les trois propriétés suivantes sont équivalentes.

i) Il existe une coupure dont la capacité est égal à la valeur du flot.
ii) Le flot *f* est maximal.
iii) Il n'existe pas de chemin améliorant par rapport à *f.*

Conséquences.

1. Puisque pour tout flot *f* et toute coupure *(A,B)*, la valeur du flot *f* est inférieure ou égal à la capacité de la coupure, i) est équivalent à l'existence d'une coupure minimale et le théorème *maxflow-mincut* est démontré.
2. Un flot est maximal s'il n'existe plus de chemin améliorant, donc l'algorithme de Ford-Fulkerson est correct (s'il termine).
3. On peut construire une coupe minimale avec l'ensemble des sommets *u* accessibles depuis *s* dans le graphe résiduel d'un flot maximal.

Terminaison de l’algorithme de Ford-Fulkerson

Hypothèse. La capacité de chaque arc est un nombre entier.

Lemme. Chaque flot intermediaire de l’algorithme de Ford-Fulkerson a une valeur entière.
Preuve. Par récurrence sur le nombre d’étapes de l’algorithme.

Lemme. Le nombre de chemin améliorant construit est inférieure à la valeur du flot maximal.
Preuve. La valeur du flot augmente d’au moins 1 à chaque chemin. Chaque flot f a une valeur bornée par la capacité de la coupe minimale.

On en déduit que l’algorithme termine toujours sous cette hypothèse et que le flot maximum est un nombre entier.

Complexité :

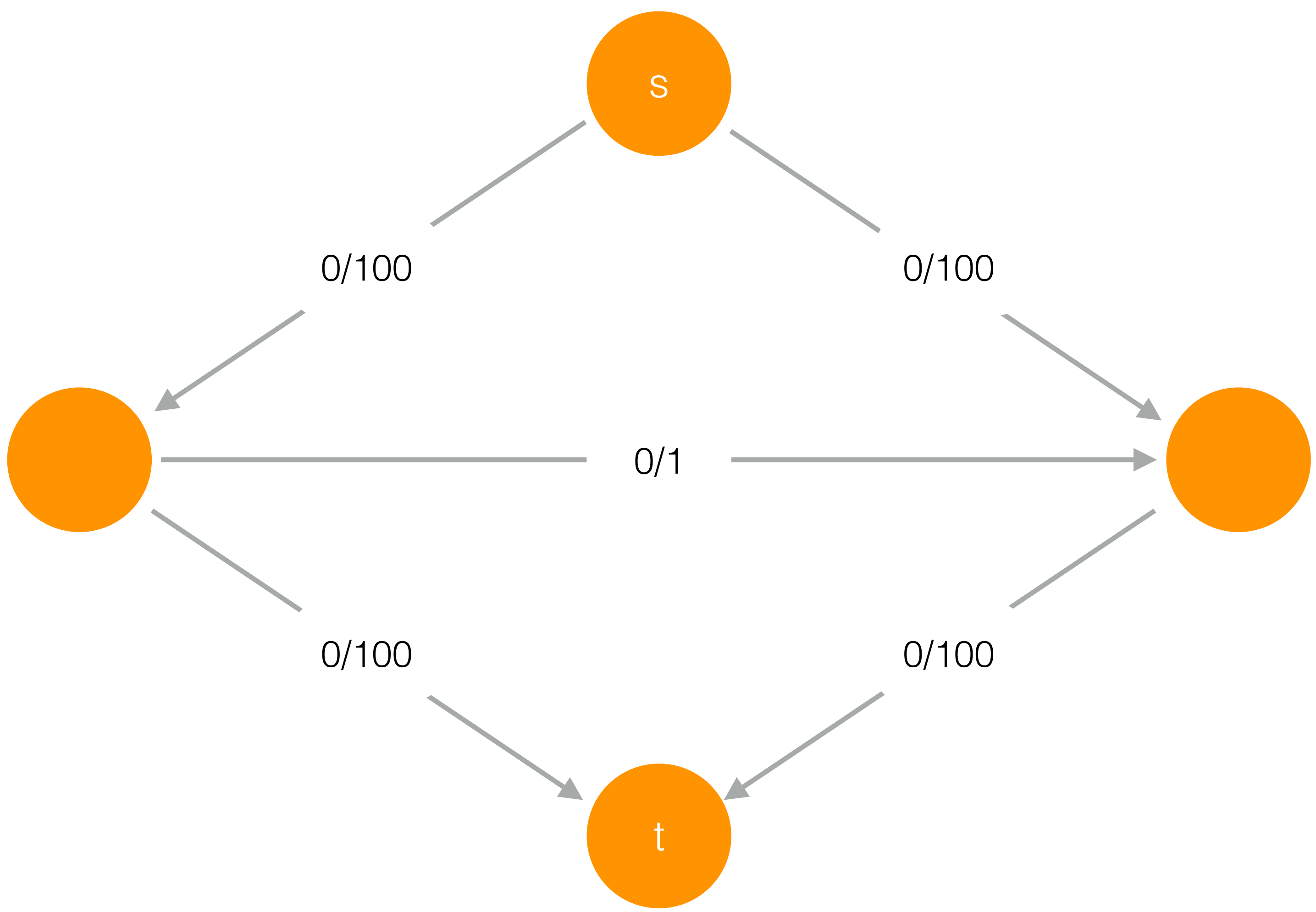
$$\mathcal{O}(A \cdot \mathrm{flot\_max})$$

une construction de chemin
= un parcours à partir de s

au plus flot_max
chemins construits

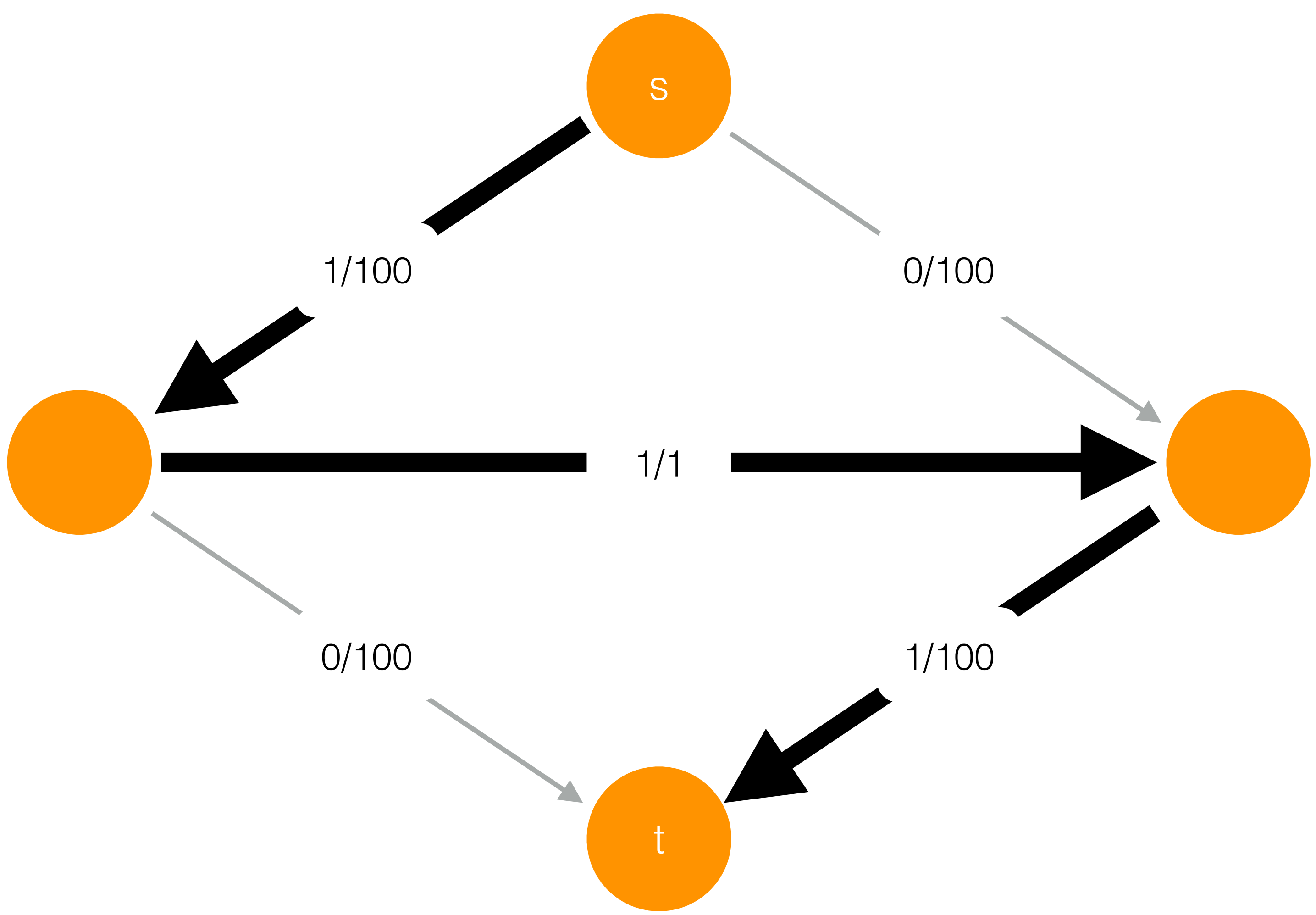
Terminaison de l'algorithme de Ford-Fulkerson

Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



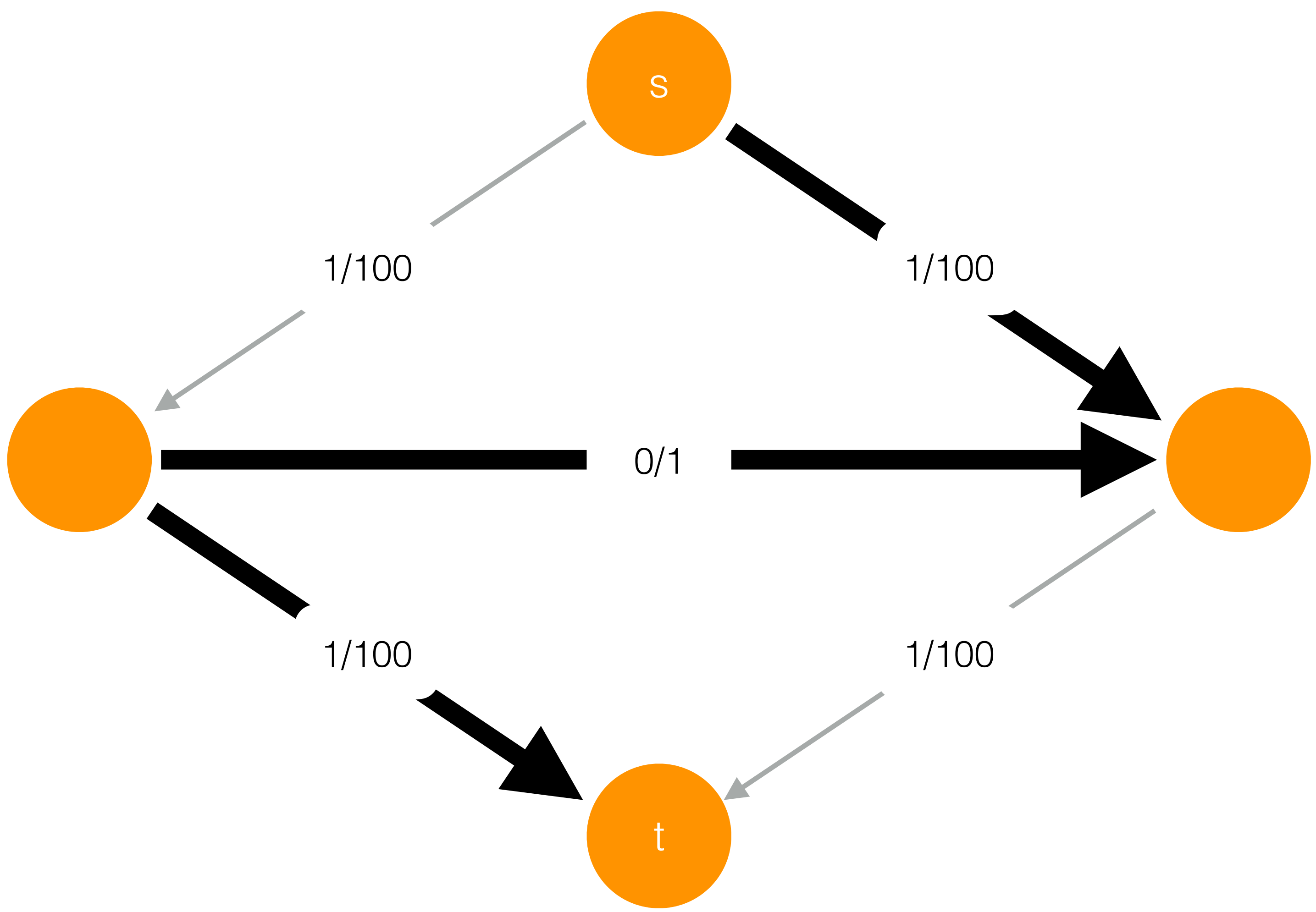
Terminaison de l’algorithme de Ford-Fulkerson

Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



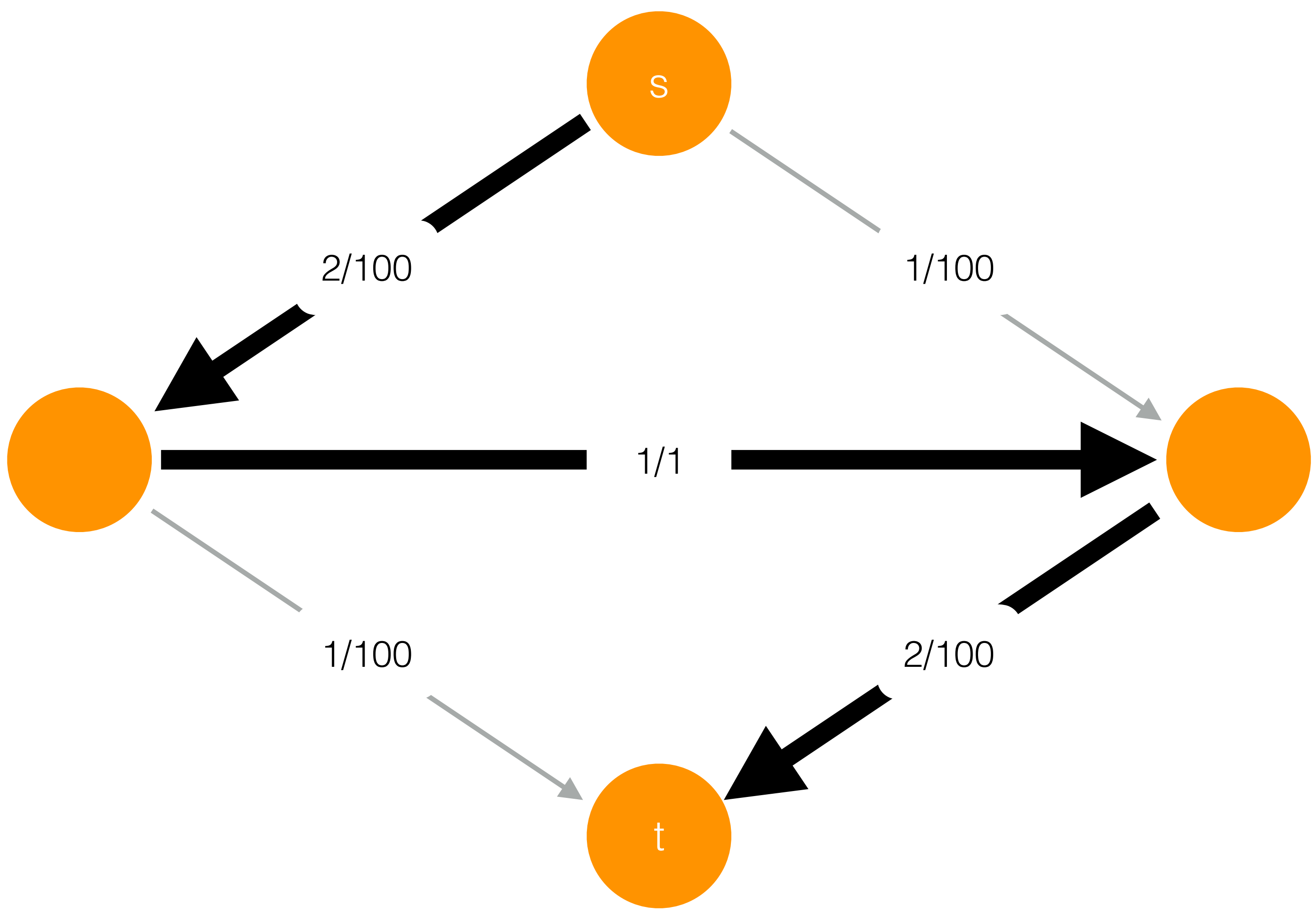
Terminaison de l'algorithme de Ford-Fulkerson

Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



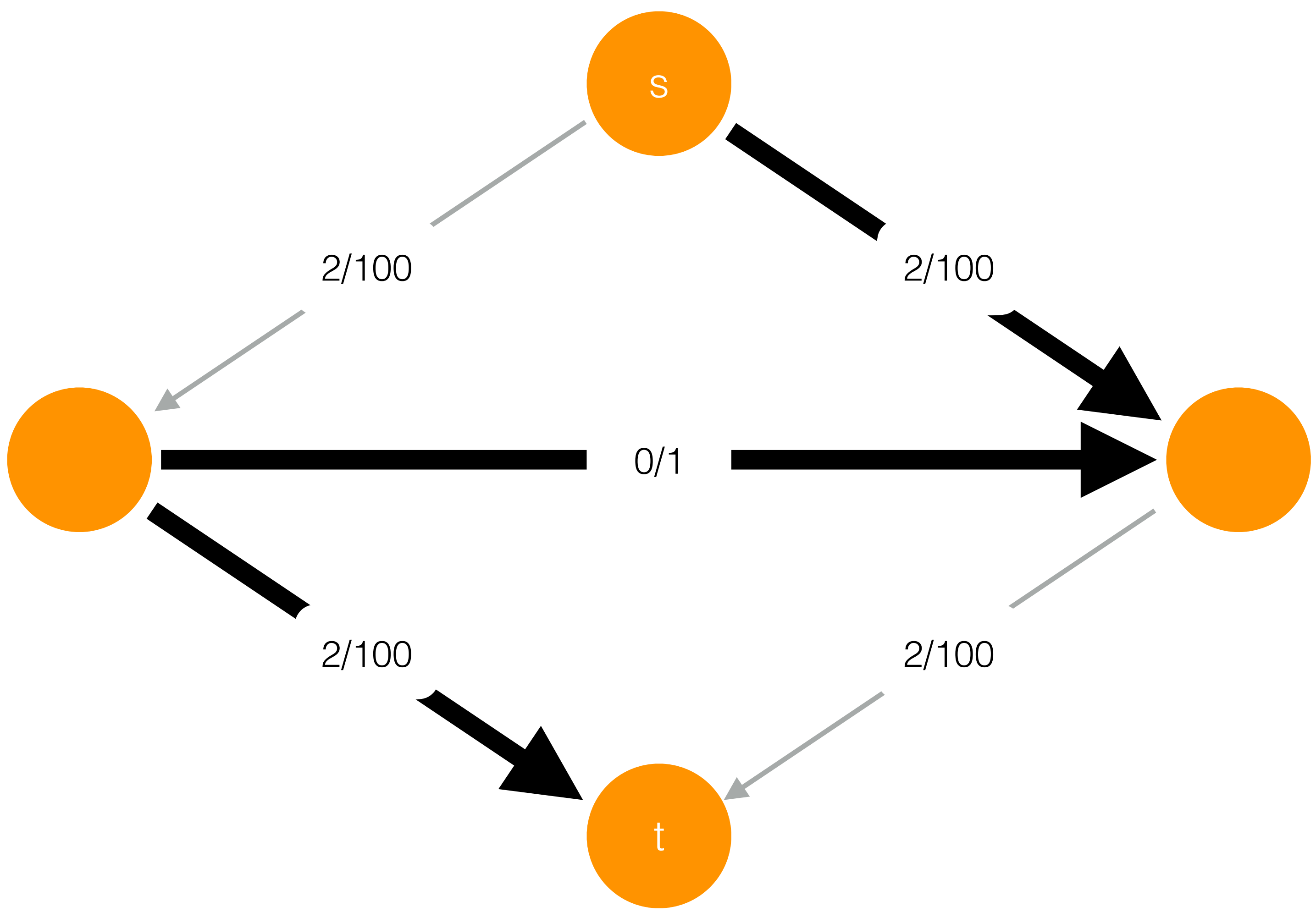
Terminaison de l'algorithme de Ford-Fulkerson

Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



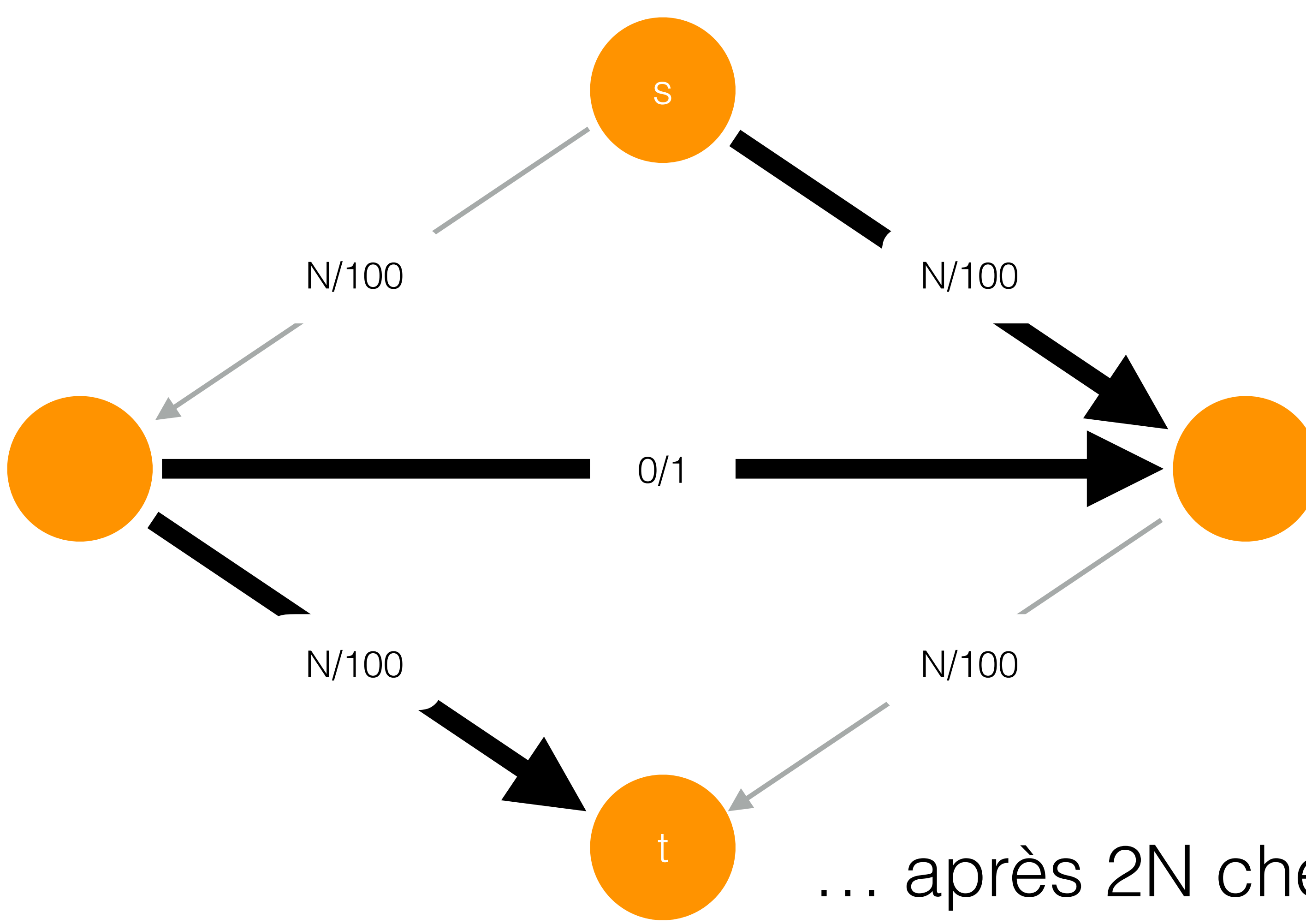
Terminaison de l'algorithme de Ford-Fulkerson

Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



Terminaison de l'algorithme de Ford-Fulkerson

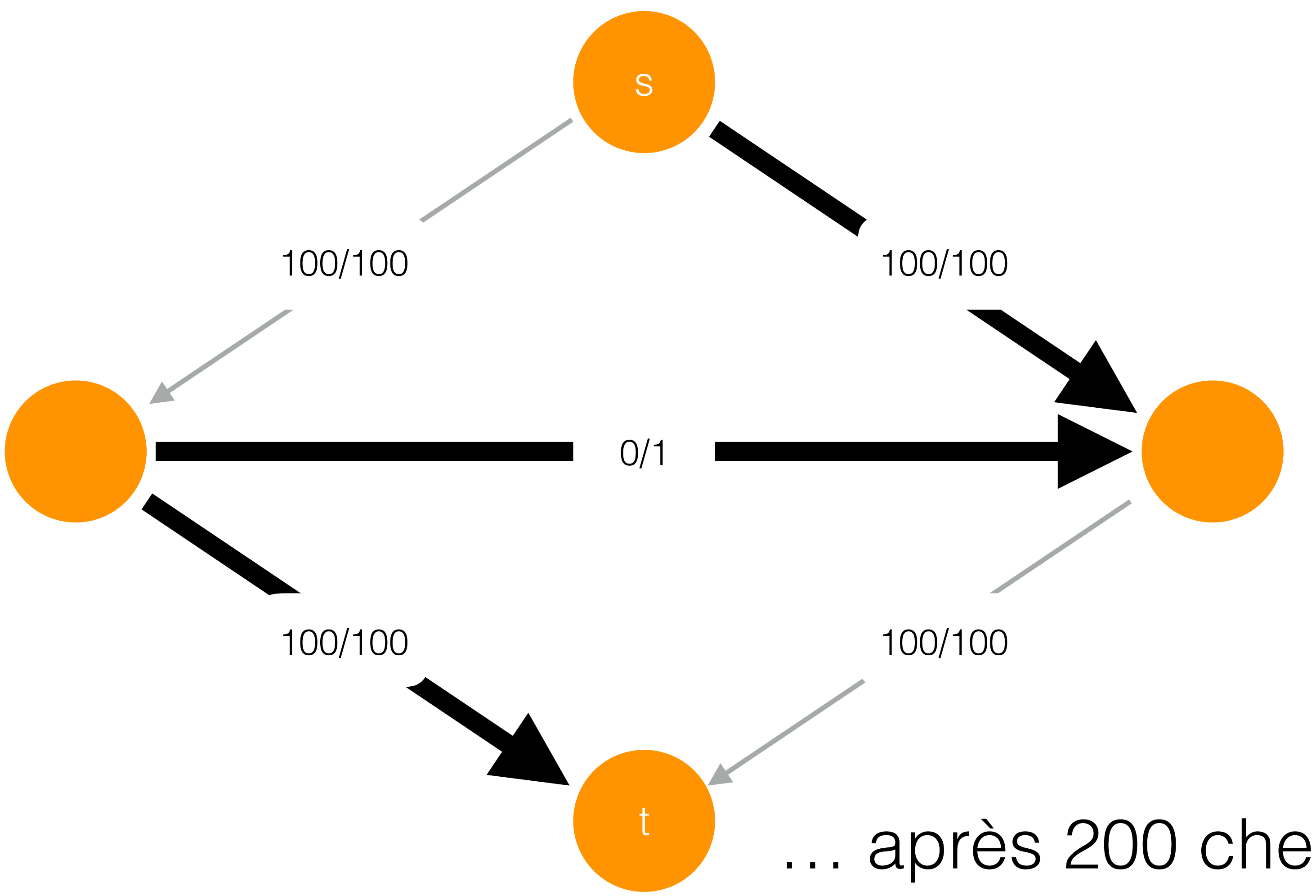
Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



… après 2N chemins améliorants !

Terminaison de l'algorithme de Ford-Fulkerson

Cas le pire. Existe-t-il un cas où le nombre maximum de chemins améliorants est considéré ?



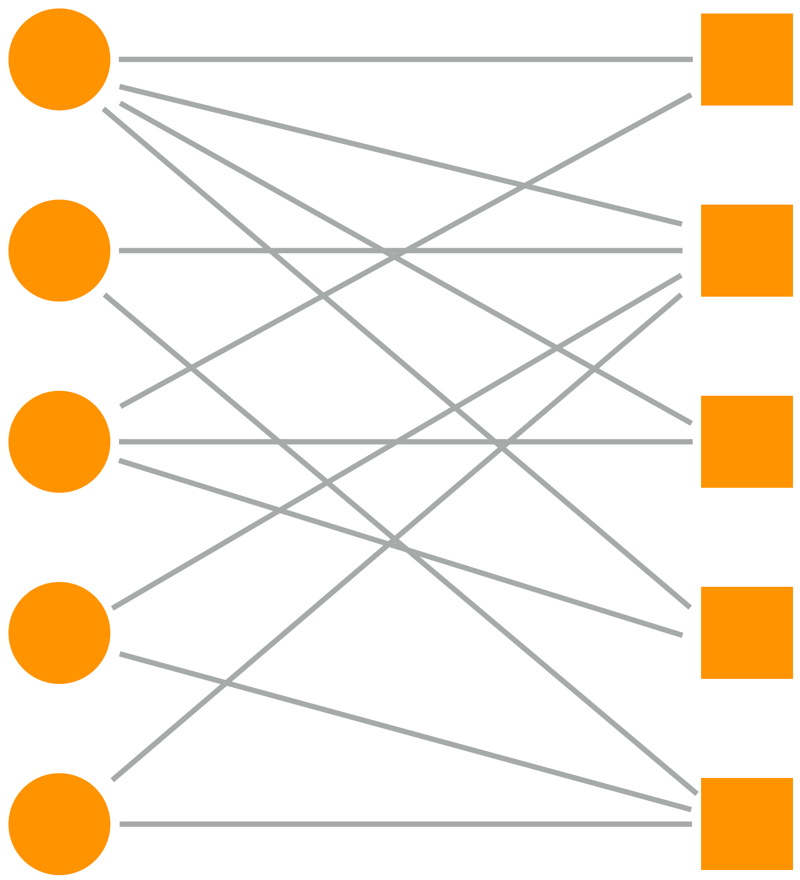
… après 200 chemins améliorants !

Application

Problème du couplage parfait

Définition. Un graphe (non-orienté) biparti est un graphe dont l'ensemble de sommet est partitioné en deux ensembles A et B tels que chaque arête ait une extrémité dans A et l'autre dans B.

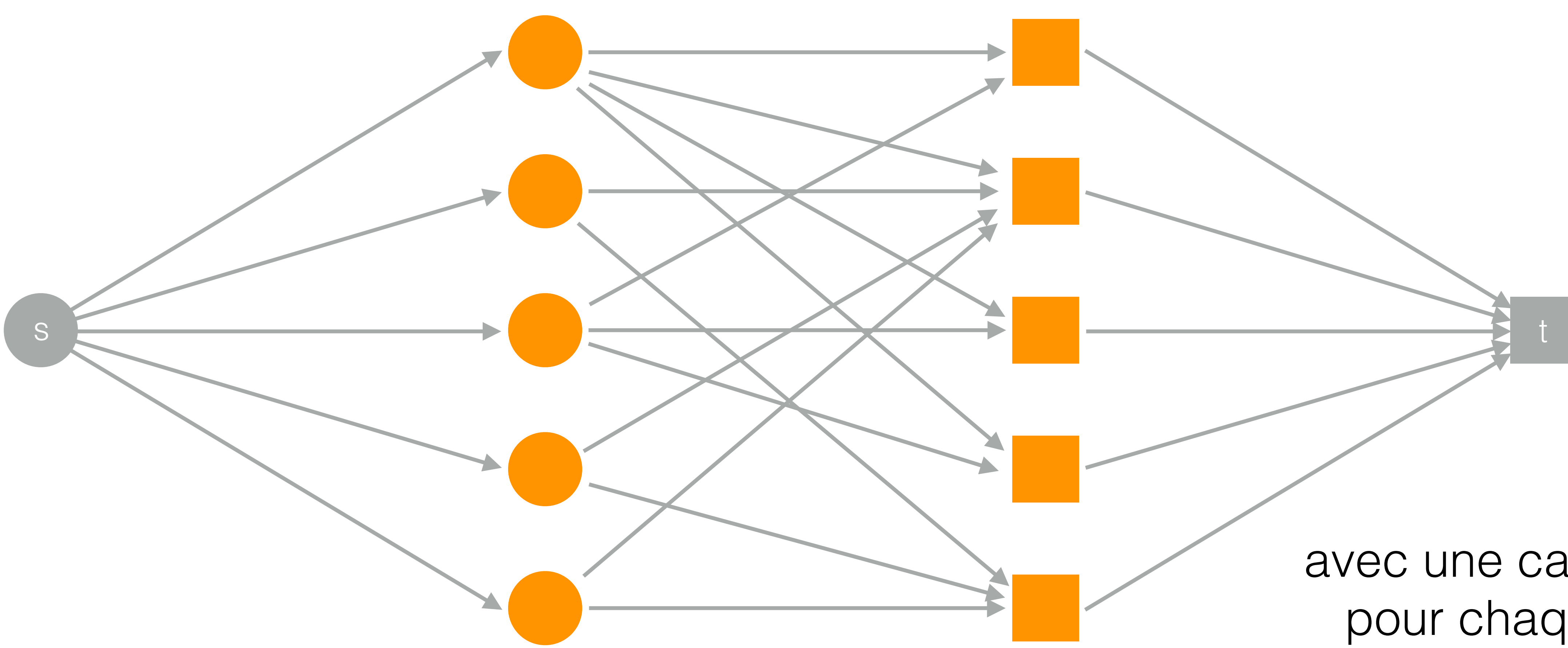
Définition. Un couplage parfait est un sous ensemble des arêtes d'un graphe biparti tel que chaque sommet est incident à exactement une arête de cet ensemble.



Problème du couplage parfait

Réduction. On peut résoudre le problème du couplage parfait en se ramenant à un problème de flot maximum.

Il existe un couplage maximal ssi le flot maximal vaut 5.

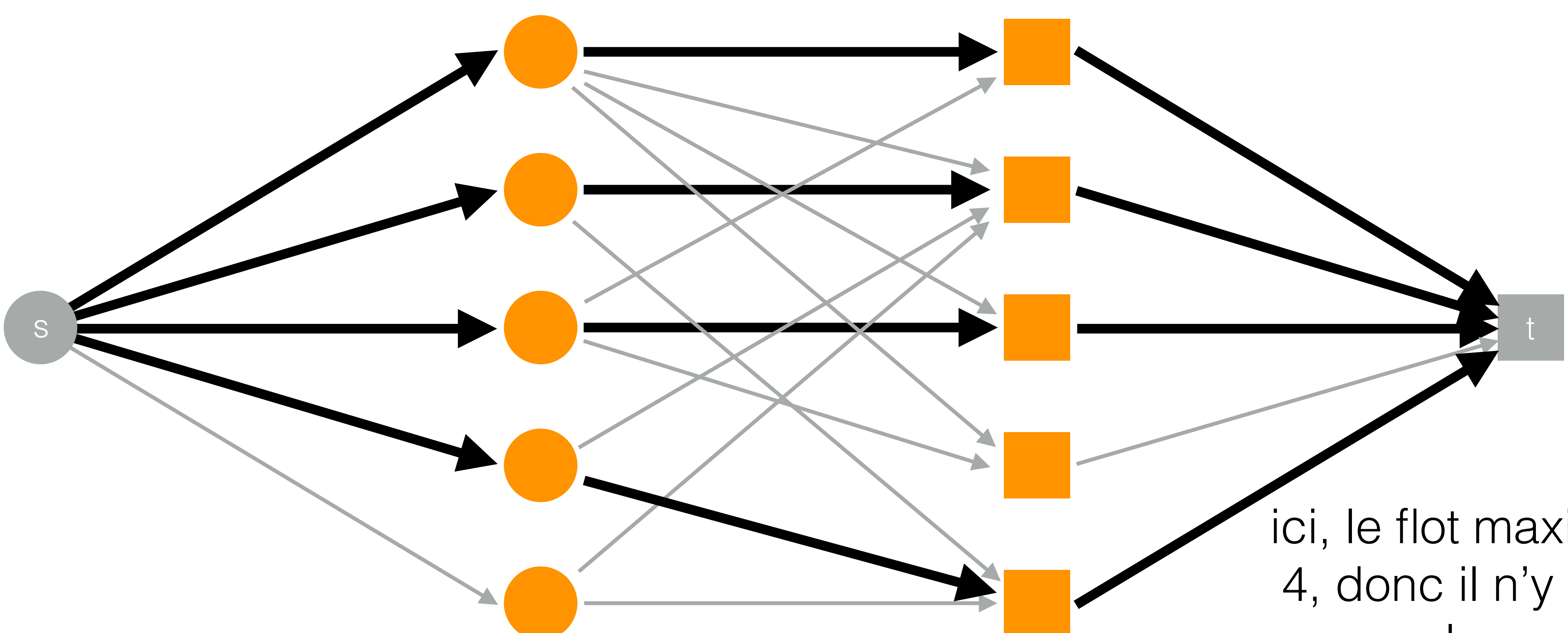


avec une capacité 1
pour chaque arc

Problème du couplage parfait

Réduction. On peut résoudre le problème du couplage parfait en se ramenant à un problème de flot maximum.

Il existe un couplage maximal ssi le flot maximal vaut 5.

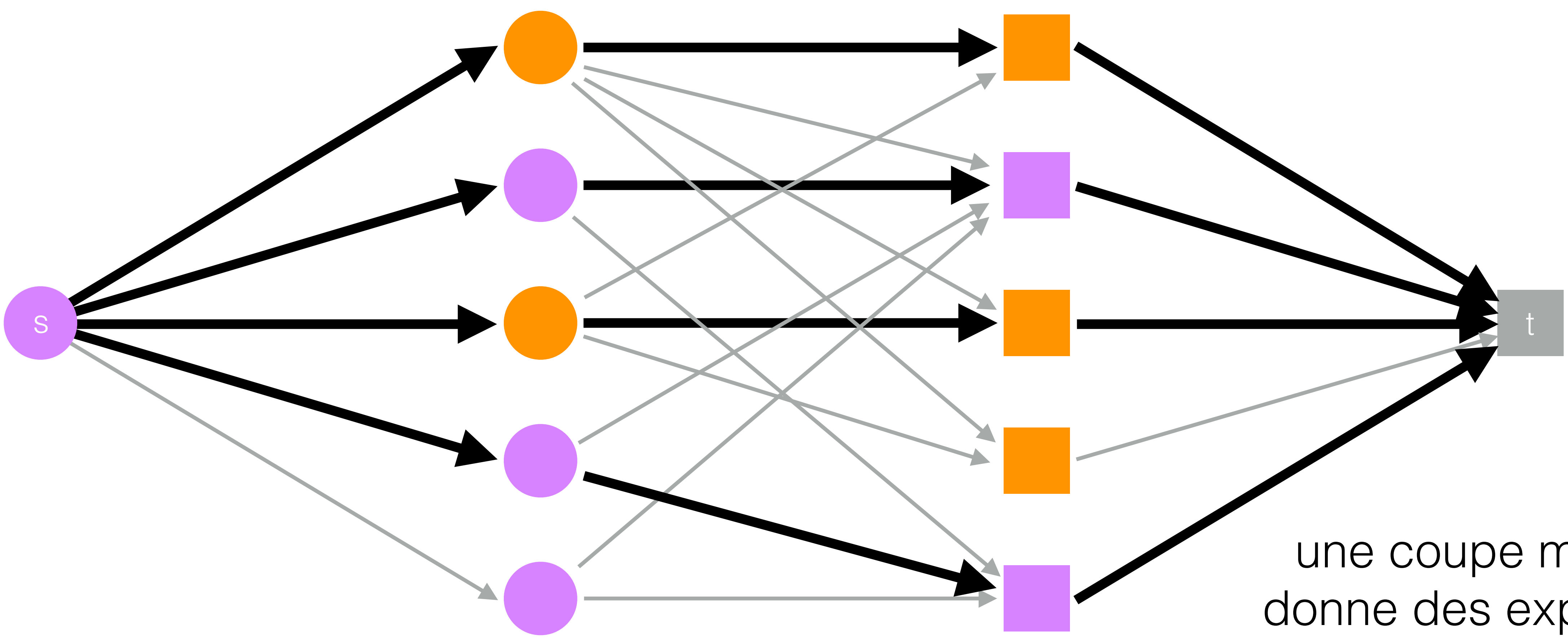


ici, le flot maximal vaut 4, donc il n'y a pas de couplage parfait

Problème du couplage parfait

Réduction. On peut résoudre le problème du couplage parfait en se ramenant à un problème de flot maximum.

Il existe un couplage maximal ssi le flot maximal vaut 5.

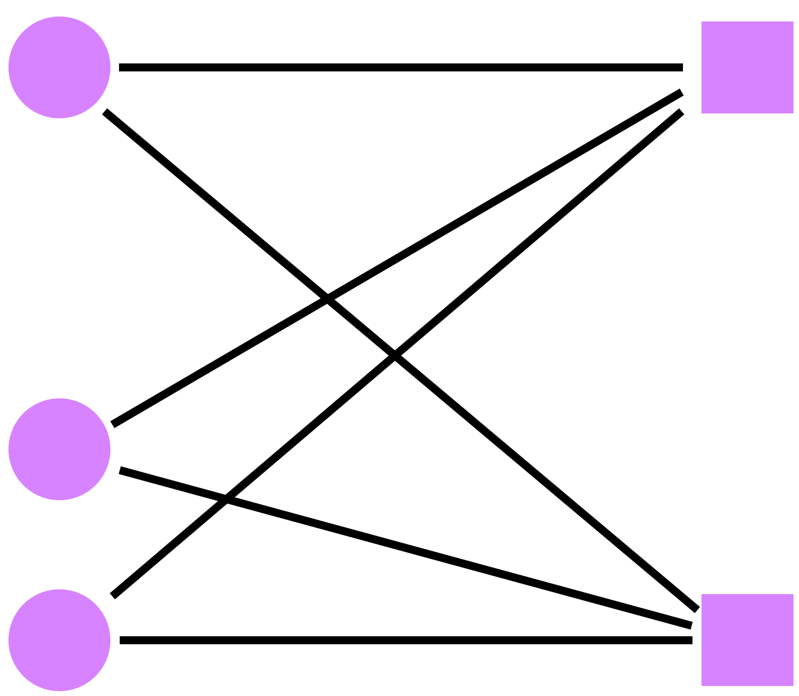


une coupe minimale
donne des explications

Problème du couplage parfait

Réduction. On peut résoudre le problème du couplage parfait en se ramenant à un problème de flot maximum.

Il existe un couplage maximal ssi le flot maximal vaut 5.



aucune chance !