Algorithmique et Complexité
TD 3/7 - Graphes à Flots

CentraleSupélec – Gif

ST2 – Gif

Plan

Introduction

Ce TD est l'occasion de s'exercer à la résolution de problèmes à base de graphes et d'approfondir la compréhension des problèmes traités en cours. Il adresse notamment le **problème du flot maximum**.

Plan

Question 1

Une société de distribution d'eau dispose de 3 pompes A, B et C, pour alimenter en eau 4 villages D, E, F et G (la demande des villages en m^3/s).

Pompe	Capacité
A	45
B	25
C	30

Village	Demande
D	30
E	10
F	20
G	30

Pompe / Village	D	E	F	G
A	10	15	10	25
B	20		15	5
C		10	5	10

La situation géographique des lieux et la configuration du réseau limitent les possibilités d'approvisionnement. En effet, le réseau d'acheminement se compose de canaux directs entre pompes et villages. Un canal relie une pompe à un village avec un débit maximum à ne pas dépasser.

Question 1 : correction

Le problème que l'on souhaite résoudre est double :

- Peut-on satisfaire la demande ?
- Comment choisir les débits que chaque pompe doit délivrer dans chaque canal ?

Question

Modéliser le problème sous forme d'un problème de flot.

Question 1 : correction

Pompe	Capacité
A	45
B	25
C	30

Village	Demande
D	30
E	10
F	20
G	30

Pompe / Village	D	E	F	G
A	10	15	10	25
B	20		15	5
C		10	5	10

Méthode

Comme pour le problème des résidences vu en cours.

➜ Un graphe à flot avec :

- un somment source s, un sommet terminal t,
- 3 sommets représentant les pompes (A, B, C),

  s relié à ces sommets par des arcs de capacité tel qu'indiqué dans le tableau 1
- 4 sommets pour représenter les villages (D, E, F G),

  relié à t par des arcs de capacité tel qu'indiqué dans le tableau 2
- Des arcs entre les sommets-pompes et les sommets-villages.

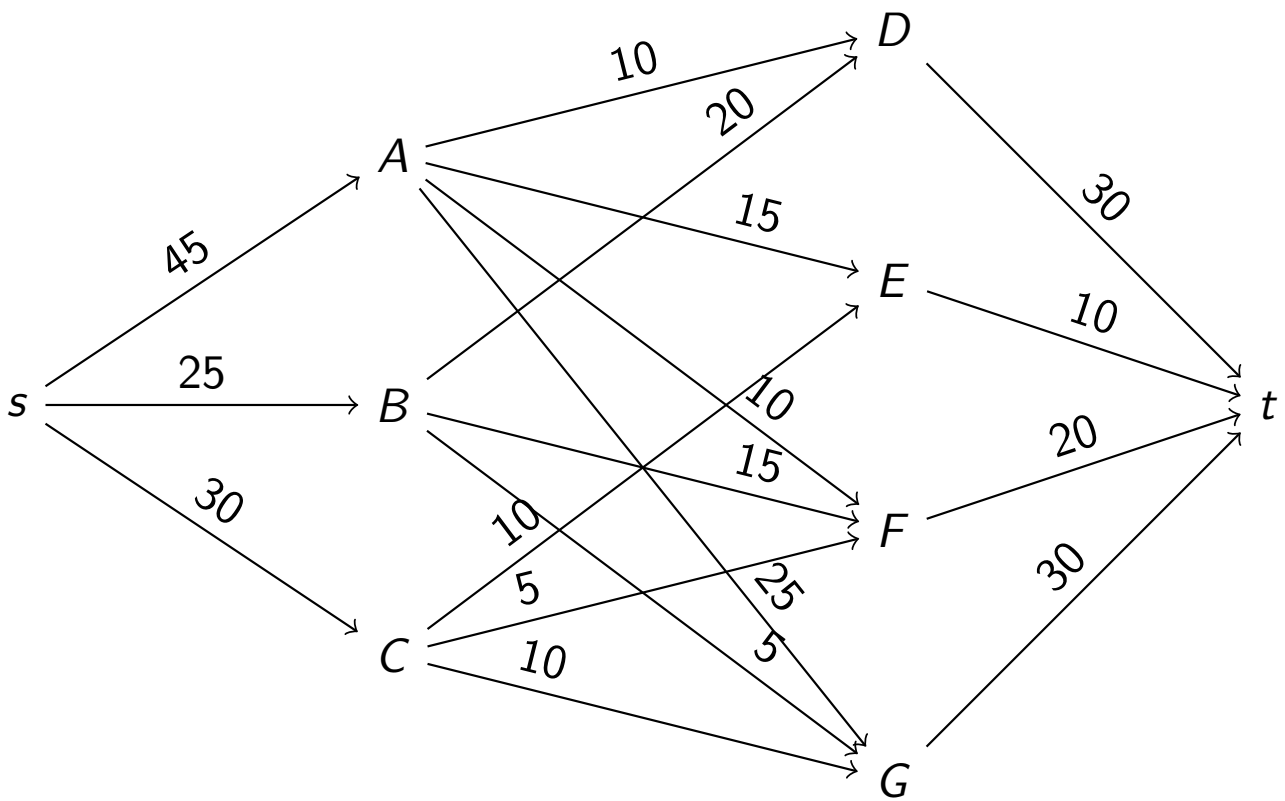
  de capacité tel qu'indiqué dans le tableau 3

Question 1 : correction

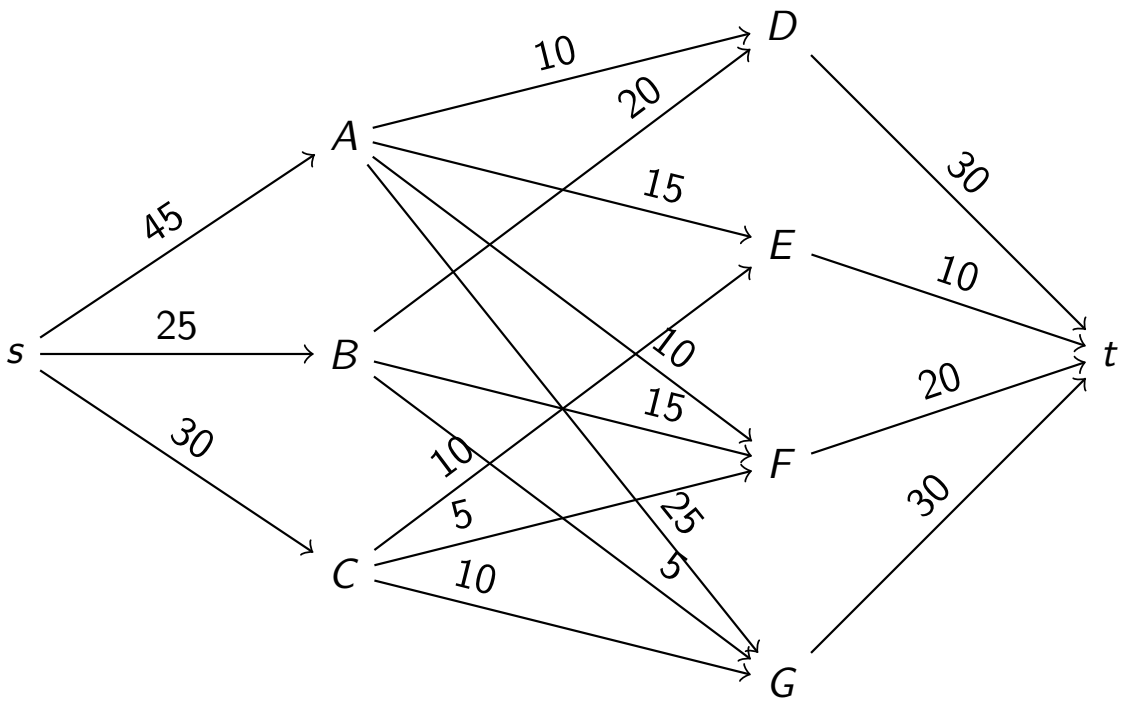
Pompe	Capacité
A	45
B	25
C	30

Village	Demande
D	30
E	10
F	20
G	30

Pompe / Village	D	E	F	G
A	10	15	10	25
B	20		15	5
C		10	5	10

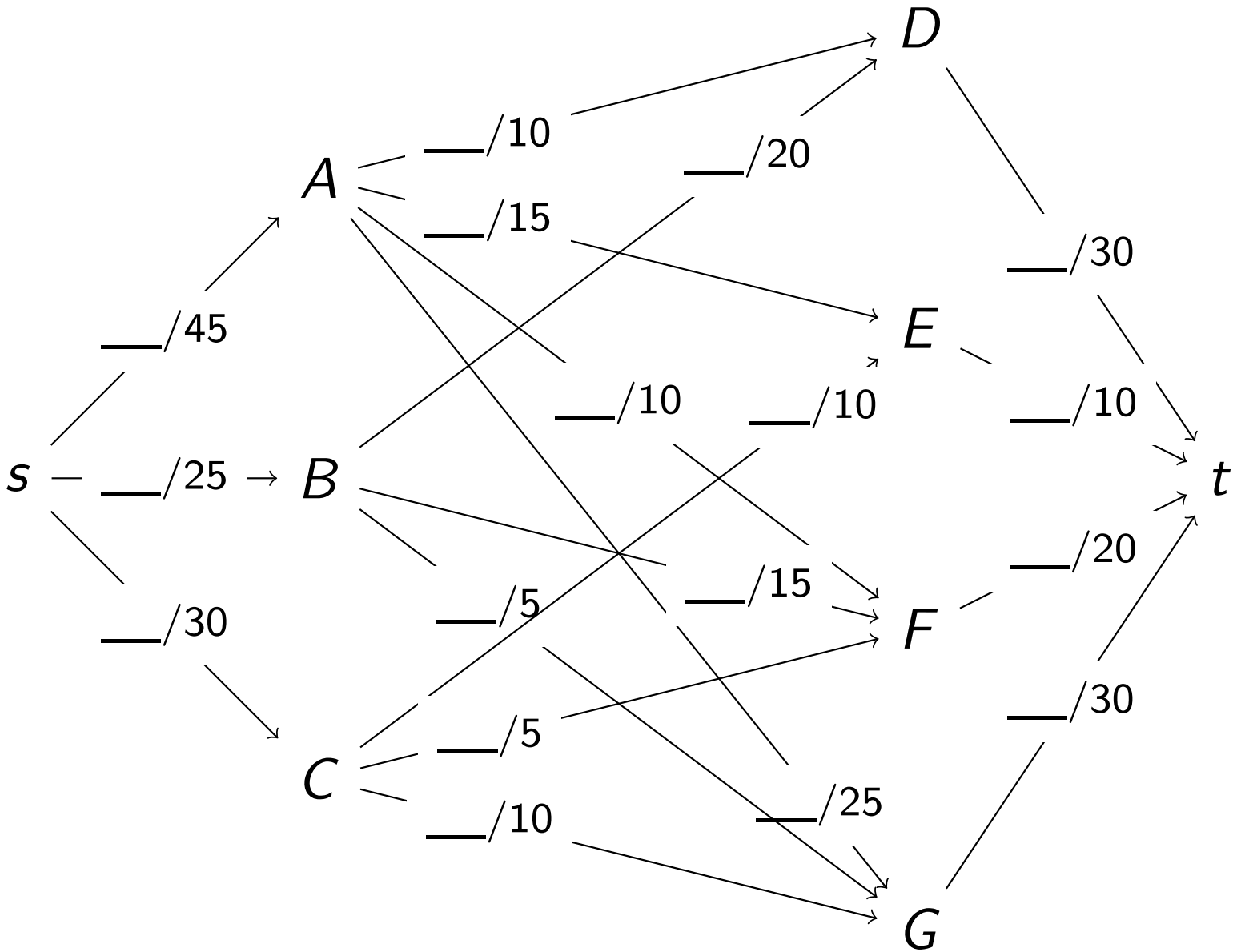


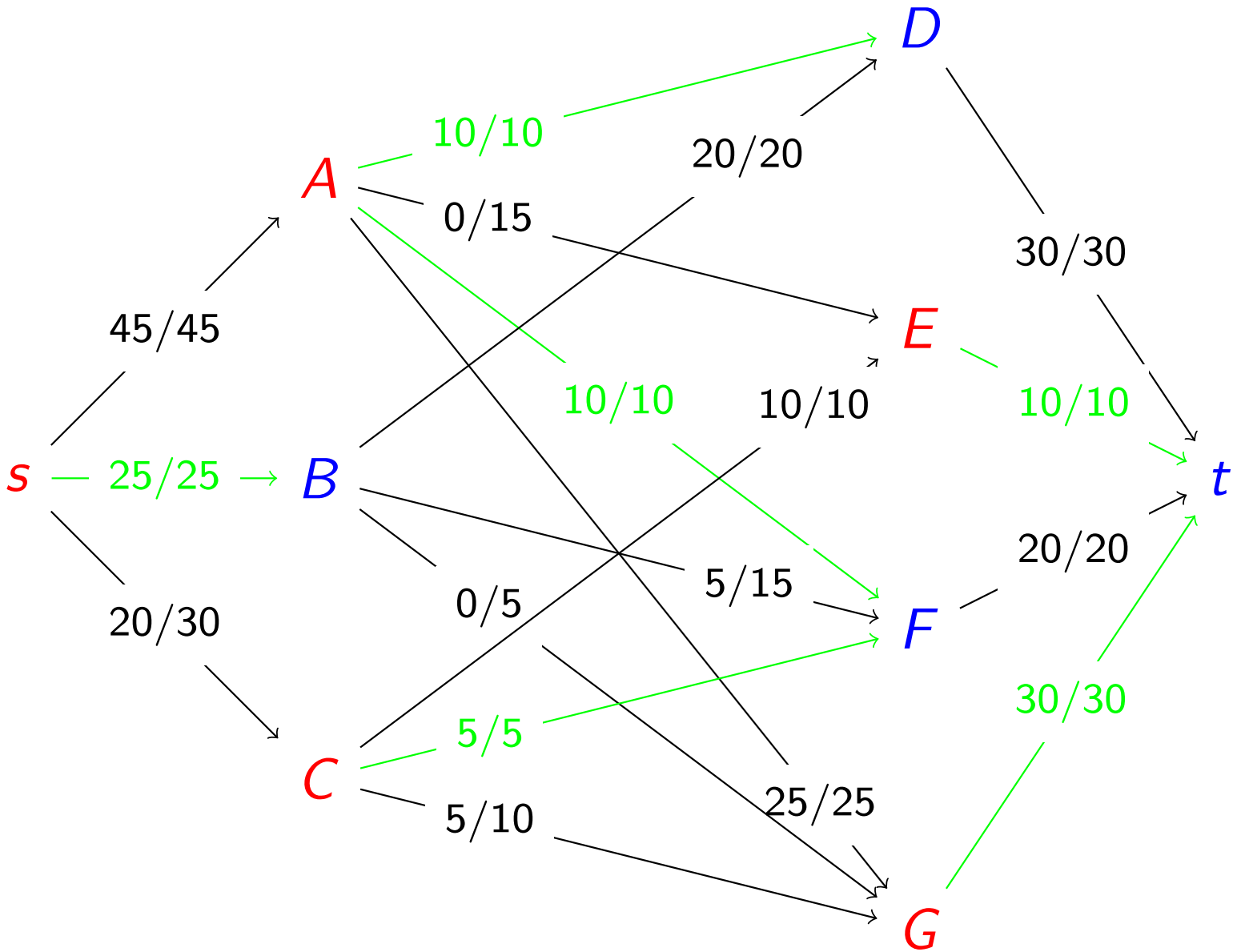
Question 2



Question

Appliquer l'algorithme de Ford-Fulkerson pour calculer une solution au problème.


D
A
/10
/20
/15
/30
/45
E
/10
/10
/10
s
/25
B
t
/20
/15
/5
F
/30
/5
C
/25
/10
G


D
10/10
A
20/20
0/15
30/30
45/45
E
10/10
10/10
10/10
s
25/25
B
t
20/20
5/15
0/5
20/30
F
30/30
5/5
C
25/25
5/10
G

Question 2 : correction

L'algo **s'arrête** et trouve par marquage la coupe de capacité min la plus proche de la source (ici il trouve la coupe est : $\{s, A, C, E, G\} \cup \{B, D, F, t\}$).

Autrement dit il s'agit de la coupe/partition dont le premier ensemble contenant s est de taille minimale.

Question 3

Question

Complexité de l'algorithme de Ford-Fulkerson ? Comment la réduire ?

Question 3 : correction

Réponse

- $\mathcal{O}(|V| + |E|) \times F)$ (vue en cours)

  F la valeur du flot max

Question 3 : correction

Réponse

- $\mathcal{O}(|V| + |E|) \times F)$ (vue en cours)

  F la valeur du flot max

- L'algo Ford-Fulkerson présenté en cours utilise un parcours en profondeur pour trouver une chaîne augmentante. Sauf qu'à l'origine l'algo de Ford-Fulkerson cherche une chaîne augmentante quelconque sans imposer un algo de parcours :

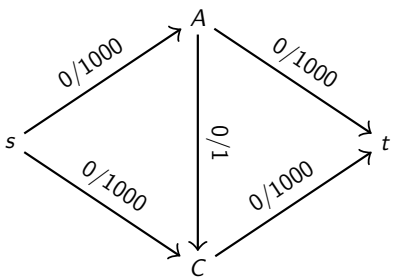
Question 3 : correction

Réponse

- $\mathcal{O}(|V| + |E|) \times F)$ (vue en cours)

  F la valeur du flot max

- L'algo Ford-Fulkerson présenté en cours utilise un parcours en profondeur pour trouver une chaîne augmentante. Sauf qu'à l'origine l'algo de Ford-Fulkerson cherche une chaîne augmentante quelconque sans imposer un algo de parcours :



  - 2000 étapes en alternant $s \to A \to C \to t$ et $s \to C \to A \to t$
  - 2 étapes ($s \to A \to t$ et $s \to C \to t$) !

Question 3 : correction

Réponse

Algorithme d'Edmonds–Karp = parcours en largeur

$\rightarrow$ Complexité $\mathcal{O}(|V| \times |E|^2)$

Question 3 : correction

Réponse

Algorithme d'Edmonds–Karp = parcours en largeur

$\rightarrow$ Complexité $\mathcal{O}(|V| \times |E|^2)$

La chaîne augmentante trouvée ainsi est le chemin le **plus court** en nombre d'arêtes qui possède du flot disponible. Ce qui résout l'exemple précédent en 2 étapes.

Question 3 : correction

Réponse

Algorithme d'Edmonds–Karp = parcours en largeur

$\rightarrow$ Complexité $\mathcal{O}(|V| \times |E|^2)$

La chaîne augmentante trouvée ainsi est le chemin le **plus court** en nombre d'arêtes qui possède du flot disponible. Ce qui résout l'exemple précédent en 2 étapes.

Cet algo ne dépend pas de la quantité du flot et présente une complexité polynomiale.

Question 3 : correction

Réponse

Algorithme d'Edmonds–Karp = parcours en largeur
$\rightarrow$ Complexité $\mathcal{O}(|V| \times |E|^2)$

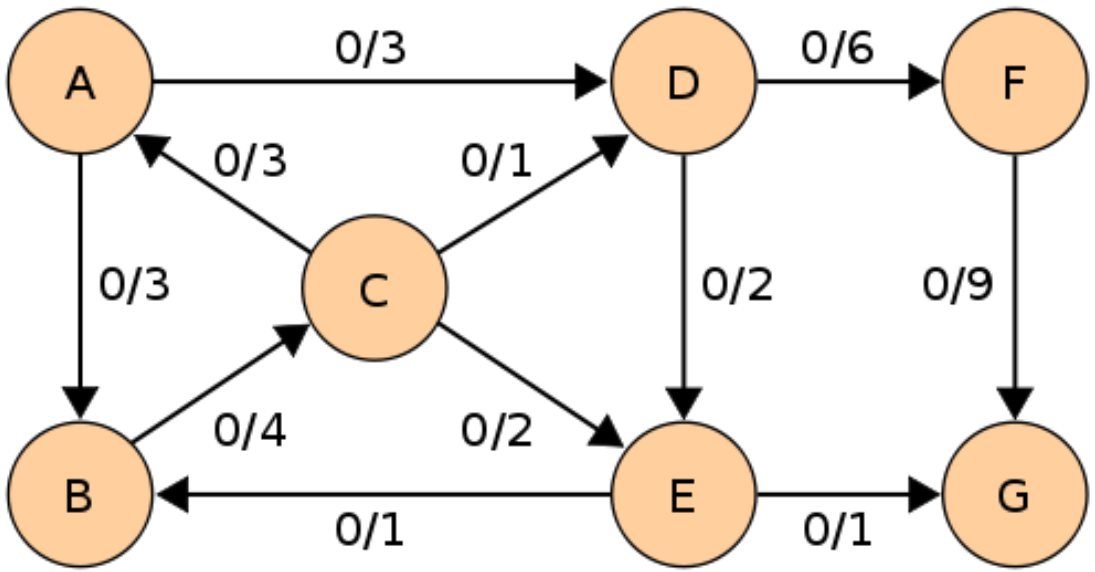
La chaîne augmentante trouvée ainsi est le chemin le **plus court** en nombre d'arêtes qui possède du flot disponible. Ce qui résout l'exemple précédent en 2 étapes.

Cet algo ne dépend pas de la quantité du flot et présente une complexité polynomiale.

La preuve de cette complexité est difficile : à chaque itération au moins un arc est saturé. Cela n'est pas définitif car il peut être dé-saturé (si on déleste du flot) dans une itération future.

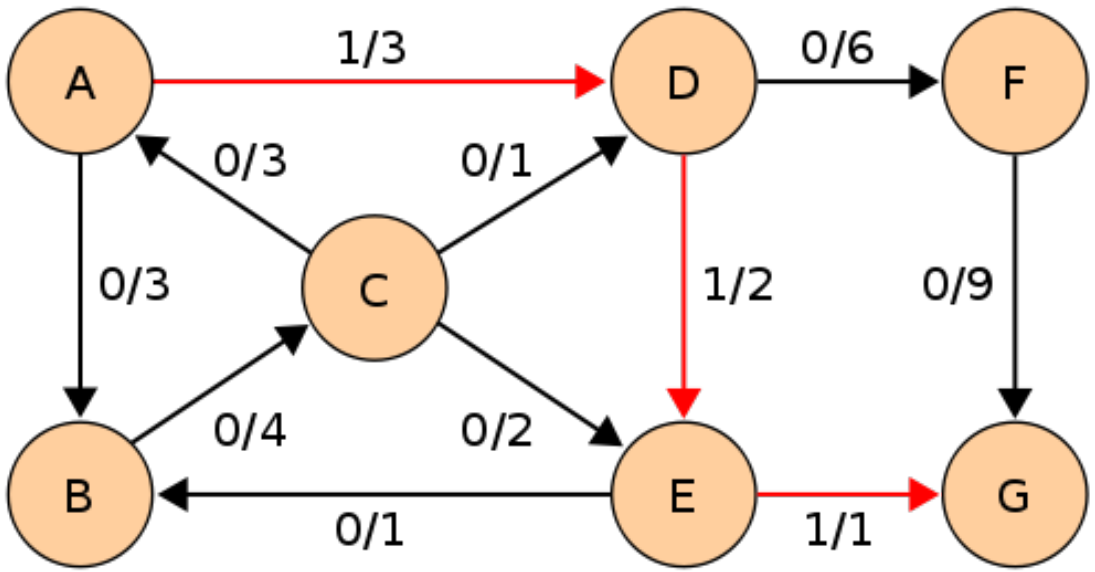
Question 3 : correction

Exemple



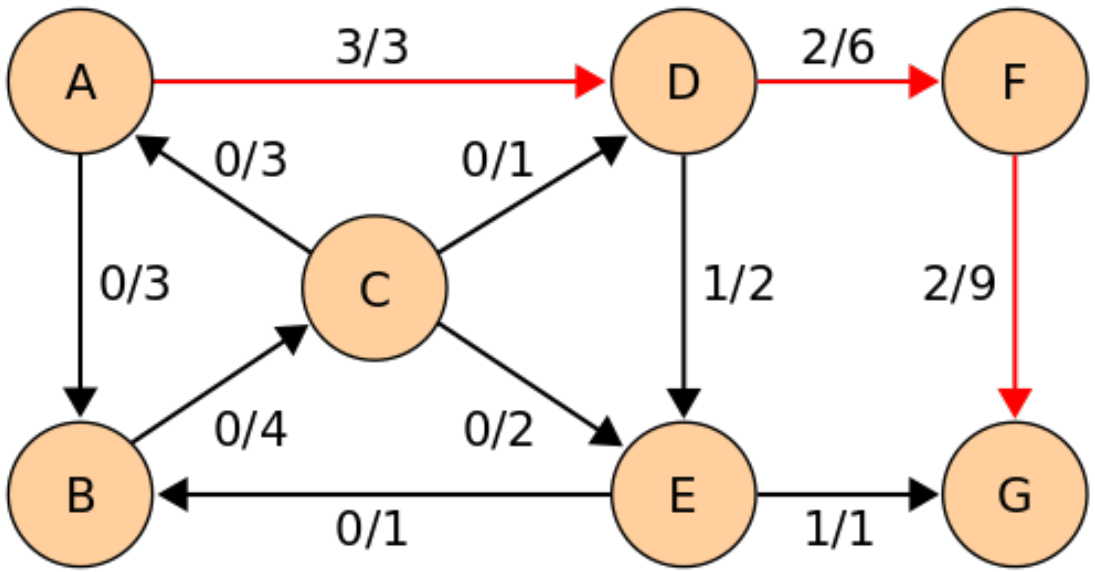
Question 3 : correction

Exemple



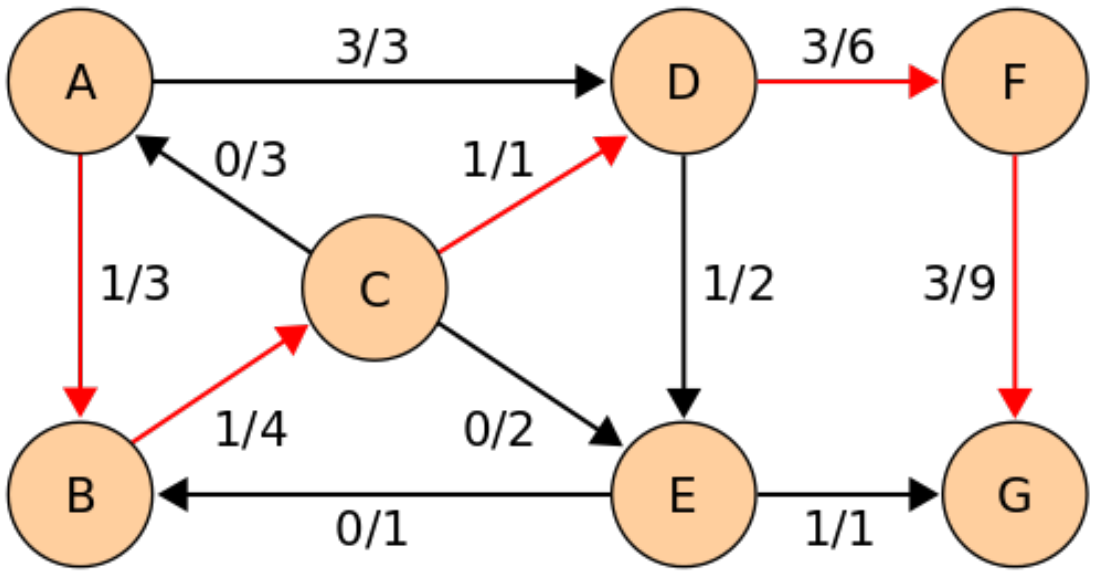
Question 3 : correction

Exemple



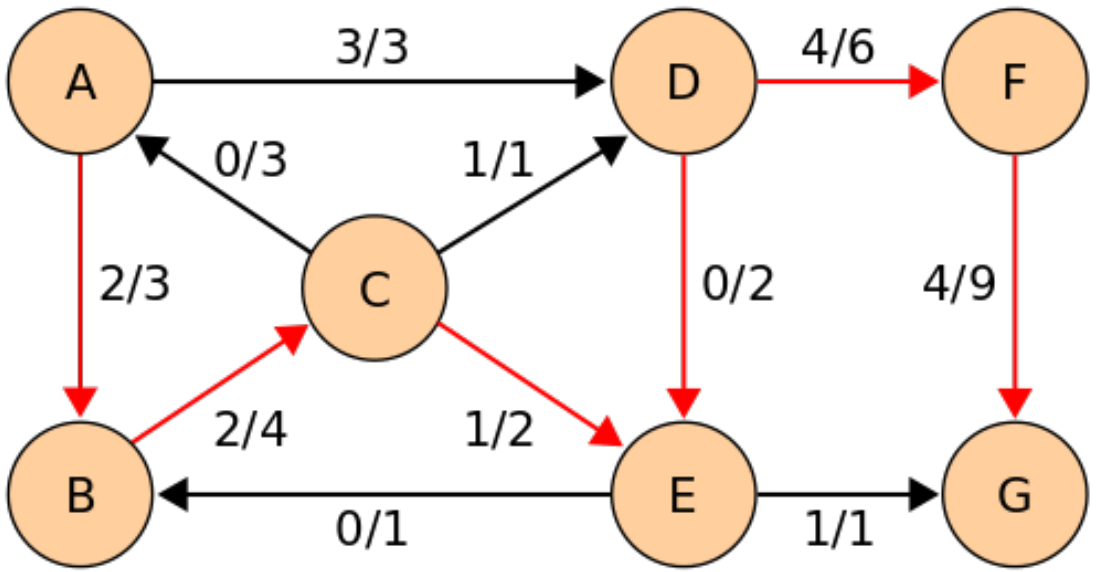
Question 3 : correction

Exemple



Question 3 : correction

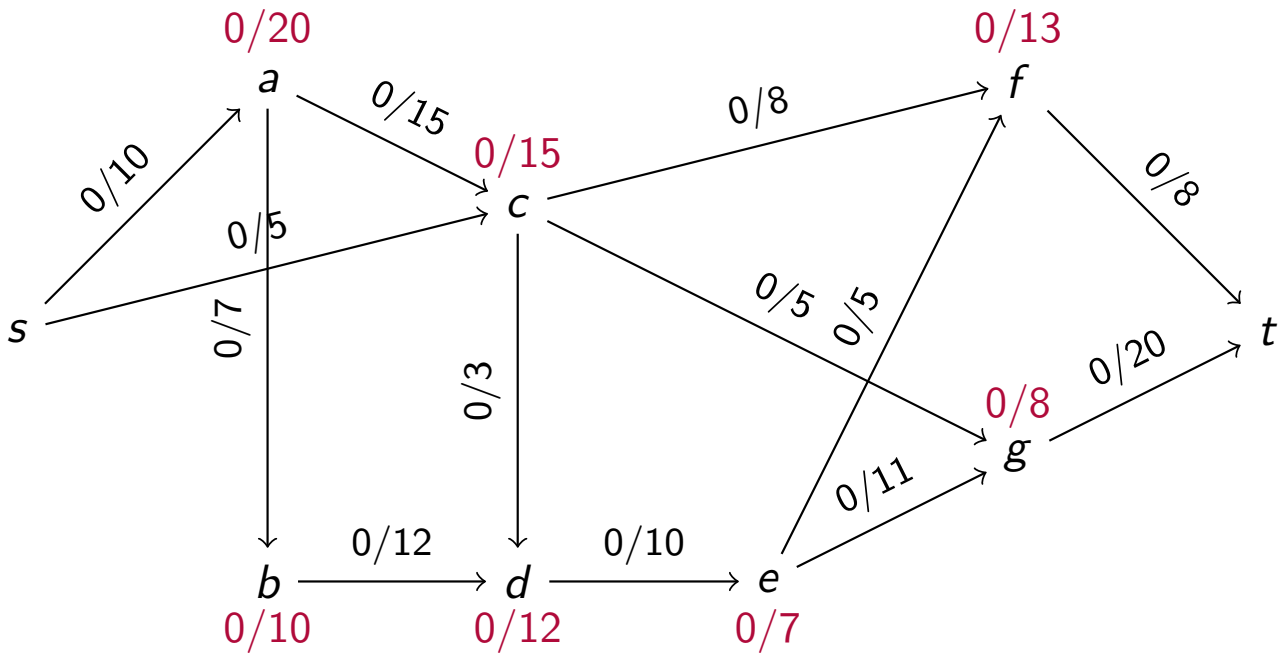
Exemple



Plan

Question 1

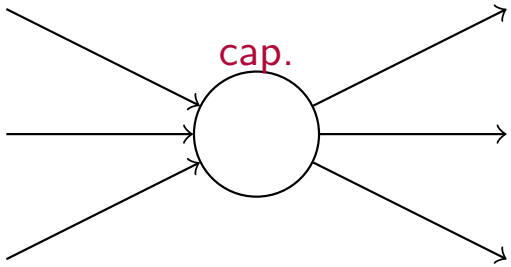
Considérons une variante au problème de flot maximum où les arêtes ainsi que les sommets ont des capacités à ne pas dépasser.



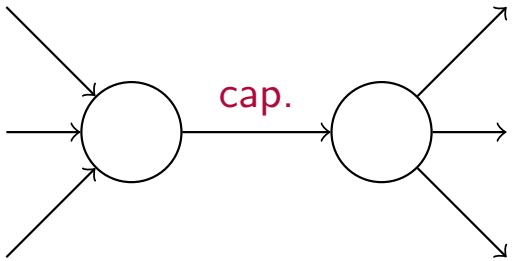
Question

Proposez une transformation pour réduire ce problème au problème ordinaire du flot maximum.

Question 1 : correction



Question 1 : correction



Question 1 : correction

Solution

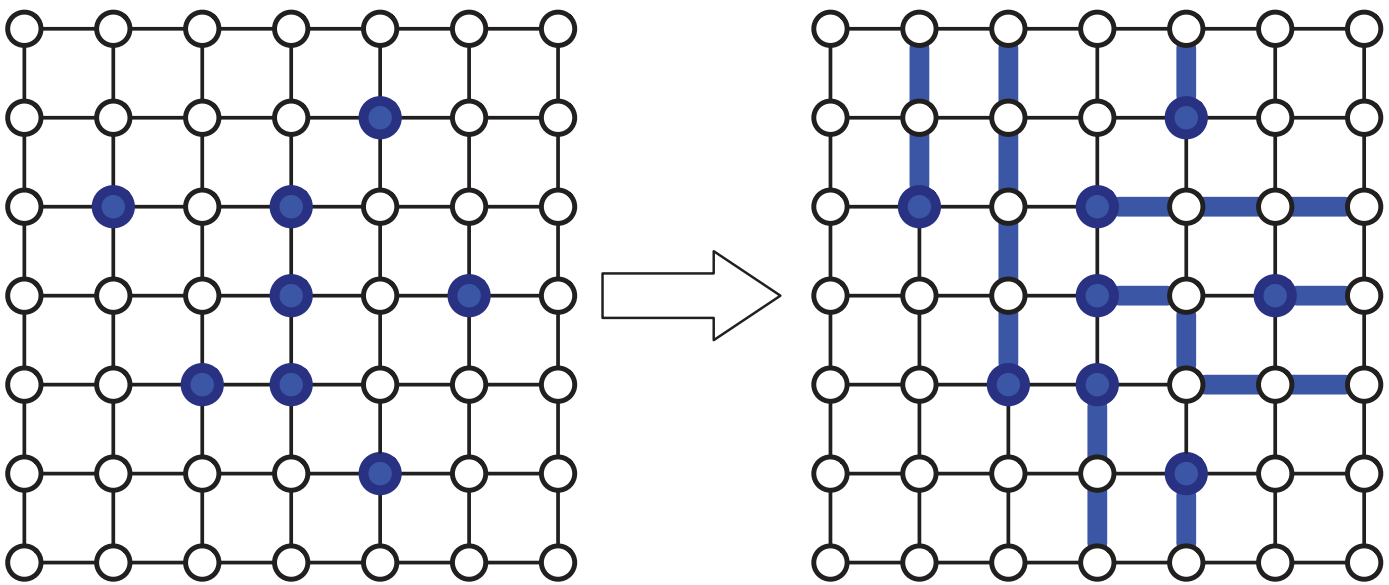
Transformer chaque sommet s en $s_1 \to s_2$

Formellement, on part d'un graphe $G = (V, E)$ avec des capacités sur les sommets $C_V : V \to \mathbb{R}$ et le arêtes $C_E : E \to \mathbb{R}$, puis on crée un graphe $G' = (V', E')$ et sa fonction de capacités $C : E' \to \mathbb{R}$:

- $V' = \{v_{in} \mid v \in V\} \cup \{v_{out} \mid v \in V\}$
- $E' = \{(u_{out}, v_{in}) \mid (u, v) \in E\} \cup \{(v_{in}, v_{out}) \mid v \in V\}$
- $\forall (u, v) \in E, \quad C(u_{out}, v_{in}) = C_E(u, v)$
- $\forall v \in V, \qquad C(v_{in}, v_{out}) = C_V(v)$

Question 2

Considérons une grille carrée de $n \times n$ points. On a m prisonniers dangereux dont les positions sont m points distincts de la grille ($m < n^2$). Le problème de l'évasion consiste à savoir s'il existe m chemins disjoints (n'ayant aucun point en commun) menant ces prisonniers à la sortie de la grille sans se croiser. La figure ci-dessous illustre un tel plan d'évasion.



Question 2

Question

Algorithme pour résoudre le problème de l'évasion

Question 2 : correction

Algorithme

Transformer en problème de flot :

(variante sommets avec capacités)

1. Ajouter un sommet source s et le connecter aux m positions de départ.
2. Ajouter un sommet terminal t et le connecter aux $4n - 4$ points de sortie.
3. Chaque arête non-orientée $\{u, v\}$ de la grille est transformée en deux arêtes orientées (u, v) et (v, u).
4. Chaque arête ainsi que chaque sommet a une capacité égale à 1.

➜ Il suffit de calculer le flot max et vérifier s'il vaut m.