Université Clermont Auvergne
BUT Info 1A
R207 - Graphes

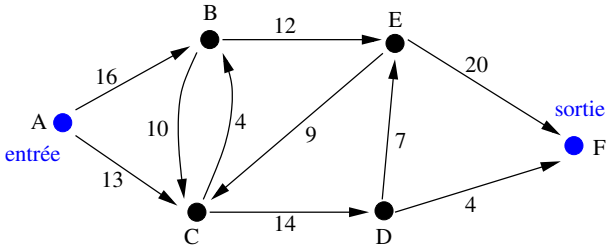
Année 2021-2022

Florent Foucaud
Malika More
Thibault Ralet
Carine Simon
Thierry Trévisan

Flots dans les graphes orientés

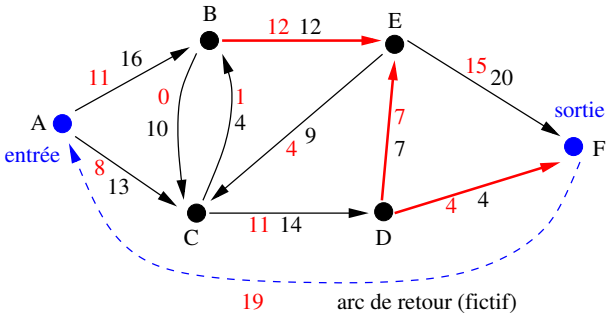
1 Introduction

Réseau de transport $R = (V, E, C, \textbf{entrée}, \textbf{sortie})$ Graphe orienté connexe sans boucle



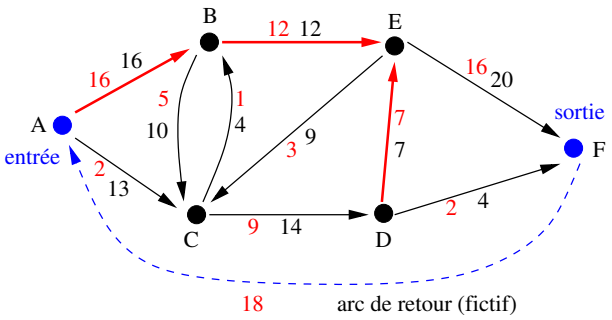
- Sommets particuliers :
  - Entrée : A (degré entrant nul)
  - Sortie : F (degré sortant nul)
- Chaque arc (u, v) possède une capacité $c(u, v) \geqslant 0$

Flot à travers un réseau



- Flux à travers l'arc (C, D) : $\varphi(C, D) = 11$
- Valeur totale du flot $\varphi_0 = 19$ (à travers l'arc de retour)
- Arcs (B, E), (D, E) et (D, F) saturés (en rouge)
- Conservation des flux : loi des nœuds ou *de Kirchhoff* (1847 circuits éléctriques)
  « En chaque sommet, ce qui entre est égal à ce qui sort. »
  - En A : $19 = 11 + 8$
  - En B : $11 + 1 = 0 + 12$
  - En C : $8 + 0 + 4 = 1 + 11$, etc.
- Ce flot est **réalisable** : pour tout arc (u, v), on a bien $\varphi(u, v) \leqslant c(u, v)$

Un autre flot réalisable à travers le même réseau



Quel est la valeur totale du flot dans cet exemple ?

Vocabulaire

- Un *réseau* est un graphe orienté avec 2 sommets particuliers s (l'*entrée*) et t (la *sortie*) et une *capacité* entière $c[a] \geqslant 0$ pour chaque arc a.
- Un *flot* dans ce réseau est une valuation entière $\phi(a) \geqslant 0$ pour chaque arc a satisfaisant la *loi des nœuds*.
- Un flot est *admissible* ssi pour tout arc a on a $\phi(a) \leqslant c[a]$.
- La *valeur du flot* ou *flot total* est égale au flot sur l'arc de retour (l'arc fictif ajouté de t vers s).

2 Problème du flot maximal

Problème

- Donnée : un réseau
- Question : trouver un *flot réalisable maximal* (dont la valeur est maximale)

Solution Algorithme de Ford-Fulkerson (1956)

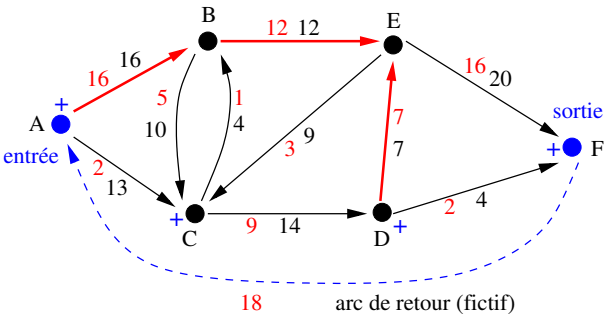
Idée générale

- Procéder par marquages successifs des sommets depuis l'entrée vers la sortie.
- On traite chaque sommet u marqué successivement.
- On essaye de marquer tous les voisins non marqués de u
  - les successeurs avec un $(+)$ *si l'arc n'est pas saturé*
  - puis les prédécesseurs avec un $(-)$ *si l'arc possède un flux non nul*
- Ceci permet de trouver des *chemins augmentants* de l'entrée vers la sortie, s'il en existe.

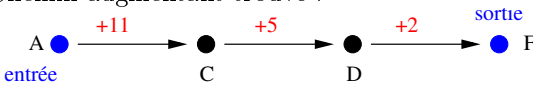
Algorithme de Ford-Fulkerson Recherche de chemins augmentants

Rappel : les arcs saturés sont en rouge

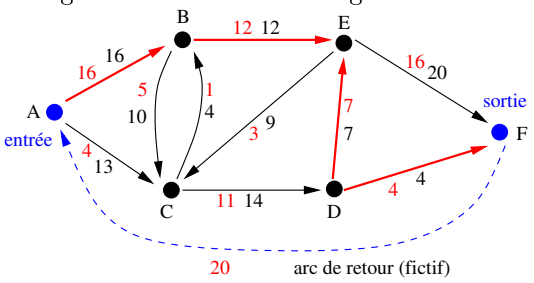
Étape 1



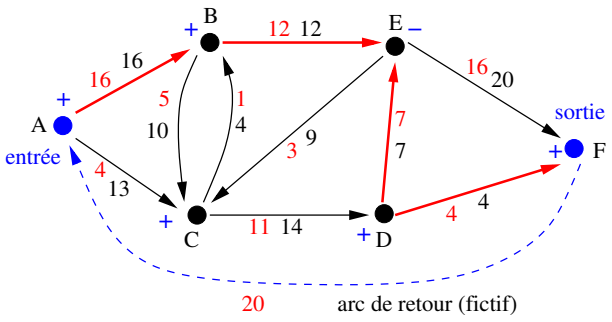
Chemin augmentant trouvé :



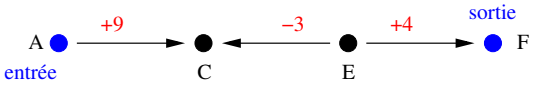
On augmente le flot de 2 le long de ce chemin :



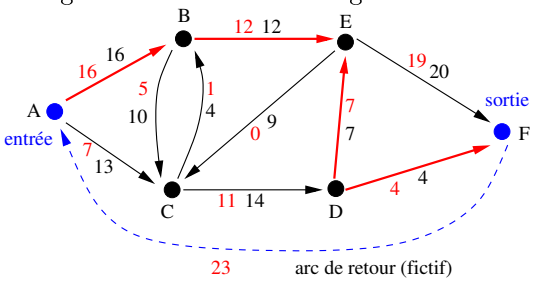
Étape 2



Chemin augmentant trouvé :

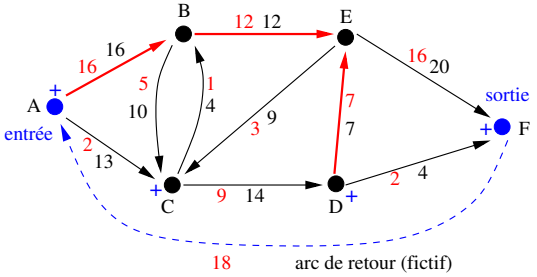


On augmente le flot de 3 le long de ce chemin :

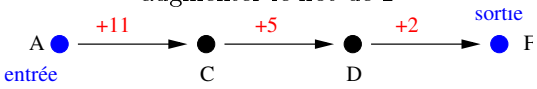


Chemins augmentants

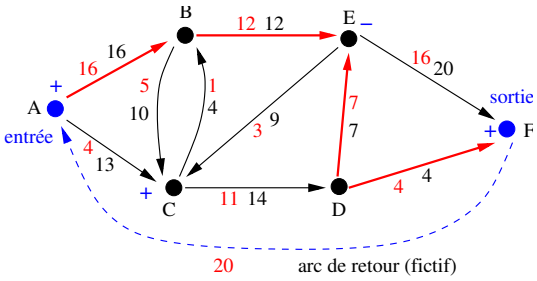
Le cas intuitif : on « ouvre des robinets ».



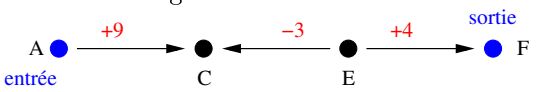
augmenter le flot de 2



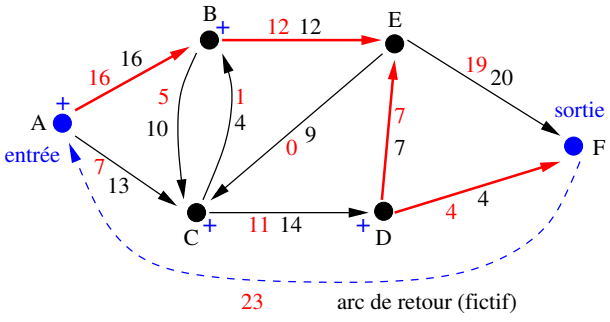
Le cas moins intuitif : on « *ferme des robinets* et on en ouvre d'autres ».



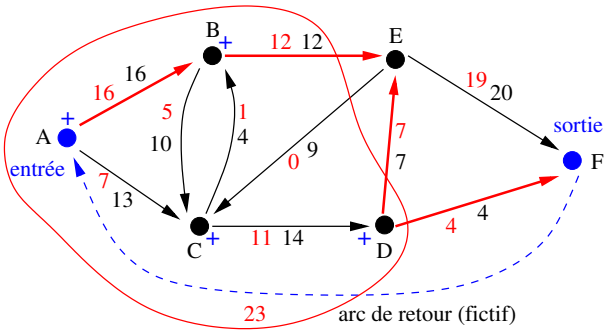
augmenter le flot de 3



Condition d'arrêt



On ne peut plus marquer la sortie : le flot obtenu est maximal



Autre façon de prouver la maximalité du flot : on a trouvé une *coupe* saturée (voir plus loin)

Algorithme 1 : Algorithme de Ford-Fulkerson

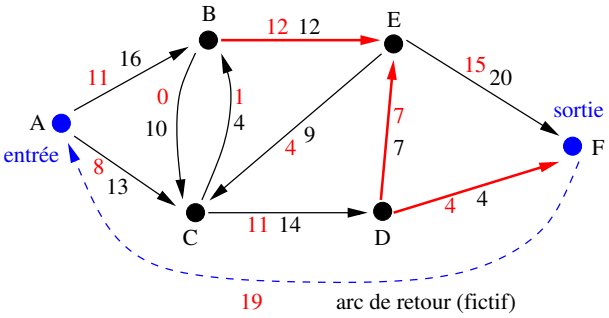
Données (G, c, s, t) *un réseau*
début
  initialisation *marquer* $+$ *le sommet entrée* s
  tant que *le flot n'est pas maximal* **faire**
    tant que *on marque des sommets* **faire**
      pour chaque *sommet marqué* u *non encore traité* **faire**
        pour chaque *arc* (u, v) **faire**
          si v *n'est pas marqué et* (u, v) n'est pas saturé **alors**
            marquer v par $(+, u)$
          fin si
        fin pour chaque
        pour chaque *arc* (v, u) **faire**
          si v *n'est pas marqué et* (v, u) a un flot non nul **alors**
            marquer v par $(-, u)$
          fin si
        fin pour chaque
      fin pour chaque
      si la sortie t n'est pas marquée **alors**
        le flot est maximal (on s'arrête)
      sinon
        augmenter le flot, démarquer les sommets (sauf l'entrée) et continuer
      fin si
    fin tq
  fin tq
fin

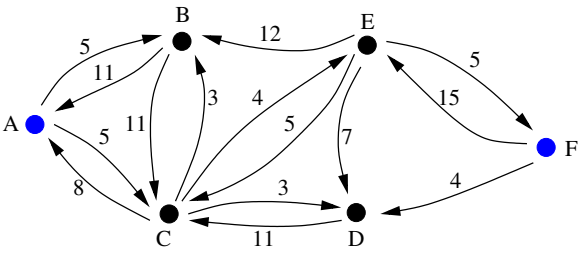
Deux notions Il y a deux notions importantes au cœur de Ford-Fulkerson.

- Chemin augmentant
- Coupe

Pour définir proprement la notion de chemin augmentant, nous allons introduire un graphe orienté associé à un flot dans un réseau : le *graphe d'écart*.

Graphe d'écart

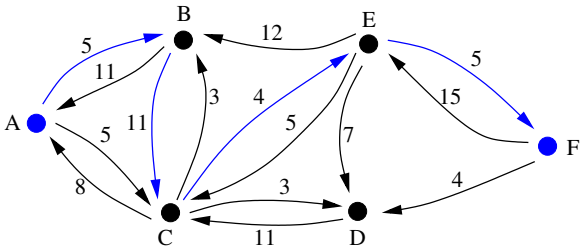


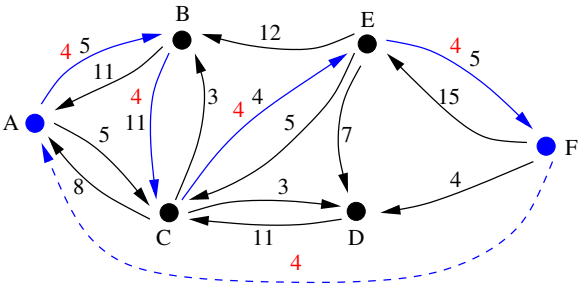


- Mêmes sommets
- Arcs dans les deux sens
- Capacités = écarts
- Supprimer les arcs de capacité nulle (saturation et flux nul)

Chemin augmentant

- Chemin de l'entrée vers la sortie dans le graphe d'écart
- Flot réalisable de valeur > 0 à travers le graphe d'écart
- Permet d'augmenter le flot initial





Retour à l'algorithme

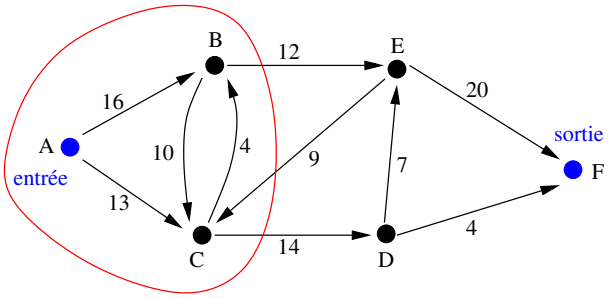
- L'algorithme de Ford-Fulkerson trouve des chemins augmentants, s'il en existe.
- Implémentation efficace ?
  - Toute la difficulté est de bien choisir les chemins augmentants.
  - Une bonne méthode : utiliser un *parcours en largeur* du graphe d'écart.
  - Permet d'atteindre plus rapidement un flot maximal.

Résumé : les notions derrière Ford-Fulkerson

- *Ford Fulkerson* procède par marquage successifs depuis le sommet s pour trouver un *chemin augmentant* par parcours en *largeur* du *graphe d'écart*
- Un *chemin augmentant* est un chemin de s à t dans le graphe d'écart ; il *permet d'augmenter le flot* dans le réseau initial.
- *Ford Fulkerson* recommence la recherche d'un chemin augmentant tant que possible.
- Lorsque ce n'est plus possible on a trouvé une *coupe* saturée.

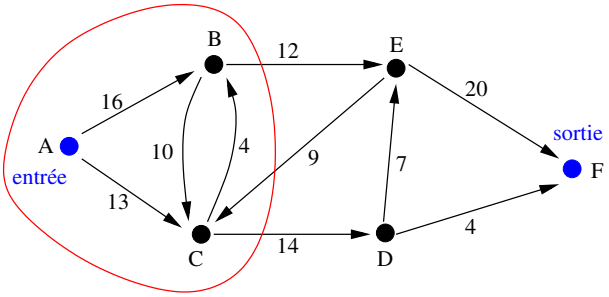
3 Théorème de la coupe

Coupe Partition des sommets en deux groupes disjoints, l'un, noté $\mathcal{E}$, contenant l'entrée (A) et l'autre la sortie (F).

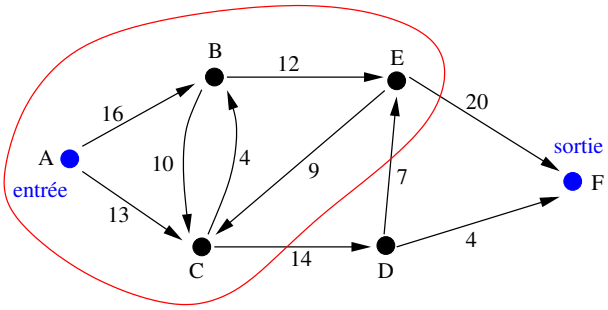


Capacité d'une coupe Somme des capacités des arcs *sortants* de $\mathcal{E}$

Arc sortant de $\mathcal{E}$ = allant d'un sommet de $\mathcal{E}$ vers un sommet extérieur à $\mathcal{E}$

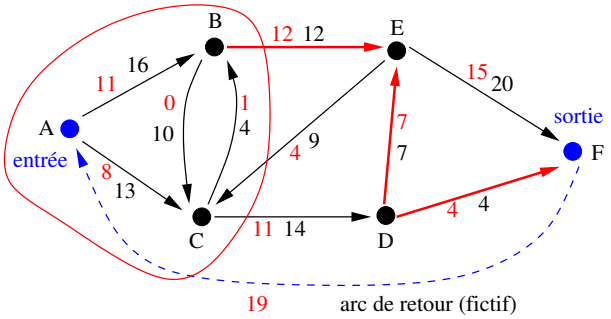


$= 12 + 14 = 26$



$= 14 + 20 = 34$

Flot net à travers une coupe Somme *algébrique* des flux sur les arcs entre $\mathcal{E}$ et l'extérieur



Capacité de la coupe = 26 Valeur du flot = 12-4+11 = 19

Corollaire important

- La valeur de *n'importe quel* flot réalisable est toujours inférieure ou égale à la capacité de *n'importe quelle* coupe.
- S'il y a égalité, c'est que le flot est maximal et la coupe est minimale.

Théorème de la coupe Les trois énoncés suivants sont équivalents :

1. Le flot est maximal
2. Le graphe d'écart ne contient pas de chemin augmentant
3. La valeur du flot est égale à la capacité d'une coupe

Preuve : avec ce qu'on a vu jusqu'ici, il est assez clair que $1 \Longrightarrow 2$ et que $3 \Longrightarrow 1$ (à détailler !).

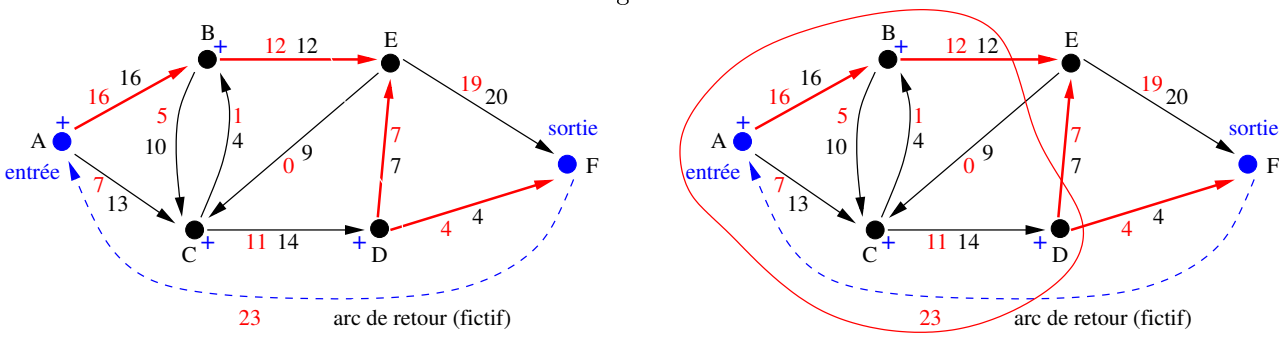
Il reste à prouver que $2 \Longrightarrow 3$. Pour cela, appelons $\mathcal{E}_0$ l'ensemble de sommets marqués par l'algorithme de Ford-Fulkerson une fois qu'il est bloqué. On voit facilement que

- L'entrée appartient à $\mathcal{E}_0$, et la sortie n'y appartient pas : c'est donc une coupe.
- Les arcs sortants de $\mathcal{E}_0$ sont forcément saturés.
- Les arcs entrants de $\mathcal{E}_0$ ont forcément un flux nul.

En conséquence, le flot à travers la coupe $\mathcal{E}_0$ est bien égal à sa capacité (coupe saturée).

Conséquences

- L'algorithme de Ford-Fulkerson calcule bien un flot maximum
- Méthode de vérification : À la fin de l'algorithme de Ford-Fulkerson



Valeur du flot maximal = 23 = Capacité de la coupe donnée par les sommets marqués

4 Conclusion

- L'*algorithme de Ford-Fulkerson* pour calculer un *flot maximal* dans un réseau.
- Cet algorithme calcule une séquence de *chemins augmentants* (permettant d'augmenter le flot total) à partir d'un flot de départ donné (par exemple le flot nul).
- *Lorsqu'il s'arrête*, cet algorithme détecte en fait une *coupe de même capacité que le flot calculé.*
- Par le *théorème de la coupe* il s'ensuit que la *coupe est minimale* et le *flot maximal.*
- Les implémentations efficaces de Ford -Fulkerson utilisent un parcours en largeur.
- Les flots permettent de résoudre efficacement d'autres problèmes : couplage maximum dans un graphe biparti, nombre maximum de chemins disjoints...