

Projet pratique Algorithme / Complexité / Calculabilité

Jean-Marc Lagniez, Viktor Lesnyak, Pierre-Alexandre
Cimbe, Ahmed Rafik

Master Informatique - Université Montpellier II

2013

Plan

- 1 Les algorithmes étudiés
 - Ford-Fulkerson
 - Edmonds-Karp
 - Dinic
 - Capacity Scaling
- 2 Experimentation et Performance
 - Performance de l'ordinateur utilisitine
 - Temps d'execution en fonction du nombre de sommets comptant le temps de crion des graphes
 - Evolutions porter
- 3 Demonstration du fonctionnement sous TIKZ
 - Demonstration du fonctionnement sous TIKZ

Plan

- 1 Les algorithmes étudiés
 - Ford-Fulkerson
 - Edmonds-Karp
 - Dinic
 - Capacity Scaling
- 2 Experimentation et Performance
 - Performance de l'ordinateur utilisitine
 - Temps d'execution en fonction du nombre de sommets comptant le temps de crion des graphes
 - Evolutions porter
- 3 Demonstration du fonctionnement sous TIKZ
 - Demonstration du fonctionnement sous TIKZ

Plan

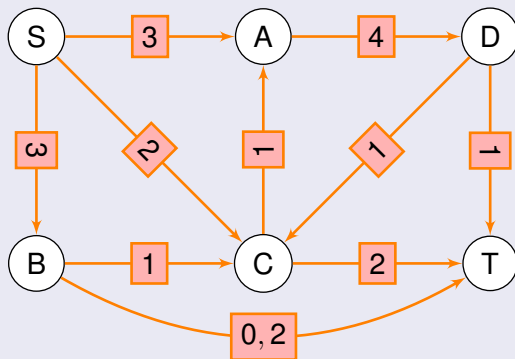
- 1 Les algorithmes étudiés
 - Ford-Fulkerson
 - Edmonds-Karp
 - Dinic
 - Capacity Scaling
- 2 Experimentation et Performance
 - Performance de l'ordinateur utilisitine
 - Temps d'execution en fonction du nombre de sommets comptant le temps de crion des graphes
 - Evolutions porter
- 3 Demonstration du fonctionnement sous TIKZ
 - Demonstration du fonctionnement sous TIKZ

Plan

- 1 Les algorithmes étudiés
 - Ford-Fulkerson
 - Edmonds-Karp
 - Dinic
 - Capacity Scaling
- 2 Experimentation et Performance
 - Performance de l'ordinateur utilisitine
 - Temps d'execution en fonction du nombre de sommets comptant le temps de crion des graphes
 - Evolutions porter
- 3 Demonstration du fonctionnement sous TIKZ
 - Demonstration du fonctionnement sous TIKZ

Algo Ford-Fulkerson

Graphe initiale

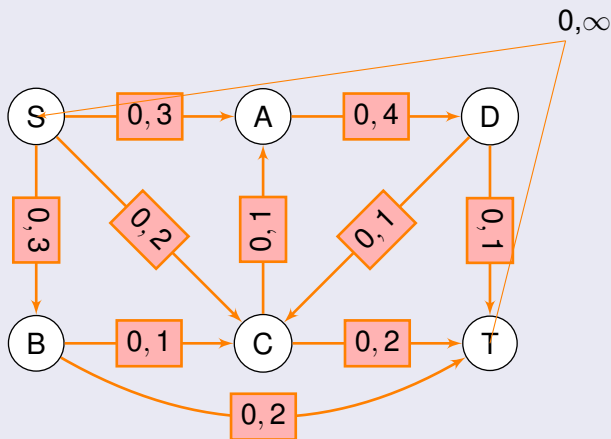


Graphe initiale

Soit $G = (V, E)$ un graphe, avec V -ensemble des arcs et E -ensemble des sommets.

Algo Ford-Fulkerson

Graphe d'ecart

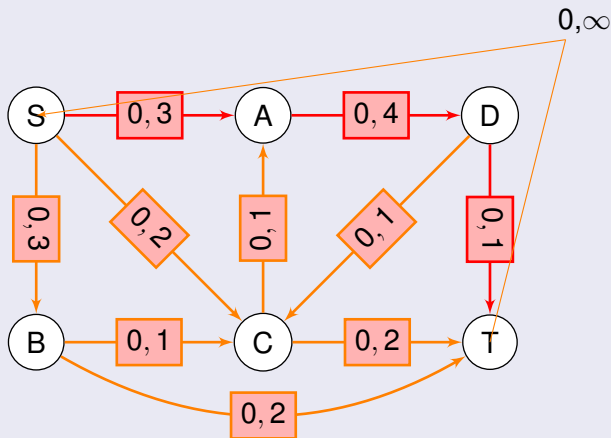


Graphe d'ecart

Pour passer de notre graphe G au Graphe d'ecart G_e on applique un flot null sur toutes les arcs et on ajout un arc qui va de la source(S) vers le puit(T).

Algo Ford-Fulkerson

Chemin ameliorant

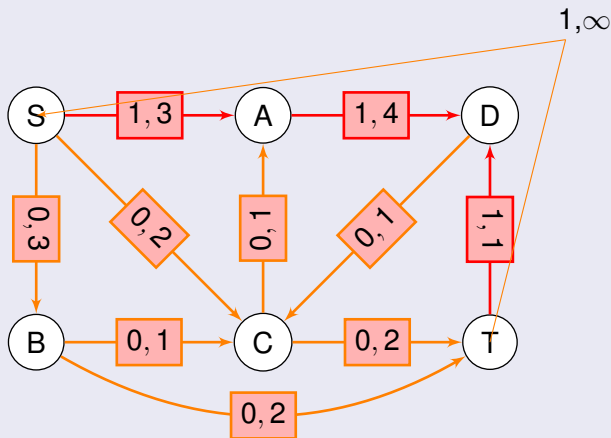


Chemin ameliorant

Ensuite on choisi un chemin ammeliorant sur le graphe d'ecart obtenue grace a un parcour en largeur.

Algo Ford-Fulkerson

Chemin ameliorant

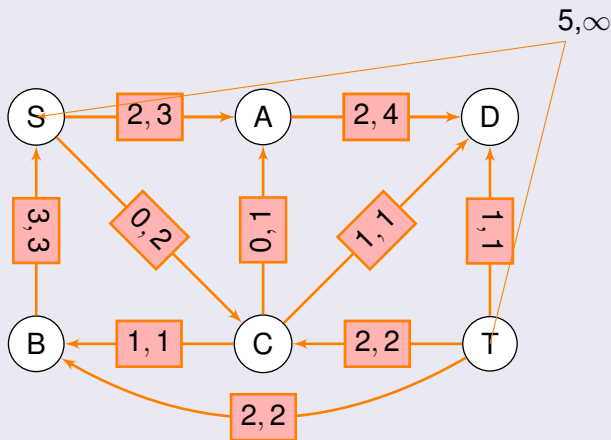


Chemin ameliorant

En utilisant le flot le plus petit de ce chemin on met à jour le graphe d'ecart.

Algo Ford-Fulkerson

Graphe finale

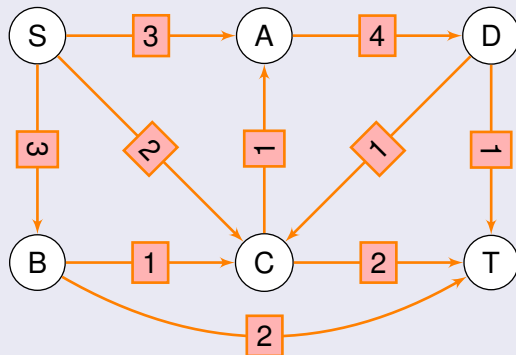


Graphe finale

Une fois tout les chemins ameliorants sont parcouru, on obtien un graphe d'ecart complet avec le flot maximal (das notre cas c'est 5).

Algo Edmonds-Karp

Graphe initiale

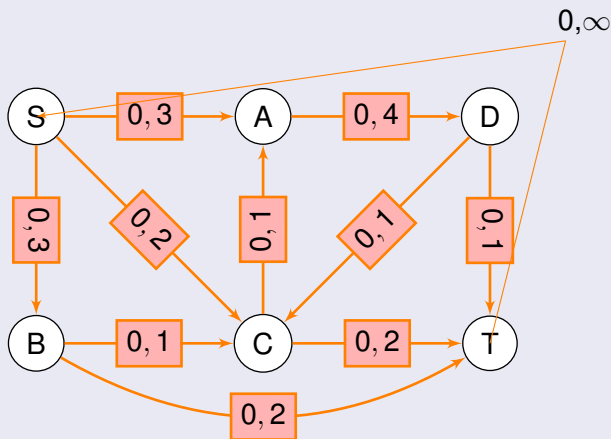


Graphe initiale

On reprends le
meme graphe
initiale.
Soit $G = (V, E)$ un
graphe, avec
 V -ensemble des
arcs et E -
ensemble des
sommets.

Algo Edmonds-Karp

Graphe d'ecart

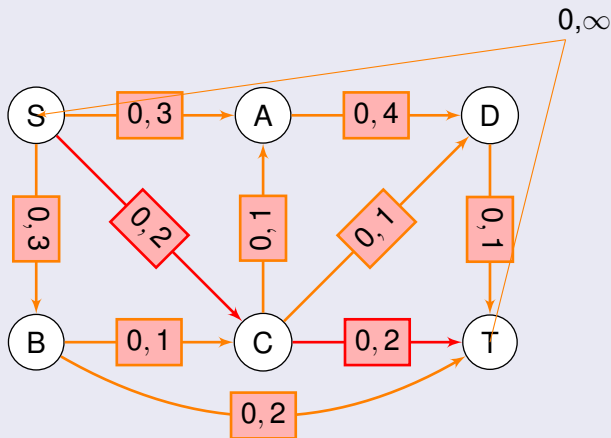


Graphe d'ecart

On refait a nouveau a partir de graphe G le Graphe d'ecart G_e on applique un flot null sur toutes les arcs et on ajout un arc qui va de la source(S) vers le puit(T).

Algo Edmonds-Karp

Chemin ameliorant



Chemin ameliorant

On choisi ici un chemin ameliorant en fonction de plus court chemin, qui dans notre cas est calcule avec l'algo de Dijkstra

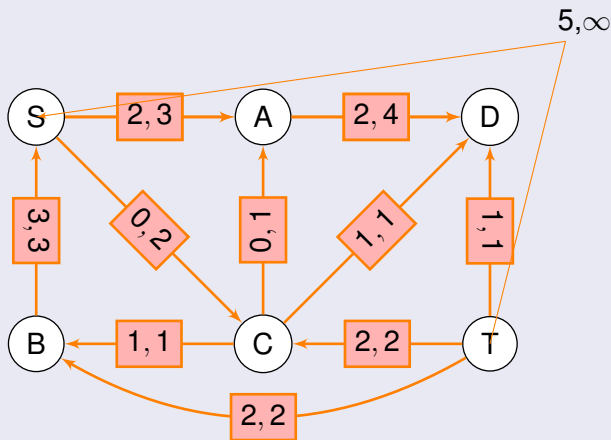
Chemin ameliorant



En utilisant le
flotte plus petit de
ce chemin on
met a jour le
graphe d'ecart.

Algo Edmonds-Karp

Graphe finale



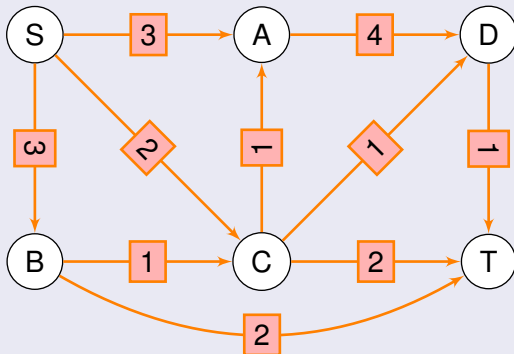
Graphe finale

Une fois tout les chemins ameliorants sont parcouru, on obtien un graphe d'ecart complet avec le flot maximal (das notre cas c'est 5).

AlgoD

Algo Capacity Scaling

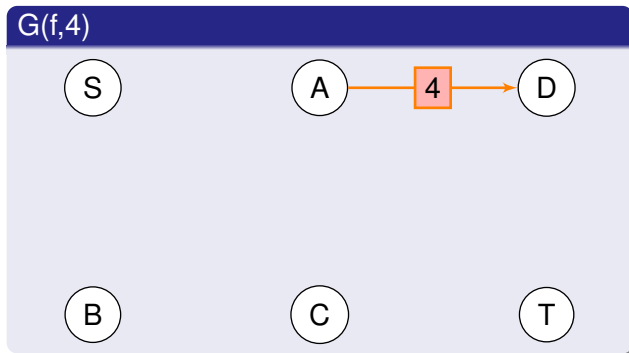
Graphe initial



Initialisation

$C=4$;
 $\Delta = 4$;
F le flot max = 0 ;

Algo Capacity Scaling



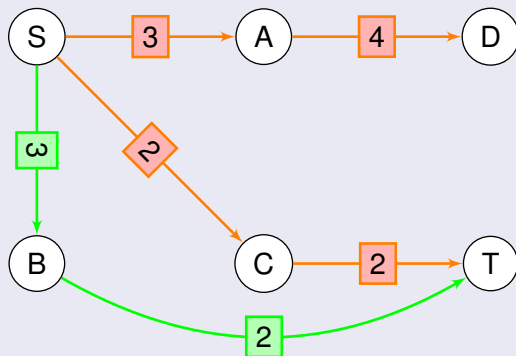
Augmentation

Pas de chemin
de s

$$\Delta = \Delta / 2 = 2$$

Algo Capacity Scaling

$G(f, 2)$



Augmentation

Premier chemin :

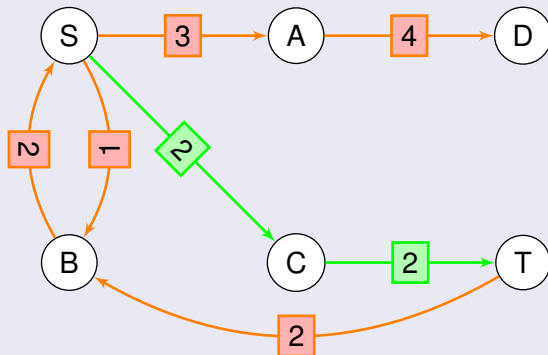
$s \rightarrow b \rightarrow t$

$\delta = 2$

$F = 2$

Algo Capacity Scaling

$G(f, 2)$



Mise à jour du
graphe d'augmenter et
Augmentation

Second chemin :

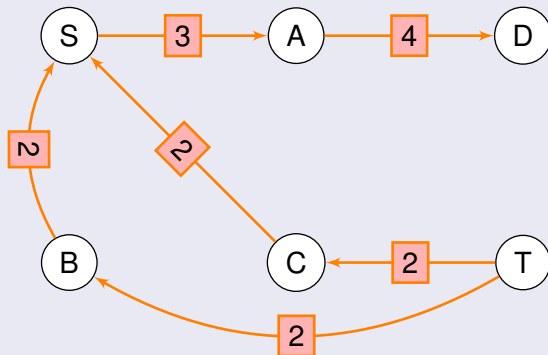
$s \rightarrow c \rightarrow t$

$\delta = 2$

$F = 4$

Algo Capacity Scaling

$G(f, 2)$

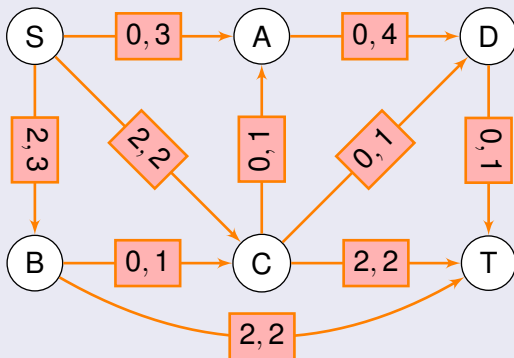


Mise à jour du
graphe d'augmenter et
Augmentation

Pas d'autre
chemin dans le
graphe.
 $\Delta = \Delta/2 = 1$

Algo Capacity Scaling

Flot actuel dans G

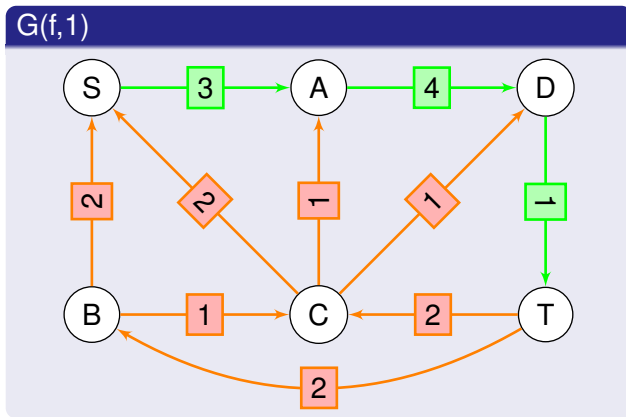


Rappel :

$$\Delta = 1$$

$$F = 4$$

Algo Capacity Scaling

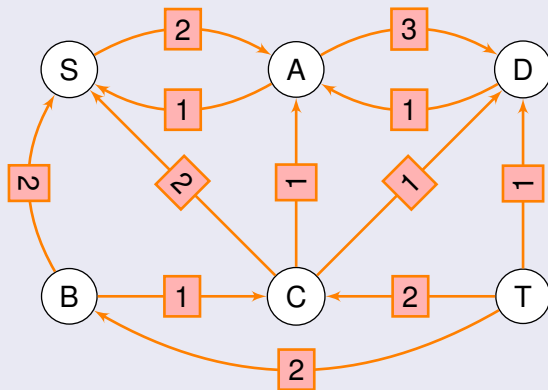


Mise à jour du
graphe résiduel et
Augmentation

Premier chemin :
 $s \rightarrow a \rightarrow d \rightarrow t$
 $\delta = 1$
 $F = 5$

Algo Capacity Scaling

$G(f, 1)$

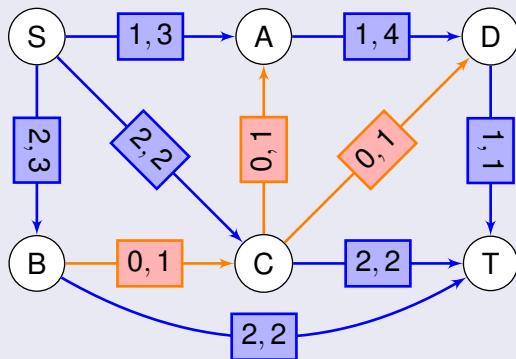


Mise à jour du
graphe résiduel et
Augmentation

Pas d'autre
chemin dans le
graphe.
L'algorithme est
terminé

Algo Capacity Scaling

Flot maximum)



Valeur du flot

F=5;

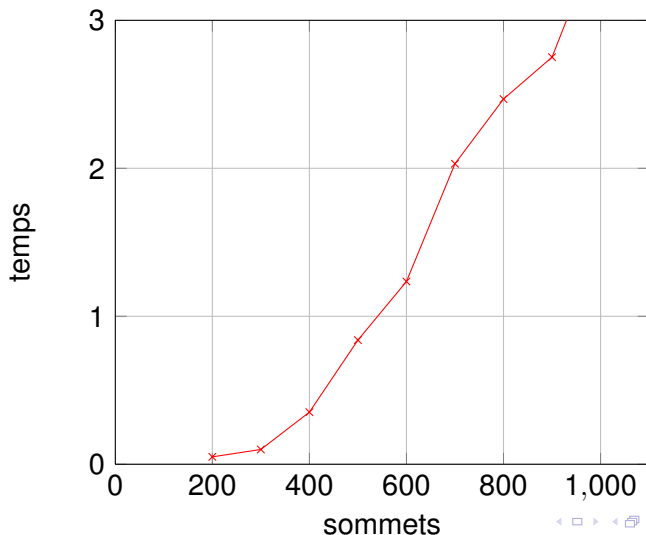
Plan

- 1 Les algorithmes étudiés
 - Ford-Fulkerson
 - Edmonds-Karp
 - Dinic
 - Capacity Scaling
- 2 **Experimentation et Performance**
 - Performance de l'ordinateur utilisitine
 - Temps d'execution en fonction du nombre de sommets comptant le temps de crion des graphes
 - Evolutions porter
- 3 Demonstration du fonctionnement sous TIKZ
 - Demonstration du fonctionnement sous TIKZ

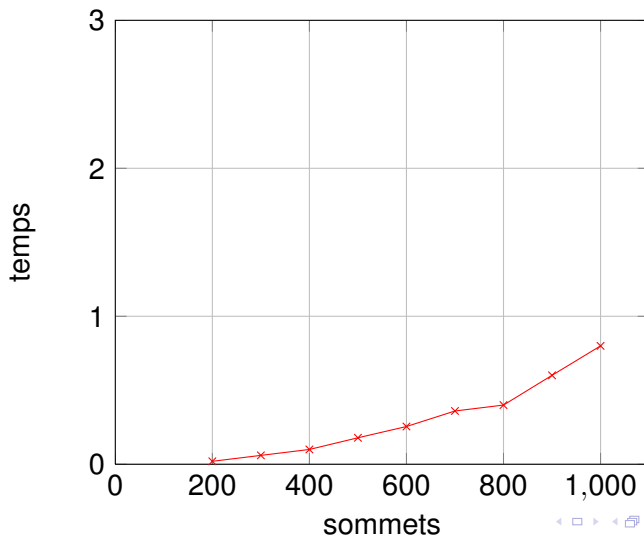
Version Ubuntu : 10.04

Intel(R) Pentium(R) Dual CPU T3200 @ 2.00GHZ

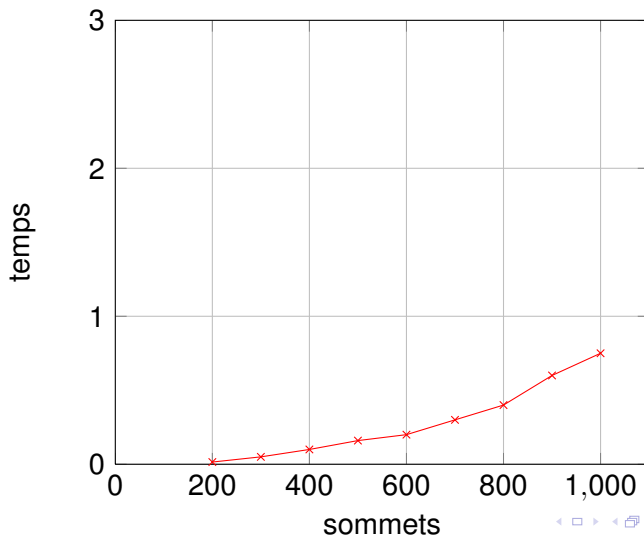
Ford-Fulkerson



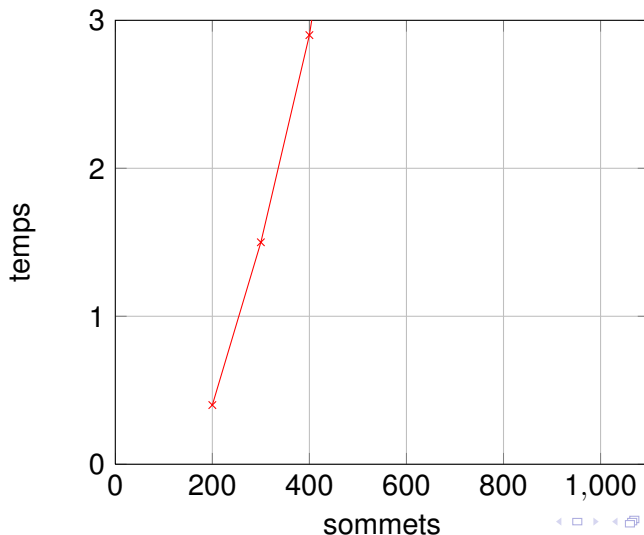
Capacity-Scaling



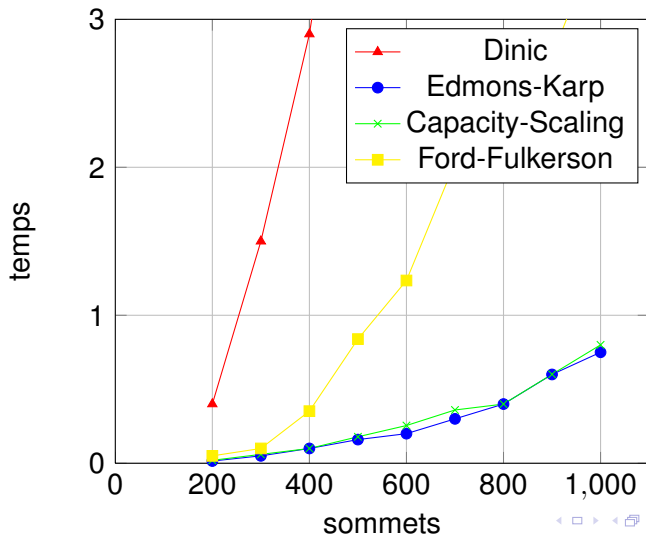
Edmons-Karp



Dinic



Comparaison des algorithmes



Evolutions

Reprntation du graphe

La reprntation sous forme de matrice nous a parue interessante dans un premier temps, de par sa simplicité d'utilisation mais nous avons ris qu'une liste de voisins aurait certainement plus représentative pour les parcours de chemin.

Graphe de couche

Le graphe de couche est peut-être mal représenté car nous l'avons représenté comme un graphe normal, dans une matrice. Ceci gêne un peu l'utilisation de ce graphe

Evolutions

Algorithme de Dinic

L'algorithme de Dinic, comme les autres, fonctionne mais il manque certainement d'optimisation

Plan

- 1 Les algorithmes étudiés
 - Ford-Fulkerson
 - Edmonds-Karp
 - Dinic
 - Capacity Scaling
- 2 Experimentation et Performance
 - Performance de l'ordinateur utilisitine
 - Temps d'execution en fonction du nombre de sommets comptant le temps de crion des graphes
 - Evolutions porter
- 3 Demonstration du fonctionnement sous TIKZ
 - Demonstration du fonctionnement sous TIKZ

Démo