

# Implémentation de l'algorithme de Dinic et d'Edmonds-Karp

TP Algorithmique, Complexité & Calculabilité (FMIN105)

William Dyce   Thibaut Marmin  
Clément Sipieter

Université Montpellier 2

15 Décembre 2011

# TP Algorithmique, Complexité & Calculabilité

Présentation du sujet

Conclusion

Implémentation

Démonstration

Tests & résultats

# TP Algorithmique, Complexité & Calculabilité

Présentation du sujet  
Algorithmes

Implémentation

Tests & résultats

Conclusion

Démonstration

# Algorithmes

Ford-Fulkerson :  $O(mnC^*)$

## Idée générale

- Trouver une chaîne améliorante.
- Augmenter le flot le long de cette chaîne.

## Défauts

- "Pseudo-exponentielle"
- "Diamond maudit"

# Algorithmes

## Edmonds-Karp : $O(n^5)$

Testing

# Algorithmes

Dinic

Testing also

# TP Algorithmique, Complexité & Calculabilité

Présentation du sujet

Conclusion

**Implémentation**

Choix Techniques

Structures

Mise en oeuvre des  
algorithmes

Démonstration

Tests & résultats

# Choix du langage de programmation

## C++

- rapidité d'exécution
- langage à objets
- connaissance du langage
- langage très répandu



# Utilisation d'un gestionnaire de version

## git - the stupid content tracker

- sauvegarde
- partage
- mise en commun

# Représentation du problème de flot maximum

## Réseau de transport

- Graphe orienté pondéré
- une source
- un puits

## Graphe d'écarts

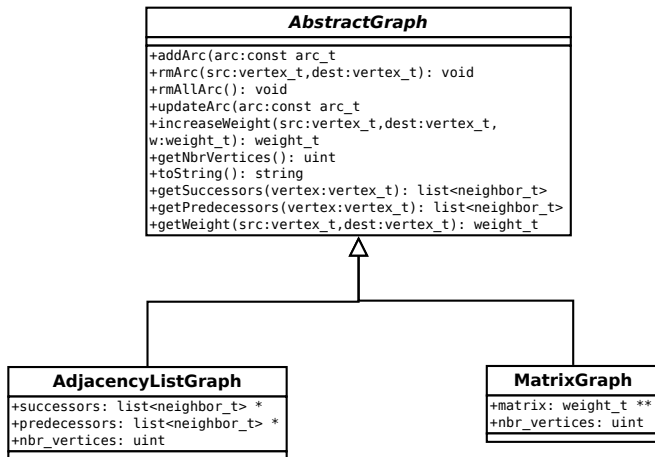
- Graphe orienté pondéré

# Représentation du problème de flot maximum

## Graphe de couches

- Graphe orienté pondéré
- Représentation des couches par un tableau de listes de sommets.

# Diagramme de classes



# Edmonds-Karp

## Recherche du plus court chemin en nombre d'arcs

### Parcours en largeur

- Graphes en listes d'adjacences :  $O(n+m)$
- Graphes en matrice d'adjacences :  $O(n^2)$

## Mise à jour du graphe d'écart

### Parcours du chemin

Décrementation du poids de chaque arc  $u,v$  du chemin

Incrémentation du poids de chaque arc  $v,u$

- Graphes en listes d'adjacences :  $O(n+m)$
- Graphes en matrice d'adjacences :  $O(n)$

# Dinic

## Génération du graphe de couches

Parcours en largeur + stockage d'une liste de parents par sommets

## Calcul du flot bloquant

Parcours en largeur + stockage d'une liste de parents par sommets

- Graphes en listes d'adjacences :  $O(nm)$
- Graphes en matrice d'adjacences :  $O(n^2m)$

## Mise à jour du graphe d'écart

Pour chaque arcs du flot bloquant

Décrementation du poids de chaque arc  $u,v$  du chemin

Incrémentation du poids de chaque arc  $v,u$

- Graphes en listes d'adjacences :  $O(m^2)$
- Graphes en matrice d'adjacences :  $O(n^2)$

# Génération de réseaux de transport aléatoires

## Deux stratégies

- Tirage aléatoire de deux sommets
- Génération de tous les arcs possibles et tirage d'un arc

# TP Algorithmique, Complexité & Calculabilité

Présentation du sujet

Conclusion

Implémentation

Démonstration

Tests & résultats

Méthode de tests

Résultats



# Méthode de tests

## Série de tests

### Complexité

- Edmonds-Karp :  $O(nm^2)$
- Dinic :  $O(n^2m)$

### Tests effectués

- Nombre de sommets : 100, 200, 300, ... 1000
- Densité du graphe : 20%, 50% et 80%

# Méthode de tests

## Profiling

### GNU gprof

Profiler, analyse du code en fonction du temps passé par chaque fonction à l'exécution.

- Compilation avec l'argument `-pg`
- Exécution du programme, génération du fichier `gmon.out`
- Exportation des statistiques en fichier texte

# Tests & résultats

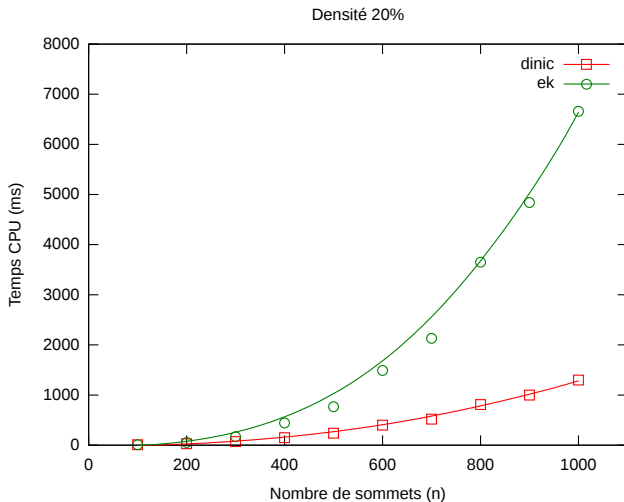
## Profiling

### GNU gprof

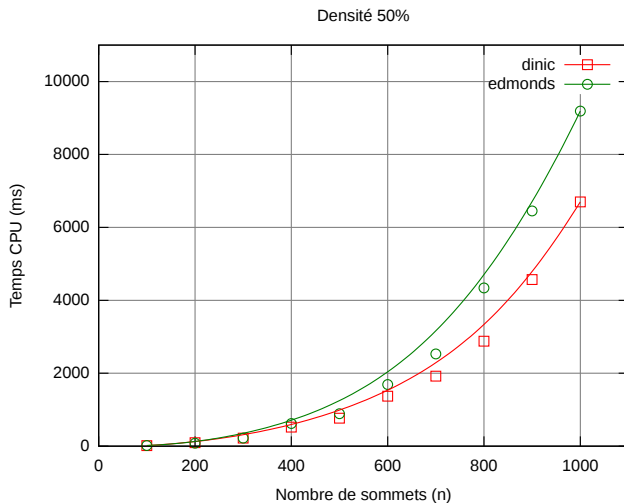
Statistique fournies pour chaque fonction :

- % temps cpu total
- temps cpu
- temps cpu par appel (de manière cumulative ou non)

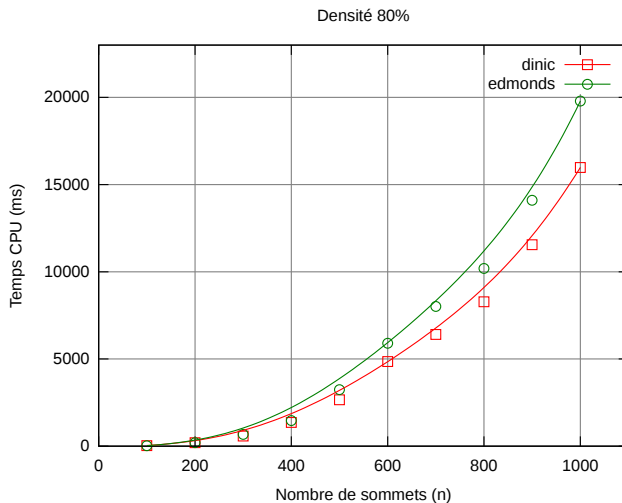
# Résultats



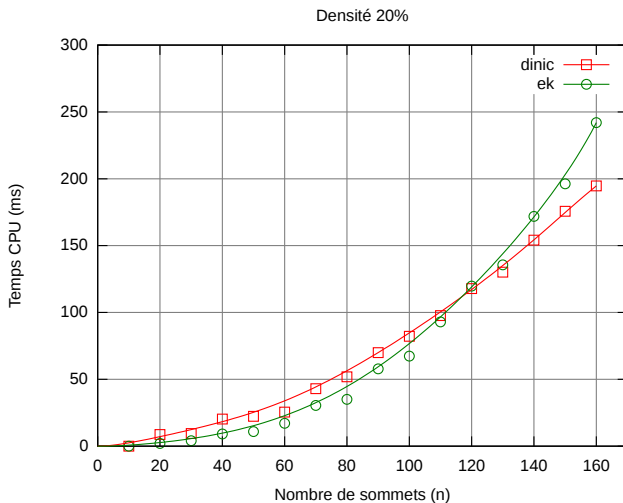
# Résultats



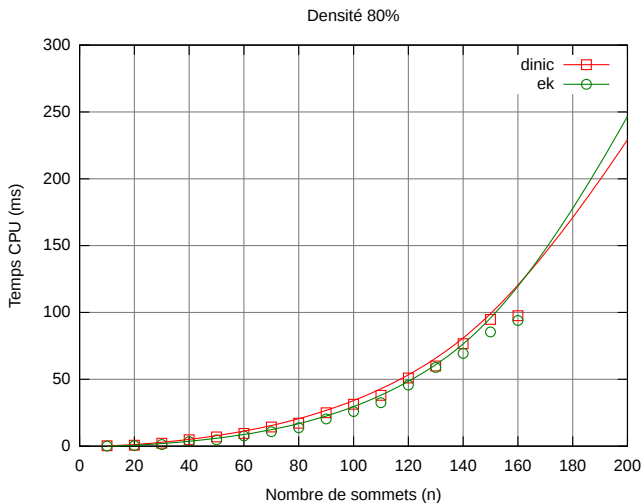
# Résultats



# Résultats



# Résultats



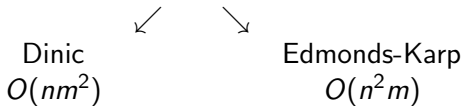


# Résultats

## Conclusion

- Dinic globalement plus rapide
- Surtout sur des graphes peu denses
- Edmonds-Karp efficace sur des petits graphes

⇒ Cohérence avec les complexités théoriques



# TP Algorithmique, Complexité & Calculabilité

Présentation du sujet

Conclusion

Conclusion

Implémentation

Démonstration

Tests & résultats

# Conclusion

# TP Algorithmique, Complexité & Calculabilité

Présentation du sujet

Conclusion

Implémentation

Démonstration

Démonstration

Tests & résultats

Présentation du sujet  
○○○

Implémentation  
○○  
○○○  
○○○

Tests & résultats  
○○○  
○○○○○○

Conclusion  
○

Démonstration  
●

# Démonstration