# Rapport sur l'étude de la complexité des algorithmes de flot (Ford-Fulkerson, Push-Relabel, Min-Cost Flow)

## Introduction

L'objectif principal de l'étude était d'évaluer les performances des trois algorithmes suivants :  
- Ford-Fulkerson (Edmond-Karp) pour le calcul du flot maximal,  
- Push-Relabel pour le calcul du flot maximal,  
- Min-Cost Flow pour un flot à coût minimal.  
  
Cette analyse a été menée sur des graphes de tailles variées (échelle logarithmique : 10, 20, 40, 100, 400, 1000, 4000, 10000) afin de comparer les performances pratiques de ces algorithmes. Nous avons utilisé deux métriques :  
1. Temps moyen pour 100 tests par taille de graphe.  
2. Temps maximal observé parmi ces 100 tests.

## 1. Génération des graphes de flot

### 1.1 Description des graphes

Pour chaque taille de graphe n, nous avons généré :  
- Une matrice de capacités C, avec des valeurs aléatoires entre 1 et 100 pour environ E = n^2 / 2 arêtes (graphes denses).  
- Une matrice de coûts D pour les graphes de flot à coût minimal, associée uniquement aux arêtes ayant une capacité non nulle.

### 1.2 Processus de génération

Un programme en langage C a été écrit pour :  
- Générer les graphes pour chaque taille n.  
- Tester les algorithmes (FF, PR, MIN) pour 100 itérations.  
- Enregistrer les temps d'exécution dans deux fichiers :  
 - results.txt : Temps pour chaque test.  
 - max\_values.txt : Temps maximal pour chaque taille n.

## 2. Analyse des résultats obtenus

### 2.1 Nuages de points (results.txt)

Les nuages de points montrent les temps d'exécution individuels pour chaque test en fonction de n. Cela permet d'observer :  
- Les variations dues à la nature aléatoire des graphes.  
- La stabilisation autour d'une moyenne pour chaque taille.

### 2.2 Courbes des temps maximaux (max\_values.txt)

Les courbes des temps maximaux montrent la progression des performances en fonction de n :  
- Ford-Fulkerson (FF) : Croissance sub-linéaire pour les petites tailles (n = 10, 20, 40), mais augmentation quadratique pour les grandes tailles.  
- Push-Relabel (PR) : Croissance plus rapide, proche d'une complexité quadratique pour n > 1000.  
- Min-Cost Flow (MIN) : Croissance exponentielle en fonction de n, influencée par le flot maximal F.

### 2.3 Observations clés

1. Ford-Fulkerson :  
 - Plus rapide pour les petits graphes, mais moins adapté aux graphes denses (n > 1000).  
2. Push-Relabel :  
 - Bien adapté aux graphes denses (E ≈ V^2).  
 - Temps d'exécution stable malgré la taille croissante.  
3. Min-Cost Flow :  
 - L'algorithme est beaucoup plus coûteux, surtout pour les grandes tailles, à cause du traitement combiné des coûts et des flots.

## 3. Comparaison avec les complexités théoriques

### 3.1 Ford-Fulkerson (FF)

- Complexité théorique : O(VE^2).  
- Observation : La croissance suit cette tendance pour les grandes tailles (n > 400).

### 3.2 Push-Relabel (PR)

- Complexité théorique : O(V^2E).  
- Observation : La courbe montre une croissance rapide, confirmant la nature quadratique avec V et linéaire avec E.

### 3.3 Min-Cost Flow (MIN)

- Complexité théorique : O(F ⋅ V ⋅ E).  
- Observation : La courbe montre une dépendance exponentielle avec F et une croissance significative avec n.

### 3.4 Comparaison des algorithmes FF et PR

Une analyse supplémentaire a été menée pour comparer les performances de Ford-Fulkerson et Push-Relabel en traçant le ratio θ\_FF(n)/ θ\_PR(n) :  
- Petits graphes (n < 100 ) : Les performances sont similaires, avec un ratio proche de 1.  
- Grands graphes ( n > 1000) : Push-Relabel surpasse Ford-Fulkerson, avec un ratio bien supérieur à 1.  
- Conclusion : Push-Relabel est plus adapté aux grands graphes denses.

## 4. Conclusions

### 4.1 Performances des algorithmes

1. Ford-Fulkerson (FF) :  
 - Adapté aux graphes petits ou clairsemés.  
 - Performances dégradées pour les grands graphes.  
2. Push-Relabel :  
 - Préférable pour les graphes denses.  
 - Meilleure gestion des grandes tailles.  
3. Min-Cost Flow (MIN) :  
 - Utile uniquement si les coûts sont un critère important.  
 - Complexité plus élevée rend cet algorithme moins pratique pour les grands graphes.

### 4.2 Recommandations

- Pour les problèmes de flot maximal sur des petits graphes : utiliser Ford-Fulkerson.  
- Pour les grands graphes denses : préférer Push-Relabel.  
- Pour les problèmes de flot à coût minimal : éviter les grandes tailles sauf si indispensable.

### 4.3 Perspectives

- Intégrer des optimisations (élagage des arêtes inutiles).  
- Tester les algorithmes sur des graphes réels (infrastructures, logistique).