20815 IRHBOULA Youssef

Optimisation de flot sur un réseau de transport

En partant du constat que la gestion d'un réseau de transport doit obéir à des contraintes de ressources, je me suis fixé pour objectif de mettre en place un algorithme permettant de maximiser le flot sur un graphe de transport pour un budget donné, en me basant sur les solutions existantes au probleme de flot maximal et au problème du sac à dos.
La gestion du flot sur un réseau de transport s'inscrit clairement dans le thème de l'année et peut avoir des applications à la fois en termes de réparation et de fluidification sur un réseau préexistant, et en aide à la décision pour le choix d'améliorations de réseaux.

Positionnement thématique (phase 2)

INFORMATIQUE (Informatique Théorique).

Mots-clés (phase 2)

Mots-Clés (en français)	**Mots-Clés** (en anglais)
Graphes	*Graphs*
Réseau de flot	*Flow network*
Optimisation	*Optimization*
Problème de flot maximal	*Max-flow problem*
Problème du sac à dos	*Knapsack problem*

Bibliographie commentée

Un réseau de flot doit répondre à des contraintes de budget en plus des contraintes de capacité usuellement prises en compte. En effet, en cas de travaux sur le réseau, que ce soit pour réparation ou pour extension, les différents projets ont un coût limité par un budget.

Les graphes de flots permettent de représenter des déplacements dans un réseau de transport entre une source et un puits en obéissant aux lois de Kirchhoff (tout circule de la source vers le puits et rien n'est créé ni ne s'arrête aux noeuds) et à des contraintes de capacités. Quand on étudie les propriétés de ces graphes, la question de l'existence du flot maximal est naturellement soulevée. Des propriétés de base sur les flots sont présentées dans l'Introduction à l'Algorithmique **[1]** nous permettent de nous familiariser avec cette structure, avec le concept de graphe résiduel et enfin de démontrer le théorème Max-Flow/Min-Cut qui répond au problème.

Reste alors à étudier les moyens de l'obtenir. En 1962, Ford et Fulkerson **[2]** présentent leur méthode pour résoudre le problème. Il s'agit d'une méthode assez naturelle basée sur le concept de chemin améliorant: après avoir initialisé à un flot nul, on augmente le flot en passant par des chemins entre la source et le puits dans le graphe résiduel. Edmonds et Karp montrent **[5]** que la recherche en largeur permet d'implémenter cette méthode en $O(VE^2)$ et d'en démontrer la terminaison. L'algorithme de Dinitz permet lui d'atteindre une complexité en $O(V^2E)$ **[3]** en passant par des graphes à couches et des flots bloquants. Une autre méthode améliorant la

complexité est la méthode pousser-réétiqueter développée par Tarjan et Goldberg **[4]**. Elle se base sur les excédents de flots et permet d'atteindre une complexité de $O(VE^2)$ puis en O(V^3) dans sa version réétiqueter-vers-l'avant.

On pense alors à appliquer la solution de programmation dynamique proposée par Garfinkel et Nemhauser **[6]** au problème du sac à dos (complexités temporelle et spatiale en O(nW)) avec des flots en posant pour valeur d'une modification la différence entre le flot maximal avec et sans. Mais cette solution s'appuie sur l'additivité des valeurs, que l'on n'a plus ici. On propose alors une solution qui calcule les sommes de manière dynamique en cherchant à optimiser le calcul des nouveaux flots avec une initialisation au flot précédent.

Problématique retenue

Comment peut-on mettre en place et implémenter un algorithme répondant à une situation relevant à la fois du problème du sac à dos et du problème de flot maximal?

Objectifs du TIPE

Je me propose:

-d'étudier les propriétés des flots et des graphes de flots
-d'analyser les algorithmes répondant au problème de flot maximal et de les implémenter en OCaml
-de mettre en place un algorithme original répondant à ma problématique en combinant problème du flot maximal et problème du sac à dos en en minimisant la complexité notamment au niveau du calcul des sommes de valeurs
-d'appliquer la solution à des situations un peu plus générales: sources et puits multiples, surgraphes..

Abstract

A transportation network can be studied using the structure of flow network which is a type of graphs. There exist algorithms that calculate the maximum flow. In real situations, a transportation network can be damaged and one can ask how, when the budget is limited, we can maximize the flow on the network. In this work, after solving the knapsack and the maxflow problems, we will propose an algorithm which is given a budget and a set of modifications of known cost, and returns the combination of maximum flow without exceeding the budget.

Références bibliographiques (phase 2)

[1] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein : Introduction to Algorithms third edition : *The MIT Press, 2009, Chapitre: V-Graph algorithms, 26-Maximum Flow*

[2] L.R. Ford Jr, D.R. Fulkerson, : Flows in Networks : *The Rand corporation, 1962*

[3] S. Even : Graph Algorithms : *Cambridge University Press, 1979, Chapitres: 5-Flows in Networks, 6-Application of Network Flow techniques*

[4] A. V. Goldberg, E. Tardos, R.E. Tarjan : Network Flow Algorithms : *Cornell University, 1989, Chapitre: 2-The Maximum flow problem*

[5] J. Edmonds, R.M. Karp : Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems : *Journal of the Association for Computing Machinery, 1972*

[6] R. Garfinkel, G.L. Nemhauser : Integer Programming : *Springer, 1972, Chapitre: 13.5.4-Reformulations based on dynamic programming*

DOT

[1] *Etude de la structure de graphe de flot et compréhension du théorème Maxflow-Mincut*

[2] *Etude des algorithmes de flot maximal: Ford-Fulkerson, pousser-réétiqueter et analyse de complexité*

[3] *Etude du problème du sac à dos et de sa solution en O(nW)*

[4] *Adaptation des deux problèmes et proposition d'une solution à la non additivité des valeurs en* $O(A(S^2+nW))$

[5] *Implémentation des algorithmes en OCaml*