TD série N°1: problème du trader

Christoph Samuel

I-Position du problème :

Lors d'une séance de trading sur un marché à terme de produits pétroliers, **un trader** lève une option d'achat sur **3 lots (A1, A2 et A3)** dont les valeurs planchers sont respectivement 50M€, 40M€ et 80M€. Il représente deux fonds d'investissement (E1 et E2) avec un **engagement maximum** de 100M€ chacun. Le trader crée deux **fonds opérationnels**, F1 (110M€ max) et F2 (60M€ max), alimentés par E1, E2 et F1. Il prévoit également un tirage direct de 30M€ max sur E1. Pour acquérir les titres, le **trader anticipe les fluctuations** et prévoit de tirer un montant maximum sur les fonds F1, F2 et E1 pour chaque opération (A1, A2 et A3).

L'objectif de ce premier TD est de construire un graphe, en y incorporant un **problème de flot maximum.** J'ai donc abordé un problème de flot optimal. Un **réseau de flot** est un graphe orienté valué positivement, possédant une **source S et un puits P** et, où chaque arc possède une capacité et peut **recevoir un flot.** La somme des flots sur une arête ne peut pas excéder sa capacité. Un graphe orienté est souvent appelé **réseau en recherche opérationnelle**. Les sommets sont appelés des nœuds et les arêtes, des arcs. De plus, on sait que tout flot rentrant dans un nœud doit en sortir, sauf pour la source et le puits, **c'est la loi de conservation.**

L'un des problèmes de l'ingénieur consiste à s'assurer que le montage respecte bien la **cohérence du réseau de flots**. Dans le cadre de ce TP, j'ai décidé de m'aider de **JGraphT**, une bibliothèque Java implémentant **l'algorithme d'Edmonds-Karp**, de Djikstra et qui propose une interface de **création et traitement de graphes**.

Elle permet entre autres de **calculer le flot maximum** et de trouver la **st-coupe.** Je me suis également servi de **yEd Graph Editor**, une application me permettant de créé et générer différents graphes, pour une représentation graphique plus adéquate du réseau de flot.

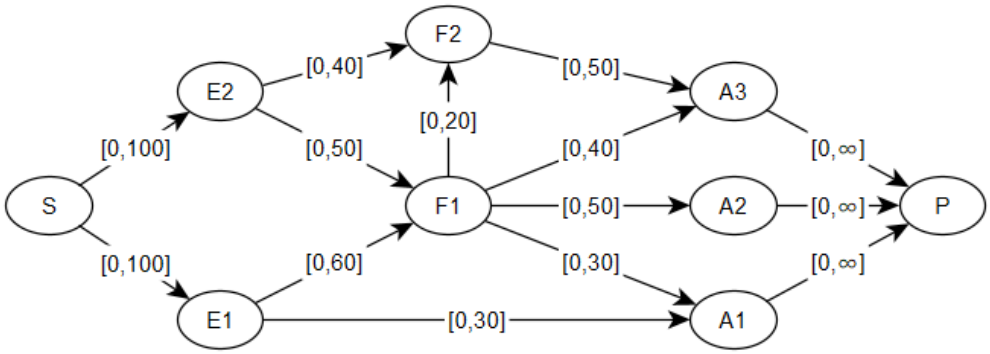
II-Réalisation :

Étude du premier problème

Le **premier problème** est de savoir si le montage proposé par **le trader** garantit, a priori, la faisabilité de l'opération **d'acquisition des trois lots** A1, A2 et A3. Lors d'une acquisition ou d'une cession, il est primordial pour le trader de connaître la **faisabilité de l'opération**. Il doit donc passer par un **réseau de flots optimal** pour résoudre son problème.

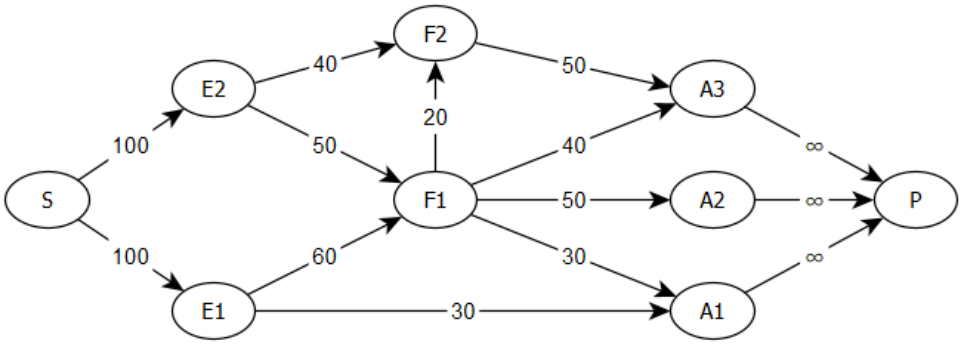
Afin de résoudre ce problème, j'ai donc ramener le problème a un problème de **transition dans le réseau de flots standard.** Soit G = (S,A), un **graphe orienté valué positivement**, avec S les nœuds du graphe, S (super source), P (super puit) et A les différents arcs du graphe définit ci-dessous, j'obtiens alors le graphe orienté suivant:

S	S, E1, E2, F1, F2, A1, A2, A3, P
A	(S → E1), (S → E2), (E1 → F1), (E1 → A1), (E2 → F1), (E2 → F2), (F1 → F2),(F1 → A1), (F1 → A2), (F1 → A3), (F2 → A3), (A1 → T), (A2 → T), (A3 → P)

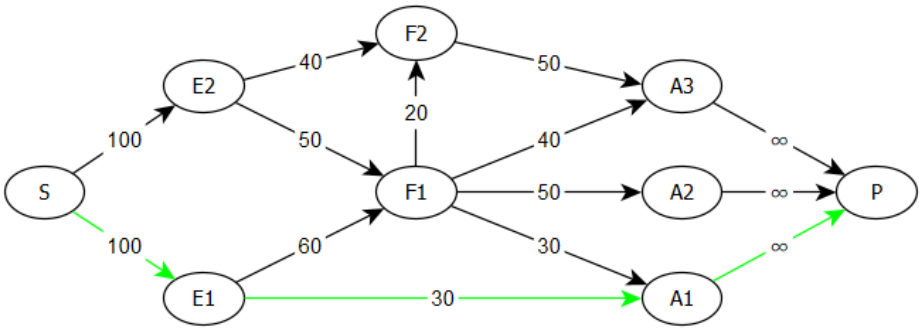


graphe G

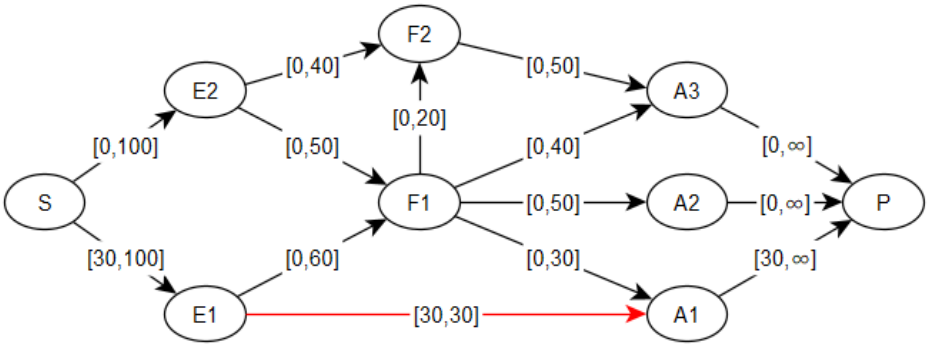
Puisqu'on ne cherche pas à maximiser le flot, je peux **augmenter le flot** dans les arcs sortant de A1, A2 et A3 jusqu'à obtenir respectivement 50M, 40M et 80M les valeurs d'acquisition de ces trois lots.



graphe d'écart de G

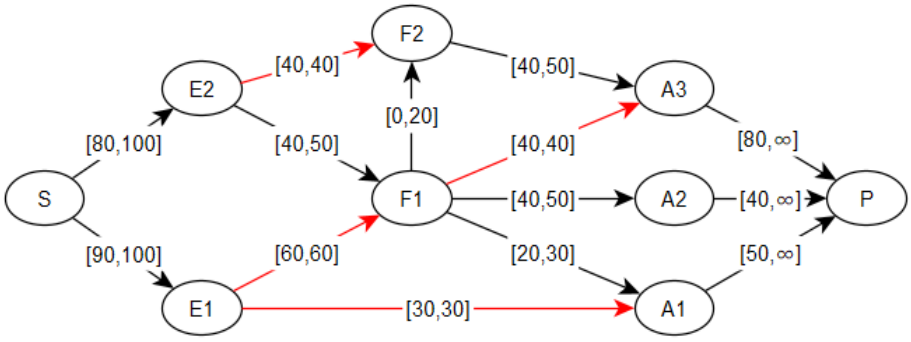


chemin augmentant n°1 (S, E1, A1, P)



première mise à jour sur le graphe G

En répétant ces étapes de **nombreuses fois**, je finis par obtenir le **résultat suivant**, via la dernière mise a jour du graphe d'écart de notre graphe G.



dernière mise a jour sur le graphe G

À partir du graphe ci-dessus, je peux **déduire** que le montage proposé par le trader **garantit, a priori, la faisabilité de l'opération d'acquisition** des trois lots A1, A2 et A3. Le premier problème est alors réglé.

Étude du second problème

Le **deuxième problème** se présente lorsque l'opérateur cédant le lot A1 (50M€), présentant une tendance haussière du marché, cherche à en tirer profit pour négocier une augmentation de la valeur de son titre. On nous demande alors quelle **augmentation maximale** peut concéder le trader sans remettre en cause son **montage initial ?**

Pour répondre a cette question, j'ai décidé d'appliquer l'algorithme **d'Edmonds-Karp** afin de **maximiser le flot passant par A1**, la procédure de l'algorithme est défini ci-dessous.

```
algorithm EdmondsKarp
    input:
        C[1..n, 1..n] (matrice des capacités)
        E[1..n, 1..?] (listes des voisins)
        s             (source)
        t             (puits)
    output:
        f             (valeur du flot maximum)
        F             (matrice donnant un flot valide de valeur maximum)
    f := 0 (le flot initial est nul)
    F := array(1..n, 1..n) (la capacité résiduelle de u à v est C[u,v] - F[u,v])
    forever
        m, P := BreadthFirstSearch(C, E, s, t, F)
        if m = 0
            break
        f := f + m
        (Backtrack et noter le flot)
        v := t
        while v ≠ s
            u := P[v]
            F[u,v] := F[u,v] + m
            F[v,u] := F[v,u] - m
            v := u
    return (f, F)
```

pseudo code de l'algorithme d'Edmonds-Karp

Cet algorithme est **identique à l'algorithme de Ford-Fulkerson**, à l'exception de l'ordre de recherche utilisé pour **déterminer un chemin augmentant**. Le chemin trouvé doit être un chemin le **plus court** (en nombre d'arcs) qui possède une capacité positive, **dans le graphe résiduel**. Un tel chemin peut être trouvé par un **parcours en largeurs** dans le graphe résiduel, en supposant que les arcs ont tous une longueur unité.

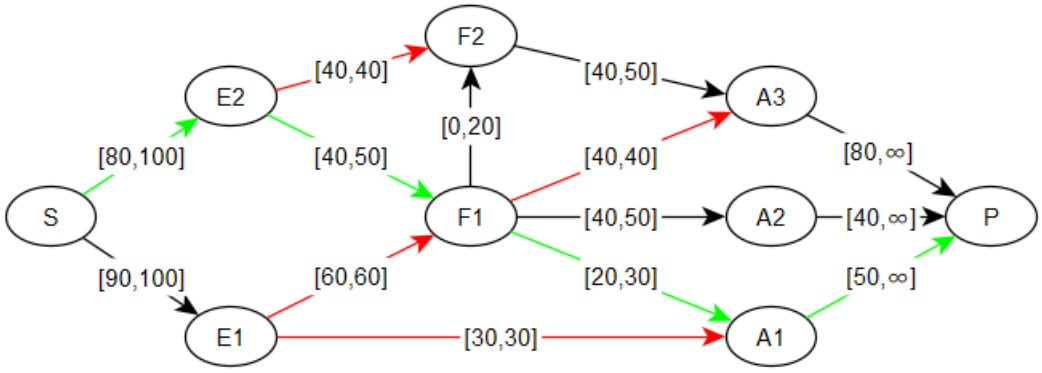
Principe de l'algorithme :

L'idée de l'algorithme de d'Edmonds–Karp consiste, partant d'un flot initial compatible, à chercher par marquage une chaîne augmentante et augmenter le flot le long de cette chaîne. Le processus s'arrête lorsqu'il y a plus de chaîne augmentante.

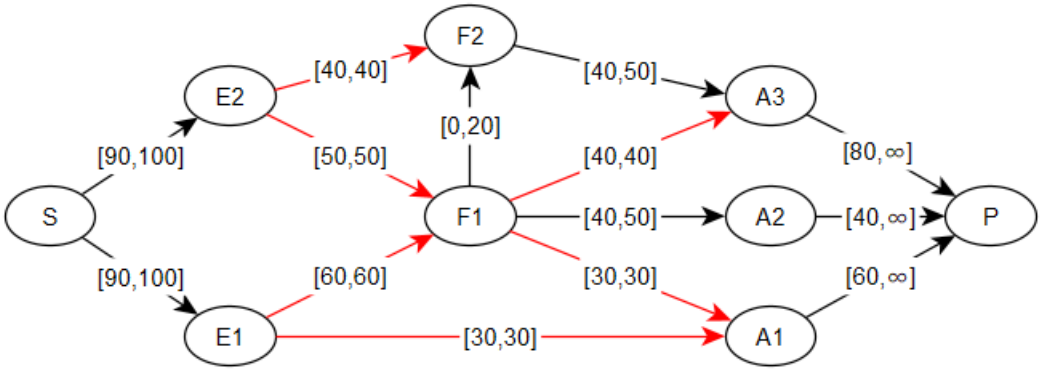
La complexité est définie telle que :

marquage O(n+m), suppression des marques O(n), examen de tous les successeurs et tous les prédécesseurs O(m), augmentation du flot: O(m).

À chaque itération, on **augmente le flot** d'une unité au moins, j'applique alors **Edmonds Karp** en partant du **graphe précédent** afin d'obtenir les **résultats** suivants.



chemin augmentant passant par A1



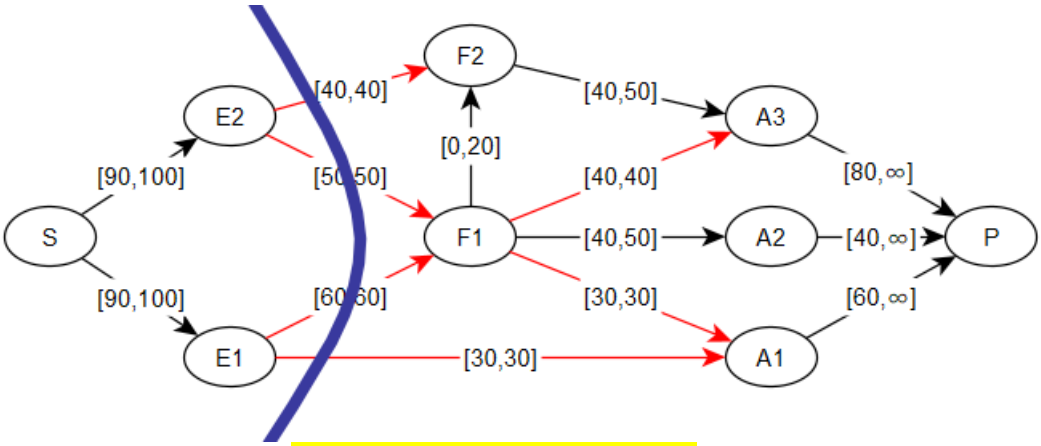
dernière mise a jour sur le graphe G

À partir des graphes ci-dessus, j'arrive à en déduire que **l'augmentation maximale** peut concéder le trader **sans remettre en cause son montage initial,** est 10M€, parce que la valeur de A1 s'augmente de **50M€ à 60M€.**

Étude du troisième problème

Le troisième problème survient lorsque l'opérateur **cédant le lot A2** (40M€), bien informé de la **négociation précédente**, décide, à son tour, une renégociation de son titre. On nous demande alors, si le **trader** peut-il concéder une nouvelle hausse **sans remettre en cause** son montage initial ?

De même que pour l'étude du **second problème,** on peut ramener le problème à un problème de recherche de **maximiser le flot passant par A2**. En appliquant l'**algorithme d'Edmonds-Karp**, et en commençant par la dernière mise à jour.



st-coupe minimale du graphe G

À partir du **graphe ci-dessus**, on peut déduire que le trader **ne peut pas concéder** une nouvelle hausse **sans remettre en cause son montage initial**, en effet, on observe l'apparition d'une **st-coupe minimale** ou les **arcs sortants de E1 et E2 sont saturés**, cette st-coupe minimale empêche au trader l'investissement de **tous les fonds disponibles** (200M€).

Pour que le trader puisse **concéder une nouvelle renégociation**, il faudrait changer les arcs de la **st-coupe minimale** par des arcs de c**apacités maximales infini** et étudier la hausse des **trois titre** A1, A2, et A3. Si l'on modifie le montage, on peut obtenir une **hausse de 10M€ en tout.**

```
-------------------< com.mycompany:Probleme_Trader >--------------------
Building Probleme_Trader 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ Probleme_Trader ---
Chemin le plus court allant de S vers P:[(S : E1), (E1 : A1), (A1 : P)]

Flot initial 0.0

Flot maximum actuel = 180.0

Flux associés à chaque arc :
Arc (S : E1) : 90.0
Arc (F1 : A3) : 30.0
Arc (E2 : F1) : 50.0
Arc (S : E2) : 90.0
Arc (A1 : P) : 60.0
Arc (E1 : A1) : 30.0
Arc (F2 : A3) : 40.0
Arc (F1 : A1) : 30.0
Arc (A2 : P) : 50.0
Arc (E1 : F1) : 60.0
Arc (A3 : P) : 70.0
Arc (E2 : F2) : 40.0
Arc (F1 : A2) : 50.0
Arc (F1 : F2) : 0.0

Arcs de la st-coupe minimale:
De : E1 à : A1 avec capacité : 30.0
De : E1 à : F1 avec capacité : 60.0
De : E2 à : F1 avec capacité : 50.0
De : E2 à : F2 avec capacité : 40.0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  2.308 s
Finished at: 2023-04-13T18:26:42+02:00
------------------------------------------------------------------------
```

terminal Apache après compilation de mon programme

On observe via le terminal Apache les différents éléments concernant l'étude de cas trader retranscrite sous la forme de réseau de flots, en effet, on observe le **chemin le plus court entre S et P**, le flot initial, le **flot maximum**, le flux associés a **chaque arc** à la fin de son passage dans d'Edmonds-Karp ainsi que les **quatre arcs composant la st-coupe minimale** avec leur flux associés.

Comme nous le montre si bien, les **résultats obtenus** après compilation de mon programme (ProblemeTrader_230323.java), le **flot maximum actuel** après négociation délivré par la super source est de **180M$** pour un flot maximum de **200M$**

L'**augmentation du flot** passe nécessairement par l'augmentation de la **capacité d'un des arcs de la st-coupe**. En prenant en compte cela, on conclu que le trader pourra concéder une augmentation maximum de **20M$ en cas de hausse** d'un des trois titres en augmentant la capacité d'un des arcs de la st-coupe et par conséquent augmenter la capacité des deux **fonds opérationnels** F1 et F2.

III-Bilan/Conclusion :

Sur le plan de l'application des modèles et algorithmes de recherche opérationnelle sur les graphes, ce TD démontre qu'il est aisé de **ramener de nombreux problèmes réels à un problème de recherche opérationnelle,** ici celui de calcul de flot optimal au sein d'un graphe et la puissance de ses solutions algorithmiques pour résoudre les-dits problèmes. J'ai appris à appliquer mes connaissances sur **le calcul de flot optimal pour réaliser un graphe** via différents algorithmes.

Sur le plan de résolution des problèmes réels, cet exemple me démontre **l'importance des réseau de flots** via l'acquisition et la cession de titres d'un trader et cela me donne un aperçu d'un futur travail qui m'attend.