Optimisation dans les graphes : sujet finance

Raphael Leonardi, Pedro Bartolomei & Vladimir Herrera Nativi

Université Paris-Saclay

17 octobre 2025

Plan

1. (i) Présentation du problème
2. (ii) Modélisation proposée
3. (iii) Algorithmes développés
4. (iv) Réponses aux questions (synthèse)

Contexte et objectif

- **Contexte** : échanges entre devises avec des taux *(sans frais)*.
- **Objectif** : maximiser un montant initial en enchaînant des échanges.
- **Contraintes** : Faut partir et revenir à la même divise et le nombre d'échanges est limité à p.

Notions clés

- *Chemin d'échanges* : séquence de devises $x_0 \to x_1 \to \cdots \to x_k$.
- *Valeur du chemin* $v(\mu)$: produit (ou somme en log) des valuations.

Graphe associé G=(X,U)

- **Orienté** : oui (les taux aller $\neq$ retour).
- **Sommets** X : ensemble des devises (EURO, USD, JP, CHF).
- **Arcs** U : (x, y) l'échange de la devise x à la devise y.
- **Valuation** $v(x, y)$: taux net d'échange $x \rightarrow y$.

Valeur d'un chemin

Définition (produit des taux)

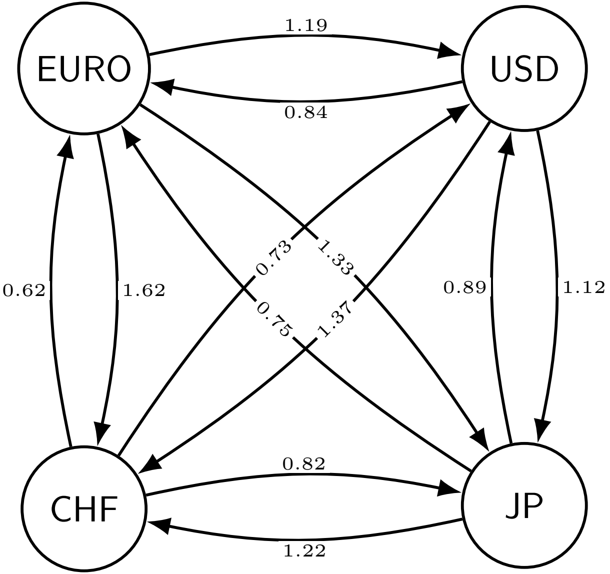
Soit $\mu : x_0 \to x_1 \to \cdots \to x_k$.

$$v(\mu) \;=\; \prod_{i=0}^{k-1} v(x_i, x_{i+1})$$

(ou, en log-additif : $\sum \log v(x_i, x_{i+1})$)

- **Recherche cible** : chemins/cycles maximisant $v(\mu)$ (ou détecter cycles de gain).

Modélisation du graphe



Vue d'ensemble des 3 algorithmes

- **A1 – Naïf (indépendant)** : exploration exhaustive limitée à p étapes.
- **A2 – Optimisé** : *pruning, tri par heuristique, mémorisation partielle*.
- **A3 – Classique (log-transform)** :
  - Si *cycles profitables* : Bellman-Ford sur coûts $c = -\log v$ (détection cycles négatifs).
  - Si *au plus p échanges* : DP / Bellman-Ford tronqué (p relaxations).

Algorithme 1 : Algorithme glouton : à chaque étape, choisir l'arc sortant de plus grande valuation, puis fermer le cycle en revenant à $s =$ EUR.

Entrées : $G = (X, U)$ graphe orienté, valuation $v : U \to \mathbb{R}_{>0}$, source $s \in X$ (EUR), limite $p \in \mathbb{N}$ (nombre de symboles / $p - 1$ transactions)

Sortie : Chemin $\pi = (x_0, \ldots, x_p)$ de longueur $p + 1$ avec $x_0 = x_p = s$

Fonction `CheminGlouton(`$G = (X, U), v, s, p$`)` **:**

> **si** $p \leq 2$ **alors**
>
> > **retourner** *"aucun chemin valide"*
>
> **fin**
>
> $\pi \leftarrow [\, s \,]$
>
> $c \leftarrow p$ `// compteur de symboles restants`
>
> **tant que** $c > 1$ **faire**
>
> > $(w, \_) \leftarrow$ MeilleurSuivant $(\pi[\text{fin}])$
> >
> > ajouter w à la fin de π
> >
> > $c \leftarrow c - 1$
>
> **fin**
>
> ajouter s à la fin de π `// forcé : retour à l'euro`
>
> **retourner** π

fin

Algorithme 2 : Enumération exhaustive (p transactions) + sélection par valeur de chemin

Entrées : $G = (X, U)$ orienté, $v : U \to \mathbb{R}_{>0}$, source $s \in X$, $p \in \mathbb{N}$ (transactions)
Sortie : $\pi^\star$ chemin de longueur p tel que $x_0 = x_p = s$ et $V^\star = \texttt{ValeurChemin}(\pi^\star)$

```
Fonction EnumPaths(s, p) :
    // Génère tous les chemins s = x0 → x1 → ··· → xp = s
    Fonction Backtrack(π) :
        si |π| = p alors
            si π[fin] = s alors
                yield π

            return
        pour chaque y ∈ X\{π[fin]} faire          // eviter les auto-boucles
            Backtrack(π · y)
    Backtrack([s])

Main :
Π ← EnumPaths(s, p)
(π*, V*) ← arg max_{π∈Π} (ValeurChemin(π))
retourner (π*, V*)
```

Pseudo-code : Algo - Tous les chemins (log)

Algorithme 3 : DP en log pour maximiser le produit des taux avec contrainte $x_t \neq x_{t-1}$ et retour à s.

Entrées : $G = (X, U)$ graphe orienté, valuation $v : U \to \mathbb{R}$, source $s \in X$ (EUR), longueur $p \in \mathbb{N}$
Sortie : Chemin optimal $\pi = (x_{\mathbf{0}}, \ldots, x_p)$ avec $x_{\mathbf{0}} = x_p = s$ **et** gain $\prod v(x_{t-\mathbf{1}}, x_t)$

Initialisation de W, dp[t][v] et prev[t][v] :

$W \leftarrow log \circ A_G$ // A_G la matrice d'adjacence de G

$\forall t \leq p, \forall v \in X, dp[t][v] \leftarrow -\infty$; $dp[0][s] \leftarrow 0$ // log(1)

$\forall t \leq p, \forall v \in X, dp[t][v] \leftarrow -1$

Transition DP (max-somme) :

pour $t \leftarrow 1$ **à** L **faire**
  pour chaque $v \in X$ **faire**
    pour chaque $u \in X \setminus \{v\}$ **faire**
      si $dp[t-1][u] + W(u, v) > dp[t][v]$ **alors**
        $dp[t][v] \leftarrow dp[t-1][u] + W(u, v)$
        $prev[t][v] \leftarrow u$
      fin
    fin
  fin
fin

$\pi \leftarrow$ reconctruction du chemin a partir de $prev[t][v]$

$gain \leftarrow \exp\left(dp[L][s]\right)$

retourner $(\pi,\ gain)$

Q1–Q2 : Définition de G=(X,U) et valeur d'un chemin

- **Q1** : G=(X,U) orienté, sommets = devises, arcs = échanges possibles, valuations $v(x, y)$.
- **Q2** : $v(\mu) = \prod v(x_i, x_{i+1})$ (ou somme de $-\log v$).

Q3–Q4 : Que rechercher dans G ? Echanges illimité, le profit ?

- **Q3** : *Meilleur chemin/cycle* maximisant le gain depuis l'EURO.
- **Q4** : *Profit illimité*.

Q6 : Meilleure séquence pour $p = 3$

- Résultat : **EURO** $\rightarrow$ **USD** $\rightarrow$ **CHF** $\rightarrow$ **EURO** (3 échanges).
- Gain obtenu : $v(\mu) = 1.010786$.

Conclusion & suites

- **Prochaines pistes** : frais variables, volatilité, risque, contraintes de liquidité.