

Les Mariages Stables

Jordann Perrotta
Aix-Marseille Université

19 mai 2017

Table des matières

1	Introduction	2
2	Définition	3
2.1	Couplage	3
2.2	Couplage parfait	3
2.3	Couplage stable	3
2.4	Ordre	3
2.5	Ordre partiel	3
2.6	Ordre total	4
3	Algorithmes et résolution	4
3.1	Basique	5
3.1.1	Pseudo-Code	5
3.2	Weakly	5
3.2.1	Pseudo-Code	5
3.2.2	Explication	5
3.3	Strong	7
3.3.1	Pseudo-Code	7
4	Algorithmes et implémentations	7
4.1	Les Structures de données	7
4.2	Implémentation basique	9
5	Installation	9
6	Results	9
7	Conclusions	9

1 Introduction

Les algorithmes pour trouver des solutions au problème du mariage stable ont des applications dans diverses situations du monde réel, peut-être la plus connue étant celle des étudiants diplômés en médecine à leurs premiers offres d'emploi. Une application importante et à grande échelle d'un mariage stable consiste à affecter les utilisateurs aux serveurs d'un grand service Internet distribué. Des milliards d'utilisateurs accèdent à des pages Web, des vidéos et d'autres services sur Internet, ce qui oblige chaque utilisateur à correspondre à l'un des centaines de milliers de serveurs du monde entier qui offrent ce service. Un utilisateur préfère des serveurs suffisamment proches pour fournir un temps de réponse plus rapide pour le service demandé, ce qui entraîne une commande préférentielle des serveurs pour chaque utilisateur. Chaque serveur préfère servir les utilisateurs à un coût inférieur, ce qui entraîne une commande préférentielle des utilisateurs pour chaque serveur. Les réseaux de distribution de contenu qui distribuent la plupart du contenu et des services du monde résolvent ce problème de mariage stable complexe entre utilisateurs et serveurs toutes les secondes afin de permettre à des milliers d'utilisateurs d'être associés à leurs serveurs respectifs qui peuvent fournir les pages Web, les vidéos demandées, et tant d'autres services.

Dans la Section 2, nous aborderons les définitions relatives au mariage stable ainsi que plusieurs exemples d'illustrations. Dans la section 3, nous expliquerons les différents types de recherche de mariage stable qu'il existe, leurs pseudo-codes, quelques propositions et théorèmes, ainsi que certaines qualités de mesure d'évaluation du mariage stable que l'on peut obtenir. Dans la section 4, nous détaillerons les algorithmes de résolution du mariage stable dans le langage Java, les structures de données utilisés ainsi que leurs utilisations. Pour finir, dans la section 5, nous verrons comment utiliser ce projet, son installation et les issues possibles de résultat.

2 Définition

Donnons nous pour ces définitions un ensemble d'hommes H et un ensemble de femmes F de même cardinalité.

2.1 Couplage

Un couplage c est une relation fonctionnelle et injective entre H et F .

En d'autres termes, c'est une bijection entre un sous-ensemble de H et un sous-ensemble de F . Chaque homme peut être marié à au plus une femme, et vice-versa.

2.2 Couplage parfait

Un couplage est parfait si c'est une bijection entre H et F . En d'autre terme, aucun homme et aucun femme ne reste célibataire.

Pour la suite de ces définitions, on suppose que chaque homme classe toutes les femmes par ordre de préférence, et inversement.

2.3 Couplage stable

Un couplage est dit stable si pour tout couple (h, f) et (h', f') , il n'existe pas une interprétation tel que h préfère f' à son partenaire actuel et f' préfère h à son partenaire actuel. De la même manière, la stabilité d'un couplage est possible dans le cas où h préfère f' à son partenaire actuel, par contre w' ne préfère pas h à son partenaire actuel.

2.4 Ordre

Un ordre, dans le cas des mariages stables, est la place qu'occupe un homme h dans les préférences d'une femme f et vice-versa. Chaque homme (respectivement femme) doit ordonner de manière décroissante ses préférences. Il existe deux type d'ordre, l'ordre partiel et l'ordre total.

2.5 Ordre partiel

Un ordre partiel, est un ordre qui admet une indifférence de personne à une position donnée dans les préférences. Pour l'exemple, un homme h_1 , en position numéro 1, préférera autant une femme f_1 qu'une autre femme f_2 .

2.6 Ordre total

Un ordre total est un ordre qui n'admet aucune indifférence de personne à une position donnée dans les préférences. Pour chaque $h \in H$, on a donc un ordre total \leq_h sur F , et pour chaque $f \in F$, un ordre total \leq_f sur H . On note $f_1 \leq_h f_2$ si h préfère f_1 à f_2 . On note $f_1 \leq_f f_2$ si f_1 préfère f_2 .

3 Algorithmes et résolution

Il existe 2 types de résolution pour le problème du mariage stable. Le premier type est l'algorithme basique de Gale/Shapley, permettant de résoudre le problème de mariage stable lorsque les préférences des hommes et des femmes sont ordonnées sans ex-æquo possible. Le deuxième type se divise en sous-type, l'algorithme de Gale-Shapley étendu, la version dite "SUPER" (que nous n'aborderons pas dans ce rapport), puis la version dite "STRONG". Ce deuxième type existe dans le sens où, il est possible d'avoir une indifférence de préférence entre tel et tel personne. Dans la vraie vie, il est tout à fait possible d'avoir des préférences ex-æquo, choisir entre une boulangerie ou une autre, un coiffeur ou un autre, un supermarché ou un autre, etc.. La version basique de résolution est un cas particulier qui peut s'appliquer uniquement si les choix réalisés sont strictement ordonnés. Ces types de résolution sont en réalité un sorte de mesure, si nous obtenons un couplage stable avec le "weakly", il est possible qu'un certain homme h peut préférer une autre femme w' que sa partenaire actuel, alors que w' ne préférera moins h à son partenaire actuel. De la même manière, nous pouvons affirmer que si un couplage stable a été trouvé par l'algorithme "STRONG", il n'existera aucun couple (h, f) et (h', f') tel que h préfère f' à son partenaire actuel et w' préfère h à son partenaire actuel. Dans la suite de cette section, nous allons voir les pseudo-codes de chaque type de résolution.

3.1 Basique

On fait évoluer un couplage partiel (i.e., pas parfait), initialement vide

3.1.1 Pseudo-Code

Algorithme 1 Basic Stable

ENTRÉES: '

$H \leftarrow$ ensemble des hommes

$F \leftarrow$ ensemble des femmes

$liste_{pref}(f) \leftarrow$ liste de préférence de f

$liste_{pref}(h) \leftarrow$ liste de préférence de h

Initialiser tout les $h \in H$ et $f \in F$ à libre

tantque il existe un homme libre m qui peut encore proposer à une femme f **faire**

$f \leftarrow$ la première femme dans la liste de h a qui h n'a pas encore propose

si f est *libre* **alors**

(h, f) devient engagé

sinon {il existe déjà un couple (h', f) }

si f préfère h à h' **alors**

h' devient libre

(h, f) s'engage

finsi

finsi

fin tantque

SORTIES: Un couplage stable

3.2 Weakly

3.2.1 Pseudo-Code

3.2.2 Explication

Cet algorithme revient à utiliser l'algorithme basique, auquel cas nous avons arbitrairement ordonné les indifférences.

Algorithme 2 Weakly Stable

ENTRÉES: ' $H \leftarrow$ ensemble des hommes $F \leftarrow$ ensemble des femmes $liste_{pref}(f) \leftarrow$ liste de préférence de f $liste_{pref}(h) \leftarrow$ liste de préférence de h Initialiser tout les $h \in H$ et $f \in F$ à libre**tantque** il existe un homme libre m qui peut encore proposer à une femme f **faire** $f \leftarrow$ la première femme dans la liste de h à qui h n'a pas encore proposé
 h propose, et devient engagé à f **si** il existe un homme h' qui est déjà engagé avec f **alors**assigner h' à libre**finsi****pour tout** successeur h'' de m dans $liste_{pref}(f)$ **faire**supprimer la paire (h'', f) **fin pour****fin tantque****SORTIES:** Un couplage stable

3.3 Strong

3.3.1 Pseudo-Code

4 Algorithmes et implémentations

Dans cette section, les algorithmes sont implémentés en Java ainsi que tout les structures de donnés. Pour commencer, afin de faciliter le choix des différents algorithmes pour la résolution d'une instance de mariage stable, le pattern design "Stratégie" semble être un bon choix. La stratégie de résolution va dépendre de plusieurs facteurs, soit du type de donné (si la gestion des indifférence doit être prise en considération), soit du choix de l'utilisateur. Si l'utilisateur décide d'utiliser une stratégie qui n'est pas optimal pour le type de donné choisie, le programme va lui indiquer. Chaque stratégie a une implémentation différente, il y en a 3, une pour l'algorithme basique de Gale-Shapley, la seconde est celle du "WEAKLY" et la dernière, l'algorithme "STRONG". Nous allons donc diviser cette section en 3 parties.

4.1 Les Structures de données

Le projet se compose de plusieurs fichiers :

- *StableStrategie.java*, une interface.
- *LireFichier.java*, une classe qui permet de lire un fichier texte contenant les préférences et de les affecter dans les *Maps* correspondante.
- *BasicStable.java*, la classe qui implémente l'algorithme basique.
- *WeaklyStable.java*, la classe qui implémente l'algorithme *weakly*.
- *StrongStable.java*, la classe qui implémente l'algorithme *strong*.
- *Instance.java*, une classe qui permet de créer une instance en fonction de la stratégie, des deux *Maps A* et *B* (des préférences) ainsi que des listes des personnes participants au couplage.
- *Main.java*, la classe principal qui exécute le programme.
- *Convertible.java*, une petite interface qui permet de convertir une *Map* de différente manière, selon la stratégie utilisé.

Algorithme 3 Strong Stable

ENTRÉES: ‘ $H \leftarrow$ ensemble des hommes $F \leftarrow$ ensemble des femmes $liste_{pref}(f) \leftarrow$ liste de préférence de f $liste_{pref}(h) \leftarrow$ liste de préférence de h Initialiser tout les $h \in H$ et $f \in F$ à libre**répéter****tantque** il existe un homme libre h qui peut encore proposer à une femme f **faire****pour tout** femme à la tête de $liste_{pref}(h)$ **faire** h propose, et devient engagé à f **pour** chaque successeur strict h' de h dans $liste_{pref}(f)$ **faire****si** h' est engagé à w **alors**

briser l'engagement

finsisupprimer la paire (h', f) **fin pour****fin pour****fin tantque****si** les couples obtenus ne forment pas un couplage parfait **alors**trouver la zone critique Z des hommes**pour tout** femme f qui est engagé à un homme dans Z **faire**briser tout les engagements impliquant f **pour tout** homme h à la fin de $liste_{pref}(f)$ **faire**supprimer la paire (h, f) **fin pour****fin pour****finsi****jusqu'à** ce que chaque $liste_{pref}(h)$ soit vide ou que tout le monde soit engagé**si** tout le monde est engagé **alors**

les engagements forment un couplage fortement stable

sinon

il n'existe pas de couplage fortement stable

finsi

4.2 Implémentation basique

5 Installation

6 Results

In this section we describe the resul

7 Conclusions

We worked hard, and achieved very little.

Références