Modèles de couplage aléatoire sur un graphe d'interaction

Nicolas Lengert, Madalina Deaconu, Antoine Lejay, Pascal Moyal

Nicolas Lengert

Institut Elie Cartan de Lorraine

Modèles de couplage aléatoire sur un graphe d'interaction

Sous la direction de Madalina Deaconu, Antoine Lejay et Pascal Moyal

Septembre 2020

Contents

1 Résumé et présentation du problème

Nous allons nous intéresser à l'étude de modèles généraux donnant les outils pour la résolution de problèmes d'affectation optimale. Un problème d'affectation, dans ce mémoire, sera considérée comme la recherche d'une solution optimale lorsque l'on veut mettre en relation différentes entités pouvant interagir entre elles ou non selon des règles prédéfinies.

Par exemple dans un contexte informatique, lorsque un client s'associe à un réseau de serveurs, certains serveurs peuvent ne pas lui être accessibles. Dans notre problématique on va considérer que de nombreux clients arrivent pour se connecter à ce réseau. On voudrait donc trouver un moyen d'associer le maximum de clients, idéalement tous les clients, à ces serveurs, en fonction de leurs admissibilités.

On va donc considérer le problème général suivant : des entités arrivent non simultanément, à un certain intervalle de temps et souhaitent s'associer entre elles. Un graphe d'appariement décrit les couplages admissibles entre les différentes entités. Il existe une politique dite de choix permettant de choisir lorsque une entité peut être associée à plusieurs autres. Une file d'attente contient les entités non couplés. Nous verrons par la suite que l'on peut établir un modèle markovien de cette file d'attente et obtenir des résultats intéressants sur le comportement en temps long du processus de couplage, en particulier nous exhiberons des conditions pour que la file d'attente se vide "régulièrement" en un certain sens que nous préciserons. Nous présenterons certains de ces résultats dans la section 3.

Nous allons adopter une vision duale de ce problème, en transformant cette problématique de file d'attente en une problématique de recherche d'un couplage dit parfait (ne laissant aucune entité non couplée) sur un graphe aléatoire contenant un grand nombre de sommets. Nous montrerons clairement le lien entre les deux modèles.

Il existe déjà des résultats forts de la théorie des graphes, que nous présenterons dans la seconde section, assurant sous certaines conditions l'existence d'un tel couplage. Il existe également un algorithme, l'algorithme *blossom*, permettant de faire cette recherche d'un couplage parfait dans un graphe. Cependant cet algorithme est coûteux en temps de calcul et l'implémentation est complexe. Nous présenterons donc dans la section 4 un algorithme en ligne, peu coûteux en temps de calcul et qui, asymptotiquement, est aussi performant que l'algorithme *blossom*.

2 Introduction à la théorie des couplages

2.1 Notations et définitions de la théorie des graphes

Avant de présenter les premiers résultats nous allons donner le vocabulaire que nous allons utiliser tout au long de ce mémoire.

Nous allons tout d'abord définir ce qu'est un graphe et préciser le vocabulaire associé.

- Un *graphe non orienté* est un couple $G = (V, E)$ avec :
  - ☐ V un ensemble de *sommets* (ou *noeuds*), ces sommets peuvent être dotés d'un label ;
  - ☐ E est un ensemble de sous-ensembles de V de cardinal 2. Les éléments $\{x, y\}$, $x, y \in V$ de E sont appelés *arêtes*. On note également l'arête $\{x, y\} =: xy$.

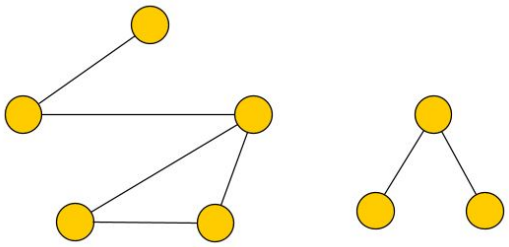


Figure 1: Un graphe quelconque.

- On appelle *voisin* ou *image* de $x \in V$ dans G tout sommet $y \in V$ tel que $\{x, y\} \in E$.
- Si $\{x, y\} \in E$, on dit que y est *adjacent* à x (et réciproquement) et on appelle *liste d'adjacence* de x l'ensemble $N(x) = \{y \in V | \{x, y\} \in E\}$.
- Deux arêtes a_1 et a_2 sont dites *adjacentes* si $\exists x$ tel que $x \in a_1$ et $x \in a_2$.
- Un sommet x est *incident* à une arête $a \in E$ si $x \in a$. On dit que a *couvre* x.
- Un graphe $G = (V, E)$ est dit *biparti* si $V = V_1 \cup V_2$ avec $V_1 \neq \emptyset$, $V_2 \neq \emptyset$, $V_1 \cap V_2 = \emptyset$ et $\forall \{x, y\} \in E$, $(x \in V_1$ et $y \in V_2)$ ou $(x \in V_2$ et $y \in V_1)$. On le note $G =: (V_1, V_2, E)$.
- Un graphe est dit *complet* si tous les sommets de ce graphe sont adjacents deux à deux. On note le graphe complet à n sommets K_n.
- Un graphe biparti $G = (A, B, E)$ est dit complet s'il est biparti et contient le nombre maximum d'arêtes. On note $K_{n,m}$ le graphe biparti complet avec $|A| = n$ et $|B| = m$.

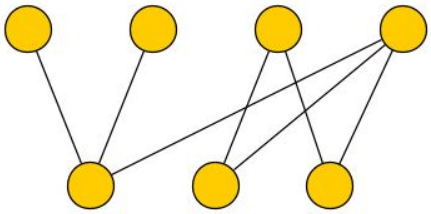


Figure 2: Un graphe biparti.

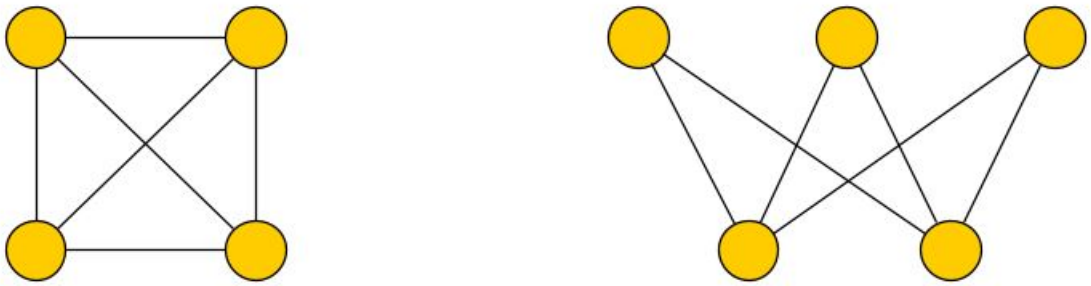


Figure 3: Le graphe K_4 et le graphe $K_{3,2}$.

Nous allons désormais présenter la notion de connexité d'un graphe et la notion de composante connexe, qui nous sera nécessaire dans la preuve de certains théorèmes. Nous présenterons également la notion de sous-graphe, une notion centrale en théorie des graphes. Nous finirons par présenter quelques notations usuelles qui interviendront dans la section 2 de ce mémoire.

- On appelle *chaîne* de G l'ensemble de sommets $\{s_0, ..., s_n\}$ où :
  $\forall i \in \{1, ..., n\}$, $\{s_{i-1}, s_i\} \in E$.
- On dit que G est *connexe* si $\forall \{x, y\} \in V^2$, il existe une chaîne de x à y.
- Soit $G = (V, E)$, $H = (W, F)$ deux graphes. Alors :
  - ☐ G est un *sous-graphe* de H si $V \subset W$ et $E = F \cap V \times V$;
  - ☐ G est un *graphe partiel* de H si $V = W$ et $E \subset F$;
  - ☐ G est un *sous-graphe partiel* de H si G est un sous-graphe d'un graphe partiel de H.
- Soit $G = (V, E)$ un graphe, soit U un sous-ensemble de V. On notera dans la suite $G - U$ le sous-graphe $(V \setminus U, E \cap (V \setminus U) \times (V \setminus U))$, c'est-à-dire le sous-graphe de G obtenu en enlevant les sommets de U.
- Une *composante connexe* d'un graphe G est un sous-graphe connexe de ce graphe.

- On notera dans la suite $o(G)$ le nombre de composantes connexes d'ordre impair du graphe G, c'est-à-dire le nombre de composantes connexes ayant un nombre impair de sommets.

Nous allons présenter désormais la notion de couplage sur un graphe qui est centrale dans ce mémoire. De nombreux problèmes d'affectation et d'optimisation peuvent être traités comme des problèmes de recherche d'un couplage maximum ou parfait sur un graphe. Ces notions sont définies ci-dessous.

- On appelle *couplage* sur un graphe $G = (V, E)$ un ensemble M d'arêtes de G deux à deux non adjacentes, c'est-à-dire $M \subset E$ et $\forall a_1, a_2 \in M$, $a_1 \cap a_2 = \emptyset$.
- Un couplage M sur un graphe $G = (V, E)$ est dit :
  - ☐ *maximal* si $\nexists a \in E$ tel que $a \notin M$ et $\forall a^* \in M$, $a \cap a^* = \emptyset$;
  - ☐ *maximum* si $\nexists N$ couplage sur G tel que $|M| < |N|$;
  - ☐ *parfait* si $\forall x \in V$, x est incident à une arête $a \in M$.

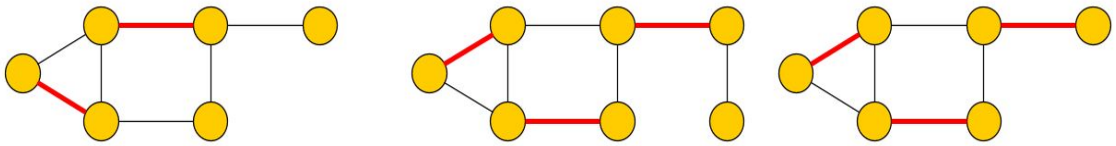


Figure 4: Des couplages (en rouge) maximal, maximum et parfait sur 3 graphes.

Nous allons définir désormais la notion de chaîne alternante et augmentante. Ces deux notions seront clés dans la preuve du théorème de Hall que nous présenterons au début de la section suivante.

- Soit M un couplage sur $G = (V, E)$. Une chaîne $\{s_0, ..., s_n\}$ est dite
  - ☐ *M-alternante* si $\forall i \in \{1, ..., n\}$, $(\{s_{i-1}, s_i\} \in M) \oplus (\{s_{i-1}, s_i\} \in E \setminus M)$;

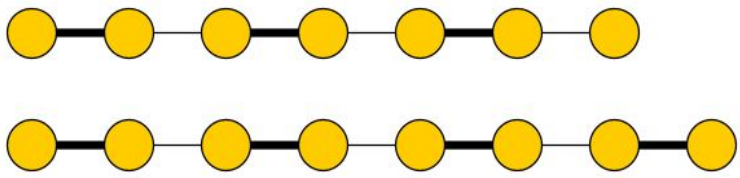


Figure 5: Deux chaînes M-alternante (le couplage M est représenté par les arêtes plus épaisses).

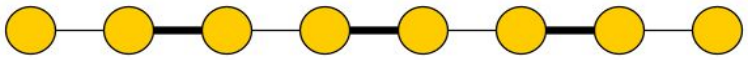


Figure 6: Une chaîne M-augmentante.

- M-*augmentante* si elle est M-alternante et si s_0 et s_n ne sont incidents à aucune arête de M, c'est-à-dire les extrémités de la chaîne ne sont pas couvertes par M.

Les chaînes M-augmentantes permettent d'augmenter le cardinal du couplage considéré en quelques étapes comme l'illustre l'exemple ci dessous.

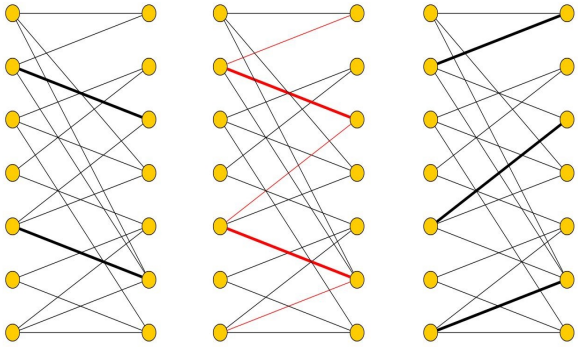


Figure 7: La chaîne M-augmentante en rouge permet de passer d'un couplage de 2 arêtes à un couplage de 3 arêtes.

- Un sommet x est M-*atteignable* d'un sommet y s'il existe une chaîne M-alternante de x à y.

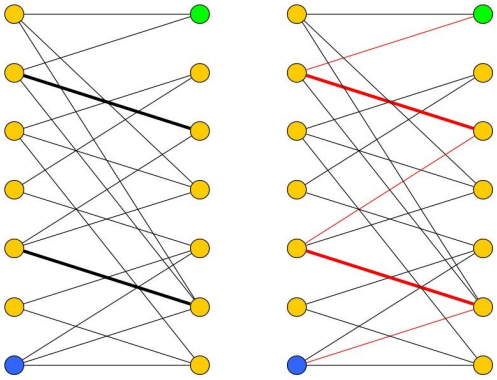


Figure 8: Le point vert est M-atteignable par le point bleu en passant par la chaîne M-augmentante en rouge

- Un sous-ensemble de sommets d'un graphe G est dit *indépendant* si cet ensemble ne contient pas de sommets adjacents deux à deux. On note I un tel ensemble et $\mathbb{I}$ l'ensemble des ensembles indépendants de G.

On aura également besoin d'une notion de distance entre deux sommets qu'on définit comme suit :

$$d_G(x, y) = \min\{|(s_0, s_1, ..., s_n)| \text{ tel que } s_0 = x, s_n = y \text{ et } s_i s_{i+1} \in E, \forall i \in \{0, .., n-1\}\}.$$

On peut montrer que c'est une distance au sens mathématique pour les graphes non orientés.

2.2 Le théorème de Hall

Le théorème de Hall (voir [1]) nous donne une condition nécessaire et suffisante à l'existence d'un couplage parfait dans un graphe biparti. Nous allons en donner une preuve constructive, c'est-à-dire que nous allons construire un couplage maximum à l'aide de l'algorithme d'Edmonds (ou *blossom algorithm*) puis montrer que si le graphe vérifie cette condition, alors le couplage construit est parfait.

Nous allons tout d'abord présenter le *lemme de Berge*[1] qui nous sera nécessaire dans l'algorithme d'Edmonds.

Théorème 2.1. *Un couplage M sur un graphe $G = (V, E)$ est maximum si et seulement si il n'existe pas de chaîne M-augmentante sur G.*

Proof. Nous allons tout d'abord démontrer par contraposée que si un couplage est maximum, alors il n'existe pas de chaîne augmentante sur ce couplage. Soit M un couplage sur G avec $|M| = k$ et supposons qu'il existe P une chaîne M-augmentante. Comme P est M-augmentante, elle contient j arêtes dans M, $j \leq k$ et $j + 1$ arêtes dans $P \setminus M$. On pose $M^* = (P \setminus M) \cup (M \setminus P)$. Ainsi $|M^*| = |M| + 1$. De plus, M^* est également un couplage car ce qui contredit le fait que M soit maximum.

Nous allons désormais démontrer directement l'autre sens de l'équivalence. Soit M un couplage sur G et supposons que G n'admet pas de chaîne M-augmentante. Supposons qu'il existe un couplage M^* de cardinal strictement plus grand que celui de M. On pose $H = (M \setminus M^*) \cup (M^* \setminus M)$. Alors les sommets de H sont incidents soit à une arête de M ou M^*, soit à deux arêtes, l'une dans M et l'autre dans M^*. Ainsi H est union disjointe de chaînes. Ces dernières peuvent être de longueur paire ou impaire. Comme M^* contient plus d'éléments que M, alors il existe une chaîne dans H contenant plus d'arêtes de M^* que d'arêtes de M et donc cette chaîne est M-augmentante ce qui contredit l'assertion initale. □

Nous allons maintenant présenter le théorème de Hall sur l'existence de couplage parfait dans les graphes bipartis.

Théorème 2.2. *Soit $G = (A \cup B, E)$ un graphe biparti. Le graphe G admet un couplage parfait si et seulement si*

$$\forall X \subset A,\ |X| \leq |N(X)|$$

avec $N(X)$ l'ensemble de sommets de B adjacents à X.
On appelle cette condition la condition de Hall.

Proof. Nous allons tout d'abord démontré que si un graphe admet un couplage parfait, alors il respecte la condition de Hall. Supposons qu'il existe un couplage parfait M sur G. Alors $\forall X \subset A$, $\forall x \in X$, x possède au moins un sommet adjacent dans B, le sommet adjacent à x dans M donc $|X| \leq |N(X)|$.

Nous allons désormais démontrer l'implication réciproque à travers une preuve constructiviste, c'est-à-dire que nous allons construire un couplage maximum sur un graphe puis nous démontrerons par contraposée que si ce couplage n'est pas parfait, alors le graphe ne respecte pas la condition de Hall.

Preuve constructiviste :

1. On construit un couplage maximum par l'algorithme d'Edmonds.
2. Supposons qu'il y ait un sommet x qui ne soit pas couplé.
3. On pose W l'ensemble des sommets de G M-atteignables par x.
4. Alors W ne respecte pas la condition de Hall.

Algorithme d'Edmonds (Blossom) (voir [2]) : on construit le couplage en améliorant à chaque itération le couplage initial vide en utilisant les chemins augmentants :

1. Entrée : M un couplage sur G.
2. Si M est maximal retourner M.
3. Sinon, soit P un chemin M-augmentant.
4. $M = (M \setminus P) \cup (P \setminus M)$.
5. Retourner à l'étape 2.

Il nous reste à montrer que l'ensemble W ne respecte pas la condition de Hall, car le cas échéant on aura montré qu'un tel ensemble implique que l'on ne puisse trouver un couplage parfait (1 sommet est laissé exposé) :
Tous les sommets de $N(W)$ sont couplés, sinon on pourrait augmenter le couplage ce qui contredit le fait qu'il soit maximum.
W contient M restreint à $N(W)$ car ces couplages sont M-atteignables par x. Ainsi $|W| = |N(W)| + 1$ car x est dans W et donc W ne respecte pas la condition de Hall. □

2.3 La formule de Tutte-Berge et le théorème de Tutte

Nous allons présenter la formule de Tutte-Berge qui donne une formule pour le cardinal du couplage maximum sur un graphe donné. Nous développerons en détail la preuve donnée dans [3].

Théorème 2.3. *Soit $G = (V, E)$ un graphe. Le cardinal du couplage maximum sur G, noté $m(G)$, vérifie :*

$$m(G) = \frac{1}{2} \min_{U \subset V} (|V| + |U| - o(G - U)).$$

Proof. L'idée est la suivante : pour tout $U \subset V$, si le sous-graphe $G - U$ admet une composante connexe d'ordre impair alors cette dernière laissera une arête hors de tout couplage possible car elle contient un nombre impair d'arêtes. Certains choix de U peuvent créer de nombreuses composantes connexes dans $G - U$ ce qui entraîne que le couplage ne pourra pas couvrir un certain nombre d'arêtes.

Nous allons tout d'abord montrer que $m(G)$ est majoré par le membre de gauche de la formule de Tutte-Berge. Directement, $\forall U \subset V$,

$$m(G) \leq |U| + m(G - U) \leq |U| + \frac{1}{2}(|V \setminus U| - o(G - U)) = \frac{1}{2}(|V| + |U| - o(G - U)).$$

On va désormais montrer que $m(G)$ est supérieur au membre de gauche de la formule de Tutte-Berge. On suppose G connexe, le cas échéant on peut appliquer cette formule sur chaque composante connexe de G. On va procéder par induction sur le cardinal de V.

Supposons tout d'abord que $|V| = 1$, alors le minimum est atteint pour $U = \emptyset$ et la formule est trivialement vérifiée.

Supposons désormais que $|V| \geq 2$ et que la formule est vraie pour les graphes ayant $|V| - 1$ sommets. On considère le premier cas suivant : il existe un sommet $v \in V$ tel que tous les couplages maximums de G couvrent v. Ceci implique directement que $m(G - \{v\}) = m(G) - 1$. Par induction on sait que la formule est vraie pour un graphe ayant $|V| - 1$ sommets donc il existe $U' \subset V \setminus \{v\}$ tel que

$$m(G - \{v\}) = \frac{1}{2}(|V \setminus \{v\}| + |U'| - o(G - \{v\} - U')).$$

On pose $U = U' \cup \{v\}$. Alors

$$\begin{aligned} m(G) &= m(G - \{v\}) + 1 \\ &= \frac{1}{2}(|V \setminus \{v\}| + |U'| - o(G - \{v\} - U')) + 1 \\ &= \frac{1}{2}(|V \setminus \{v\}| + |U \setminus \{v\}| - o(G - \{v\} - (U - \{v\}))) + 1 \\ &= \frac{1}{2}(|V| - 1 + |U| - 1 - o(G - U)) + 1 \\ &= \frac{1}{2}(|V| + |U| - o(G - U)). \end{aligned}$$

La formule est donc "héréditaire", elle est donc vraie pour tout $|V|$.

Supposons désormais que pour chaque sommet $v \in V$, il existe un couplage maximum M tel que v n'est pas couvert par M. Nous allons démontrer que dans ce cas, $m(G) = \frac{1}{2}(|V| - 1)$,

c'est-à-dire que le couplage maximum ne couvre pas exactement 1 sommet (sinon on aurait $m(G) = \frac{|V|}{2}$). En admettant que ce soit juste, en prenant $U = \emptyset$ on obtient la formule de Tutte-Berge.

On suppose par contraposée que toutes paires de sommets u et v sont laissées non couvertes par un couplage maximum de G, c'est-à-dire que pour tout couple $\{u, v\} \in V^2$, il existe N couplage maximum de G tel que les sommets u et v ne sont pas couverts par le couplage. On pose

$$d(M, u, v) = d_G(u, v).$$

avec u et v non couverts par M un couplage maximum sur G. On choisit M couplage maximum tel que

$$M = \arg\min\{d(M, u, v) \text{ avec } M \text{ couplage max et u,v non couverts par } M\}.$$

Alors $d(M, u, v) > 1$ car si $d_G(u, v) = 1$, on peut ajouter l'arête uv au couplage M ce qui contredit sa maximalité. Soit t un sommet sur le plus court chemin de u à v dans G. t est couvert par M.

Soit N un couplage maximum de G ne couvrant pas t tel que

$$|M \cap N| = \max\{|M \cap M^*| \text{ tel que } M^* \text{ couplage maximum sur G}\}.$$

c'est-à-dire que la différence symétrique $M \Delta N$ est la plus petite possible au sens du cardinal.

Le sommet u n'est pas couvert par N car sinon $d(N, t, u) < d(M, u, v)$ ce qui contredit le fait que M est minimal pour $d(M, u, v)$. Il en est de même pour v. Comme $|M| = |N|$ et N ne couvre pas t, alors par hypothèse il existe un sommet $x \neq t$ non couvert par N. x est nécessairement couvert par M par hypothèse, en particulier $x \notin uv$. Soit $y \in V$ tel que $xy \in M$. Comme N est un couplage maximum, il existe $z \in V$ tel que $yz \in N$. On en déduit que $z \neq x$ car x n'est pas couvert par N.

Ainsi, $N^* = N \setminus \{yz\} \cup \{xy\}$ est un couplage maximum laissant t non couvert et tel que

$$|M \cap N^*| > |M \cap N|$$

ce qui contredit la relation de maximalité du cardinal de $|M \cap N|$. □

On présente ici le théorème de Tutte sur l'existence de couplage parfait dans un graphe général, vu comme conséquence de la formule de Tutte-Berge.

Théorème 2.4. *Un graphe fini $G = (V, E)$ a un couplage parfait si et seulement si*

$$o(G - U) \leq |U|, \ \forall U \subset V.$$

On appelle cette condition la condition de Tutte.

Proof. La condition de Tutte implique dans la formule de Tutte-Berge que $m(G) \geq \frac{1}{2}|V|$ or un couplage parfait possède un cardinal $\frac{1}{2}|V|$ donc il existe un couplage parfait sur G. □

Remarque 2.1. On peut démontrer directement le théorème de Tutte sans passer par la formule de Tutte-Berge, voir par exemple [1].

2.4 Lien entre le théorème de Tutte et le théorème de Hall

Nous allons montrer ici que la condition de Tutte définie au théorème 2.4 dans le cas d'un graphe biparti (ayant le même nombre de sommets dans chacun des ensembles de la bipartion des sommets) implique la condition de Hall définie dans le théorème 2.2.

Corollaire 2.5. *Soit $G = (A, B, E)$ un graphe biparti avec $|A| = |B|$. Si le graphe G respecte la condition de Tutte, alors il respecte également la condition de Hall.*

Proof. Soit $G = (A, B, E)$ un graphe biparti avec $|A| = |B|$.
On va raisonner par contraposée. Soit $S \subset A$ un sous-ensemble ne respectant pas la condition de Hall, c'est-à-dire $|S| > |N(S)|$. Tout élément de S est adjacent à un élément de $N(S)$ par définition. Il suit donc que tous les sommets de S dans le graphe de $G - N(S)$ sont isolés (n'ont pas de voisin dans $G - N(S)$). Il y a donc au moins $|S|$ sommets isolés dans $G - N(S)$ ce qui implique que $o(G - N(S)) > |N(S)|$ ce qui contredit la condition de Tutte.

□

3 Résultats sur les files d'attentes dans des modèles d'appariement

Il existe deux grands modèles de couplage à temps discret étudiés dans la littérature. Le premier, le plus général (on peut ramener l'étude du modèle que l'on présentera dans la section 3.4 à l'étude de ce modèle, voir [6] pour ce résultat), est le *bipartite matching model*. Le modèle que nous présentons a été étudié par exemple dans [4] et dans [5].

3.1 Le modèle de couplage stochastique sur un graphe biparti, ou Bipartite Matching Model (BMM)

Le modèle est le suivant : on considère tout d'abord un graphe biparti $G = (A, B, E)$ dont les sommets sont des labels. L'ensemble A représente les labels d'entités appelées traditionnellement *clients* et l'ensemble B représente les labels d'entités que l'on nomme *serveurs*. Les arêtes du graphe représentent les couplages admissibles entre les labels de ensemble A avec les labels de l'ensemble B. On appelle ce graphe *graphe d'interaction* ou *graphe d'appariement*. On considère le système suivant : un interface reçoit des couples d'entités *client-serveur*.

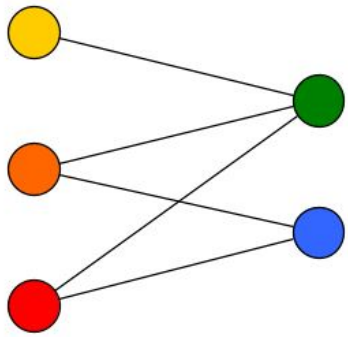


Figure 9: Exemple d'un graphe d'appariement

Ces arrivées se font non-simultanément et à temps discret. Leurs labels respectifs sont tirés au sort selon une certaine loi de probabilité , notée μ, sur $A \times B$. On suppose que ce tirage au sort est indépendant du tirage au sort des labels du couple précédemment rentré dans l'interface. Le système apparie les entités présentes dans l'interface en fonction des compatibilités décrites par le graphe G. Si plusieurs choix sont possibles pour l'appariement d'un client ou d'un serveur, le système utilise une *politique de choix*, notée Φ, afin de déterminés quelle entité sera appariée (nous détaillerons précisément ce qu'est une politique de choix dans la suite et nous donnerons deux exemples de politiques de choix). Le système conserve dans une double file d'attente les entités de type client et les entités de type serveur non appariées. On notera $X_n = (X_n^A, X_n^B)$ l'état de cette file d'attente à l'instant n. On montrera dans la suite que cette file d'attente est une chaîne de Markov, on détaillera l'espace d'états en fonction de la politique de choix que l'on utilisera.

Remarque 3.1. Nous considérerons toujours un graphe d'appariement connexe. Si le graphe d'appariement n'est pas connexe, on peut toujours se ramener à l'étude de chacune des composantes connexes du graphe. De même, nous exclurons les graphes de configuration complets, leur étude est triviale puisque l'on peut toujours coupler les entités qui arrivent avec n'importe quel élément de la file d'attente.

On va tout d'abord démontrer le lemme suivant :

Lemme 3.1. *X_n^A et X_n^B contiennent le même nombre d'éléments.*

Proof. Tout d'abord les entrées d'entités dans l'interface se font toujours par paire client-serveur. A l'instant n il y a donc $2n$ entités dans l'interface, n sont de type client et n sont de type serveur. De plus, comme le graphe d'appariement est biparti, il n'y a pas de couplage possible entre deux entités de type client ou entre deux entités de type serveur. Ainsi les entités de l'interface sortent toujours du système par paire client-serveur.

Il suit donc que, à l'instant n,

$$|X_n^A| = n - \#\{\text{paires client-serveur sorties du système à l'instant } n\} = |X_n^B|.$$

Cela conclut la démonstration. □

3.2 Politiques de choix admissibles

Soit $G = (A, B, E)$ un graphe biparti, $\mu = \mu_a \otimes \mu_b$ une mesure de probabilité sur $A \times B$.

Informellement, une politique de choix est dite admissible si :

- elle tient compte uniquement de l'état présent de la file d'attente (propriété de Markov)
- elle choisit en priorité des éléments présents dans la file d'attente

Dans cette partie nous étudierons deux politiques de choix pour la file d'attente : la politique First-In First-Out (FIFO) et la politique Match the Longest (ML).

3.2.1 Politique First-In First-Out

Un mot fini sur A (resp. B) est une suite finie $(a_1, ..., a_k) = a_1...a_k$ d'éléments de A (resp. B). On note A^* l'ensemble des mots finis sur A et on note $(A^*, \cdot)$ la structure composée de l'ensemble des mots finis sur A^*, muni de l'opération de concaténation usuelle. Cette opération est associative mais n'est pas commutative.

Soit $a_i \in A$, soit $U \subset A$. On note :

- $N(a_i) = \{b \in B | \{a_i, b\} \in E\}$;
- $N(U) = \bigcup_{a \in U} N(a)$.

On note A^k, B^k l'ensemble des mots de longueur k sur A, B. Alors on peut prendre pour espace d'états du processus (X_n) l'ensemble :

$$\mathcal{E} = \left\{ (a,b) \in \bigcup_{k\geq 0} A^k \times B^k | N(a) \cap \{b\} = \emptyset \text{ et } N(b) \cap \{a\} = \emptyset \right\}.$$

Nous allons désormais présenter la fonction de mise à jour de la file d'attente en fonction de la politique FIFO. Pour rappel, si $(a,b) \in \mathcal{E}$ alors $|a| = |b|$.

Soit $a \in A^k$, $b \in B^k$ deux mots de longueurs $k \leq n$ et $X_n = (a,b)$ l'état de la chaîne à l'instant n. De plus, on pose $X_0 = \emptyset$ et $(Y_n)_{n\geq 1}$ une suite de v.a. i.i.d. de loi $\mu_a \otimes \mu_b$ sur $A \times B$. On note $a_{-i} = a_1 ... a_{i-1} a_{i+1} ... a_k$.

La suite $(Y_n)_{n\geq 1}$ est indépendante de X_0. On pose $X_{n+1} = \Phi(X_n, Y_{n+1})$ où

$$\begin{aligned} \Phi \ : \ & \mathcal{E} \times (A \times B) \longrightarrow \mathcal{E} \\ & ((a,b),(a^*,b^*)) \longmapsto \Phi((a,b),(a^*,b^*)) \end{aligned}$$

avec

$$\begin{aligned} \Phi((a,b),(a^*,b^*)) =& (a,b) \text{ si } b^* \notin N(a) \text{ et } a^* \notin N(b) \text{ et } (a^*,b^*) \in E \\ =& (aa^*, bb^*) \text{ si } b^* \notin N(a) \text{ et } a^* \notin N(b) \text{ et } (a^*,b^*) \notin E \\ =& (a_{-i^*}, b_{-j^*}) \text{ si } b^* \in N(a) \text{ et } a^* \notin N(b) \\ =& (a_{-i^*}a*, b) \text{ si } b^* \in N(a) \text{ et } a^* \notin N(b) \\ =& (a, b_{-j^*}b^*) \text{ si } b^* \notin N(a) \text{ et } a^* \in N(b). \end{aligned}$$

en notant $i^* = \arg\min\{a_k, a_k \in N(b^*)\}$ et $j^* = \arg\min\{b_k, b_k \in N(a^*)\}$. La politique FIFO est admissible car elle définit une chaîne de Markov et choisit en priorité des éléments de la file d'attente. La difficulté de l'utilisation de cette politique vient de l'espace d'états, c'est pourquoi nous utiliserons la politique *Match the Longest* qui permet de travailler sur des espaces usuels.

3.2.2 Politique Match the Longest

Avec cette politique de choix, nous ne stockons pas des mots dans la file d'attente mais nous comptons simplement les occurrences de chaque élément n'ayant pas pu être couplé dans le processus et on garde cette information dans un vecteur. Formellement, on pose $x \in \mathbb{N}^{|A|}$ avec x_k le nombre d'éléments de A de type k, $y \in \mathbb{N}^{|B|}$ avec y_k le nombre d'éléments de B de type k. Nous avons démontré précédemment qu'il y a autant d'éléments non stockés d'un type $a \in A$ que d'éléments stockés d'un type $b \in B$. On en déduit que

$$\sum_{k=1}^{|A|} x_k = \sum_{k=1}^{|B|} y_k.$$

De plus, si $(a,b) \in E$, alors $x_a y_b = 0$. En effet si $x_a y_b \neq 0$, alors cela signife que l'on peut ajouter au couplage des arêtes ab jusqu'à ce qu'il n'y ait plus l'un ou l'autre label dans la file d'attente. On considère donc la file d'attente $(X_n)_{n\geq 0}$ ayant pour espace d'états :

$$\mathcal{E} = \left\{ (x,y) \in \mathbb{N}^{|A|} \times \mathbb{N}^{|B|} \ ; \ \sum_{k=1}^{|A|} x_k = \sum_{k=1}^{|B|} y_k, \ \forall (a,b) \in E, \ x_a y_b = 0 \right\}.$$

La politique ML consiste, lorsqu'on a le choix entre plusieurs sommets, à choisir celui en plus grand nombre dans la file d'attente. On pose $(Y_n)_{n\geq 1}$ une suite de v.a. i.i.d. de loi $\mu_a \otimes \mu_b$ sur $A \times B$ et $X_0 = 0 \in \mathbb{N}^{|A|} \times \mathbb{N}^{|B|}$. La suite $(Y_n)_{n\geq 1}$ est indépendante de X_0. On pose $X_{n+1} = \Phi(X_n, Y_{n+1})$ où

$$\begin{aligned} \Phi \ : \ & \mathcal{E} \times (A \times B) \longrightarrow \mathcal{E} \\ & ((x,y),(a,b)) \longmapsto \Phi((x,y),(a,b)) \end{aligned}$$

avec, en posant $e_i \in \mathbb{N}^{|A|}$ tel que $(e_i)_j = \delta_{i,j}$ et $f_i \in \mathbb{N}^{|B|}$ tel que $(f_i)_j = \delta_{i,j}$:

$$\begin{aligned} \Phi((x,y),(a,b)) &= (x,y) \text{ si } x_{N(b)} = 0 \text{ et } y_{N(a)} = 0 \text{ et } (a,b) \in E \\ &= (x + e_a, y + e_b) \text{ si } x_{N(b)} = 0 \text{ et } y_{N(a)} = 0 \text{ et } (a,b) \notin E \\ &= (x - e_{\phi(x,b)}, y - e_{\psi(y,a)}) \text{ si } x_{N(b)} \neq 0 \text{ et } y_{N(a)} \neq 0 \\ &= (x - e_{\phi(x,b)} + e_a, y) \text{ si } x_{N(b)} \neq 0 \text{ et } y_{N(a)} = 0 \\ &= (x, y - e_{\psi(y,a)} + e_b) \text{ si } x_{N(b)} = 0 \text{ et } y_{N(a)} \neq 0. \end{aligned}$$

avec $\phi(x,b) \sim \mathcal{U}(\arg\max\{x_k | x_k \in N(b)\})$ et $\psi(y,a) \sim \mathcal{U}(\arg\max\{y_k | y_k \in N(a)\})$. Il est à noté que $x_{N(b)}$ et $y_{N(a)}$ sont potentiellement des vecteurs.

La file d'attente $(X_n)_{n\geq 0}$ est donc une chaîne de Markov. Soit (x,y) un état de la chaîne. Partant de (x,y), on peut toujours retourner à l'état $(0,0)$ en un nombre fini d'étape en tirant successivement des paires de sommets pouvant être couplés avec des éléments de la file d'attente. De même, partant de $(0,0)$, on peut arriver en un nombre fini d'étape dans tout $(x,y) \in \mathcal{E}$. La chaîne est donc irréductible. De plus, pour tout $(x,y) \in \mathcal{E}$, $p_{(x,y),(x,y)} > 0$, la chaîne est donc apériodique.

3.3 Comportement de la file d'attente en temps long

Nous allons tout d'abord donner quelques définitions avant de présenter le grand théorème de cette section qui nous donne des conditions sur la loi μ pour que la file d'attente ne contiennent aucune entité à coupler et cela pour une infinité d'étape. On pourra trouver plus de détails sur les chaînes de Markov dans [7].

Définition 3.1. Soit $(X_n)_{n\geq 0}$ une chaîne de Markov irréductible, soit i un état de la chaîne. On note V_n^i le nombre de passage de la chaîne dans l'état i à l'instant n, défini par

$$V_n^i = \sum_{1\leq k\leq n} \mathbf{1}_{\{X_k=i\}}.$$

On note également V^i le nombre de passages total dans l'état i, c'est-à-dire

$$V^i = \sum_{k\geq 1} \mathbf{1}_{\{X_k=i\}}.$$

On a la dichotomie suivante : soit

$$\mathbb{P}(V^i = +\infty) = 1$$

et dans ce cas on dit que l'état i est *récurrent*, soit

$$\mathbb{P}(V^i = +\infty) = 0$$

et dans ce cas on dit que l'état i est *transient*. On distingue pour un état i la notion de *récurrence positive* où dans ce cas

$$\lim_{n\longrightarrow+\infty} \frac{V_n^i}{n} = c_i > 0$$

et la notion de *récurrence nulle* où dans ce cas

$$\lim_{n\longrightarrow+\infty} \frac{V_n^i}{n} = 0.$$

Il est à noté que lorsque la chaîne est irréductible, tous les états de la chaîne sont de même nature. Dans notre modèle, on souhaite que la file d'attente soit récurrente pour que l'état de la file où cette dernière est vide soit visités une infinité de fois.

Nous considérerons pour la suite que les lois marginales de μ que sont μ_A et μ_B sont indépendantes, c'est-à-dire $\mu = \mu_A \otimes \mu_B$.

Définition 3.2. On dit que la mesure μ vérifie la condition *NCOND* si :

$$\forall U \subset A,\ \mu_A(U) < \mu_B(N(U)),\ \forall V \subset B,\ \mu_B(V) < \mu_A(N(V)).$$

Soit $\mathcal{M}^+$ l'ensemble des mesures de probabilités sur $A \times B$. On peut ainsi définir l'ensemble

$$NCOND(G) = \left\{\mu = \mu_A \otimes \mu_B \in \mathcal{M}^+ \mid \forall U \subset A,\ \mu_A(U) < \mu_B(N(U)),\ \forall V \subset B,\ \mu_B(V) < \mu_A(N(V))\right\}.$$

On définit également l'ensemble

$$STAB(G,\Phi) = \left\{\mu \in \mathcal{M}^+ \mid (X_n) \text{ est récurrente positive } \right\}.$$

On dit enfin que la politique de choix est maximale lorsque

$$STAB(G,\Phi) = NCOND(G).$$

Nous allons désormais présenter le théorème principal de cette section. Nous donnerons une preuve alternative et détaillée de celle présentée en [5].

Théorème 3.1. *Soit $G = (A, B, E)$ un graphe d'appariement biparti et μ une loi de probabilité sur $A \times B$ telle que $\mu = \mu_A \otimes \mu_B$. Supposons que X_n soit récurrente positive. Alors μ vérifie NCOND sur G.*

Proof. Nous allons raisonner par contraposée. Supposons qu'il existe $U \subsetneq A$ tel que $\mu_A(U) > \mu_B(N(U))$. On pose

$$U_n = \sum_{k=1}^{n} \mathbf{1}_{\{Y_k^1 \in U\}}$$

et

$$V_n = \sum_{k=1}^{n} \mathbf{1}_{\{Y_k^2 \in N(U)\}}.$$

La variable $\mathbf{1}_{\{Y_k^1 \in U\}}$ est une variable de Bernoulli de paramètre $\mu_A(U)$ et la variable $\mathbf{1}_{\{Y_k^2 \in N(U)\}}$ est une variable de Bernoulli de paramètre $\mu_B(N(U))$. Par la loi forte des grands nombres, il suit que :

$$\lim_{n \longrightarrow +\infty} \frac{U_n}{n} = \mathbb{E}[\mathbf{1}_{\{Y_1^1 \in U\}}] = \mu_A(U) \text{ p.s.}$$

et

$$\lim_{n \longrightarrow +\infty} \frac{V_n}{n} = \mathbb{E}[\mathbf{1}_{\{Y_1^2 \in N(U)\}}] = \mu_B(N(U)) \text{ p.s.}$$

On pose X_n^U le nombre d'éléments de U présent dans la file d'attente à l'instant U. Immédiatement, $X_n^U \geq U_n - V_n$.

Supposons que $\frac{X_n^U}{n} \longrightarrow c > 0$. Alors il suit que $X_n^U = O(n)$ et donc $X_n^U \longrightarrow +\infty$. Or la chaine est récurrente positive, donc l'état $\emptyset$ est visité une infinité de fois. Il suit donc que X_n^U ne peut tendre vers l'infini. Donc si X_n est récurrente positive alors $\frac{X_n^U}{n} \longrightarrow 0$.

On remarque immédiatement que

$$X_n^U \geq U_n - V_n$$

donc en passant à la limite, p.s. :

$$\frac{X_n^U}{n} \longrightarrow \mu_A(U) - \mu_B(N(U)) > 0.$$

Ceci implique que la file d'attente est transiente. Donc si (X_n) est récurrente positive,

$$\mu_A(U) \leq \mu_B(N(U)), \ \forall U \subsetneq A.$$

Traitons désormais le cas $\mu_A(U) = \mu_B(N(U))$. Nous allons montrer que dans ce cas la chaîne est récurrente nulle. Supposons qu'il existe $U \subsetneq A$ tel que

$$\mu_A(U) = \mu_B(N(U)). \quad (1)$$

Nécessairement, $N(U) \neq B$ car sinon $\mu_B(N(U)) = 1 = \mu_A(U)$ ce qui contredit le fait que U est strictement inclus dans A. On pose $V = B \setminus N(U)$. Alors :

$$\mu_A(U) = \mu_B(N(U)) \iff \mu_A(A \setminus U) = \mu_B(V)$$

Il suit que $(U \times V) \cap E = \emptyset$, i.e. il n'y a pas d'arêtes entre les sommets de U et les sommets de V.

$$\mu(U \times V) = \mu_A(U)\mu_B(B \setminus N(U)) = \mu_A(U)(1 - \mu_B(N(U))) = \mu_A(U) - \mu(U \times N(U)).$$
$$\mu((A \setminus U) \times N(U)) = \mu_A(A \setminus U)\mu_B(N(U)) = (1 - \mu_A(U))\mu_B(N(U)) = \mu_B(N(U)) - \mu(U \times N(U)).$$

En utilisant (1), on obtient :

$$\mu(U \times V) = \mu((A \setminus U) \times N(U)).$$

On définit U_n, V_n et X_n^U comme précédemment. On appelle D_n le nombre de départ d'éléments de l'ensemble $(A \setminus U) \times N(U)$ présents dans la file d'attente. On pose $Z_n = U_n - V_n$. Si la file d'attente reçoit un élément de $U \times V$ alors U_n augmente et V_n reste inchangé donc $Z_n = Z_{n-1} + 1$. De même, si la file reçoit un élément de $(A \setminus U) \times N(U)$, alors $Z_n = Z_{n-1} - 1$. Les deux autres cas ne changent pas la valeur de Z_n (on ajoute ou non un élément à U_n et V_n). Il suit que

$$X_n \geq U_n - V_n + D_n \geq U_n - V_n = Z_n.$$

Comme précédemment on utilise la loi des grands nombres pour avoir des résultats asymptotiques.

Supposons tout d'abord que $\mu(U \times V) > 0$. Alors $\mu((A \setminus U) \times N(U)) > 0$ et $\lim\limits_{n \longrightarrow +\infty} \frac{U_n - V_n + D_n}{n} = \mu_A(U) - \mu_B(N(U)) + \mu((A \setminus U) \times N(U)) > 0$ donc la file $X_n^U = O(n)$ et donc la chaîne est transiente.

Supposons enfin que $\mu(U \times V) = 0$. Par indépendance des marginales, $\mu_A(U) = 0$ ou $\mu_B(V) = 0$ ce qui est impossible puisque cela reviendrait à supprimer U ou V du graphe d'interaction. □

3.4 Le modèle de couplage stochastique sur un graphe général ou General Stochastic Matching Model (GSMM)

Le GSMM (nous nous appuierons sur le document [6]) est une variante du BMM où l'on considère que les entrées d'entités se font une par une, il y a donc un seul type d'entité. On suppose avoir un graphe d'appariement $G = (A, E)$ où les sommets A sont les labels des entités qui vont entrer dans le système. Les arêtes du graphe représente les couplages possibles entre les différents labels. On considère le système suivant : un interface reçoit à chaque étape (à temps discret) une entité. Son label est tiré selon une loi de probabilité μ sur A. Les tirages au sort des labels sont supposés indépendant.

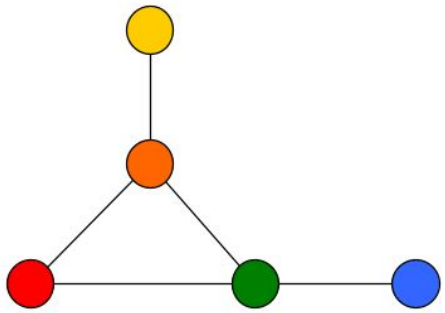


Figure 10: Exemple d'un graphe d'appariement général.

Le système apparie les entités présentes dans l'interface selon les couplages possibles décrit par le graphe d'appariement et si plusieurs choix sont possibles, une politique de choix Φ détermine quelle entité sera choisie dans le couplage. Le système conserve dans une file d'attente les entités non appariées. On notera X_n l'état de la file d'attente à l'instant n.

Comme précédemment une politique de choix est admissible si elle induit un comportement Markovien de la file d'attente et si elle choisit en priorité les éléments déjà présents dans la file.

Politique First-In First-Out :

L'espace d'état de la file d'attente (X_n) est l'ensemble suivant :

$$\mathcal{E} = \left\{ a = (a_1 ... a_k) \in \bigcup_{k \geq 0} V^k \mid \forall i < k; a_i a_k \notin E \right\}.$$

On pose (X_n) l'état de la chaîne à l'instant n, $X_0 = \emptyset$ et (Y_n) suite de v.a. i.i.d. de loi μ sur V. La suite (Y_n) est indépendante de X_0. On pose $X_{n+1} = \Phi(X_n, Y_{n+1})$ avec :

$$\begin{aligned} \Phi \ : \ & \mathcal{E} \times V \longrightarrow \mathcal{E} \\ & (a,v) \longmapsto \Phi((a,v)) \end{aligned}$$

telle que

$\Phi((a,v)) = av$ si il n'existe pas de $i \leq k$ tel que $a_i v \in E$;
$\Phi((a,v)) = a_{-i^*}$ si il existe $\{i_1, ..., i_p\}$ tel que $a_i v \in E \ \forall i \in \{i_1, ..., i_p\}$ avec $i^* = \arg\min\{a_{i_1}, ..., a_{i_p}\}$.

La file d'attente (X_n) est donc une chaîne de Markov irréductible et périodique de période 2 (voir la remarque précédente).

Politique Match the Longest :

On pose $x \in \mathbb{N}^{|V|}$ avec x_k le nombre d'éléments de type k stockés dans la file d'attente. $\forall i \in V$, soit $e_i \in \mathbb{N}^{|V|}$ tel que $(e_i)_j = \delta_{i,j}$. On considère le processus (X_n) à valeurs dans

$$\mathcal{E} = \left\{ x \in \mathbb{N}^{|V|} \mid \forall (i,j) \in E \ : \ x_i.x_j = 0 \right\}.$$

On pose (X_n) l'état de la chaîne à l'instant n, $X_0 = 0 \in \mathbb{N}^{|V|}$ et (Y_n) suite de v.a. i.i.d. de loi μ sur V. La suite (Y_n) est indépendante de X_0. On pose $X_{n+1} = \Phi(X_n, Y_{n+1})$ avec :

$$\begin{aligned} \Phi \ : \ \mathcal{E} \times V &\longrightarrow \mathcal{E} \\ (x, v) &\longmapsto \Phi((x, v)) \end{aligned}$$

tel que

$$\begin{aligned} \Phi(x,v) &= x + e_v \text{ si } \forall i \in N(v), x_i = 0 \\ &= x - e_j \text{ sinon, avec } j = \arg\max\{i \in N(v); x_i \neq 0\}. \end{aligned}$$

La file d'attente ainsi définit est donc également une chaîne de Markov irréductible et périodique de période 2.

3.5 Résultats existants sur le *Stochastic Matching Model*

Soit $G = (V, E)$ un graphe, μ une loi de probabilité sur V et Φ une politique de choix. On peut définir comme précédemment les ensembles

$$NCOND(G) = \{\mu \text{ probabilité sur V telle que } \mu(U) < \mu(N(U)) \ \forall U \subsetneq V\}$$

et

$$STAB(G, \Phi) = \{\mu \text{ probabilité sur V telle que } (X_n) \text{ est récurrente positive pour la politique } \Phi\}.$$

On va présenter différents résultats existant sur ce modèle sans preuve. Tout d'abord un résultat analogue au théorème 3.1 dans le cas général. Les preuves de ces résultats sont disponibles dans [6].

Théorème 3.2. *Soit G un graphe général, soit μ une loi de probabilité sur les sommets de G. Si la file d'attente (X_n) est récurrente positive alors μ vérifie NCOND sur G.*

Remarque 3.2. Pour la plupart des graphes (sauf cas simples avec peu de sommets), on peut construire une politique de choix Φ tels que

$$STAB(G, \Phi) \subsetneq NCOND(G).$$

Cela signifie que sur ce graphe G on peut trouver une probabilité sur les sommets μ respectant la condition *NCOND* et ne rendant pas la file d'attente (X_n) récurrente positive. Un tel contre-exemple est donné dans [8].

On dispose d'un résultat qui donne une équivalence entre la condition *NCOND* et une autre condition plus simple à vérifier. On dit que la mesure μ vérifie la condition *NCOND** sur le graphe $G(A, E)$ si :

$$\text{Pour tout ensemble indépendant } I \subsetneq A,\ \mu(I) < \mu(N(I)).$$

On appelle

$$NCOND^*(G) = \{\mu \text{ probabilité sur V telle que } \mu(I) < \mu(N(I))\ \forall I \in \mathbb{I}\}.$$

Le résultat suivant est prouvé dans [6].

Proposition 3.1. *NCOND(G)=NCOND*(G).*

La condition *NCOND* en pratique est difficile à vérifier car le nombre de sous-ensembles de sommets croît exponentiellement avec le nombre de sommets du graphe. La condition *NCOND** concerne un plus faible nombre de sous-ensembles et est donc plus rapide à vérifier.

On dispose enfin d'un résultat sur le comportement de la file d'attente en temps long lorsque l'on considère la politique de choix ML.

Théorème 3.3. *Si G n'est pas biparti alors les politiques de choix FIFO et Ml sont maximales, c'est-à-dire que toute loi de probabilité sur V vérifiant NCOND est une loi rendant la file d'attente récurrente positive en utilisant l'une des deux politiques de choix FIFO ou ML.*

Le résultat pour ML est prouvé dans [6] et le résultat pour FIFO est donné dans [9].

Corollaire 3.4. *Si G est biparti alors la file d'attente (X_n) n'est pas récurrente positive ($STAB(G, \Phi) = \emptyset,\ \forall \Phi$ politique admissible).*

Ainsi, si on considère un graphe biparti alors pour toute politique de choix, la file d'attente ne sera pas récurrente positive. On ne peut donc travailler avec des arrivés de sommet unique. Il faut nécessaire se placer dans le cadre du *Bipartite Stochastic Matching Model* où les sommets arrivent deux par deux, avec un tirage pour chaque ensemble de la bipartition à chaque étape.

4 Algorithme de construction d'un couplage sur un graphe aléatoire avec un grand nombre de sommets

La recherche d'algorithme *en ligne* pour la construction de couplage parfait est une voix de recherche qui est exploré depuis le début des années 90 (voir par exemple [10]). Nous allons réinterpréter les résultats de file d'attente comme des résultats donnant la construction d'un algorithme de couplage de labels *en ligne* puis nous présenterons les liens entre ce modèle et les modèles de file d'attente précédemment décrit. On va construire tout d'abord une suite de graphe aléatoire (G_n) de la manière suivante : à chaque étape, une nouvelle entité (ou couple d'entités) rentre dans le système. Son label est tiré au sort et on construit un graphe en ajoutant à chaque étape les labels tirés sous forme de sommets ainsi que les couplages admissibles sous forme d'arêtes entre les labels. On construit sur ce graphe un couplage à l'aide d'un algorithme que l'on va présenter. On cherche à ce que ce couplage soit parfait. On pourrait mettre en oeuvre l'algorithme *blossom*, mais le faire à chaque nouvelle arrivée demande beaucoup de temps de calcul. Nous allons donc présenter un algorithme en ligne permettant la construction d'une suite de couplages maximums sur cette suite de graphes aléatoires. Nous allons, grâce aux résultats du modèle de file d'attente, montrer que cet algorithme est aussi bon que l'algorithme *blossom* sur un grand nombre d'étapes, c'est-à-dire que lorsqu'il y a eu un très grand nombre d'arrivées notre algorithme construit des couplages parfaits presque sûrement.

4.1 Algorithme général de construction d'un couplage sur un graphe aléatoire biparti ayant un grand nombre de sommets

On considère un graphe d'interaction biparti $G = (A, B, E)$ représentant les liens possibles entre deux classes d'objets ou d'individus. Prenons pour exemple le graphe suivant :

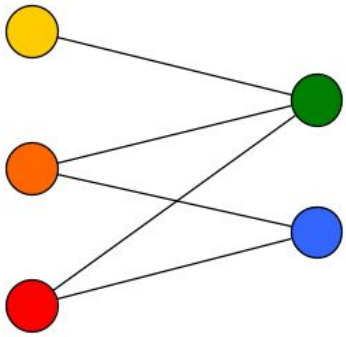


Figure 11: Graphe d'appariement biparti.

On considère la dynamique du BMM. On va tout d'abord construire une suite de graphes $(G_n = (A_n \cup B_n, E_n))_{n\geq 0}$ avec les couples de labels obtenus selon l'algorithme suivant :

- On initialise G_0 comme étant le graphe vide;

- $\forall n \geq 1$:
  - On tire un couple de labels (a_n, b_n) selon la loi $\mu_a \otimes \mu_b$;
  - $A_n = A_{n-1} \cup \{a_n\}$, $B_n = B_{n-1} \cup \{b_n\}$;
  - $\forall m \leq n$:
    * Si a_n et b_m sont adjacents (voisins) dans G alors $E_n = E_{n-1} \cup (a_n, b_m)$;
    * Si b_n et a_m sont adjacents dans G alors $E_n = E_{n-1} \cup (a_m, b_n)$.

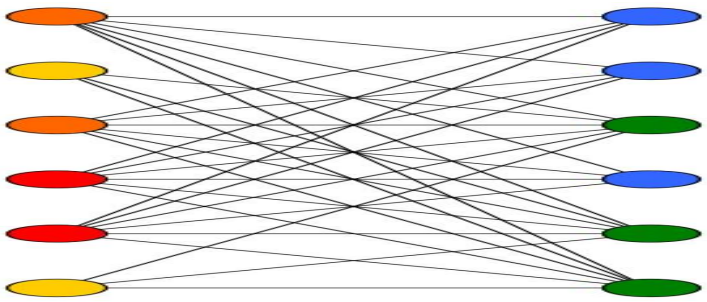


Figure 12: Exemple d'un graphe possible G_6 en respectant le graphe d'appariement présenté précédemment.

Remarque 4.1. A_n et B_n sont des multi-ensembles, ils peuvent contenir plusieurs fois le même label.

On va construire une suite de couplages maximums $(M_n = (A_n \cup B_n, F_n))_{n\geq 0}$ sur les graphes $(G_n)_{n\geq 0}$.

- M_0 est le couplage vide
- $\forall n \geq 1, \forall m \leq n$
  - On pose $N(a_n) \subset B_n$ l'ensemble des sommets adjacents à a_n dans G_n et $N^*(a_n)$ le sous-ensemble de $N(a_n)$ contenant les sommets non précédemment couplés. Trois cas sont possibles :
    * Si $N^*(a_n) = \emptyset$ alors on garde en attente a_n dans une file d'attente X_n^A. On concatène par la droite les nouveaux sommets dans la file d'attente. $M_n = (A_n \cup B_n, F_{n-1})$;
    * Si $|N^*(a_n)| = 1$ alors il existe un sommet voisin de a_n non couplé que l'on désigne par s, $1 \leq s \leq n$. On ajoute l'arête (a_n, s) dans notre couplage, c'est-à-dire $M_n = (A_n \cup B_n, F_{n-1} \cup (a_n, s))$;
    * Si $|N^*(a_n)| > 1$ alors on choisit le sommet s à coupler avec a_n parmi les sommets disponibles selon une règle de choix Φ. $M_n = (A_n \cup B_n, F_{n-1} \cup (a_n, s))$;

– On répète l'algorithme pour b_n sauf s'il a été couplé avec a_n dans l'étape précédente, et en stockant les sommets non couplés dans une file d'attente X_n^B.

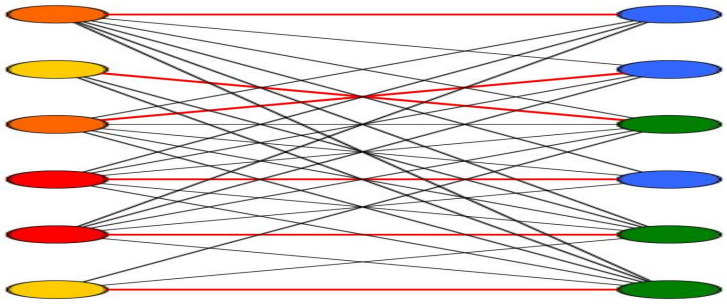


Figure 13: Exemple du couplage obtenu sur G_6 par l'algorithme, avec la politique de choix FIFO.

4.2 Algorithme général de construction d'un couplage sur un graphe aléatoire général ayant un grand nombre de sommets

On considère la dynamique du GSMM. On construit le graphe en fonction des arrivées et à chaque étape

On va tout d'abord construire à la volée une suite de graphes $(G_n = (V_n, E_n))_{n\geq 0}$ avec les couples de sommets obtenus selon l'algorithme suivant :

- On initialise G_0 comme étant le graphe vide;
- $\forall n \geq 1$:
  - On tire un sommet v_n selon la loi μ;
  - $V_n = V_{n-1} \cup \{v_n\}$;
  - $\forall m < n$: si v_n et v_m sont adjacents (voisins) dans G alors $E_n = E_{n-1} \cup (v_n, v_m)$.

On va construire une suite de couplages maximums $(M_n = (V_n, F_n))_{n\geq 0}$ sur les graphes $(G_n)_{n\geq 0}$.

- M_0 est le couplage vide;
- $\forall n \geq 1, \forall m < n$
- On pose $N(v_n) \subset V_n$ l'ensemble des sommets adjacents à v_n dans G_n et $N^*(v_n)$ le sous-ensemble de $N(v_n)$ contenant les sommets non précédemment couplés. Trois cas sont possibles :

- Si $N^*(v_n) = \emptyset$ alors on "stocke" v_n dans une file d'attente X_n. $M_n = (V_n, F_{n-1})$;
- Si $|N^*(v_n)| = 1$ alors il existe un sommet voisin de v_n non couplés que l'on désigne par s, $1 \leq s < n$. On ajoute l'arête (v_n, s) dans notre couplage, c'est-à-dire $M_n = (V_n, F_{n-1} \cup (v_n, s))$;
- Si $|N^*(v_n)| > 1$ alors on choisit le sommet s à coupler avec v_n parmi les sommets disponibles selon une règle de choix Φ. $M_n = (V_n, F_{n-1} \cup (v_n, s))$.

Remarque 4.2. Le processus de file d'attente est périodique de période 2. En effet la file ne peut pas rester dans le même état en 1 pas de temps car soit le nouveau sommet arrivant est couplé avec un ancien et la file se réduit de 1, soit le nouveau sommet ne peut être couplé avec un élément de la file et donc la taille de la file augmente de 1. Il faut donc nécessairement un nombre pair d'étapes (au minimum 2) pour que la file se retrouve dans son état initial.

4.3 Lien entre l'algorithme et le modèle de file d'attente

Nous allons montrer le lien existant entre la construction de couplage parfait sur la suite de graphes construite par l'algorithme décrit précédemment et le comportement de la file d'attente en temps long, décrit dans la section 3.

Avant de présenter les résultats, nous allons introduire quelques notations. On considère un graphe $G = (V_1, V_2, E)$ et $(Y_n)_n$ une suite de v.a. i.i.d. de loi μ sur $V_1 \times V_2$. On pose, pour tout $U \subsetneq V_1$ (resp. $U \subsetneq V_2$),

$$P_n(U) =: \sum_{1 \leq k \leq n} \mathbf{1}_{\{Y_k \in U\}}.$$

On peut désormais construire l'évènement $A_n(U)$ comme suit : pour tout $U \subsetneq V_1$ (resp. V_2), on pose :

$$A_n(U) = \{P_n(U) < P_n(N(U)\},$$

et on définit l'évènement

$$A_n = \bigcap_{U \subsetneq V_1} A_n(U).$$

Si l'évènement A_n est réalisé pour un graphe biparti, alors cela signifie qu'à l'instant n, il existe un couplage parfait sur le graphe G_n construit par notre algorithme en ligne (puisque cet évènement implique la condition de Hall). En particulier l'algorithme *blossom* utilisé sur le graphe G_n construit un couplage parfait.

On va également définir l'évènement B_n par

$$B_n = \{X_n = 0\}.$$

Si l'évènement B_n est réalisé, alors notre file d'attente est vide et donc notre algorithme en ligne a construit un couplage parfait sur G_n.

Proposition 4.1. *Soit $G = (V, E)$ un graphe d'appariement. On considère le système dynamique précédemment décrit. Si (X_n) est récurrente positive alors G_n admet un couplage parfait pour une infinité d'indice n, c'est-à-dire*

$$\mathbb{P}(\limsup B_n) = 1.$$

En notant $T_0 = \inf\{k \geq 0|\ |M_k| = \frac{|V|}{2}\}$ le temps du premier couplage parfait et pour tout entier $i \geq 0$,

$$T_0^i = \inf\left\{k \geq T_0^{i-1}|\ |M_k| = \frac{|V|}{2}\right\},$$

le temps du $i+1$-ième couplage parfait, on a qu'il existe $c_0 > 0$

$$\mathbb{E}[T_0^i] = \frac{1}{c_0}$$

c'est-à-dire que l'on construit un couplage parfait en moyenne toutes les $\frac{1}{c_0}$ étapes.

Proof. Supposons que la file d'attente (X_n) soit récurrente positive. Alors $\forall e \in \mathcal{E}$ avec $\mathcal{E}$ l'espace d'états de la file d'attente,

$$\lim_{n\longrightarrow+\infty} \frac{\sum_{k\geq 1} \mathbf{1}_{\{X_k=e\}}}{n} \longrightarrow c_e,$$

où c_e est une constante strictement positive. Cela signifie en particulier que la chaîne passe une fraction non nulle c_0 de son temps dans l'état 0, et donc que la file d'attente est vide une fraction positive du temps. Or si $X_n = 0$, M_n couvre donc tous les sommets de G_n et est donc un couplage parfait. On obtient donc qu'en moyenne, toutes les $\frac{1}{c_0}$ étape, notre algorithme construit un couplage parfait. Ainsi, l'évènement B_n est vérifié pour une infinité d'indices n, et donc

$$\mathbb{P}(\limsup B_n) = 1.$$

Cela achève la preuve. □

Proposition 4.2. *Soit $G = (V_1, V_2, E)$ un graphe d'appariement biparti. Supposons que $\mu \in NCOND(G)$ et que $(X_n)_n$ soit récurrente positive. Alors*

$$\mathbb{P}(\liminf A_n) = 1.$$

Cela signifie que presque-sûrement, à partir d'un certain rang aléatoire n_0, pour tout $n \geq n_0$, G_n admet un couplage parfait.

Proof. Supposons que (X_n) soit récurrente positive. Comme $\mu \in NCOND(G)$, alors par la loi forte des grands nombres, pour tout $U \subsetneq V_1$ (resp. V_2)

$$\lim_{n\longrightarrow\infty} \left(\frac{P_n(U)}{n} - \frac{P_n(N(U))}{n}\right) = \mu(U) - \mu(N(U)) < 0 \text{ p.s.}$$

En particulier, il existe p.s. n_0^U tel que pour tout $n \geq n_0^U$, $P_n(U) < P_n(N(U))$ p.s. On pose $n_0 = \max\{\max_{U \subsetneq V_1}\{n_0^U\}, \max_{U \subsetneq V_2}\{n_0^U\}\}$. Alors, p.s. pour tout $n \geq n_0$, pour tout $U \subsetneq V_1$ (resp. V_2), $P_n(U) < P_n(N(U))$. En d'autres termes, on a

$$\mathbb{P}(\liminf A_n) = 1,$$

c'est-à-dire que p.s. il existe un couplage parfait sur G_n à partir d'un certain rang. □

Remarque 4.3. En théorie il serait donc possible d'obtenir à partir d'un certain rang aléatoire un couplage parfait en utilisant l'algorithme *blossom*, mais il serait bien trop coûteux en pratique de l'exécuter à chaque étape. L'algorithme en ligne, bien que moins efficace dans le sens où l'on ne peut assurer la construction d'un couplage parfait à chaque étape à partir d'un certain rang, permet néanmoins de construire "régulièrement" (dans le sens définit dans la proposition 4.1) des couplages parfaits.

Nous allons désormais faire le lien entre les résultats du GSMM vu dans la section précédente et notre problématique.

1er résultat du GSMM : Si G n'est pas biparti alors ML est maximale.

ML est maximale donc en particulier la file d'attente est récurrente positive. On a de plus que (X_n) récurrente positive pour une certaine mesure $\mu \iff \mu$ vérifie $NCOND$. Ainsi, la file d'attente passe une fraction non négligeable de son temps dans l'état 0, i.e. $\exists c_0$ tel que

$$\frac{\sum_{k=1}^{n} \mathbf{1}_{\{X_k=0\}}}{n} \longrightarrow c_0 > 0$$

On aura donc construit un couplage parfait pour une infinité d'indice n. De plus, en moyenne toutes les $\frac{1}{c_0}$ étapes on construira un couplage parfait.

Cela signifie que pour ces indices n où la file d'attente est vide, il existe un couplage parfait sur G_n et donc par le théorème de Tutte,

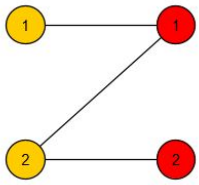
$$o(G_n - U) \leq |U| \ \forall U \subsetneq V_n$$

2nd résultat du GSMM : Si G est biparti alors la file d'attente (X_n) n'est pas récurrente positive ($STAB(G, \Phi) = \emptyset, \ \forall \Phi$ politique admissible)

Ce résultat implique que le temps moyen passé par la file d'attente dans l'état 0 tend vers 0. Ainsi, plus le graphe G_n est grand et plus il sera difficile de construire un couplage parfait sur ce dernier avec l'algorithme présenté en début de chapitre. Ceci est dû au fait qu'on aura des trajectoires du processus (Y_n) faisant de trop grandes excursions dans l'un ou l'autre des ensembles de la bipartition. Le graphe G_n pour n sera donc déséquilibré d'un coté de la bipartition ou de l'autre ce qui induit l'existence d'ensembles ne respectant pas la condition de Hall.

4.4 Etude d'un cas particulier

On va considérer un graphe de configuration simple, le plus simple possible étant un graphe biparti à 4 sommets.
On va donc considérer le graphe suivant : Les labels sont les numéros représentés sur les



sommets. On va supposer que $\mu_A(1) = p,\ \mu_A(2) = 1 - p$ et que $\mu_B(2) = q,\ \mu_B(1) = 1 - q$.
Les ensembles indépendants ne saturant pas l'ensemble A ou l'ensemble B sont les suivants :

$$\mathbb{I} = \{\{1\}, \{2\}, \{3\}, \{4\}, \{1, 4\}\}$$

En utilisant la condition *NCOND* sur les ensembles indépendants, on obtient le système suivant :

$$\begin{cases} p < 1 - q \\ 1 - q < 1 \\ 1 - p < 1 \\ q < 1 - p \\ pq < (1 - p)(1 - q) \end{cases}$$

ce qui nous donne

$$\begin{cases} p + q < 1 \\ p, q > 0. \end{cases}$$

Si l'algorithme utilise la politique de choix Match the Longest, alors on a équivalence entre *NCOND* et la récurrence positive de la file d'attente et donc

$$STAB(G, ML, \mu) = \{\mu = \mu_A \otimes \mu_B \mid \mu_A(1) + \mu_B(2) < 1\}$$

4.5 Application numérique

Nous allons implémenter l'algorithme de couplage en ligne pour un graphe général puis nous comparerons l'efficacité des deux politiques de choix que nous avons présenté dans ce mémoire, à savoir FIFO et ML. Les codes sont réalisés sous Python 3.7 et sont disponibles en annexes.

Nous travaillerons tout d'abord sur le graphe suivant :

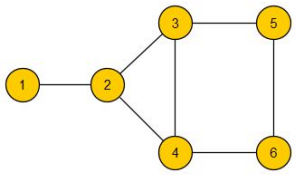


Figure 14: Graphe d'appariement

On utilisera les probabilités suivantes : $\mu(1) = \frac{1}{8}$, $\mu(2) = \frac{1}{4}$, $\mu(3) = \frac{1}{4}$, $\mu(4) = \frac{1}{4}$, $\mu(5) = \frac{3}{32}$ et $\mu(6) = \frac{1}{32}$. On peut vérifier que μ ainsi définie vérifie la condition *NCOND*.

On simule l'arrivée de 100 000 entités et on obtient que pour la politique FIFO, la file d'attente est vide, c'est-à-dire que l'algorithme a construit un couplage parfait 5539 fois, c'est-à-dire $5,539\%$ du temps. Pour la politique MF, la vide d'attente a été vidé 4980 fois, soit $4,98\%$ du temps. Les résultats sont donc très similaires entre ces deux politiques. Ci-dessus, nous avons tracé la trajectoire de la file d'attente (en terme de sommets restant à coupler) pour 100 arrivées pour les deux politiques de choix.

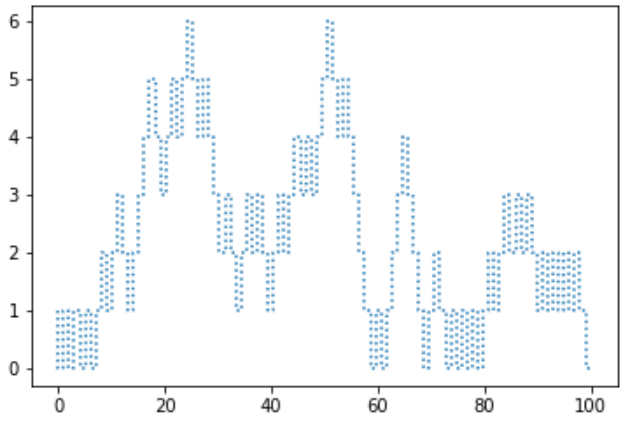


Figure 15: Nombre de sommets dans la file d'attente pour la politique FIFO

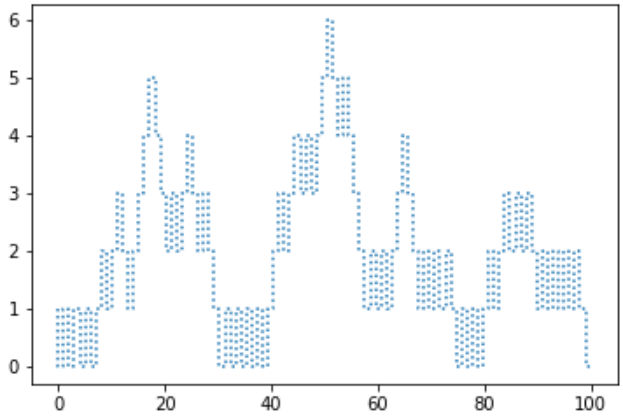


Figure 16: Nombre de sommets dans la file d'attente pour la politique ML

Nous allons désormais étudier le cas du graphe d'appariement suivant :

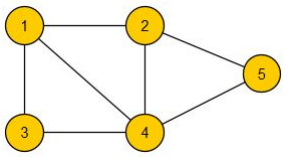


Figure 17: Graphe d'appariement

On utilisera les probabilités suivantes : $\mu(1) = \frac{1}{7}$, $\mu(2) = \frac{5}{21}$, $\mu(3) = \frac{1}{7}$, $\mu(4) = \frac{1}{3}$, $\mu(5) = \frac{1}{7}$. La loi de probabilité μ vérifie *NCOND*. On obtient, après avoir simulé 100 000 arrivées, que FIFO vide la file d'attente (et donc construit un couplage parfait) 20114 fois, soit $20,114\%$ du temps et ML vide la file d'attente 19756 fois, soit $19,756\%$ du temps.

Ci-dessus, nous avons tracé la trajectoire de la file d'attente pour 100 arrivées pour les deux politiques de choix.

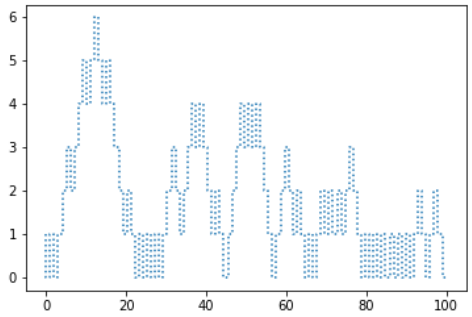


Figure 18: Nombre de sommets dans la file d'attente pour la politique FIFO

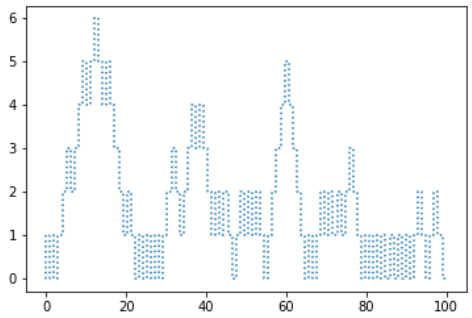


Figure 19: Nombre de sommets dans la file d'attente pour la politique ML

Enfin, nous allons réaliser une dernière fois ces calculs sur le graphe d'appariement suivant :

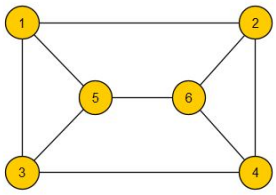


Figure 20: Graphe d'appariement

On utilisera les probabilités suivantes : $\mu(1) = \frac{1}{12}$, $\mu(2) = \frac{1}{12}$, $\mu(3) = \frac{1}{12}$, $\mu(4) = \frac{1}{12}$, $\mu(5) = \frac{1}{3}$ et $\mu(6) = \frac{1}{3}$. la loi μ ainsi définie vérifie *NCOND*.

On obtient, après avoir simulé 100 000 arrivées, que FIFO vide la file d'attente (et donc construit un couplage parfait) 8576 fois, soit $8,576\%$ du temps et ML vide la file d'attente

9104 fois, soit $9,104\%$ du temps.

Ci-dessus, nous avons tracé la trajectoire de la file d'attente pour 100 arrivées pour les deux politiques de choix.

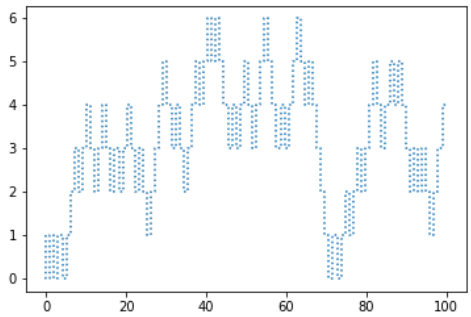


Figure 21: Nombre de sommets dans la file d'attente pour la politique FIFO

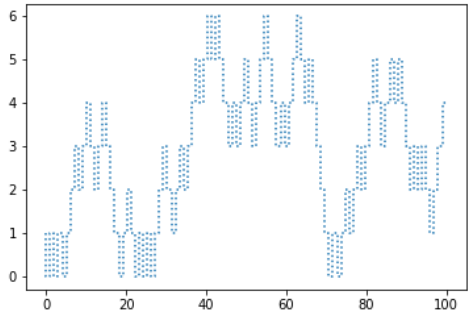


Figure 22: Nombre de sommets dans la file d'attente pour la politique ML

On ne peut donc pas conclure sur l'efficacité supérieure d'une de ces deux politiques par rapport à l'autre, le résultat dépendant du graphe d'appariement. On observe cependant que les trajectoires de la file d'attente pour les deux politiques sont proches et sont globalement autant efficace.

5 Conclusion

Nous avons pu constater que l'efficacité de notre algorithme en ligne est très dépendante des caractéristiques intrinsèques du graphe. Les politiques de choix FIFO et ML, qui sont toutes les deux maximales, amènent en pratique à des résultats très similaire. Il pourrait être intéressant d'étudier en détail pour quelle classe de graphe l'une ou l'autre est plus efficace,

en supposant qu'un tel résultat existe.

De plus, nous n'avons pas évoqué la difficulté de vérifier *NCOND* ou *NCOND** pour des graphes d'appariement ayant un très grand nombre de sommets. L'article [6] présente un résultat en ce sens en prouvant qu'il est possible de vérifier *NCOND* en temps polynomial, de l'ordre du nombre de sommets au cube.

Une ouverture intéressante serait d'établir un résultat analogue à la proposition 4.2 dans le cadre d'un graphe général, cela implique qu'il faudrait trouver un lien fort entre la condition de Tutte et la condition *NCOND*.

References

[1] J.A. Bondy and U.S.R. Murty. Graph theory with applications (vol. 290). *London : Macmillan*, 1976

[2] E.W. Weisstein, Blossom algorithm. *https://mathworld.wolfram.com/*, 2011

[3] W. T. Tutte. The Factorization of Linear Graphs. *J. London Math. Soc.* **22**: 107-111, 1947.

[4] R. Caldentey, E.H. Kaplan, and G. Weiss. FCFS infinite bipartite matching of servers and customers. *Adv. Appl. Probab* **41**(3): 695–730, 2009.

[5] A. Bušić, V. Gupta, and J. Mairesse. Stability of the bipartite matching model. *Adv. Appl. Probab.* **45**(2): 351–378, 2013.

[6] J. Mairesse and P. Moyal. Stability of the stochastic matching model. *Journal of Applied Probability* **53**(4): 1064–1077, 2016.

[7] J.R. Norris, Markov chains (No. 2), *Cambrige university press*, 1998

[8] P. Moyal and O. Perry. On the Instability of matching queues. *Annals of Applied Probability* **27**(6): 3385-3434, 2017.

[9] P. Moyal, A. Bušić and J. Mairesse (2018). A product form for the general stochastic matching model. *Journal of Applied Probability* **58**(2): 449-468, 2021.

[10] R. M. Karp, U. V. Vazirani and U. V. Vazirani. An optimal algorithm for on-line bipartite matching. *Proc. of the ACM symposium on Theory of computing*: 352-358, 1990.

6 Annexe

Nous présentons ici tout d'abord la classe Graphe que nous avons implémenté en Python 3.7.

```
class Graphe:

def __init__(self,dictionnaire):
    """le graphe sera initialisé par un objet de type dict"""
    if dictionnaire == None:
        """Si None est rentré comme parametre, on utilise un dictionnaire vide"""
        dictionnaire = {}
    self.graph=dictionnaire

def sommets(self):
    return list(self.graph.keys())

def __liste_arete(self):
    """methode privée qui nous donne la liste des aretes du graphe"""
    arete = []
    for sommet in self.graph:
        for voisin in self.graph[sommet]:
            if  {voisin,sommet} not in arete:
                arete.append({sommet,voisin})
    return arete

def aretes(self):
    return self.__liste_arete()

def add_sommet(self,sommet):
    if sommet not in self.graph:
        self.graph[sommet]=[]

def add_arete(self,arete):
    arete = set(arete)
    [sommet_1,sommet_2]=list(arete)
    if sommet_1 in self.graph and sommet_2 in self.graph:
        self.graph[sommet_1].append(sommet_2)
        self.graph[sommet_2].append(sommet_1)
    if sommet_1 in self.graph and sommet_2 not in self.graph:
        self.add_sommet(sommet_2)
        self.graph[sommet_1].append(sommet_2)
        self.graph[sommet_2].append(sommet_1)
```

```
        if sommet_1 not in self.graph and sommet_2 in self.graph:
            self.add_sommet(sommet_1)
            self.graph[sommet_1].append(sommet_2)
            self.graph[sommet_2].append(sommet_1)
        else:
            self.graph[sommet_1]=[sommet_2]
            self.graph[sommet_2]=[sommet_1]

    def affichage(self):
        res = "sommets: "
        for sommet in self.graph:
            res += str(sommet) + " "
        res += "\naretes: "
        for arete in self.__liste_arete():
            res += str(arete) + " "
        return res

"""test de la classe"""

graphe=Graphe({ "a" : ["d"],
          "b" : ["c"],
          "c" : ["b", "c", "d", "e"],
          "d" : ["a", "c"],
          "e" : ["c"]
        })

"""print(graphe.sommets())

print(graphe.aretes())
graphe.add_arete(("x","y"))
print(graphe.sommets())

print(graphe.aretes())

print(graphe.affichage())
graphe.add_arete(("a","b"))
print(graphe.affichage())

graphe.add_arete(("a","z"))
print(graphe.affichage())

print(graphe.aretes())"""
```

Nous présentons enfin l'algorithme de couplage avec les deux politiques de choix. Nous avons appliqué les deux politiques dans la même fonction afin de tester ces politiques sur un jeu de données identique.

```
    from Graphe import Graphe
from random import random
import matplotlib.pyplot as plt
import numpy as np
import copy


graphe=Graphe({ "1" : ["2"],
          "2" : ["1","3","4"],
          "3" : ["2", "4","5"],
          "4" : ["2", "3","6"],
          "5" : ["3","6"],
          "6" : ["4","5"]
        })

graphe2=Graphe({ "1" : ["2","3","4"],
          "2" : ["1","5","4"],
          "3" : ["1", "4"],
          "4" : ["2", "3","5","1"],
          "5" : ["4","2"]
        })

graphe3=Graphe({ "1" : ["2","3","5"],
                 "2" : ["1","4","6"],
                 "3" : ["1","4","5"],
                 "4" : ["2","3","6"],
                 "5" : ["1","3","6"],
                 "6" : ["2","4","5"]
      })

print(graphe.affichage())

"""probabilité associée à l'ensemble des sommets"""
mu=[1/8,3/8,5/8,7/8,29/32,1]
mu2=[1/7,8/21,11/21,18/21,1]
mu3=[1/12,2/12,3/12,4/12,8/12,1]
def tirage(loi):
    """le vecteur loi contient les probas des sommets a tirer.
    On retourne le label d'un sommet"""
```

```
    i = len(loi)
    u = random()
    for j in range(0,i):
        if u < loi[j]:
            return j+1
    return i

def couplage(graphe,loi,nbrSommet):
    """On prend en entrée un graphe d'appariement, un vecteur contenant les poids
    des sommets (loi de probabilité) et le nombre de sommets
    à tirer au hasard"""

    """On va effectuer sur les mêmes arrivées l'algo de couplage avec FIFO et l'algo
    de couplage avec ML, afin de pouvoir comparer les performances sur le meme
    jeu de donnée"""

    """on initialise le couplage à l'ensemble vide et la file d'attente par un
    premier sommet tiré au hasard pour FIFO"""
    sommet = tirage(loi)
    X_FIFO = [sommet]
    Couplage_FIFO = []
    file_FIFO = [len(X_FIFO)]
    nbrFileVide_FIFO = 0
    """On initialise pour la politique ML"""
    X_ML = len(loi)*[0]
    X_ML[sommet-1] = 1
    Couplage_ML = []
    file_ML = [sum(X_ML)]
    nbrFileVide_ML = 0
    i = 1
    while i < nbrSommet :
        sommet = tirage(loi)
        """on implémente ici FIFO en parcourant
        la file d'attente de la première à la dernière valeur"""
        var = False
        if len(X_FIFO) == 0:
            nbrFileVide_FIFO += 1
            X_FIFO.append(sommet)
        else:
            for j in range(len(X_FIFO)):
                if str(sommet) in graphe.graph[str(X_FIFO[j])]:
                    Couplage_FIFO.append({X_FIFO[j],sommet})
```

```
                del X_FIFO[j]
                var = True
                break
        if var == False:
            X_FIFO.append(sommet)
    file_FIFO.append(len(X_FIFO))
    """On implémente ici la politique ML"""
    if sum(X_ML) == 0:
        nbrFileVide_ML += 1
        X_ML[sommet-1] += 1
    else:
        var2 = copy.deepcopy(X_ML)
        while sum(var2) > 0:
            if str(sommet) in graphe.graph[str(np.argmax(var2)+1)]:
                Couplage_ML.append({np.argmax(var2)+1,sommet})
                X_ML[np.argmax(var2)] -= 1
                break
            else:
                var2[np.argmax(var2)] = 0
        if sum(var2) == 0:
            X_ML[sommet-1] += 1
    file_ML.append(sum(X_ML))
    i = i+1
return [X_FIFO,Couplage_FIFO,file_FIFO,nbrFileVide_FIFO,X_ML,
Couplage_ML,file_ML,nbrFileVide_ML]
```