Rapports d'approximation effectifs d'ordonnancements bicritères d'intervalles sur des machines identiques.

Fabien Baille, Evripidis Bampis, Christian Laforest †

LaMI, CNRS UMR 8042, Université d'Evry, Tour Evry 2, 523, places des terrasses 91000 Evry, France
mail : {fbaille,bampis,laforest}@lami.univ-evry.fr

Dans cet article, nous nous intéressons au problème d'ordonnancement d'intervalles (tâches particulières avec des dates d'arrivées et de terminaisons très contraintes) sur des machines identiques. Chaque intervalle ayant un poids associé, notre objectif est de construire des ordonnancements maximisant simultanément le poids ET le nombre de tâches ordonnancées. Ce problème est une modélisation naturelle du problème du contrôle par un fournisseur de l'accès à son réseau (découpé en sous canaux), sous contraintes de tarification.
La maximisation bicritère étant en général impossible, nous nous intéressons à l'*approximation bicritère*. Dans un travail récent ([BBL03]) nous avons proposé un algorithme FUSION permettant de construire un ordonnancement bicritère approché à partir de deux ordonnancements monocritères. Nous avons montré que les rapports d'approximation (études des pires cas) de notre méthode sont optimaux.
Cependant, l'analyse conduisant à la preuve de l'optimalité des rapports d'approximation de FUSION laisse entrevoir que dans le cas d'instances autres que des instances pathologiques, il est possible d'obtenir une meilleure connaissance de la qualité de l'ordonnancement produit. Cet article présente cela et montre donc que FUSION peut être aussi très bon "en pratique" (i.e. au cas par cas, et pas seulement pour les pires cas).

Keywords: QoS, Ordonnancement, Algorithme d'approximation

1 Introduction

Considérons un canal de communication divisé en plusieurs sous-canaux (par exemple une fibre optique ou une liaison satellite). Un fournisseur d'accès peut consacrer une partie (voire la totalité) de ses sous-canaux à des services qui nécessitent une connexion sans interruption dans un intervalle de temps relativement long, réservé à l'avance par le client (des services de type visio-conférence par exemple). Pour utiliser le médium de communication dans son intégralité, l'opérateur peut décider de faire passer des connexions plus courtes. Lorsque le client utilise le médium, il paye à l'opérateur un certain prix fixé par ce dernier (N.B. le prix peut être fonction de paramètres multiples est n'est pas nécessairement le même pour chaque client). Ainsi, certaines demandes de connexions rapportent plus au fournisseur d'accès que d'autres. Par contre, pour ce qui est de la fidélisation du client, le fournisseur aura tout intérêt à accepter le plus de requêtes possibles. Ainsi, il devra trouver un bon compromis entre bénéfice à courts termes (accepter en priorité les requêtes générant un fort profit) et bénéfices à long termes (fidéliser le client en l'acceptant même s'il rapporte peu).

Dans notre étude, on s'intéresse au cas où toutes les demandes de connexions sont connues à l'avance et où une demande de connexion doit être satisfaite à sa date d'arrivée ou être refusée (une connexion ne peut pas être retardée). De plus on supposera que les sous-canaux sont tous identiques. Donc, quelque soit le canal utilisé, le service rendu est le même. Pour modéliser ces données, nous nous ramenons à un problème d'ordonnancement d'intervalles sur plusieurs machines identiques. Détaillons cette modélisation.

† Travail partiellement financé par le projet français *AcTAM* du CNRS, le réseau thématique *APPOL II* de l'Union Européenne et le projet *MULT-APPROX* France-Berkeley

Nous associons à chaque canal de communication une *machine* et à chaque demande de connexion un *intervalle*. De manière plus formelle, un intervalle est un triplet (r_i,p_i,w_i) où $r_i \geq 0$ représente la *date d'arrivée de l'intervalle*, $p_i > 0$ sa *durée* et $w_i > 0$ son *poids*. Nous voulons donc placer des intervalles dans un système de k machines identiques.

On dira qu'un intervalle est *accepté* ou *ordonnancé* si une machine l'exécute sans interruption entre les dates r_i et $p_i + r_i$. Dans le cas contraire, l'intervalle sera dit *refusé*. Notons qu'ici, à tout instant, une machine n'exécute qu'un seul intervalle (la préemption n'est pas permise).

Étant donné un ensemble σ d'intervalles, un *ordonnancement valide* de σ est un ordonnancement de certains intervalles de σ respectant les contraintes d'exécution précédentes.

Nous nous intéressons à la maximisation *simultanée* de deux critères:

– **Le nombre :** maximisation du nombre d'intervalles acceptés.

– **Le poids :** maximisation de la somme des poids des intervalles acceptés.

Soit σ une instance composée de n intervalles et O un ordonnancement valide de σ (O ne contient donc pas nécessairement tous les intervalles de σ). Nous noterons $N(O)$ (resp. $W(O)$) le nombre d'intervalles (resp. le poids total des intervalles) acceptés par l'ordonnancement O. Nous noterons $N^*(\sigma)$ (resp. $W^*(\sigma)$) le nombre *maximal* d'intervalles (resp. le poids maximal d'intervalles) de l'instance σ ordonnançables dans le système. Dans la suite, nous appellerons *rapport d'approximation effectif* d'un ordonnancement O pour le nombre (resp. pour le poids) la valeur $\frac{N^*(\sigma)}{N(O)}$ $\left(\text{resp. } \frac{W^*(\sigma)}{W(O)}\right)$. Nous appellerons aussi *rapport d'approximation théorique* (pour le poids et pour le nombre) la valeur maximale de tous les rapports d'approximation effectifs associés à tous les ordonnancements valides O de toute instance σ. De manière plus formelle, un algorithme d'ordonnancement A est dit α-approché pour le nombre (resp. le poids), si pour toute instance σ, on a $\alpha N(A(\sigma)) \geq N^*(\sigma)$ (resp. $\alpha W(A(\sigma)) \geq W^*(\sigma)$).

On peut noter que le problème de maximisation du nombre est un sous-problème du poids. En effet, il suffit de poser $w_i = 1$ pour tous les intervalles pour retrouver le nombre optimal en maximisant le poids. Toutefois, maximiser un des critères peut conduire à un pauvre résultat pour l'autre. Pour s'en persuader, il suffit de prendre une instance où des intervalles avec un grand p_i et un grand w_i doivent s'exécuter en même temps que des intervalles de courte durée et avec de petits poids. Un ordonnancement maximisant le nombre aura tendance à contenir les intervalles de faible durée tandis qu'un ordonnancement maximisant le poids aura tendance à contenir les intervalles les plus longs au détriment des plus courts.

Dans le cadre monocritère, une solution optimale peut être trouvée en temps polynomial en utilisant les algorithmes de Carlisle et Lloyd [CL95] ou d'Arkin et Silverberg [AS87] pour le poids (et donc pour le nombre) et ceux de Carlisle et Lloyd [CL95] ou de Faigle et Nawijn [FN95] pour le nombre.

Dans le cadre de l'optimisation bicritère, il devient impossible de trouver une solution ayant *simultanément* une solution optimale pour les deux critères. Cependant, nous avons proposé dans [BBL03] un algorithme polynomial FUSION ayant les meilleurs couples de rapports d'approximation possibles. Toutefois, même si les résultats théoriques sur les rapports d'approximation que nous obtenons sont optimaux, nous montrons ici que pour de nombreuses instances, notre algorithme donne un ordonnancement bicritère dont les rapports d'approximation effectifs sont proches des rapports idéaux (le couple $(1,1)$) et bien meilleurs que les rapports d'approximation théoriques. Dans cet article, nous montrons ce phénomène et nous l'expliquons.

2 L'algorithme FUSION

Dans cette section, nous rappelons notre algorithme FUSION et une partie de nos résultats théoriques basés sur une étude au pire cas obtenus dans [BBL03]. L'idée majeure de l'algorithme FUSION est la construction d'un ordonnancement bicritère approché à partir de deux ordonnancements monocritères optimaux.

Soit σ un ensemble d'intervalles, soit r un entier tel que $1 \leq r < k$.

FUSION (σ, r)

- Construire $A_N(\sigma)$ un ordonnancement de σ maximisant le nombre grâce à l'algorithme de Faigle et Nawijn [FN95]

- Construire $A_W(\sigma)$ un ordonnancement de σ maximisant le poids grâce à l'algorithme de Carlisle et Lloyd [CL95]
- Choisir dans $A_N(\sigma)$ les r machines contenant le plus d'intervalles. Soit B_N l'ensemble des intervalles acceptés par ces r machines.
- Choisir dans $A_W(\sigma)$ les $k-r$ machines dont la somme des poids des intervalles acceptés est maximale. Soit B_W l'ensemble des intervalles contenus dans ces $k-r$ machines.
- Retourner l'ordonnancement qui place les intervalles de B_N sur les r premières machines de la même manière que dans $A_N(\sigma)$, et ceux de B_W/B_N sur les $k-r$ machines restantes de la même manière que dans $A_W(\sigma)$.

Théorème 1 ([BBL03]) *Pour le problème d'ordonnancement d'intervalles, l'algorithme* FUSION *a des rapports d'approximation de* $\frac{k}{r}$ *pour le nombre* **et** *de* $\frac{k}{k-r}$ *pour le poids.*

Le paramètre r permet à l'utilisateur de favoriser un des critères tout en garantissant tout de même un rapport d'approximation (donc une performance) pour le second.

Idées de preuve. Dans B_N, il y a au moins $\frac{N(A_N(\sigma))}{\frac{k}{r}}$ intervalles. En effet, B_N est l'ensemble des intervalles qui s'exécutent sur les r machines qui acceptent le plus d'intervalles. Comme A_N est optimal, on obtient un rapport d'approximation pour le nombre qui est de $\frac{k}{r}$.

De la même manière, dans B_W, la somme des poids des intervalles est au moins $\frac{W(A_W(\sigma))}{\frac{k}{k-r}}$. Or, lors de la dernière phase de l'algorithme, il se peut qu'une grande partie des intervalles de B_W n'appartienne pas à B_W/B_N. Cependant, ces intervalles appartiennent à B_N et leur poids peut alors être compté. Comme $A_W(\sigma)$ est optimal, on obtient un rapport d'approximation pour le nombre qui est de $\frac{k}{k-r}$.

Ces rapports d'approximation sont les meilleurs que l'on puisse espérer trouver, en effet :

Théorème 2 ([BBL03]) *Sur un système de k machines, on a :*
Pour tout entier r tel que $1 \leq r < k$, *et pour tout* $\varepsilon > 0$, *il existe un ensemble* σ *d'intervalles tels que pour tout ordonnancement O de* σ, *on a : si* $N(O) > \frac{N^*(\sigma)}{\frac{k}{r}}$, *alors* $W(O) < \frac{k-r}{k} W^*(\sigma)(1+\varepsilon)$.

Ce dernier résultat permet de définir des couples de rapports d'approximation qui sont inatteignables. Dans [BBL03], nous donnons les preuves complètes de ces deux théorèmes.

Ici, il est essentiel de noter que pour atteindre ces rapports d'approximation, nous n'analysons que le nombre (resp. le poids) des intervalles de B_N (resp. de B_W/B_N). L'impact sur le nombre des intervalles de B_W/B_N et l'impact sur le poids de ceux de B_N sont difficilement estimables de manière théorique même si le théorème 2 montre qu'ils sont négligeables dans les pires cas. Ce qui nous laisse envisager que dans les cas courants, l'algorithme donne des rapports d'approximation effectifs meilleurs que les rapports théoriques. C'est ce que nous montrons dans la prochaine section.

3 Étude de cas

Dans cette section, nous allons étudier un cas particulier d'instances, où clairement, une recherche de compromis est nécessaire. Ainsi, nous allons commencer par décrire les instances et ensuite, nous allons nous intéresser aux rapports effectifs de nos solutions dans le cas où $r = \frac{k}{2}$, nous allons les comparer aux rapports théoriques et enfin, nous allons expliquer en quoi cette valeur de r semble constituer le meilleur compromis.

3.1 Les instances

Dans nos instances, nous nous donnons trois paramètres : k, le nombre pair de machines du système, c, le facteur de proportionnalité du poids pour les intervalles dits *courts* et n, un entier.

Étant donnés ces paramètres, nous décrivons une instance σ comme suit :

- kn intervalles *courts* ont les paramètres suivants : $p_i = 1$, $w_i = \frac{n}{c}$ et des r_i tels que $\frac{k}{2}$ intervalles commencent en 0, $\frac{k}{2}$ commencent en 1 et ainsi de suite jusqu'au temps $2n-1$. Il est facile de voir que ces intervalles peuvent être ordonnancés totalement sur $\frac{k}{2}$ machines entre les dates 0 et $2n$.
- k intervalles *longs* avec des paramètres : $p_i = n^2$, $w_i = n^2$ et $r_i = 0$.
- k intervalles *modestes* avec des paramètres $p_i = 2n - \frac{1}{2}$, $w_i = 1$ (d'où leur nom), et avec des r_i tels que $\frac{k}{2}$ intervalles commencent en 0 et $\frac{k}{2}$ commencent en $2n - \frac{1}{2}$. Clairement, les intervalles modestes peuvent s'ordonnancer sur $\frac{k}{2}$ machines entre les dates 0 et $4n-1$.

La figure 1 nous donne un exemple d'instance avec $k = 6$, $n = 20$ et $c = 3$. Ici, on distingue 120 intervalles courts tous de poids $\frac{20}{3}$, 6 intervalles modestes de poids 1 et 6 intervalles longs de poids 400.

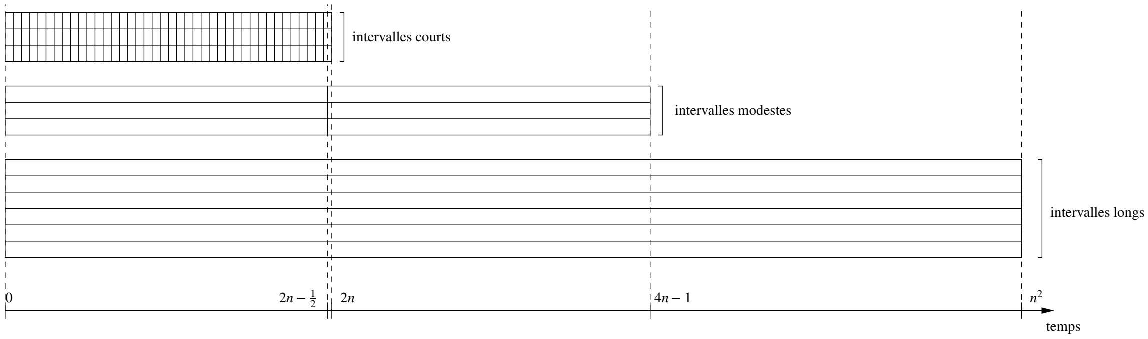

FIG. 1 – *Exemple d'instance avec $k = 6$, $n = 20$ et $c = 3$*

3.2 Optimiser un seul critère n'est pas satisfaisant

Le nombre. Le seul ordonnancement optimal pour le nombre O_N^* est de choisir tous les intervalles *courts* (sur les $\frac{k}{2}$ premières machines par exemple) et tous les intervalles *modestes* (sur les $\frac{k}{2}$ dernières machines par exemple). Il est aisé de voir que $N^*(\sigma) = N(O_N^*) = k(n+1)$ et $W(O_N^*) = k\left(\frac{n^2}{c} + 1\right)$. La figure 2 illustre l'ordonnancement O_N^* avec les valeurs $k = 6$, $n = 20$ et $c = 3$. On obtient alors $N(O_N^*) = 126$ et $W(O_N^*) = 806$.

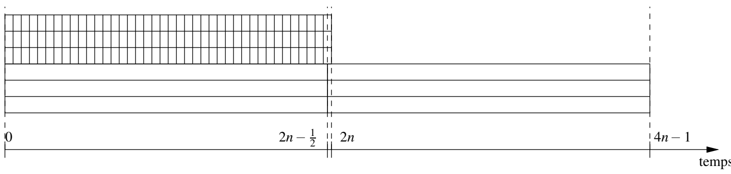

FIG. 2 – *O_N^* avec l'instance de la figure 1 ($k = 6$, $n = 20$ et $c = 3$)*

Le poids. Tout aussi clairement, le seul ordonnancement optimal pour le poids O_W^* consiste à placer tous les intervalles *longs*. Nous obtenons alors $W^*(\sigma) = W(O_W^*) = kn^2$ et $N(O_W^*) = k$. La figure 3 illustre O_W^* avec les valeurs $k = 6$, $n = 20$ et $c = 3$. On obtient alors $N(O_N^*) = 6$ et $W(O_N^*) = 2400$.

Rapports d'approximations effectifs pour O_W^* et pour O_N^*. Le rapport d'approximation effectif pour le poids de O_N^* est alors $\frac{cn^2}{n^2+c}$ qui tend vers c lorsque c est négligeable devant n^2.

Le rapport d'approximation effectif pour le nombre de O_W^* est $n+1$.

Lorsqu'on veut maximiser le nombre pour ce type d'instance, on s'écarte donc d'un facteur c de la solution optimale pour le poids, et si on veut maximiser le poids, on s'écarte d'un facteur $n+1$ de la

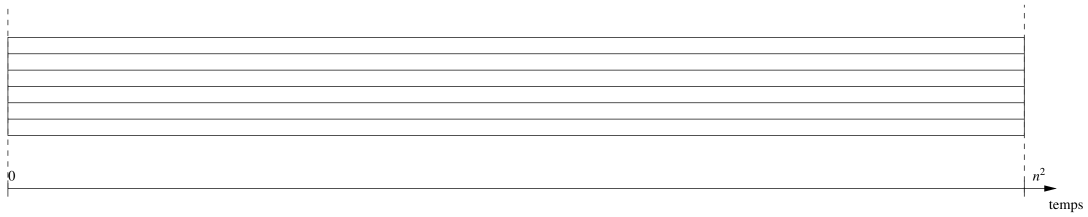

FIG. 3 – *O^*_W avec l'instance de la figure 1 ($k = 6$, $n = 20$ et $c = 3$)*

solution optimale pour le nombre. Il devient clair que dans le cadre de l'optimisation bicritère, ces solutions triviales ne sont pas satisfaisantes.

3.3 Le compromis apporté par FUSION

Appliquons FUSION à l'instance décrite ci-dessus avec $r = \frac{k}{2}$. Soit O l'ordonnancement produit. De manière évidente, O est constitué de $\frac{k}{2}$ intervalles longs et des nk intervalles courts. Ainsi, $N(O) = \frac{k}{2} + nk$ et $W(O) = \frac{kn^2}{2} + \frac{kn^2}{c}$. Ce qui nous donne un rapport effectif de $\frac{k(n+1)}{kn+\frac{k}{2}} = \frac{2n+2}{2n+1}$ pour le nombre, qui tend vers 1 quand n est élevé. Le rapport effectif pour le poids est de $\frac{kn^2}{k\frac{n^2}{c}+\frac{k}{2}n^2} = \frac{2c}{2+c}$, proche de 1 pour certaines valeurs de c. Ainsi, avec $2 < c \ll n$, nous obtenons un couple de rapports d'approximation qui est proche de $(1{,}1)$. On note que pour $r = \frac{k}{2}$, l'approche théorique de l'algorithme FUSION nous avait conduit à un couple de rapports d'approximation de $(2{,}2)$. Nous avons donc un couple de rapports effectifs meilleur pour ce type d'instance que le couple de rapports théoriques. Ainsi, toujours avec l'exemple de la figure 1, FUSION retourne l'ordonnancement de la figure 4. On observe alors que $N(O) = 123$ et $W(O) = 2000$.

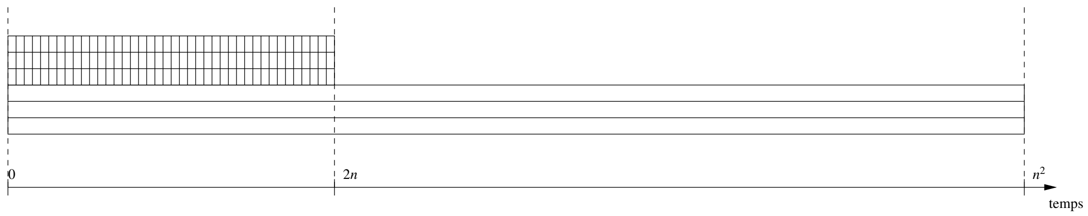

FIG. 4 – *Résultat de FUSION avec l'instance de la figure 1*

Nous pouvons, à ce stade de l'article, nous demander d'une part si les rapports effectifs sont toujours meilleurs que les rapports théoriques et d'autre part si la valeur de r que nous avons choisie représente le meilleur compromis.

3.4 Un cas particulier

Pour tenter de répondre à ces questions, nous avons expérimenté notre algorithme sur plusieurs instances et nous avons remarqué qu'effectivement, les couples de rapports effectifs sont meilleurs que les couples de rapports d'approximation théoriques. Mieux encore, on peut trouver des instances pour lesquelles FUSION donne un résultat proche du couple de rapports d'approximation idéal (parce qu'inatteignable dans ce cas) $(1{,}1)$. De plus, la valeur $r = \frac{k}{2}$ donne le compromis le plus satisfaisant.

Nous présentons ici les résultats de l'instance avec $n = 1000$, $c = 7$, et $k = 20$ qui, bien qu'étant un cas particulier illustre le comportement général de ce type d'instance. On note que pour ces valeurs, le rapport effectif pour le nombre de la solution optimale pour le poids est d'environ 1000 et le rapport d'approximation effectif pour le poids de la solution optimale pour le nombre est environ 7. Ce qui est inacceptable pour un opérateur. En effet, il perd 7 fois ce qu'il aurait pu gagner s'il avait choisi ses clients autrement et il refuse 1000 fois plus de clients qu'il en aurait accepté s'il avait ordonnancé les demandes de connexions autrement.

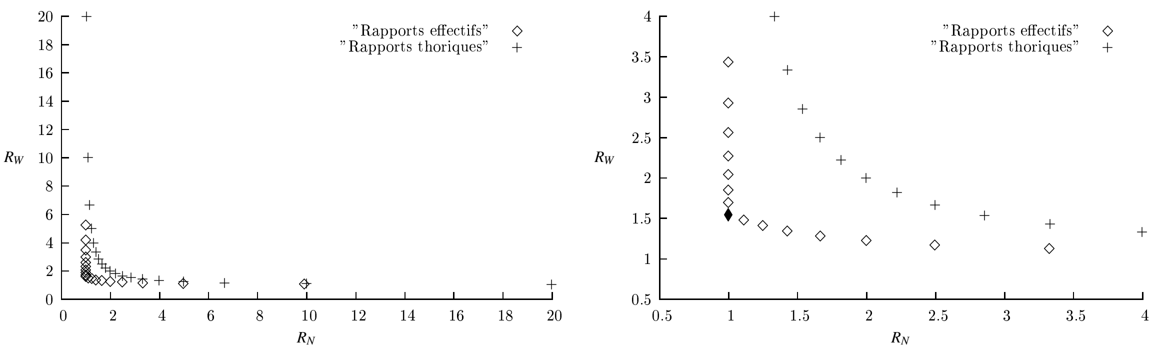

FIG. 5 – *Rapports théoriques et effectifs pour toutes les valeurs de r ($n = 1000$, $k = 20$, $c = 7$)*

Notre algorithme donne un rapport d'approximation de 1,5 environ pour le poids et de 1 environ pour le nombre, avec $r = \frac{k}{2}$. La partie gauche de la figure 5 montre la position des rapports d'approximation théoriques par rapport aux rapports effectifs. Celle de droite (qui est un zoom de la figure de gauche) montre plus en détail le fait que pour $r = 10$ (le point noir sur la figure), le couple de rapports d'approximation effectifs est le plus satisfaisant.

Pour information, nous donnons dans le tableau 1 les valeurs numériques exactes correspondantes.

r	Rapports effectifs	Rapports théoriques
1	(9.915800,1.037137)	(20.000000,1.052632)
2	(4.982578,1.077138)	(10.00000,1.111111)
3	(3.327239,1.120349)	(6.666667,1.176471)
4	(2.497505,1.167172)	(5.000000,1.250000)
5	(1.999001,1.218079)	(4.000000,1.333333)
6	(1.666389,1.273629)	(3.333333,1.428571)
7	(1.428673,1.334489)	(2.857143,1.538462)
8	(1.250312,1.401456)	(2.500000,1.666667)
9	(1.111543,1.475500)	(2.222222,1.818182)
10	(1.000500,1.557803)	(2.000000,2.000000)
11	(1.000450,1.689370)	(1.818182,2.222222)
12	(1.000400,1.845210)	(1.666667,2.500000)
13	(1.000350,2.032723)	(1.538462,2.857143)
14	(1.000300,2.262658)	(1.428571,3.333333)
15	(1.000250,2.551247)	(1.333333,4.000000)
16	(1.000200,2.924213)	(1.250000,5.000000)
17	(1.000150,3.424897)	(1.176471,6.666667)
18	(1.000100,4.132459)	(1.111111,10.000000)
19	(1.000050,5.208500)	(1.052632,20.000000)

TAB. 1 – *Valeurs des couples de rapports théoriques et des couples de rapports effectifs*

4 Conclusion - Perspectives

Dans cet article nous avons décrit des instances pour lesquelles les rapports d'approximation effectifs de FUSION étaient éloignés des rapports théoriques. Cependant, les rapports théoriques ne peuvent pas être améliorés en vue de ces résultats puisque pour les instances "pathologiques" servant à montrer le théorème 2, ils sont atteints. Cet article tend donc à montrer que FUSION donne *aussi* de bons résultats en pratique.

Pour pouvoir confirmer cela de manière plus large nous sommes à la recherche de modèles probabilistes permettant de générer des instances "réalistes" du point de vue réseau.

Références

[AS87] E. Arkin and B. Silverberg. Scheduling jobs with fixed start and end times. *Discrete Applied Mathematics*, 18:1–8, 1987.

[BBL03] F. Baille, E. Bampis, and Ch. Laforest. A note on optimal bicriteria approximation schedules for simultaneous maximization of weight and profit, en cours de soumission. 2003.

[CL95] Martin C. Carlisle and Errol L. Lloyd. On the k-coloring of intervals. *Discrete Applied Mathemetics*, 59:225–235, 1995.

[FN95] U. Faigle and M. Nawijn. Note on scheduling intervals on-line. *Discrete Applied Mathematics*, 58:13–17, 1995.