RAIRO. RECHERCHE OPÉRATIONNELLE

ALIX MUNIER

Résolution d'un problème d'ordonnancement cyclique à itérations indépendantes et contraintes de ressources

RAIRO. Recherche opérationnelle, tome 25, n° 2 (1991), p. 161-182

<http://www.numdam.org/item?id=RO_1991__25_2_161_0>

Recherche opérationnelle/Operations Research
(vol. 25, n° 2, 1991, p. 161 à 182)

RÉSOLUTION D'UN PROBLÈME D'ORDONNANCEMENT CYCLIQUE A ITÉRATIONS INDÉPENDANTES ET CONTRAINTES DE RESSOURCES (*)

par Alix MUNIER ([1])

Résumé. – *Un problème d'ordonnancement cyclique est un problème d'ordonnancement où un nombre fini de tâches génériques liées par des contraintes de précédence ou de ressources doivent être exécutées une infinité de fois. Malgré de nombreuses applications, ce type de problème a été peu étudié jusqu'ici en tant que problème d'optimisation combinatoire.*

Dans cet article, on dispose d'un ensemble fini de tâches génériques à exécuter une infinité de fois sur des processeurs spécialisés. On suppose d'autre part l'existence de contraintes de précédence entre les tâches d'une même itération. La fonction objectif est la maximisation de la fréquence d'exécution des tâches.

On propose pour ce problème une caractérisation des solutions réalisables et un algorithme polynomial basé sur un parcours topologique du graphe de précédence pour la recherche d'une solution optimale.

Mots clés : Ordonnancement cyclique; fréquence d'exécution des tâches; contraintes de précédence; machines spécialisées.

Abstract. – *A cyclic scheduling problem is defined by a finite set of generic tasks with resource or precedence constraints to be executed an infinity of times. Although these problems have important applications, few were up to now studied as combinatorial optimization problems.*

In this paper, we consider a set of dependent generic tasks and dedicated classes of processors. The objective is the maximization of the frequency of the task occurrences.

We propose a characterization of the feasible solutions and an efficient polynomial algorithm, based on a topological sort on the precedence graph, that provides an optimal schedule of the tasks.

Keywords : Cyclic Scheduling; frequency of the tasks occurrences; precedence constraints; dedicated processors.

(*) Reçu octobre 1989.
([1]) Laboratoire MASI, Université Pierre-et-Marie-Curie, 4, place Jussieu, 75252 Paris Cedex 05.

Recherche opérationnelle/Operations Research, 0399-0559/91/02 161 22/$ 4.20

INTRODUCTION

Un problème d'ordonnancement cyclique est un problème d'ordonnancement où un nombre fini de tâches génériques liées par des contraintes de précédence ou de ressources doivent être exécutées une infinité de fois. Ce type de problème admet de nombreuses applications industrielles ou informatiques mais n'a été étudié que récemment en tant que problème d'optimisation combinatoire.

La fonction objectif généralement utilisée est la maximisation de la fréquence d'exécution des tâches. Les résultats obtenus jusqu'ici ont surtout concerné la version cyclique du problème central de l'ordonnancement (PERT) [1, 2] et ont permis de prendre en compte des contraintes de ressources particulières dans le cas d'ateliers flexibles [5].

Cependant, les problèmes avec contraintes de ressources ont surtout été étudiés en imposant la répétition d'un même ordonnancement des tâches génériques. Ainsi, dans le cadre des architectures pipelines, de nombreux travaux ont été consacrés aux tables de réservations [6, 4]. D'autre part, pour les ateliers flexibles, plusieurs fonctions objectifs ont été retenues suivant le modèle d'atelier à machines dédicacées considéré; pour des problèmes de job-shop cycliques particuliers, Graves *et al.* [3] propose une heuristique pour minimiser la durée du job pour une période de valeur donnée et Roundy [8] considère un compromis entre la minimisation de la durée d'un job et de la période. Matsuo a étudié le problème de flow-shop cyclique à deux machines avec comme fonction objectif la minimisation du temps total des attentes entre les passages d'une machine à une autre [7].

Dans cet article, il faut exécuter une infinité de fois un ensemble fini de tâches génériques liées par des contraintes de précédence. D'autre part, les machines sont classées en types disjoints et toutes les occurrences d'une même tâche ne peuvent être effectuées que par les machines d'un type donné. Le critère de l'ordonnancement est la maximisation de la fréquence d'exécution de l'ensemble des tâches génériques.

Dans le premier paragraphe, on propose une description des données et du critère d'optimisation considéré. On démontre alors qu'il est licite de limiter notre étude aux ordonnancements cycliques dont les dates d'exécutions des tâches sont des suites K-périodiques. Le paragraphe suivant développe un ordonnancement optimal dans le cas particulier où toutes les machines sont du même type. On démontre alors dans le troisième paragraphe que cette solution peut être étendue au cas où les machines sont de type différents lorsque les tâches ne sont plus soumises à des contraintes de précédence.

Dans le dernier paragraphe, on démontre que les propriétés de périodicité de l'ordonnancement proposé permettent de relâcher ces contraintes. Elles n'interviennent plus alors que pendant une phase de démarrage dont la durée est évaluée.

1. SPÉCIFICATION DU PROBLÈME D'ORDONNANCEMENT

(*a*) *Définition des tâches*

On dispose d'un ensemble fini de n tâches génériques $T_1, T_2, \ldots, T_n$ de durées strictement positives $p_1, p_2, \ldots, p_n$.

A chaque tâche générique T_i, on associe l'ensemble des exécutions successives de T_i notées $T(i, 1), T(i, 2) \ldots T(i, p), \ldots$, où $T(i, p)$ correspond à la p-ième exécution de la tâche T_i.

On note t_i^p la date de début d'exécution de la tâche $T(i, p)_{i \in I,\ p>0}$.

L'itération p est le sous-ensemble de tâches $\{T(i, p), i \in I\}$ et sa durée est de $\max_I (t_i^p + p_i) - \min_I t^{p_i}$.

On suppose que deux exécutions d'une même tâche générique T_i ne peuvent être exécutées en même temps sur deux machines différentes. Par conséquent, comme les durées p_i sont strictement positives, la suite (t_i^p) est strictement croissante.

(*b*) *Contraintes de succession*

Les tâches génériques sont liées par un graphe de précédence $G = (I, U)$ sans circuit qui définit un ordre partiel sur ces tâches et qui doit être vérifié pour toutes les tâches d'une même occurrence.

$$(i.e. : (i, j) \in U \Leftrightarrow \forall p > 0\ t_i^p + p_i \leqq t_j^p)$$

(*c*) *Contraintes de ressources :*

On dispose de m types de machines indexés sur $J = \{1 \ldots m\}$. Pour tout j élément de J, on connaît m_j le nombre de machines de type j numérotées de 1 à m_j et la fonction AFF de I dans J, qui à toute tâche génériques T_i associe AFF (i) le type de machine sur lequel toutes les exécutions successives de la tâche T_i doivent être effectuées.

(*d*) *Critère d'optimisation et mode d'exécution des tâches*

Le critère d'optimisation retenu est la maximisation de la fréquence minimale d'exécution des tâches; si, pour tout $N>0$, on note D_N la durée de l'ordonnancement des N premières exécutions de toutes les tâches génériques, on maximise, sous réserve d'existence de la limite, la valeur :

$$\lim_{N \to \infty} \frac{N}{D_N}.$$

On considère qu'une allocation des machines qui satisfait les contraintes du problème et qui maximise la fréquence minimale d'exécution des tâches est optimale.

Par la suite, en utilisant des propriétés topologiques sur l'ensemble des solutions d'une instance de notre problème d'ordonnancement, on montre que l'on peut se restreindre à une sous-classe de solutions caractérisée à l'aide de la K-périodicité dont on rappelle la définition :

K-périodicité

Étant donné un entier strictement positif K, u_n est une suite K-périodique :

- si elle est croissante au sens large;
- s'il existe un entier N_0 et un réel positif w tel que :

$$\forall n \geqq N_0 : \quad u_{n+K} = u_n + w.$$

K est appelé le facteur de périodicité : c'est à partir du rang N_0 le nombre d'éléments de la suite sur un intervalle d'amplitude w;

w est la période de la suite;

K/w est la fréquence de la suite.

On note alors E l'ensemble des ordonnancements cycliques réalisables associés au problème posé et qui admettent une fréquence d'exécution finie, $\mathscr{P}(E)$ l'ensemble des parties de E et $\mathscr{F}$ la fonction définie de E dans R^{+*} qui à tout ordonnancement o appartenant à E associe $\mathscr{F}(o)$ la fréquence minimale d'exécution des tâches de l'ordonnancement o. On considère le sous-ensemble $\mathscr{T}$ inclus dans $\mathscr{P}(E)$ défini par :

$$\mathscr{T} = \{ \mathscr{F}^{-1}(\theta),\ \theta \text{ ouvert de } R^{+*} \}.$$

$(E, \mathcal{T})$ constitue un espace topologique sur E dont les ouverts sont les éléments de $\mathcal{T}$. On démontre alors la proposition suivante :

PROPOSITION 1 : *Le sous-ensemble de E noté A constitué des ordonnancements cycliques dont les dates d'exécutions des tâches t_i^p sont des suites K-périodiques constitue une partie dense de E.*

Preuve : Soit o, un élément quelconque de E. Il suffit de démontrer que l'intersection de tout voisinage de o noté $\mathcal{V}(o)$ avec A n'est pas vide.

$\mathcal{V}(o)$ est un voisinage de o. Il existe donc un ouvert de $\mathcal{T}$ contenant o contenu dans $\mathcal{V}(o)$. Donc, il existe h élément de R^{+*} tel que $F^{-1}(]\mathcal{F}(o)-h, \mathcal{F}(o)+h[)$ soit inclus dans $\mathcal{V}(o)$.

D'autre part, par définition de la fréquence d'un ordonnancement :

$$\lim_{N \to \infty} \frac{N}{D_N} = \mathcal{F}(o) \quad \Leftrightarrow \quad \forall \varepsilon > 0, \quad \exists M / \forall N \geqq M, \left| \frac{N}{D_N} - \mathcal{F}(o) \right| < \varepsilon.$$

Si $\varepsilon = h - \eta$, on peut alors déterminer M fonction de $h-\eta$ tel que (*fig.* 1) :

$$\forall N \geqq M, \quad \mathcal{F}(o) - h + \eta < \frac{N}{D_N} < \mathcal{F}(o) + h - \eta.$$

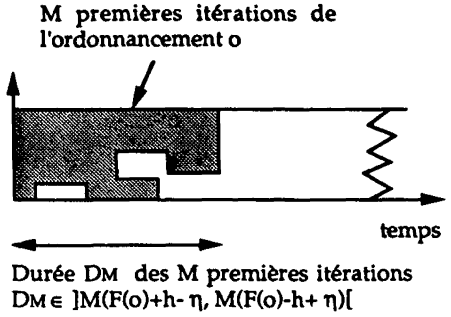

Figure 1

Si l'on considère alors l'ordonnancement o' qui consiste à répéter toutes les D_M unités de temps la restriction de l'ordonnancement o à ces M premières itérations (*fig.* 2) o' est un ordonnancement réalisable K-périodique de fréquence M/D_M appartenant au voisinage de o, $\mathcal{V}(o)$. La proposition est donc démontrée.

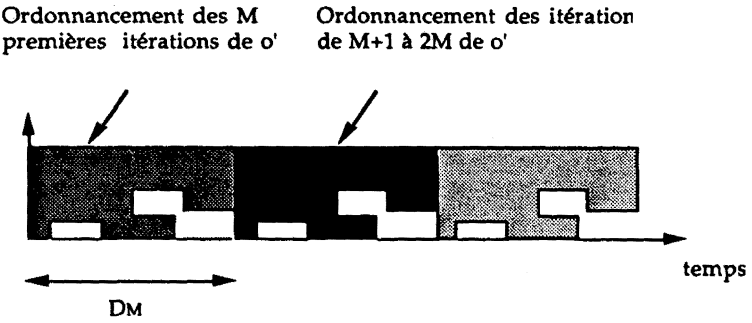

Figure 2

La proposition précédente permet donc de se limiter à l'étude des ordonnancements cycliques dont les dates d'exécution des tâches sont des suites K-périodiques.

De plus, si on permet à deux de ces suites t_i^N et t_j^N, correspondant aux exécutions des tâches T_i et T_j, d'avoir des fréquences différentes, la différence $(t_i^N - t_j^N)$ n'est pas bornée et donc, la durée d'une itération tend vers l'infini. L'ordonnancement n'est alors pas exploitable.

Par la suite, les solutions réalisables d'un problème cyclique seront les ordonnancements K-périodiques dont toutes les tâches ont la même fréquence. Ces ordonnancements seront alors appelés K-périodiques stables.

2. ÉTUDE D'UN ORDONNANCEMENT SUR UN TYPE DE MACHINE

Dans ce chapitre, on propose un ordonnancement K-périodique optimal dans le cas où l'on dispose de r machines identiques.

(*a*) *Principe de la solution proposée*

Soit $S=(T_1 \ldots T_n)$ une liste topologique des tâches génériques pour l'ordre partiel (I, U). On note alors, pour tout q strictement positif, $S(q)=(T(1, q) \ldots T(i, q) \ldots T(n, q))$ la séquence correspondante des tâches de l'itération q.

Le principe de la solution est d'une part l'exécution séquentielle et sans attente des tâches d'une même itération sur une même machine, d'autre part l'allocation des machines aux itérations par indice croissant. On montre par la suite que malgré le fait que la liste topologique S soit fixée, l'ordonnancement obtenu est optimal.

Deux exécutions d'une même tâche générique ne peuvent être simultanées. Donc, si l'on exécute la première séquence $S(1)$ sur la première machine au temps $t=0$, la séquence $S(2)$ (itération 2) pourra commencer au plus tôt à la date $\max p_i$. Il en est de même pour les séquences $S(3) \ldots S(r)$ qui commenceront au plus tôt aux dates $2 \max p_i, \ldots, (r-1) \max p_i$.

Pour exécuter l'itération $S(r+1)$, il faut alors considérer deux cas :

Cas 1 :

$$\sum_{i=1}^{n} p_i \leqq r \times \operatorname*{Max}_{i \in \{1 \ldots n\}} p_i$$

alors, au temps $t = r \times \operatorname{Max} p_i$, la première séquence est achevée.

On peut exécuter $S(r+1), \ldots, S(2r)$ après un décalage de $r \times \operatorname{Max} p_i$ par rapport aux séquences précédentes sur les mêmes machines, et réitérer pour les séquences suivantes (*fig.* 3).

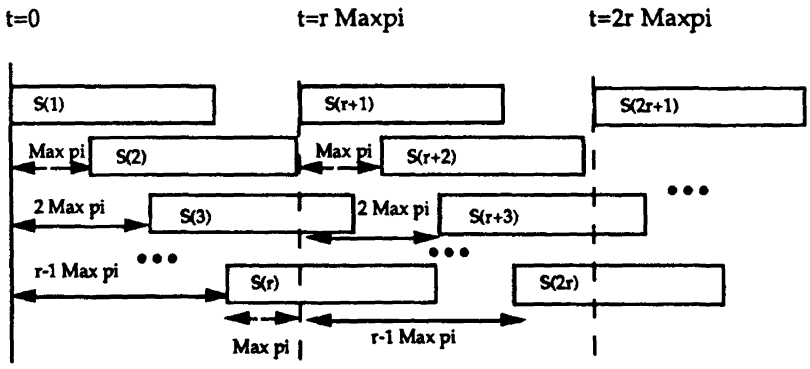

Figure 3

Cas 2 :

$$\sum_{i=1}^{n} p_i > r \times \operatorname*{Max}_{i \in \{1 \ldots n\}} p_i$$

pour tout $l = 1, \ldots, r$ on exécute alors $S(r+l)$ immédiatement après $S(l)$ sur la machine l et l'on réitère avec les séquences suivantes (*fig.* 4).

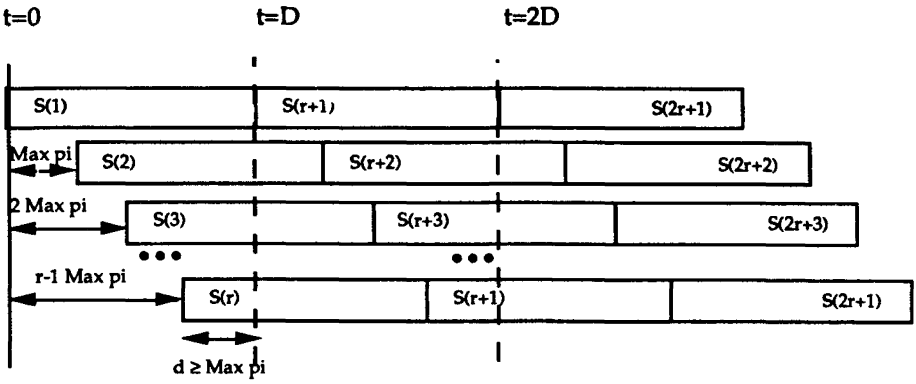

Figure 4

(*b*) *Dates de l'ordonnancement proposé*

Nous pouvons regrouper en un seul formalisme les deux cas précédents.

En effet, si l'on pose :

$D = \sum_{i=1}^{n} p_i$, la somme des durées des tâches génériques;

$P = \operatorname{Max}_{i \in I} p_i$ la durée de la tâche générique la plus longue,

et $M = \operatorname{Max}(P, D/r)$

alors, l'expression de la date t^N de lancement de la séquence $S(N)$, pour $N > 0$, est donnée par $t^N = kr\,M + q\,P$ où k et q sont respectivement le quotient et le reste de la division euclidienne de $N-1$ par r.

Donc, pour $i \in I$, la N-ième exécution de la tâche T_i commence à l'instant :

$$t_i^N = \sum_{s=1}^{i-1} p_s + kr\,M + q\,P \tag{1}$$

et est exécutée sur la machine $(q+1)$. On a donc d'une manière générale :

$$t_i^{N+r} = t_i^N + r\,M. \tag{2}$$

(*c*) *Respect des contraintes*

Montrons que l'ordonnancement proposé précédemment satisfait les contraintes sur les tâches et les ressources et qu'il constitue une solution K-périodique optimale du problème à r machines identiques.

(α) *Les contraintes sur les tâches*

Les tâches d'une même occurrence vérifient, par construction des séquences, l'ordre partiel induit par le graphe de précédence.

D'autre part, si l'on considère deux séquences quelconques $S(p)$ et $S(q)$ avec $q>p$, alors $S(q)$ est lancée après $S(p)$ et la différence entre leurs dates de lancement est d'au moins P. Donc, deux occurrences d'une même tâche ne sont pas exécutées en même temps sur deux machines différentes.

(β) *Les contraintes de ressources*

Si l'on considère les deux séquences successives sur la même machine $S(p)$ et $S(p+r)$, alors, l'intervalle de temps alloué à $S(p)$ sur cette machine est de $M\,r=\text{Max}\,(r\,P,\,D)\geqq D$.

Donc, $S(p)$ a le temps de s'exécuter entièrement sur la machine considérée.

(γ) *K-périodicité, stabilité et optimalité de l'ordonnancement*

L'ordonnancement proposé est K-périodique stable d'après l'expression des dates de l'ordonnancement (2) données au paragraphe précédent.

De plus :

- Si $(D/r)\geqq P$,

alors, le facteur de périodicité, la période et la fréquence ont respectivement pour valeur r, D et r/D.

D'autre part, les machines sont dans ce cas utilisées au maximum de leur capacité. Donc, l'ordonnancement est optimal de fréquence r/D.

- Sinon, les valeurs du facteur de périodicité, de la période et de la fréquence sont alors de 1, P et $1/P$. Or, comme deux exécutions successives de la plus longue tâche sont disjointes dans le temps par hypothèse, les dates d'exécution forment une suite de fréquence inférieure ou égale à $1/P$. L'ordonnancement est donc de fréquence optimale.

En conclusion, l'ordonnancement proposé est K-périodique de fréquence optimale

$$\varphi=\frac{1}{M}.$$

Nous avons donc montré que dans le cas où toutes les machines sont du même type, le problème d'ordonnancement cyclique possédait une solution optimale K-périodique. Nous généralisons maintenant cette approche au cas où les machines sont de types différents et les tâches génériques indépendantes.

3. PLUSIEURS TYPES DE MACHINES, TÂCHES INDÉPENDANTES

Dans cette partie, on suppose que les machines sont regroupées en types distincts et que les tâches génériques sont indépendantes. On démontre alors qu'il est possible de généraliser la solution proposée précédemment et d'en déduire un ordonnancement K-périodique stable.

Pour cela, on détermine une borne supérieure de la fréquence optimale en appliquant à chaque type de machine la méthode décrite au chapitre précédent, puis on propose un ordonnancement optimal de même fréquence pour tous les types de machines.

(*a*) *Borne supérieure de la fréquence*

Pour tout indice j de J, on note :

$D_j = \sum_{\text{AFF}\ (i)=j} p_i$, la somme des durées des tâches affectées à une machine de type j

et $P_j = \operatorname{Max}_{\text{AFF}\ (i)=j} p_i$, la durée maximale des tâches affectées à une machine de type j.

Pour obtenir une borne supérieure de la fréquence optimale, nous proposons un ordonnancement K-périodique du problème obtenu en relâchant la contrainte d'égalité des fréquences des tâches génériques.

Cet ordonnancement est construit comme suit : pour chaque type de machine j de J, nous déterminons une séquence S_j composée de toutes les tâches génériques qui doivent être exécutées sur une machine de type j. Nous utilisons alors la méthode précédente pour ordonnancer les séquences S_j (1), S_j (2), . . . , S_j (n).

Les tâches génériques étant indépendantes, l'ordonnancement ainsi construit est réalisable. D'autre part, si on note pour toute tâche générique T_i, $j = \text{AFF}\ (i)$ alors la fréquence d'exécution de T_i a pour valeur :

$$\varphi_j = \frac{1}{M_j} \qquad \text{avec} \quad M_j = \operatorname{Max}\left(P_j, \frac{D_j}{m_j}\right).$$

Considérons maintenant le problème de départ. Une solution de ce problème est aussi une solution du problème relâché et toutes les tâches génériques y sont exécutées avec une même fréquence φ. Il en résulte que $\forall j$, $\varphi \leqq \varphi j$, et donc que $\min_{j \in J} \varphi_j$ est une borne supérieure de la fréquence.

Nous montrons maintenant que, moyennant des décalages sur les séquences des tâches, on peut construire un ordonnancement dont toutes les tâches ont pour fréquence $\varphi = \min_{j \in J} \varphi_j$.

(*b*) *Solution proposée*

Posons $M = 1/\varphi$.

Soit j un type de machine et O_j l'ordonnancement des tâches allouées aux machines de type j suivant la méthode du paragraphe précédent. Nous transformons O_j en O'_j de telle sorte que la fréquence d'exécution des tâches dans O'_j soit égale à φ. O'_j est obtenu à partir de O_j en effectuant un décalage supplémentaire de $m_j (M - M_j)$ unités de temps pout toutes les séquences sauf les premières de chaque machine (*fig.* 5).

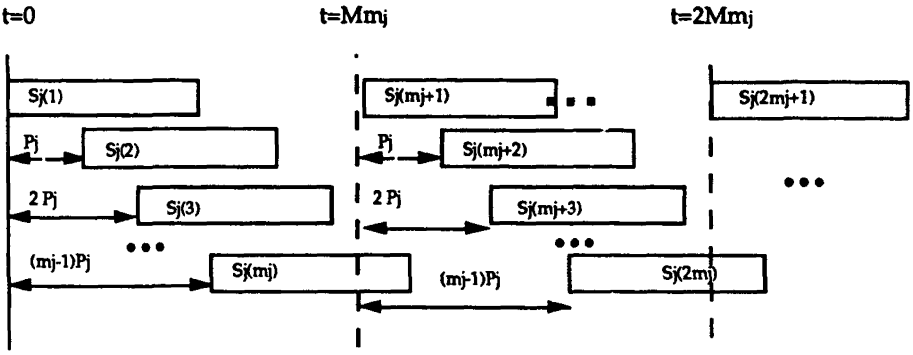

Figure 5

L'expression de la date t^N de lancement de la séquence $S_j(N)$, pour $N > 0$, est alors donnée par $t^N(j) = k_j m_j M + q_j P_j$ où k_j et q_j sont respectivement le quotient et le reste de la division euclidienne de $N-1$ par m_j. Soit

$$t_i^N = \left[\frac{N-1}{m_j} \right] m_j (M - P_j) + P_j (N-1).$$

(*c*) *Optimalité et K-périodicité de l'ordonnancement proposé*

Pour chaque type de machine j de J, les dates de lancement des séquences constituent une suite K-périodique de période et de facteur de périodicité donnés par :

$$K_j = 1 \quad \text{et} \quad w_j = M \quad \text{si } M = P_j$$
$$K_j = m_j \quad \text{et} \quad w_j = m_j M \quad \text{sinon.}$$

Par conséquent, la suite D_N des dates de fin d'itération est K-périodique de facteur de périodicité $K = \text{PPCM}\ (m_j)$, de période $w = KM$ et de fréquence optimale φ et l'ordonnancement proposé est K-périodique stable.

(*d*) *Exemple*

On considère six tâches génériques indépendantes $T_1 \ldots T_6$ et trois types de machines indicés de 1 à 3. Les durées des tâches p_i, la fonction d'affectation des tâches aux machines et le nombre de machines par type sont donnés par les figures 6 et 7.

Ti	T1	T2	T3	T4	T5	T6
pi	3	1	5	6	3	2
AFF(i)	1	2	1	2	3	2

Figure 6

j	1	2	3
mj	2	2	1

Figure 7

On définit, pour tout type j de machines, une séquence de tâches S_j, et l'on calcule les paramètres de l'ordonnancement D_j/m_j et P_j (*fig.* 8).

j	Sj	Dj/mj	Pj
1	T1T3	2	5
2	T2T4T6	2	6
3	T5	1	3

Figure 8

La fréquence de l'ordonnancement φ est de 1/6.

L'ordonnancement proposé est alors K-périodique stable de facteur de périodicité $K = \text{PPCM}\ (m_j) = 2$ et de période $w = 12$.

Pour tout j, si on note k_j et r_j respectivement le quotient et le reste de la division euclicienne de $N-1$ par m_j, le planning des séquences est donné par la figure 9. L'ordonnancement est représenté en figure 10.

j	1	2	3
tN(j)	12*kj + 5*rj	12*kj + 6*rj	6*kj

Figure 9

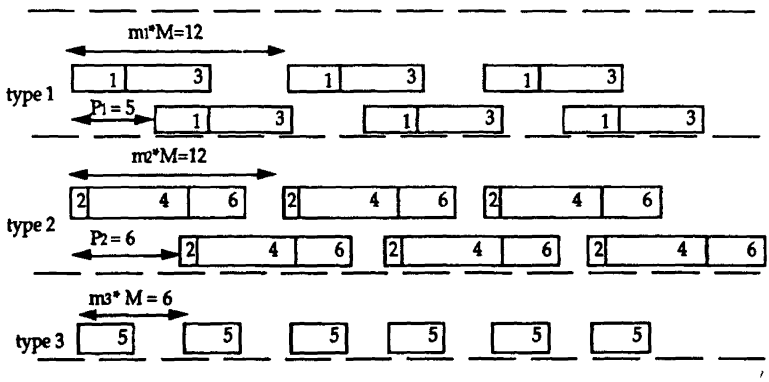

Figure 10

Nous avons donc montré que dans le cas de plusieurs types de machines et de tâches génériques indépendantes il existe un ordonnancement K-périodique stable optimal.

Nous introduisons dans le paragraphe suivant des contraintes de précédence et nous montrons comment déduire par de simples décalages d'indice sur les suites de la solution du problème relâché un ordonnancement optimal pour le problème contraint.

4. PLUSIEURS TYPES DE MACHINES, TÂCHES DÉPENDANTES

Dans ce chapitre, on démontre qu'une solution optimale du problème général peut être construite à partir de l'ordonnancement optimal du problème relaxé de ses contraintes de succession en supprimant simplement de cette solution pour chaque tâche générique T_i ses Δ_i premières exécutions. Il en résulte que l'ordonnancement optimal possède un régime transitoire de durée finie.

La construction des suites K-périodiques de la solution optimale est fondée sur la propriété suivante.

PROPRIÉTÉ : Soient u_n et v_n deux suites K-périodiques strictement croissantes de même fréquence,

(*a*) il existe un entier δ tel que pour tout entier Δ supérieur ou égal à δ, la suite $v'_n = v_{n+\Delta}$ domine la suite u_n;

(*b*) si $v'_n = v_{n+\Delta}$ domine u_n, alors pour tout Δ_1, Δ_2 tel que $\Delta_2 - \Delta_1 \geqq \Delta$, $v_{n+\Delta_2}$ domine $u_{n+\Delta_1}$.

Preuve : Soient u_n et v_n, deux suites K-périodiques de même fréquence. On suppose que les propriétés de périodicité de ces suites débutent au rang 0. On a alors :

$$\forall n > 0, \quad v_{n+K} = v_n + w, \text{ et } u_{n+k'} = u_n + w' \qquad \text{avec} \quad w, w', K \text{ et } K',$$

des entiers strictement positifs tels que $w/K = w'/K'$.

Si on note $P = \text{PPCM}\ (K, K')$, on remarque que les suites u_n et v_n sont K-périodiques de facteur de périodicité P et de période w/K.

D'autre part, les valeurs $u_0, u_1, \ldots, u_{p-1}$ sont bornées et la suite v_n tend vers l'infini quand n tend vers l'infini. Donc, il existe un entier δ positif tel que pour tout $n \in \{1, \ldots, \bar{P} - 1\}$, $v_{n+\delta} \geqq u_n$.

On a alors, pour tout $n > 0$, et pour tout $\Delta \geqq \delta$, $v_{n+\Delta} \geqq u_n$. Donc, la suite $v'_n = v_{n+\Delta}$ domine u_n.

De plus, pour tout couple d'entiers positifs (Δ_1, Δ_2) tels que $\Delta_2 - \Delta_1 \geqq \Delta$, $v_{n+\Delta_2} \geqq v_{n+\Delta+\Delta_1}$ par croissance de v_n, et $v_{n+\Delta+\Delta_1} \geqq u_{n+\Delta_1}$ d'après la première partie de la proposition, donc, la suite $v_{n+\Delta_2}$ domine $u_{n+\Delta_1}$. □

La proposition précédente permet d'affirmer qu'il est possible à partir de l'ordonnancement K-périodique t_i^N du problème relaxé de ses contraintes de précédence d'associer à chaque arc (i, j) du graphe de précédence une valeur Δ_{ij} telle que les suites $t_i'^N = t_i^N$ et $t_j'^N = t_j^{N+\Delta_{ij}}$ vérifient $t_i'^N + p_i \leqq t_j'^N$.

Si (i, j) est un arc du graphe de précédence et si les occurrences des tâches T_i et T_j sont à exécuter sur des machines de type respectivement u et v, alors $\Delta_{ij} = \text{PPCM}\ (m_u, m_v)$ convient.

Le problème consiste alors à associer à chaque tâche générique T_i un décalage Δ_i tel que

$$\forall (i, j) \in U, \quad \Delta_j - \Delta_i \geqq \Delta_{ij}.$$

Les Δ_i constituent donc un ensemble de potentiels du graphe $G = (I, U)$, valué par les Δ_{ij}. Une solution possible est donnée par l'ensemble de potentiels

minimaux :

Δ_i = Valeur maximale d'un chemin d'extrémité i.

En conclusion, les suites $\theta_i = t_i^{N+\Delta_i}$ constituent un ordonnancement K-périodique optimal de fréquence φ pour le problème général.

Exemple : En reprenant l'exemple donné au chapitre précédent, on considère maintenant que les tâches sont soumises à des contraintes de précédence définies par le graphe G (*fig.* 11).

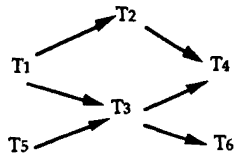

Figure 11

Les décalages d'indice sont alors donnés par la figure 12 :

Ti	T1	T2	T3	T4	T5	T6
Δi	1	2	2	3	1	3

Figure 12

Le planning définitif des tâches est représenté par la figure 13 :

Le régime transitoire a donc une durée de 24.

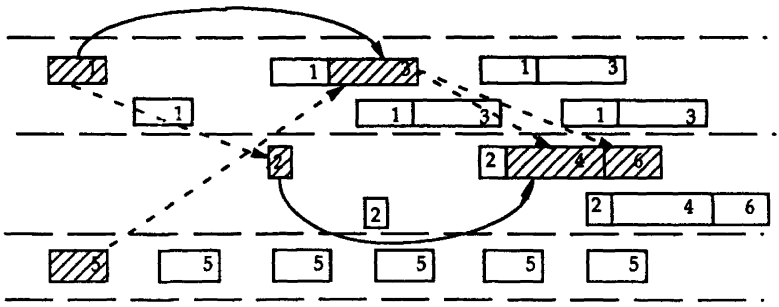

Figure 13

La durée du régime transitoire que l'on cherche à minimiser dépend essentiellement du choix des Δ_{ij}. La proposition suivante fournit une borne inférieure pour ces décalages qui permet de réduire sensiblement la durée du régime transitoire.

Pour limiter la durée du régime transitoire, on peut alors minimiser le décalage pour des séquences d'exécutions fixées en considérant la proposition suivante.

PROPOSITION 2 : *Si T_i et T_j sont deux tâches génériques telles que toutes les occurrences de la tâche T_i doivent être exécutées avant celles de la tâche T_j, si u et v sont les indices des types de machines auxquelles elles sont affectées, si d_i (resp. d_j) est la somme des durées des tâches situées avant T_i (resp. avant T_j) dans la séquence des tâches affectées aux machines de type u S_u (resp. de type v S_v), alors, Δ_{ij} est un décalage sur les indices entre les exécutions de T_i et T_j si et seulement si :*

$$\Delta_{ij} M + \text{mod}\,(\Delta_{ij} - 1, \text{PGCD}\,(m_u, m_v))(M - \text{Max}\,(P_u, P_v)) \geqq d_i - d_j + p_i + (m_v - 1)(M - P_v).$$

Preuve : En annexe.

Exemple (suite) : En utilisant, toujours sur le même exemple, l'inégalité ci-dessus pour le calcul de Δ_{ij}, on obtient alors comme valeurs pour les décalages

Ti	T1	T2	T3	T4	T5	T6
Δi	1	3	3	5	1	5

Figure 14

le tableau donné en figure 14. Le planning définitif des tâches est alors représenté par la figure 15 :

Le régime transitoire a une durée de 18.

CONCLUSION

Dans cet article, nous avons proposé une solution polynomiale à un problème d'ordonnancement cyclique avec contraintes de ressources. Cette étude permet d'obtenir des bornes supérieures pour les fréquences d'exécution des tâches génériques dans le cas de problèmes plus généraux. De plus, elle montre que des versions cycliques de certains problèmes *NP*-difficiles sont des problèmes polynomiaux. Ces résultats constituent le point de départ d'une

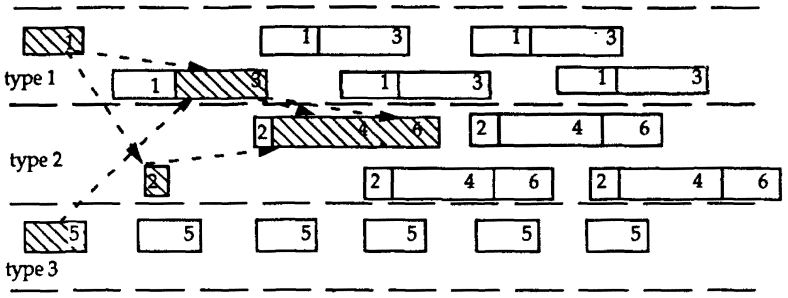

Figure 15

étude systématique des versions cycliques de problèmes d'ordonnancement classiques et de leur complexité.

BIBLIOGRAPHIE

[1] J. CARLIER et Ph. CHRETIENNE, Problèmes d'ordonnancement : modélisation, complexité, algorithmes. Masson, Paris, *Collection Et. et Rech. en Informatique.*

[2] G. COHEN, D. DUBOIS, J. P. QUADRAT et M. VIOT, A Linear System Theoric View of Discrete Events Processes and its Use for Performance Evaluation in Manufacturing, *IEEE Trans on Aut Control*, 1985, *AC 30*, p. 210-220.

[3] S. C. GRAVES, H. C. MEAL, D. STEFEK et A. H. ZEGHMI, Scheduling of Re-Entrant Flow Shops, *Journal of Operations Management*, August 1983.

[4] C. HANEN, Problèmes d'ordonnancement des architectures pipelines : modélisation, optimisation, algorithmes, *Thèse d'université*, Rapport MASI n° 193, Université Paris-VI, September 1987.

[5] H. HILLION et J. M. PROTH, Performance Evaluation of Job-Shop Systems Using Timed Event Graphs, *IEEE Trans on Aut Control*, 1989, *AC 1*, p. 3-9.

[6] P. M. KOGGE, The Architecture of Pipelined Computers, New York, McGraw Hill, 1981.

[7] H. MATSUO, Cyclic Sequencing Problems in the two-Machine Permutation Flow-shop: Complexity, Worst Case and Average Case Analysis, *Graduate School of Business at the University of Texas at Austin*, Technical Report, January 1988.

[8] R. ROUNDY, Cyclic Shedules for Job Shops with Identical Jobs, *School of operations research and industrial engineering*, Cornell University, Technical Report No. 766, July 1988.

ANNEXE

DÉMONSTRATION DE LA PROPOSITION 2

Soient Δ_i et Δ_j les décalages d'indices pour les exécutions des tâches T_i et T_j tels que chaque exécution de la tâche T_i soit effectuée avant celle de la tâche T_j.

Si on pose $\Delta_i = q^i m_u + r^i$ avec q^i et r^i quotient et reste de la division euclidienne de Δ_i par m_u, la première exécution de la tâche T_i est effectuée à la date :

$$t_i'^{\,1} = q^i m_u M + r^i P_u + \sum_{q \in Q_i} p_q$$

avec $Q_i = \{ T_{k,\ k \in I} / \text{AFF}(k) = u$ et T_k a été placé avant T_i dans la séquence $S_u \}$.

La date d'exécution de la N-ième occurrence de la tâche T_i a alors pour expression :

$$t_i'^{\,N} = t_i'^{\,1} + q'^{\,i} m_u M + r'^{\,i} P_u + m_u (M - P_u)\ 1_{\{r'^{\,i} + r^i < m_u\}}$$

où 1 représente l'indicatrice d'ensemble et $r'^{\,i}$ et $q'^{\,i}$ respectivement le quotient et le reste de la divion euclidienne de N par m_u.

Pour tout $N > 0$, $T(i, N)$ doit être exécutée avant $T(j, N)$. Donc :

$$\forall N > 0, \quad t_i'^{\,N} + p_i \leqq t_j'^{\,N} \quad \Leftrightarrow \quad \forall N > 0, \qquad t_j'^{\,N} - t_i'^{\,N} \geqq p_i \tag{1}$$

$(1) \Leftrightarrow \forall N > 0$, $q'^{\,i}$ et $r'^{\,i}$ (resp. $q'^{\,j}$ et $r'^{\,j}$) étant quotient et reste de la division euclidienne de $N-1$ par m_u (resp. m_v),

$$t_j'^{\,1} - t_i'^{\,1} \geqq p_i + M (q'^{\,i} m_u - q'^{\,j} m_v) + (r'^{\,i} P_u - r'^{\,j} P_v) + m_u (M - P_u)\ 1_{\{r'^{\,i} + r^i < m_u\}} - m_v (M - P_v)\ 1_{\{r'^{\,j} + r^j < m_v\}}.$$

On cherche alors à évaluer le second membre de cette inégalité. Pour cela, on pose :

$$F(x, r^i) = m_u (M - P_u)\ 1_{\{x \geqq m_u - r^i\}} + x (P_u - M), \qquad x \in \{0, \ldots, m_u - 1\}$$

$$G(x, r^j) = -m_v (M - P_v)\ 1_{\{x \geqq m_v - r^j\}} - x (P_v - M), \qquad x \in \{0, \ldots, m_v - 1\}.$$

L'équivalence a alors pour expression :

$(1) \Leftrightarrow \forall N > 0$, $r'^{\,i}$ et $r'^{\,j}$ étant les restes de la division euclidienne de $N-1$ par m_u et m_v.

$$t_j'^{\,1} - t_i'^{\,1} \geqq p_i + F(r'^{\,i}, r^i) + G(r'^{\,j}, r^j).$$

Si on définit
$D = \{ (x, y) \in \{0, m_u - 1\} * \{0, m_v - 1\} / \ni N > 0$, x et y sont les restes de la division euclidienne de $N-1$ par m_u et $m_v \}$, on a alors :

$$(1) \quad \Leftrightarrow \quad t_j'^{\,1} - t_j'^{\,1} \geqq p_i + \operatorname*{Max}_{(x,\ y) \in D} (F(x, r^i) + G(y, r^j)). \tag{2}$$

On procède alors en trois étapes : dans un premier temps, on démontre qu'il est possible de donner une définition de D qui ne fasse intervenir que les quantitées de machines m_u et m_v, et qui, de ce fait, soit plus facilement exploitable. On propose alors un couple d'entiers dont on démontre qu'il appartient à D et qu'il maximise à r^i et r^j fixés la somme des deux fonctions F et G. Enfin, on remplace la valeur de cette somme dans l'inéquation (2) pour obtenir le résultat de la proposition 2.

1. SIMPLIFICATION DU DOMAINE D

On propose une expression simplifiée du domaine D donnée par le lemme suivant :

LEMME 1 : *Si on note*

$$Q_{uv} = \text{PGCD}\ (m_u, m_v)$$

$$D' = \{(p, q) \in \{0, m_u - 1\} \times \{0, m_v - 1\} / p - q \equiv 0\ (Q_{uv})\}$$

alors $D = D'$.

Preuve : On démontre le lemme par une double inclusion.

$(D' \supset D)$.

Si $(p, q) \in D$.

Alors, il existe α et β entiers naturel tels que $\alpha m_u + p = \beta m_v + q$.

Donc

$$p - q = \beta m_v - \alpha m_u = Q_{uv}\left(\beta \frac{m_v}{Q_{uv}} - \alpha \frac{m_u}{Q_{uv}}\right).$$

Donc $p - q \equiv 0\ (Q_{uv})$.

$(D \supset D')$

Si $(p, q) \in D'$.

Alors, il existe un entier relatif k tel que $p = q + Q_{uv} k$.

D'autre part, $\dfrac{m_u}{Q_{uv}} \Lambda \dfrac{m_v}{Q_{uv}} = 1$.

Donc, d'après le théorème de Bezout, il existe deux entiers relatifs z et w tels que $zm_u + wm_v = Q_{uv}$.

Donc

$$p = q + (zm_u + wm_v)\ k \Rightarrow zm_u k + q = -wm_v k + p$$

$$\Rightarrow \forall k' \in Z,\ (zk + m_v k')\ m_u + q = (-wk + m_u k')\ m_v + p.$$

Si on choisit alors $k' \geqq \text{Max } (-zk/m_v,\ wk/m_u)$ et que l'on pose

$$N-1=(zk+k'm_v)\ m_u+p=(-wk+k'\ m_u)\ m_v+q$$

alors p et q sont les restes de la division euclidienne de $N-1$ par respectivement m_u et m_v. □

2. CALCUL DU MAXIMUM DE LA SOMME DE F ET G

On propose une expression du maximum de la somme sur D à r^i et r^j fixés des fonctions F et G donnée par le lemme suivant :

LEMME 2 : *Le maximum de la somme des fonctions F et G à r^i et r^j fixés sur le domaine D est atteint pour le couple :*

$$(p,\ q)=(\text{mod } (\alpha\, 1_{\{P_u \geqq P_v\}}-r^i,\ m_u),\ \text{mod } (-\alpha\, 1_{\{P_v > P_u\}}-r^j-1,\ m_v))$$

et a pour expression :

$$\underset{(p,\ q)\ \in\ D}{\text{Max}}\ (F\ (p,\ r^i)+G\ (q,\ r^j))=(r^i-\alpha\, 1_{\{P_u \geqq P_v\}})\ (M-P_u) + \text{mod } (-r^j-1-\alpha\, 1_{\{P_u < P_v\}},\ m_v)\ (M-P_v).$$

Preuve : On démontre la proposition dans le cas où $P_u \geqq P_v$, la démonstration étant similaire dans le cas contraire.

Pour cela, on procède en trois parties : on montre que le couple (p, q) donné est bien élément de D, puis on calcule pour ce couple la somme des deux fonctions à maximiser et on montre que l'on a alors atteint le maximum.

Le couple (p, q) est élément de D

$p-q=\text{Mod } (\alpha-r^i,\ m_u)-\text{Mod } (-r^j-1,\ m_v)$.

Donc $p-q \equiv \alpha-r^i+r^j+1\ (Q_{uv})$.

Par définition de α, $p-q \equiv 0\ (Q_{uv})$.

Le couple (p, q) est donc élément de D.

Calcul de la somme de deux fonctions pour (p, q) donné

Par définition de la fonction F :

$$F\ (p,\ r^i)=m_u\ (M-P_u)\ 1_{\{\text{mod } (\alpha+m_u-r^i,\ m_u) \geqq m_u-r^i\}} + \text{mod } (\alpha+m_u-r^i,\ m_u)\ (P_u-M).$$

Donc,

$$F\ (p,\ r^i)=(M-P_u)\ (-\alpha+r^i).$$

D'autre part, pour la fonction G :

$$G(q, r^j) = -m_v (M - P_v) 1_{\{m_v - r^j - 1 \geq m_v - r^j\}} - (m_v - r^j - 1)(P_v - M).$$

Donc $G(m_v - r^j - 1, r^j) = \text{mod}(m_v - r^j - 1, m_v)(M - P_v)$.

Donc, $F(p, r^i) + G(q, r^j)$ a bien la valeur donnée par la proposition.

La somme des deux fonctions F et G est maximale en (p, q)

On démontre ce résultat par l'absurde : si on suppose qu'il existe $(p', q') \in D$, tels que $F(p', r^i) + G(q', r^j) > F(p, r^i) + G(q, r^j)$ avec $(p, q) \neq (p', q')$.

Par définition de D, $p - p' \equiv q - q' \ (Q_{uv})$. On considère alors trois cas :

Si $q' = q$ *et* $p' \neq p$, alors $F(p', r^j) > F(p, r^j)$

– Si $M = P_u$, alors $F(x, r^i)$ est constante égale à 0 pour $x \in \{0, m_u - 1\}$. Donc, il y a contradiction.

– Sinon, par définition de la fonction F :

$$F(p', r^i) > F(p, r^i) \quad \Rightarrow \quad p' \in \{\text{mod}(m_u - r^i, m_u) \ldots \text{mod}(\alpha - 1 + m_u - r^i, m_u)\}.$$

Soit alors β un entier positif ou nul et strictement inférieur à α tel que

$$p' = \text{mod}(\beta - r^i, m_u).$$

$$p - p' = \text{mod}(\alpha - r^i, m_u) - \text{mod}(\beta - r^i, m_u) \equiv q - q' \ (Q_{uv})$$

Donc $p - p' \equiv 0 \ (Q_{uv}) \Rightarrow \alpha - \beta \equiv 0 \ (Q_{uv})$.

Ce qui est impossible puisque $0 \leq \beta < \alpha < Q_{uv}$.

Si $q' \neq q$ *et* $p' = p$

– Si $M = P_v$, alors $M = P_u$ et les deux fonctions sont identiquement nulles. Donc tous les couples de D sont solutions. L'hypothèse est donc contradictoire.

– Sinon, $G(q', r^j)$ étant le maximum de G à r^j fixé, l'hypothèse de départ est absurde.

Si $q' \neq q$ *et* $p' \neq p$

– Si $M = P_v$, alors comme précédemment, l'hypothèse est contradictoire.

– Sinon, la fonction G est maximale à r^j fixé en q. Donc $F(p', r^i) > F(p, r^i)$ est une condition nécessaire pour que l'hypothèse de départ soit vérifiée.

- Si $M = P_u$, ceci est impossible.
- Sinon, $p' \in \{\text{mod}(m_u - r^i, m_u) \ldots \text{mod}(\alpha - 1 + m_u - r^i, m_u)\}$.

Soit $\beta \geqq 0$ et strictement inférieur à α tel que $p' = \text{mod}\,(\beta - r^i, m_u)$ $p - p' = \text{mod}\,(\alpha - r^i, m_u) - \text{mod}\,(\beta - r^i, m_u)$.

Donc $p - p' \equiv \alpha - \beta\ (m_u)$ avec

$$m_u \geqq \alpha - \beta > 0 \quad \text{et} \quad F(p', r^i) - F(p, r^i) = (\alpha - \beta)(M - P_u).$$

D'autre part,

$$p - p' \equiv q - q'\ (Q_{uv})$$
$$\Rightarrow \quad q - q' \equiv \alpha - \beta\ (Q_{uv}) \quad \Rightarrow \quad \text{mod}\,(q - q', m_v) = \alpha - \beta + k\,Q_{uv}$$

avec $k \geqq 0$, puisque $\text{mod}\,(q - q', m_v) \geqq 0$ et $Q_{uv} \geqq \alpha - \beta$.

Donc

$$G(q, r^j) - G(q', r^j) = (\alpha - \beta + k\,Q_{uv})(M - P_v).$$

Ainsi,

$$F(p', r^i) - F(p, r^i) > G(q, r^j) - G(q', r^j)$$
$$\Rightarrow \quad (\alpha - \beta)(M - P_u) > (\alpha - \beta + Q_{uv}\,k)(M - P_v)$$
$$\Rightarrow \quad (\alpha - \beta)(P_v - P_u) > Q_{uv}\,k\,(M - P_v).$$

Le terme de gauche est négatif, celui de droite positif ou nul.

Donc, l'hypothèse de départ ne peut être vérifiée. □

3. DÉMONSTRATION DE LA PROPOSITION

Par définition :

$$t'^1_j - t'^1_i = q^j m_j M + r^j P_v + \sum_{q \in Q_j} p_q - q^i m_u M - r^i P_{\text{AFF}\,(a)} - \sum_{q \in Q_i} p_q$$

$$t'^1_j - t'^1_i = (\Delta_j - \Delta_i)\,M - r^j (M - P_v) + r^i (M - P_u) + \sum_{q \in Q_j} p_q - \sum_{q \in Q_i} p_q$$

En remplaçant cette valeur et celle du maximum de F et G dans la condition ci-dessus, on obtient :

$$(\Delta_j - \Delta_i)\,M + \sum_{q \in Q_j} p_q - \sum_{q \in Q_i} p_q \geqq -\alpha\,(M - P_i)\,1_{\{P_i \geqq P_j\}}$$
$$+ (\text{mod}\,(-r^j - 1 - \alpha\,1_{\{P_i < P_j\}}, m_j) + r^j)(M - P_v).$$

De plus :

$$\text{mod}\,(-r^j - 1 - \alpha\,1_{\{P_i < P_j\}}, m_j) + r^j = m_j - 1 - \alpha\,1_{\{P_i < P_j\}}.$$

Donc, on obtient l'inégalité donnée par la proposition.