RAIRO. RECHERCHE OPÉRATIONNELLE

C. PRINS

Deux problèmes d'ordonnancement en télécommunications par satellite

RAIRO. Recherche opérationnelle, tome 25, n° 3 (1991), p. 341-358

<http://www.numdam.org/item?id=RO_1991__25_3_341_0>

Recherche opérationnelle/Operations Research
(vol. 25, n° 3, 1991, p. 341 à 358)

DEUX PROBLÈMES D'ORDONNANCEMENT EN TÉLÉCOMMUNICATIONS PAR SATELLITE (*)

par C. PRINS [1]

Résumé. – *Cet article présente deux problèmes NP-difficiles rencontrés dans les télécommunications par satellite. Le premier consiste à ordonnancer des paquets de données sur les répéteurs d'un satellite, sous des contraintes de ressources, de façon à minimiser la durée de transmission. Il est bien résolu grâce à une méthode sérielle.*

Le second est la minimisation du nombre d'équipements nécessaires aux stations terriennes pour transmettre un ensemble donné de paquets. Une procédure de recuit simulé le résoud de manière quasi-optimale.

Des logiciels basés sur nos méthodes sont utilisés quotidiennement pour l'exploitation du système à satellite d'EUTELSAT. Notre approche est applicable à certains problèmes d'atelier et de localisation.

Mots clés : Ordonnancement; contrainte de ressources; méthode sérielle; recuit simulé; télécommunications par satellite.

Abstract. – *This paper presents two NP-hard problems related to satellite communications. The first consists of scheduling data packets thru satellite repeaters, under resource constraints, to minimize transmission time. It is well solved using a list scheduling algorithm.*

The second is to minimize the amount of equipment required at earth stations for transmitting a given set of bursts. A simulated annealing procedure solves it quasi-optimally.

Software programs implementing these methods are used in the day-to-day operation of the EUTELSAT system. Our approach could be applied to similar problems encountered in shop planning and in facility location.

Keywords : Scheduling; resource constraints: list algorithm; simulated annealing; satellite communications.

(*) Reçu en mai 1990.
(1) Institut de Mathématiques Appliquées, B.P. n° 808, 49008 Angers Cedex 01.

Recherche opérationnelle/Operations Research, 0399-0559/91/03 341 18/$ 3.80

I. INTRODUCTION

Les réseaux à satellite utilisant l'AMRT (Accès Multiple à Répartition dans le Temps) posent des problèmes d'optimisation variés, décelés dès les années 70, alors que de tels systèmes étaient encore à l'état de projets.

Les premiers travaux publiés ont traité des modèles simplifiés de systèmes, avec une seule fonction économique: voir par exemple les articles d'Inukai [9] et de Bongiovanni [10]. En fait, pour des raisons matérielles et logicielles, la mise en place des systèmes réels a révélé des objectifs multiples et des contraintes plus nombreuses que prévu.

Dans un premier temps, notre contribution au domaine a été justement de traiter un vrai système, celui de l'organisation européenne EUTELSAT, depuis la modélisation jusqu'aux logiciels utilisés pour l'exploitation. Peu d'auteurs ont ainsi abordé un cas réel dans sa globalité et sa complexité: citons les approches intégrées de King [11], et de Minoux et Brouder [12], pour le système concurrent INTELSAT.

Plus récemment, en sortant du cas EUTELSAT, nous avons répertorié et modélisé les problèmes d'optimisation AMRT, évalué leur complexité, et proposé des méthodes de résolution [1].

Nous présentons dans cet article deux problèmes représentatifs, pour lesquels nous avons amélioré considérablement nos méthodes initiales décrites dans [2, 3]. Le système EUTELSAT qui en est l'origine est brièvement décrit dans la partie II. La partie III définit les deux problèmes étudiés, qui sont très liés, à savoir l'*ordonnancement des paquets* et la *minimisation du nombre de convertisseurs de fréquence*. A notre connaissance, aucun autre auteur n'a traité le second problème. Les méthodes de résolution sont exposées dans les parties IV et V.

Le domaine d'origine des problèmes traités nécessite quelques notions simples de télécommunications, qui ont été réduites ici au strict minimum. Les modélisations considérées s'appliquent en fait à une classe de problèmes débordant du cadre des télécommunications.

II. DESCRIPTION SIMPLIFIÉE DU SYSTÈME

II.1. Satellite et répéteurs

L'élément-clé du système EUTELSAT est un *satellite géostationnaire*, couvrant les pays européens avec les *faisceaux* de ses *antennes*. L'antenne de réception, unique, permet au satellite de recevoir des *paquets* de données

numériques émis par des stations au sol. Cette antenne alimente un ensemble d'amplificateurs appelés *répéteurs*, ayant chacun sa propre fréquence de réception.

Les paquets sont retransmis par trois antennes émettrices dont chacune couvre une partie de la zone de desserte. Les trois faisceaux ainsi définis sont appelés Ouest, Est et Atlantique. Chaque répéteur est connecté en sortie à une de ces antennes, et plusieurs répéteurs peuvent être affectés au même faisceau de destination.

II.2. Stations et paquets

Les stations terriennes reçoivent des données numériques des réseaux terrestres et les assemblent en paquets, qui seront échangés via le satellite. Pour émettre un paquet $A \rightarrow B$ de la station A vers la station B, la station A doit choisir un répéteur connecté au faisceau couvrant B.

La transmission s'effectue simplement en émettant le paquet vers le satellite, sur la fréquence de réception du répéteur choisi. Ce dernier réémet le paquet vers le faisceau de destination. Toute station C de ce faisceau pouvant aussi recevoir le paquet de B, il est permis de définir des *paquets multi-destinations* comme $A \rightarrow B, C$.

II.3. Mode d'exploitation

Le système EUTELSAT ne réorganise pas ses paquets en temps réel. Ceci vient du fait que la mission principale du système est la téléphonie numérique. La mise en service de nouvelles lignes doit s'effectuer par étapes, car elle requiert une planification bilatérale entre pays, et des travaux de connexion dans les centraux téléphoniques.

En pratique, les pays participants fixent ensemble le trafic (nombre de lignes) à assurer pour une période de 6 à 12 mois. Le système est alors configuré pour ce trafic et va rester stable jusqu'à la période suivante. Comme les nouvelles valeurs du trafic sont confirmées quelques semaines à l'avance, il est possible d'optimiser soigneusement l'utilisation du satellite et les équipements nécessaires dans les stations terriennes.

Les systèmes AMRT en général posent des problèmes d'optimisation fort variés, et cet article n'en considère que deux en particulier (pour une présentation générale, *voir* [1]). Dans la suite, on suppose qu'un ensemble de paquets et un nombre adéquat de répéteurs ont déjà été définis, en tant que solutions d'autres problèmes d'optimisation décrits dans [2, 3].

III. LES DEUX PROBLÈMES CONSIDÉRÉS

Les stations se partagent le satellite à tour de rôle, au cours d'un cycle de 2 ms appelé *trame AMRT*. Comme le débit numérique des répéteurs est constant (120 Mbit/s), la trame est graduée en bits. La discipline de partage des répéteurs est imposée par les exploitants du système. Il suffit de la définir pour une trame, puisque qu'elle est identique pour les trames suivantes.

Plus précisément, chaque paquet doit avoir un répéteur et un instant d'émission dans la trame, tels que (*fig.* 1):

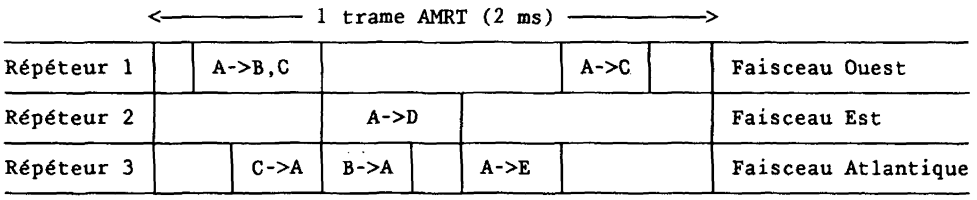

Figure 1. – Ordonnancement simplifié avec trois répéteurs (un par faisceau) B, C sont dans le faisceau Ouest, D dans l'Est, et A, E dans l'Atlantique.

– le faisceau du répéteur choisi couvre effectivement les destinations du paquet;

– aucune station n'émette ni ne reçoive plus d'un paquet à la fois (une station peut cependant émettre un paquet pendant qu'elle en reçoit un autre, comme *A* sur la figure 1);

– aucun paquet n'en chevauche un autre dans le même répéteur;

– la capacité de la trame ne soit pas dépassée.

Le *Problème* 1 est la minimisation de la durée totale de transmission C_{MAX}, de façon à réserver un maximum de capacité en fin de trame pour des tests ou du trafic imprévu. Il s'agit d'un *problème d'ordonnancement à contraintes de ressources*, connu pour être NP-difficile à partir de trois répéteurs, même pour des paquets de durées égales (problème SS10 de Garey et Johnson [4]).

Ici, les répéteurs sont les machines, les émetteurs et récepteurs des stations sont les ressources renouvelables, et les paquets sont les tâches se partageant les ressources. Une station peut être vue comme un couple de deux ressources, un émetteur et un récepteur. Aucune contrainte de précédence n'existe entre les tâches. Les faisceaux partitionnent les machines en sous-ensembles de machines équivalentes, une tâche devant être exécutée sur une machine de classe appropriée. Nous résolvons le Problème 1 dans la partie IV avec une *méthode sérielle*.

Le *Problème* 2 consiste à minimiser le nombre de convertisseurs de fréquence. Le convertisseur est un appareil coûteux des stations terriennes, synthétisant la fréquence d'accès à un répéteur. Il représente le coût fixe à payer par un émetteur ou un récepteur pour pouvoir gérer un nombre non limité de paquets sur un répéteur choisi.

Par exemple, dans la figure 1, l'émetteur de la station A utilise trois convertisseurs pour émettre ses paquets dans les répéteurs 1, 2 et 3. De même, le récepteur de A a besoin d'un convertisseur pour recevoir les paquets $B \rightarrow A$ et $C \rightarrow A$ du répéteur 3. L'ordonnancement de la figure 1 nécessite en tout 10 convertisseurs. Si la station D se situait dans le faisceau Ouest, un convertisseur pourrait être économisé pour A, en déplaçant $A \rightarrow D$ vers le répéteur 1.

Les problèmes 1 et 2 sont très liés. Ils pourraient fusionner en un problème d'ordonnancement consistant à minimiser la dispersion des ressources sur les machines (ici, des stations sur les répéteurs). En fait, les contraintes du système imposent de connaître assez tôt le nombre de convertisseurs, car il faut commander et installer ces équipements dans les stations. L'ordonnancement peut par contre n'être calculé qu'au dernier moment, car il n'implique pas de matériel supplémentaire (il est simplement téléchargé sous forme de table dans l'ordinateur contrôlant chaque station).

Pour ces raisons, le problème 2 est traité en pratique dans une étape de calcul optionnelle, avant l'ordonnancement: les paquets sont préaffectés aux répéteurs de façon à minimiser le nombre de convertisseurs. Sans cette étape, l'algorithme d'ordonnancement garde le choix du répéteur, quand le faisceau de destination d'un paquet est desservi par plusieurs répéteurs équivalents. Même isolé sous cette forme, le problème 2 est NP-difficile [1].

Nous introduisons les notations ci-dessous pour la suite de l'article:

r, n, m, b sont respectivement les nombres de stations, de paquets, de répéteurs et de faisceaux;

S_k est le faisceau $(1 \ldots b)$ auquel le répéteur k $(1 \ldots m)$ est connecté;

M_k est le nombre de répéteurs connecté au faisceau k;

F est la taille (durée) de la trame AMRT, en bits.

Chaque paquet peut être défini par (e_j, R_j, p_j, s_j), dans lequel:

e_j est la station émettrice du paquet $(1 \ldots r)$;

R_j est *l'ensemble* des stations de destination du paquet (chacune dans $1 \ldots r$);

p_j est la taille (durée) du paquet (nombre entier de bits);

s_j est le faisceau de destination $1 \ldots b$, j devant être retransmis dans un répéteur k tel que $S_k = s_j$.

Les valeurs maximales suivantes caractérisent le système EUTELSAT et donnent une idée de la taille des problèmes: $m=7$, $r=32$, $n=256$, $b=3$, $|R_j|=8$, $F=241664$ bits, avec des p_j dans [832,136768].

IV. RÉSOLUTION DU PROBLÈME 1

IV.1. Principe des méthodes sérielles en général

Puisqu'aucun algorithme polynômial ne peut être trouvé pour le Problème 1 (à moins que P=NP), nous avons conçu une *méthode sérielle* (appelée aussi *algorithme de liste*) pour le résoudre de manière approchée.

Les méthodes sérielles classent au préalable les tâches selon un ordre de priorité, puis construisent itérativement un ordonnancement à partir de la date de début. A chaque itération, la tâche disponible de plus grande priorité est affectée à la première machine libérée.

Ces méthodes construisent facilement des ordonnancements admissibles dans lesquels aucune machine n'attend inutilement. La qualité des solutions obtenues dépend bien sûr de la liste choisie pour les tâches.

IV.2. Évaluation des méthodes sérielles

Pour une donnée d, la qualité d'un algorithme de liste A se mesure souvent par sa *performance relative* $R_A(d)=C_{MAX}(d)/C^*_{MAX}(d)$, quotient de la durée totale, calculée par l'algorithme, par l'optimum réel. On peut parfois établir le *comportement au pire* P_A, maximum des performances relatives sur l'ensemble des données d possibles. Les résultats suivants sont ainsi connus [7] pour l'ordonnancement de tâches indépendantes sur m machines identiques (problème connu sous le nom de M-processeurs):

$$P_A=2-\frac{1}{m} \quad \text{pour une liste quelconque} \tag{1}$$

$$P_A=\frac{4}{3}-\frac{1}{3m} \quad \text{pour des tâches triées par durées décroissantes} \tag{2}$$

Comme on peut s'y attendre, l'introduction de ressources renouvelables dégrade fortement le comportement au pire de M-processeurs:

$$P_A=\frac{m+1}{2} \quad \text{pour une liste quelconque.} \tag{3}$$

Graham [8] montre que cette borne est atteinte, mais sur des cas où les tâches utilisent chacune un grand nombre de ressources, ce qui n'est pas le cas de notre Problème 1. Avec une liste bien choisie, on peut donc certainement améliorer P_A. De plus un mauvais P_A n'interdit pas un bon comportement moyen, surtout sur des données réelles ayant une structure spécifique.

Le calcul d'un comportement au pire en présence de contraintes de ressources est complexe [8], et nous n'avons pas encore pu le réussir pour la méthode sérielle présentée en IV.4. Nous devons donc nous contenter des performances relatives *moyennes* obtenus sur des données réelles. Les pires cas rencontrés, soigneusement enregistrés, sont à 16% de l'optimum pour $m=6$, ce qui est très en deçà du pessimisme de (3).

Le Problème 1 étant NP-difficile, l'optimum $C^*_{\text{MAX}}(d)$ est souvent inconnu. La performance relative d'un algorithme de liste A peut cependant être bornée *a posteriori* si on dispose d'une bonne borne inférieure B pour l'optimum:

$$R_A(d)=\frac{C_{\text{MAX}}(d)}{C^*_{\text{MAX}}(d)} \leqq \frac{C_{\text{MAX}}(d)}{B(d)} \tag{4}$$

En particulier, un ordonnancement tel que $B(d)=C_{\text{MAX}}(d)$ est optimal. Nous définissons maintenant une telle borne pour le Problème 1, puis présentons l'algorithme de liste retenu.

IV.3. Une borne inférieure pour la durée totale

Soient $D_E(i)$ et $D_R(i)$ les quantités totales de trafic respectivement émises et reçues par une station i. Comme i ne peut émettre ni recevoir plus d'un paquet à la fois, il s'agit de bornes inférieures pour C^*_{MAX}:

$$D_E(i)=\sum_{\{j \mid e_j=i\}} p_j, \qquad i=1\ldots r \tag{5}$$

$$D_R(i)=\sum_{\{j \mid i \in R_j\}} p_j, \qquad i=1\ldots r \tag{6}$$

D'autres bornes résultent de la charge moyenne par répéteur dans chaque faisceau k, $D_s(k)$:

$$D_s(k)=\left\lceil \frac{\sum_{\{j \mid s_j=k\}} p_j}{M_k} \right\rceil, \qquad k=1\ldots b \tag{7}$$

Le maximum de toutes ces bornes fournit la borne inférieure B recherchée, une condition nécessaire d'existence d'un ordonnancement étant $B \leqq F$.

$$B = \text{MAX}\left(\underset{i}{\text{MAX}}(D_E(i)),\ \underset{i}{\text{MAX}}(D_R(i)),\ \underset{k}{\text{MAX}}(D_S(k))\right) \tag{8}$$

IV.4. Exemple de calcul de bornes

Nous illustrons le calcul de bornes par un exemple simple, dans lequel 4 stations échangent 8 paquets à travers trois répéteurs. Chaque répéteur dessert l'un des trois faisceaux notés W (Ouest), E (Est) et A (Atlantique). Les stations 2 et 3 sont dans le faisceau W, 4 dans E, la station 1 se trouvant à la limite de W et A. La station 1 peut ainsi être atteinte par un répéteur W (paquet 8) ou A (paquets 4 et 6).

Les émetteurs et récepteurs de chaque paquet, ainsi que leur durée dans une unité ici arbitraire, sont donnés. Les *tableaux I à III* résument le calcul de bornes (les deux dernières colonnes du tableau I servent d'exemple pour le paragraphe IV.5 uniquement).

TABLEAU I

Liste de paquets

N° j	Paquet $e_j \rightarrow R_j$	Taille p_j	Faisceau s_j	Vecteur de priorité V_j	Rang
1	1 → 2	20	W	(100, 100, 60)	6
2	1 → 2, 3	30	W	(100, 100, 80, 60)	5
3	1 → 4	10	E	(60, 60, 60)	7
4	2 → 1	50	A	(180, 130, 60)	2
5	2 → 4	10	E	(60, 60, 60)	8
6	3 → 1	80	A	(180, 130, 120)	1
7	3 → 4	40	E	(120, 60, 60)	4
8	4 → 1, 2, 3	50	W	(180, 100, 100, 80, 50)	3

TABLEAU II

Liste des stations.

Station i	Trafic émis $D_E(i)$	Reçu $D_R(i)$
1	60	180
2	60	100
3	120	80
4	50	60

On trouve une borne finale $B = 180$, due au trafic reçu par la station 1 (souligné dans le tableau II). Tout ordonnancement des paquets du tableau I

TABLEAU III

Liste des faisceaux.

Faisceau k	Charge moyenne par répéteur $D_s(k)$
W	100
E	60
A	130

a donc une durée au moins égale à 180. Si la taille de trame F était limitée par exemple à 160, il n'existerait aucun ordonnancement admissible.

IV.5. Liste de priorité

L'idée de la borne (8) est de considérer les durées d'activité minimales des trois types de ressources impliquées dans la transmission des paquets: les émetteurs, les récepteurs, et les faisceaux. La même idée peut s'étendre aux paquets pour définir une *mesure de priorité*. A chaque paquet j émis par i vers d destinations $j_1, j_2, \ldots, j_d$ situées dans un faisceau k, nous associons un *vecteur de priorité* V_j qui tient compte des ressources du paquet:

$$V_j = (D_E(i), D_R(j_1), D_R(j_2), \ldots, D_R(j_d), D_S(k)) \tag{9}$$

Un *ordre de priorité* est obtenu en triant les composantes de chaque vecteur par valeurs décroissantes, puis en comparant les vecteurs de manière lexicographique (comme des chaînes de caractères): un paquet i sera dit plus prioritaire qu'un autre j si $V_j \leqq V_i$. Si deux paquets ont le même vecteur, le plus long sera le plus prioritaire.

Les deux dernières colonnes du tableau I donnent les vecteurs et l'ordre obtenus pour l'exemple du IV.4: le paquet le plus prioritaire est le n° 6.

IV.6. Explication préliminaire de l'algorithme de liste

Notre algorithme, dont le texte suit, utilise la liste de paquets triée par priorités décroissantes, en tenant compte des conflits de ressources.

Une itération de l'algorithme (lignes 5-17) place dans la trame soit un paquet, soit un temps mort (espace inutilisé à cause des conflits). Elle commence par calculer la plus petite date t à laquelle un ou plusieurs répéteurs se libèrent, ces répéteurs formant un ensemble X (ligne 5). Les émetteurs et récepteurs actifs à t dans les autres répéteurs forment des ensembles Y_E et Y_R (ligne 6).

Les paquets possibles (ensemble Z ligne 7) ont leurs émetteurs et récepteurs libres à t, et leur faisceau est desservi par un répéteur de X. Si Z est non vide, le paquet candidat i de plus forte priorité est ordonnancé à t, dans un répéteur libre de faisceau compatible. Ce répéteur est désormais libre à $t+p_i$ (lignes 13-15). Si Z est vide, on calcule la plus petite date t' après t à laquelle un paquet déjà ordonnancé se termine. L'intervalle $[t, t']$ est condamné, et le répéteur n'est considéré libre qu'à t' (lignes 10-11). Ceci termine une itération.

Après un placement de paquet, si les bornes et les vecteurs de priorité sont recalculés pour la liste des paquets restant à transmettre, on obtient une version dite *dynamique* de l'algorithme: l'ordre des paquets dans la liste résiduelle peut varier d'une itération à l'autre. Une version plus simple mais moins performante, dite *statique*, est obtenue si on ne touche pas au classement initial (suppression de la ligne 16).

IV.7 Algorithme de liste dynamique A

La trame est supposée graduée en bits de 1 à F.

```
1.  Début
2.    Mettre à 1 la 1re date libre de chaque répéteur k, Ak;
3.    Calculer les bornes (formules (5) à (8)) et les vecteurs de priorité Vj de chaque paquet j;
4.    Répéter
5.      Calculer  t = Min(Ak), puis l'ensemble X des répéteurs libres à t, et enfin l'ensemble
        S(X) des faisceaux atteints par X;
6.      Construire les ensembles YE and YR des émetteurs et récepteurs actifs à t;
7.      Calculer l'ensemble de paquets Z = {j | ej ∉ YE, Rj ∩ YR = ∅, et sj ∈ S(X)};
8.      Si Z = ∅ alors
9.        (* On condamne l'intervalle inutilisable [t, t' − 1]*)
10.       Calculer t' = Min(Ak > t);
11.       Pour chaque répéteur k de X faire Ak := t'
12.     Sinon
13.       Soit i un paquet tel que Vi = Min(Vj);
14.       Soit un répéteur k de X avec Sk = si;
15.       Ak := Ak + pi;
16.       Mettre à jour les bornes et les vecteurs de priorité
17.     FinSi
18.   Jusqu'à ce que tous les paquets soient transmis;
19.   Sortir l'ordonnancement, et CMAX = Max(Ak) − 1;
20.   Sortir la distance à l'optimum CMAX/B
20. Fin.
```

IV.8. Exemple d'itération de l'algorithme *A*

On reprend l'exemple du IV.4 pour illustrer une première itération de A. Au départ tous les répéteurs sont libres à la date 1, et sont donc dans X. Aucune station n'est active (Y_E et Y_R sont vides), et tous les paquets sont dans Z. Le paquet n° 6, de la station 3 vers 1, est le plus prioritaire. Il est

donc placé dans le seul répéteur de faisceau compatible, l'unique répéteur Atlantique.

La durée du paquet (80) est soustraite aux bornes $D_E(3)$, $D_R(1)$, et $D_S(A)$, pour mettre à jour les durées résiduelles d'activité des ressources impliquées dans le paquet 6. Les vecteurs de priorité peuvent être alors recalculés pour déterminer le nouvel ordre des paquets restants, ce qui donne les *tableaux IV à VI*, résultant des tableaux I à III au terme de cette première itération.

TABLEAU IV

Liste des paquets restant à transmettre

N° j	Paquet $e_j \rightarrow R_j$	Taille p_j	Faisceau s_j	Vecteur de priorité V_j	Rang
1	1 → 2	20	W	(100, 100, 60)	3
2	1 → 2, 3	30	W	(100, 100, 80, 60)	2
3	1 → 4	10	E	(60, 60, 60)	5
4	2 → 1	50	A	(100, 60, 50)	4
5	2 → 4	10	E	(60, 60, 60)	6
7	3 → 4	40	E	(60, 60, 50)	7
8	4 → 1, 2, 3	50	W	(100, 100, 100, 80, 50)	1

TABLEAU V

État des stations.

Station i	$D_E(i)$ résiduel	$D_R(i)$ résiduel
1	60	100
2	60	100
3	50	80
4	50	60

TABLEAU VI

Liste des faisceaux.

Faisceau k	Charge moyenne résiduelle $D_s(k)$
W	100
E	60
A	50

Pour la seconde itération, le paquet le plus prioritaire (n° 8) ne peut être placé, car son récepteur 1 est occupé sur le répéteur Atlantique. Le paquet 2, second dans l'ordre des priorités, est finalement choisi, ce qui donne l'ordonnancement partiel de la figure 2.

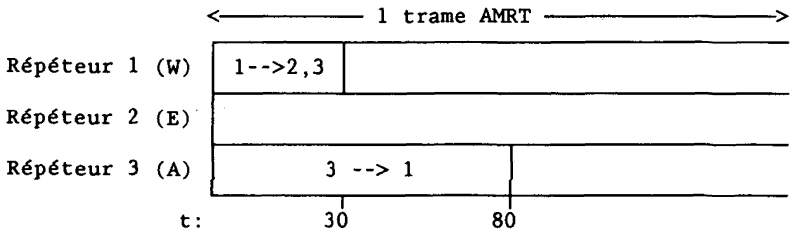

Figure 2. – Ordonnancement partiel pour l'exemple IV.8.

IV.9. Algorithme répétitif *A'*

Pour les gros trafics, des paquets avec beaucoup de destinations peuvent créer des *temps morts* importants dans certains répéteurs: les seuls paquets susceptibles d'être transmis en même temps ont été déjà traités plus tôt dans la trame.

L'algorithme répétitif *A'* qui suit a été conçu pour traiter ces situations, en utilisant la version statique de *A* en sous-programme. Après un appel à *A*, l'ordonnancement obtenu est analysé, les paquets transmis pendant les temps morts voient leur rang augmenté d'une position dans la liste des priorités, puis *A* est exécuté à nouveau. Le processus s'arrête quand un ordonnancement satisfaisant est obtenu, ou après un nombre maximal d'essais.

A' élimine beaucoup de temps morts et parvient même à vaincre certains cas serrés non résolus par *A*. En pratique, on exécute d'abord la version dynamique de *A*. *A'* n'est utilisé que si l'ordonnancement obtenu a une durée totale insatisfaisante due à des temps morts excessifs.

```
1.  Algorithme répétitif A'
2.  Début
3.    Calculer la borne B et la liste triée des vecteurs de priorité;
4.    Répéter
5.      Exécuter l'algorithme A dans sa version statique;
6.      Garder l'ordonnancement s'il est améliorant;
7.      Pour chaque temps mort H dans l'ordonnancement faire
8.        Repérer les paquets en cours de transmission pendant H;
9.        Les avancer d'une position dans la liste de priorité
10.     FinPour
11.   Jusqu'à C_MAX = B ou la fin d'un nombre choisi d'essais;
12.   Sortir le meilleur ordonnancement
13. Fin.
```

IV.10 Exemple simple pour A'

Soulignons que A' utilise la version statique de A, c'est-à-dire que l'ordre des priorités des paquets reste le même pendant toute l'exécution de A. L'apport de A' consiste simplement à analyser chaque ordonnancement rendu par A comme dans la figure 3.

Un temps mort (trou) étant constaté, les paquets en cours de transmission pendant le temps mort (paquets 2 et 9 de la figure) sont avancés d'une position dans le classement statique par priorité, puis A est rappelé.

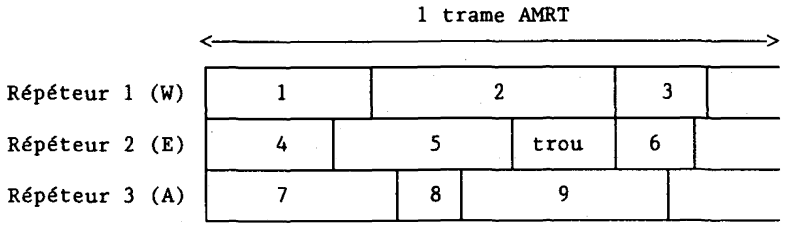

Figure 3. – Analyse d'un ordonnancement de A par A'.

IV.11. Résultats

Pour les premières années d'exploitation du système EUTELSAT (problèmes à 30-80 paquets, 3-4 répéteurs, 16 stations), A et A' trouvent l'optimum : la borne inférieure B est atteinte. Des simulations avec le trafic prévu pour les années 90 (jusqu'à 256 paquets, 7 répéteurs et 32 stations) donnent une performance relative moyenne égale à 1,03 pour A, 1,025 pour A', et 1,08 pour la version statique de A (inutilisée en pratique).

Le pire résultat trouvé sur des données réelles est de 1,16 pour A, et de 1,10 pour A'. A' n'améliore que peu la performance moyenne de A, mais se comporte mieux sur les cas difficiles et donne des ordonnancements comportant moins de temps morts. En pratique, A' n'améliore plus C_{MAX} après 20-40 itérations. Les toutes premières itérations de A' conduisent souvent à un tassement spectaculaire de l'ordonnancement initial.

V. RÉSOLUTION DU PROBLÈME 2

V.1. Préliminaire

Le problème 2 ne se pose pas pour un faisceau desservi par un répéteur unique: toute station émettant vers cette zone, ou en recevant des paquets, aura besoin de toute façon d'un convertisseur. Il est ainsi trivial dans le cas de la figure 1, chaque paquet n'ayant pas d'autre répéteur possible. Le problème d'optimisation apparaît seulement dans les faisceaux à plusieurs répéteurs en parallèle, et peut être résolu séparément faisceau par faisceau.

Pour le cas à deux répéteurs par faisceau, nous avons présenté dans [2] une méthode exacte arborescente, basée sur la déconnexion progressive d'un graphe biparti valué décrivant les interactions stations-paquets. Appliquée à des données réelles (faisceaux de 25-40 paquets ayant de 1 à 8 destinations chacun), cette méthode a toujours donné l'optimum rapidement, après l'exploration de 20 à 60 sommets de l'arborescence. Malheureusement, il semble difficile de l'étendre à plus de deux répéteurs sans une explosion des temps de calcul...

Depuis [2], le cas général a pu être résolu grâce à une *procédure de recuit simulé*. L'origine de ces méthodes en optimisation est l'article classique [5].

V.2. Procédure de recuit simulé

Les m répéteurs sont considérés comme des récipients de taille F (la taille de la trame), auxquels les n paquets doivent être affectés de manière à minimiser le nombre de convertisseurs. Une affectation admissible initiale P est tout d'abord calculée. Dans le cas général, cette recherche initiale est NP-complète, puisqu'elle équivaut à un problème d'ordonnancement à M processeurs (*voir* Garey et Johnson [4]).

Cependant, les données réelles ont peu de très grands paquets, et les répéteurs offrent une capacité totale toujours suffisante: $\sum p_j \leqq m.F$. Des heuristiques connues très simples, comme «FIRST FIT» ou «BEST FIT» pour le problème de «BIN PACKING» n'ont dans ces conditions aucune difficulté à trouver l'affectation initiale. Le coût initial C de P (nombre de convertisseurs) est calculé facilement par comptage des émetteurs et récepteurs actifs dans chaque répéteur.

Un *voisinage de P* est défini par les affectations obtenues à partir de P par deux types de transformations élémentaires: des déplacements et des permutations de paquets entre les répéteurs. Dans ce qui suit, L_k est la charge actuelle du répéteur k.

Déplacement: un paquet i est déplacé de son répéteur actuel u vers un autre v. Une telle transformation est admissible si $L_v + p_i \leqq F$.

Permutation: un paquet i d'un répéteur u est permuté avec un paquet j d'un répéteur v. La permutation est réalisable si $L_u - p_i + p_j \leqq F$ et $L_v - p_j + p_i \leqq F$.

Une *recherche locale* simple (sans recuit) consisterait à effectuer tant que c'est possible des déplacements ou permutations améliorant le nombre de convertisseurs. Une telle recherche locale a été programmée et ses résultats ont été comparés dans le cas $m=2$ avec la méthode arborescente: la recherche locale est en moyenne à 10 % de l'optimum, qu'elle trouve seulement exceptionnellement. Les résultats sont encore pire si seuls des déplacements ou des permutations sont considérés.

Nous décrivons maintenant un schéma de recuit pour améliorer la recherche locale. On fixe une «température» initiale T. En partant de l'affectation initiale P, une transformation admissible (déplacement ou permutation) est tirée au sort, et la variation résultante du coût ΔC est calculée. La transformation est acceptée sans condition si $\Delta C \leqq 0$, sinon, on l'accepte avec une probabilité $\exp(-\Delta C/T)$.

T est diminuée suivant une fonction en escalier: toutes les U transformations, T est remplacé par $k.T$, k étant une constante positive proche de 1. Le recuit est stoppé après un nombre maximal de tirages N, ou quand la probabilité d'accepter une transformation non améliorante devient inférieure à un seuil fixé ε.

V.3. Résultats

Le réglage des paramètres d'un recuit simulé est une tâche délicate, dépendant largement du problème étudié. La référence [6] discute de cette question de manière détaillée.

Dans le cas d'EUTELSAT, le refroidissement doit être mené lentement, avec au moins 2000 tirages. Les meilleurs résultats ont été obtenus avec $U=250$, $T=1$, $N=10000$, $K=0{,}925$ et $\varepsilon = 5.10^{-4}$. La plupart du temps, la valeur définitive du coût est acquise vers 7000 tirages. A chaque étape, on tire au sort des transformations jusqu'à en trouver une qui soit admissible: il est trop coûteux de dresser au préalable la liste des transformations admissibles.

La procédure de recuit retrouve les optimums calculés par notre méthode arborescente exacte, dans le cas $m=2$. Pour plus de deux répéteurs, nous n'avons pas de méthodes exactes pour comparer, ni de bonnes bornes inférieures pour l'optimum, mais le recuit se comporte toujours mieux que plusieurs

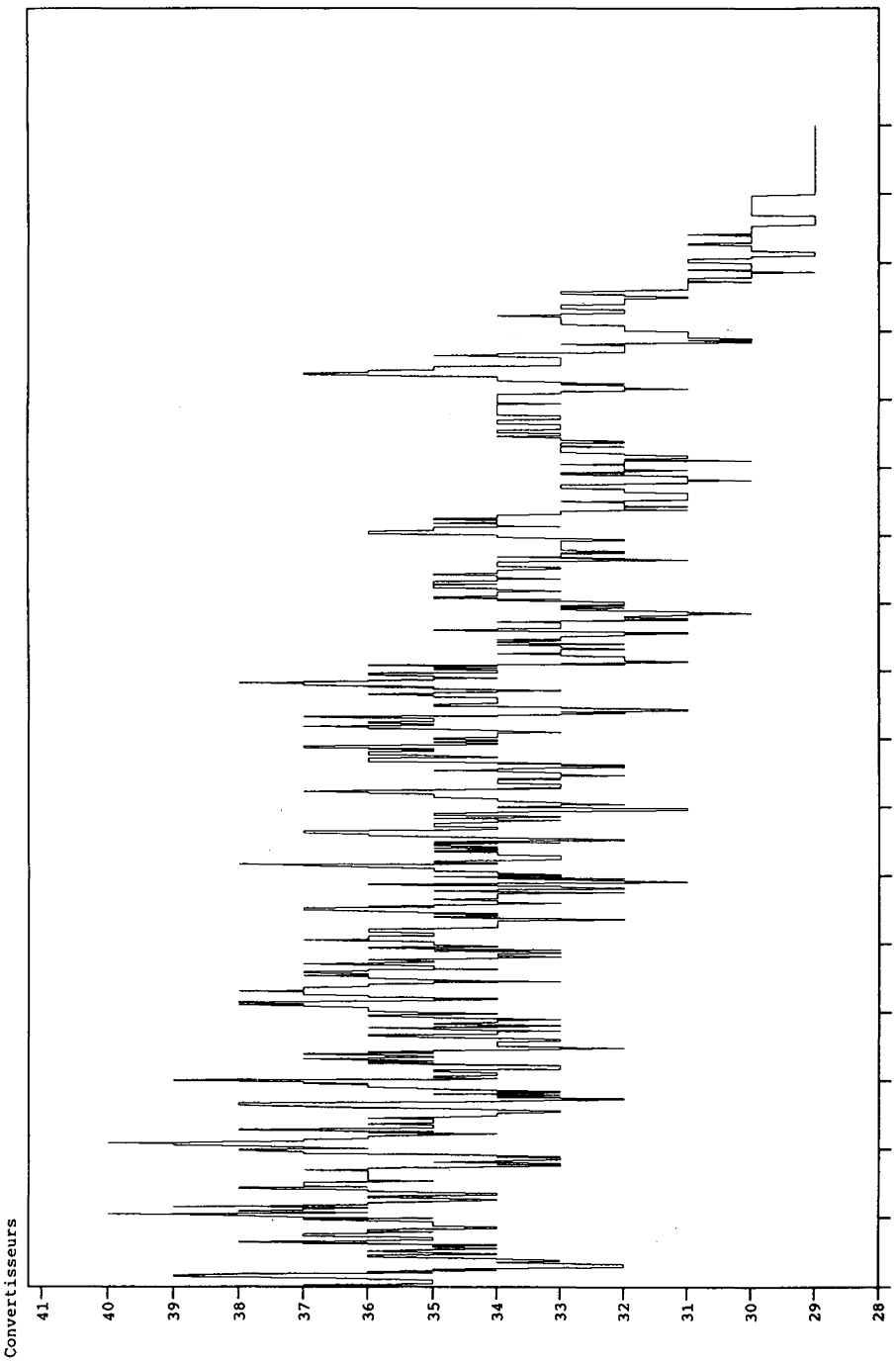

Figure 4. – **Exemple typique de courbe de recuit.** Le problème traité à 2 répéteurs et 27 paquets impliquant 24 émetteurs et récepteurs. Le nombre entier de convertisseurs oscille entre 40 et l'optimum 29, atteint vers 1 500 tirages (seuls les tirages admissibles ont été figurés sur environ 7 000 tirages au total). Dans l'affectation résultante des paquets aux répéteurs, 5 des stations (29 moins 24) interviennent sur les 2 répéteurs et nécessitent 2 convertisseurs.

autres heuristiques essayées sur le problème. Comme aucune de ces heuristiques n'est capable de trouver l'optimum pour $m=2$, nous soupçonnons la méthode de recuit d'être quasi optimale pour $m>2$.

Une méthode de recuit est applicable à un problème quand on peut passer d'une solution réalisable à une autre par des transformations simples et rapides, ce qui est le cas ici. Les déplacements, permutations et variations de coût sont calculables rapidement, au point de pouvoir traiter en 15 minutes sur un PC un problème de 60 à 100 paquets avec $m=3$. Un tracé de la fonction coût sur un écran graphique est assez spectaculaire. La figure 4 est un exemple dans lequel on n'a figuré que les transformations admissibles (environ 1 700) : le nombre réel de tirages approche 7 000.

VI. CONCLUSION

Le système AMRT à satellite de l'organisation EUTELSAT est en service commercial depuis mars 1986, grâce à un logiciel basé sur une version antérieure de nos méthodes [2]. Le perfectionnement apporté par le recuit simulé a permis d'améliorer le logiciel, qui a pu ainsi s'adapter à une augmentation continue du trafic, donnant des problèmes de plus en plus grands (5 répéteurs et 100 paquets au début 90).

Le lecteur avisé aura remarqué que le Problème 1 peut modéliser des problèmes d'atelier avec contraintes de ressources renouvelables. Notre algorithme de liste, qui a prouvé son bon comportement moyen, peut donc être utile dans un domaine ou la grande majorité des problèmes sont NP-difficiles, et pour lequel le manque d'heuristiques efficaces se fait cruellement sentir.

Le problème 2 semble plus spécifique aux télécommunications par satellite. Il peut pourtant servir de modèle à des problèmes de stockage à long terme, avec coût de maintenance minimal.

Dans ces problèmes, les répéteurs deviennent des entrepôts de « capacités » connues, et les paquets, des matériels devant y être stockés pour une longue durée. Un exemple serait le stockage de matériel militaire de réserve en temps de paix. Pour chaque matériel, on donne la « taille » (surface au sol par exemple) et un ensemble d'outils spécialisés requis pour son entretien.

S'agissant d'un entretien préventif, il suffit d'installer un seul exemplaire d'un outil dans un dépôt, qu'il y ait un ou plusieurs matériels en ayant besoin dans le dépôt. Dans notre problème 2, l'ensemble des outils possibles était celui des stations, et un convertisseur de fréquence correspondait à l'installation d'un exemplaire de l'outil dans l'entrepôt.

Le problème général consiste à affecter les matériels aux entrepôts, de façon à respecter la capacité de ces derniers et à minimiser le nombre total d'outils de maintenance à installer... Il s'apparente à certains problèmes de localisation multi-produit avec capacités, mais nous n'avons pas trouvé de travaux publiés le concernant.

BIBLIOGRAPHIE

1. C. PRINS, Problèmes d'optimisation de ressources dans les systèmes de télécommunications par satellite utilisant l'A.M.R.T., *Doctorat de l'Université Paris-VI*, juin 1988.
2. C. PRINS et J. CARLIER, Resource Optimization in a T.D.M.A./D.S.I. System: the EUTELSAT Approach, *7th Int. Conf. on Digital Satellite Communications* (*ICDSC-7*), Munich, mai 1986, p. 511-518.
3. J. CARLIER et C. PRINS, Optimisation des plans de trame dans le système A.M.R.T./C.N.C. d'EUTELSAT, *Ann. Télécommun.*, 1988, *43*, n° 9-10.
4. M. R. GAREY et D. S. JOHNSON, Computers and Intractability: a Guide to the Theory of NP-completeness, *Freeman*, San Francisco, 1979.
5. S. KIRKPATRICK, C. D. GELATT et M. P. VECCHI, Optimization by Simulated Annealing, *Science*, 1983, *22*, p. 671-680.
6. D. S. JOHNSON, C. R. ARAGON, L. A. MCGEOCH et C. SCHEVON, Optimization by Simulated Annealing: an Experimental Evaluation, *Oper. Res.*, 1989, *37*, p. 865-892.
7. R. L. GRAHAM, Bounds on Multiprocessing Time Anomalies, *S.I.A.M. J. Appl. Math.*, 1969, *17*, p. 416-429.
8. M. R. GAREY et R. L. GRAHAM, Bounds for Multiprocessor Scheduling with Resource Constraints, *S.I.A.M. J. on Comp.*, 1975, *4*, p. 187-200.
9. T. INUKAI, An Efficient SS/TDMA Time Slot Assignment Algorithm, *I.E.E.E. Trans. Comm.*, 1979, *27*, p. 1449-1455.
10. G. BONGIOVANNI, D. COPPERSMITH et C. K. WONG, An Optimum Time Slot Assignment Algorithm for a SS/TDMA System with Variable Number of Transponders, *I.E.E.E. Trans. Comm.*, 1981, *29*, p. 1025-1036.
11. C. A. KING, P. TRUSTY, J. JANKOWSKY, R. DUESING et P. ROACH, INTELSAT TDMA/DSI Burst Time Plan Development, *Int. J. Satellite Comm.*, 1985, *3*, p. 35-43.
12. M. MINOUX et C. BROUDER, Models and Algorithms for Optimal Traffic Assignment in SS/TDMA Switching Systems, *Int. J. Satellite Comm.*, 1987, *5*, p. 33-47.