MACHNET un reseau performant pour le haut debit ou les communications longues distances

Philippe Jacquet, Paul Mühlethaler

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél. (1) 39 63 55 11

Rapports de Recherche

N° 684

MACHNET UN RESEAU PERFORMANT POUR LE HAUT DEBIT OU LES COMMUNICATIONS LONGUES DISTANCES

Philippe JACQUET
Paul MÜHLETHALER

JUIN 1987

MACHNET UN RESEAU PERFORMANT POUR LE HAUT DEBIT OU LES COMMUNICATIONS LONGUES DISTANCES.

Philippe Jacquet
Paul Mühlethaler

INRIA
Projet SCORE
Domaine de Voluceau
BP 105 - Rocquencourt
78153 LE CHESNAY CEDEX
FRANCE

Mai 1987

Machnet a simple access protocol for high speed or long haul communications.

Philippe Jacquet and Paul Mühlethaler.

Abstract: This paper introduces a simple access protocol for high speed networks or long haul communications. Based on a tree algorithm with defered collisions detections and resolutions, this protocol includes mechanisms which reduce collisions and increase the throughput. An analytical model and simulations provide performances of this protocol.

Machnet un protocole d'accès simple pour communications haut débit ou longues distances.

Philippe Jacquet et Paul Mühlethaler.

Résumé : Ce papier propose un protocole d'accès simple pour des réseaux haut-débit ou longue distance. Basé sur un algorithme en arbre à détections et résolutions de collisions différés ce protocole comporte des méchanismes qui reduisent les collisions et augmentent le débit. Un modèle analytique et des simulations fournissent les performances du protocole.

Mots clefs :- Protocole d'accès - Collision - Débit - Délai.

Introduction

L'augmentation de la puissance des sytèmes informatiques, le besoin croissant de communication entre ceux-ci montrent le réel besoin de réseaux locaux à haut-débit.

D'une part biensur il s'agit d'étudier de nouvelles technologies centrées surtout sur l'optique pour pouvoir véhiculer sur un canal des informations avec une vitesse de l'ordre du Gbit/s (les réseaux les plus rapides travaillant maintenant à une vitesse de l'ordre de la centaine de Mbit/s), mais aussi semble-t-il nécessaire de proposer de nouveaux algorithmes pour partager de façon efficiente le canal car les protocles classiques existant ne conviennent pas .

L'accès multiple pour des systemes à haut-débit ou pour des réseaux longue distance pose des problèmes voisins; cependant certaines solutions utilisant des topologies particulières sont spécifiques.

Le but de ce rapport est de présenter un protocole qui puisse résoudre ces deux problèmes en cherchant ;

(i) à utiliser convenablement la bande passante allouée.

(ii) à avoir pour les paquets échangés des délais moyens d'attente bien meilleurs que le TDMA classique.

(iii) à garder la flexibilité et la simplicité d'usage pour un réseau local tout en tolérant les fautes.

Le protocole de MACHNET apporte les réponses suivantes à ces problèmes.

(i) il permet l'utilisation de 100 % de la bande passante offerte quelques soient le délai de propagation et le nombre de sites actifs.

(ii) les délais d'attente restent finis et indépendants du nombre de sites actifs tant que le débit est inférieur à 40 %; au dela le système tend vers un TDMA sur les stations actives.

(iii) deux compteurs linéaires (compteur de session, compteur de session en préparation) suffisent pour gérer dynamiquement le protocole, l'insertion d'une nouvelle station sur le réseau est une opération simple et ne dérange pas les autres stations. Les fautes peuvent altérer les performances du système mais rares sont celles qui peuvent durablement les dégrader.

Ce papier ouvre le problème des protocoles pour réseaux haut-débit à résolution de collisions. A notre connaissance rien n'a été publié à ce sujet à ce jour.

Ce rapport comporte trois parties : dans la première, on pose le problème du haut-débit et des longues distances et on précise les notations et hypothèses ensuite utilisées tout au long de la présentation. Dans la seconde, on présente le protocole. Dans la troisième, on analyse ces propriétés et ces performances. En conclusion, on montrera que ce protocole permet d'envisager toute une classe de protocoles parents.

En annexe, se trouvent des résultats techniques et des développements mathématiques dont l'utilité n'est pas immédiatement indispensable à la compréhension du document.

I. Les hypothèses

a. Notations :

On note W la bande passante présente du canal utilisé enbit/s et B le nombre maximum de bits du paquet, r sera pris égal au temps de propagation entre les stations les plus éloignés.

Il est pratique d'introduire la grandeur sans dimension a : rapport de la durée maximale de propagation par la durée maximum du paquet.

$$a = \frac{rW}{B} .$$

On montre [1] que les protocoles classiques (par exemple Ethernet ou le bus à jeton) autorisent une capacité maximale:

$$c = \frac{1}{1 + O(a)} \qquad O(a) \# a .$$

Le cas du haut debit correspond au cas où W est grand celui des longues distances étant caractérisé par r grand. Dans les deux cas la capacité maximum utilisable est faible, ce qui est rédibitoire.

b. Les bases de départ

Dans le cadre des protocoles à résolution de collisions et si l'on veut satisfaire les longs délais de propagation, il est clair qu'il faut envisager un protocole synchrone.

On fera l'hypothèse du canal sloté ; la durée du slot étant celle de la trame la plus longue autorisée. On supposera aussi que pour un paquet émis sur un slot le résultat succès,blanc ou collision n'est connu par un feed-back que b slots après, b étant indépendant de la station. De façon plus précise si un paquet est émis au slot x on supposera que le résultat n'est connu qu'au début du slot b + x (cette information est immédiatement utilisable par un algorithme). Si cela n'est pas le cas on peut toujours s'y ramener en intervenant soit au niveau matériel ou soit au niveau logiciel .

De très nombreuses topologies satisfont à ces hypothèses, citons la topologie en étoile avec branches de même longueur et station centrale détectrice de collisions, les topologies Expressnet avec canal d'écriture et canal de lecture, les communications satellite. *(Voir FIG 1).*

II. Le protocole big. mach.

a. Les deux idées conductrices du protocole :

La première réside dans l'organisation du processus de résolution de collision qui permet grâce à une technique de résolution par arbre binaire et une méthode de multiplexage des ces arbres [2] d'utiliser les slots qui auraient pu être perdus à cause des délais de propagation.

La seconde consiste en un mécanisme original d'allocation qui permet de réduire les collisions lorsque le débit sur le canal augmente et d'accepter sur celui-ci des charges arbitrairement voisines de 100 %.

Les deux paragraphes qui suivent sont consacrés à la présentation de ces deux idées, le troisième rassemble le tout en insistant sur le déroulement temporel de l'algorithme et les contraintes qui en découlent.

b. Le mécanisme de résolution de collision

Le mécanisme conduit à distinguer deux types de slot:

-les slots où des paquets peuvent être émis pour la première fois

-les slots dédiés à la réémission des paquets ayant déjà subi une collision.

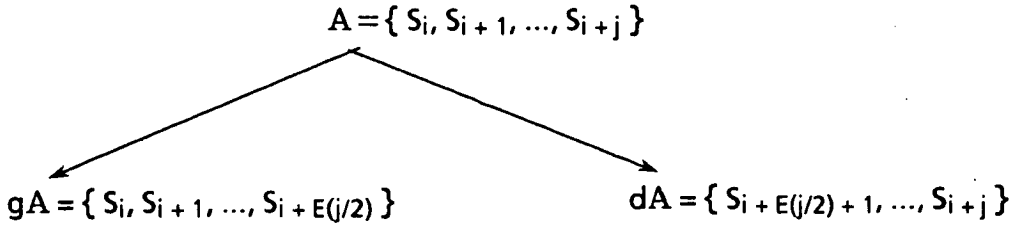
Pour la première catégorie les paquets émis sont tous émis pour la première fois. Dans ce cas on dit qu'on ouvre une session. On appelle session un ensemble de stations ayant tenté la première émission de leur paquet sur un même slot. Plus précisément une session est donc un ensemble de couple de la forme (paquet, station),en sachant que relativement à une session une station impliquée est équivalente à un paquet et réciproquement. On notera la session (M,S) où M est l'ensemble des paquets et S des stations (la relation station paquet étant implicite). On utilisera le mot session indifféremment pour désigner les paquets ou les stations.

Si une session ne comporte qu'un paquet aucune réémission n'est nécessaire. Si au contraire une session a plus de deux stations, le feed back qui arrivera b slots après l'émission des paquets entrés en collision initialisera le processus de réémission. Une session est fermée lorsque l'algorithme de résolution de la collision intiale est terminé. Entre son ouverture et sa fermeture une session est dite en cours.

Il est commode d'introduire la notion d'arbre, si $S_1, \ldots, S_n$ sont les stations connectées on appellera arbre un sous ensemble de stations de la forme $A = \{ S_i, S_{i+1}, \ldots, S_{i+j} \}$ où l'on a $1 \leq i \leq i+j \leq n$. A cet arbre dans le cas où $j \geq 1$, on associe

-le sous arbre gauche $gA = \{ S_i, S_{i+1}, \ldots, S_{i+E(j/2)} \}$

-le sous arbre droit $dA = \{ S_{i+E(j/2)+1}, \ldots, S_{i+j} \}$.



* E(p) désigne la partie entière de p

A chaque slot on associe un couple (session,arbre) par exemple ((M,S),A). Les paquets émis sur ce slot seront les paquets de M dont les stations appartiennent à $S \cap A$. On dira que l'on a émis sur ce slot (M,A).

Le mécanisme de résolution de collisions est géré par une pile dont les éléments sont composés d'un ensemble de paquets et d'un arbre. Soit l'émission sur un slot de (M,A), si plus d'un paquet sont émis sur ce slot, la réception du feed back collision provoque la mise en pile de (M,gA),puis de (M,dA).

A chaque slot et chaque fois que la pile est non vide on dépile l'élément du sommet et on l'émet.Par conséquent l'ouverture de nouvelles sessions nécessite une pile vide.Dans ce cas de nouveaux paquets peuvent-être émis sur le canal et l'arbre de ce slot initial est l'arbre complet $\{S_1, ..., S_n\}$.On peut montrer (*Cf* annexe) qu'à un instant donné b + 1 sessions au plus sont en cours.

Sous forme condensée l'algorithme s'écrit :

```
-Faire à chaque slot :
    Début
    Si la pile est vide
        Alors
            Ouvrir une nouvelle session.
        Sinon
            Dépiler le sommet de la pile puis l'émettre.

    Si le feed-back est une collision provenant d'une émission (M,A)
        Alors
            Début
            Empiler (M,dA) dans la pile de réémission.
            Empiler (M,gA) dans la pile de réémission.
            Fin
    Fin
```

On donne *FIG* 2 le mécanisme de gestion de la pile. *FIG* 3 on suit la résolution d'une collision lorsque huit stations connectées ont toutes un paquet à émettre. Une session est ouverte au slot 0 avec une collision d'ordre 8. Dans la pile on a omis l'ensemble des paquets pour ne figurer que l'arbre ; (1 à 4 signifie { 1, 2, 3, 4 }), les slots 1, 2, 5, 16, 17 pourraient permettre d'ouvrir d'autres sessions dans ce cas l'apparition de nouvelles collisions modifiera la résolution de la collision envisagée et la pile contiendra des paquets de sessions différentes.

L'algorithme précédent est décrit de façon globale, or l'essence d'un protocole d'accès est d'être un algorithme local. Dans ce qui suit on montre comment il est possible de l'exécuter localement. Pour une station on a vu que plusieurs sessions pouvaient être en cours de résolution, donc le mécanisme de résolution de collision va se présenter simplement sous forme de tâches parrallèles,chacune d'elles étant chargée d'un paquet d'une session. A ces tâches parralèles il convient d'en ajouter une autre qui gère l'ouverture de nouvelles sessions.

La tâche "Session"consiste a émettre le paquet puis à attendre le résultat. En cas de succès la tâche se termine sinon il faut réémettre le paquet .Si le tirage de l'index donne 1 la réémission aura lieu le slot suivant sinon on va suivre la position du paquet dans la pile grâce à un compteur de collision c initialisé à 2 à la réception du feed-back. A chaque slot c est incrémenté de 1 à chaque réception de feed-back collision décrémenté de 1 sinon. Le passage de c à 0 provoque l'émission du paquet.

De façon plus formelle écrivons la tâche "Session"

Tâche " Session"

```
-Faire jusqu'à désactivation
    Début
    Emission du paquet sur le slot à venir.
    Attente de b slots.
    Si (feed-back ≠ collision)
        Alors
            désactivation.
        Sinon
            Début
            tirage de l'indexe.
            Si (indexe ≠ 1)
                Alors
                    Début
                    c = 2.
                    Faire à chaque slot tant que c ≠ 1.
                        Début
                        Si (feed-back = collision)
                            Alors
                                c = c + 1.
                            Sinon
                                c = c-1.
                        Fin
                    Fin
                Fin
            Fin
    Fin
```

La tâche "Ouverture de session" consiste à suivre le nombre d'éléments dans pile; quand celui-ci est nul une nouvelle session peut être ouverte.On suit la hauteur de la pile par un compteur s. On suppose que s est initialisé à 0,ensuite à chaque slot si le feed-back reçu est une collision on fait (s = 2 si s = 0 sinon s = s + 1)dans le cas contraire s = max(0,s-1) .Le passage de s à 0 autorise l'ouverture d'une nouvelle session.Résumons:

Tâche "Ouverture de session"

-Faire à chaque slot :

```
    Début
    Si (s = 0)
        Alors
            Activation d'une tâche "Session ".
    Si (feed-back = collision)
        Alors
            Si (s = 0)
                Alors
                s = 2.
                Sinon
                s = s + 1.
        Sinon
            s = max(0,s-1).
    Fin
```

L'algorithme décrit fournit un protocole complet d'accès au médium cependant à forte utilisation du canal la capacité utilisée est égale à: n/(2n-1): on doit en effet visiter tous les noeuds avant d'arriver aux feuilles. Ceci nous conduit donc à introduire un nouveau mécanisme succeptible de réduire le nombre des collisions à forte charge, collisions qui sont pénélisantes pour le débit mais aussi pour les délais ceci d'autant plus que b est grand.

c. Le mécanisme d'allocation

Son but est double: réduire les collisions et permettre de fortes charges sur le réseau (jusqu'à 100 % de la bande passante).

En ce qui concerne la réduction des collisions son mode d' action peut se comparer à celui des protocoles à fenêtre temporelle[3].[4]. Avec le mécanisme précédent lors de l'ouverture d'une session tous les paquets en attente participent à la session à venir. On distribue la population en attente sur plusieurs sessions en préparation. Les sessions en préparation sont les sessions appelées à être ouvertes dans l'avenir. Le mécanisme de création et de peuplement de ces sessions est le suivant. Chaque succès reçu en feed-back donne le signal de la création d'une nouvelle session en préparation appelée session en préparation courante.Cette dernière reçoit désormais les nouveaux paquets. En régime stationnaire il est évident qu'il y a autant de succès que de création de nouvelles sessions. En conséquence le nombre moyen de paquets par session en préparation (donc par sessions ouvertes sur le canal) est de l'ordre de l'unité; ce qui permet effectivement la réduction du nombre des collisions. Pour gérer ce processus il suffit d'un seul compteur dynamique: le compteur de session en préparation courante No. Ce compteur évolue de la manière suivante: on l'incrémente de 1 à chaque succès reçu en feed-back, et on le décrémente de 1 à chaque ouverture de nouvelle session. Les sessions en préparation sont donc numérotées de 1 à No. Quand le compteur de session s passe à 0, on ouvre sur le canal la session en préparation numéro 1 et l'on décrémente les numéros d'ordre des autres sessions en préparation.

Pour permttre de fortes charges on rajoute à ce mécanisme deux nouveaux principes. Le premier principe est un contrôle des arrivées: chaque station n'a le droit de placer qu'un seul paquet sur l'ensemble des sessions en préparation. Le deuxième principe est un mécanisme original de réservation sans déclaration. Chaque fois que les stations reçoivent le feed-back d'un succès par exemple pour l'émission d'un paquet de la station S_i ,la session courante devient réservée à la station S_i, qui sera seule autorisée à y déposer un paquet dans l'avenir. On dit que cette session est réservée pour la station S_i. Une nouvelle session en préparation sera créée pour servir de session courante. On s'inspire donc de la réservation chez le dentiste;deux personnes ayant un rendez vous ne peuvent arriver sur la même tranche horaire mais l'accès chez le dentiste peut se faire sans rendez-vous (un soudain mal de dents), on peut surgir à l'improviste sur une tranche horaire qui a été par ailleurs déjà réservée (collision). Lorsque la charge sur le canal augmente la proportion de paquets passant hors places réservées diminue. A la limite dans chaque session en préparation il n'y aura plus qu'un seul paquet qui occupera sa place réservée, le protocole se comportera comme un TDMA sur les stations actives.

Voici une description détaillée et locale du mécanisme global. Chaque station va devoir gérer trois compteurs:
Le compteur de session en préparation courante No.
Le compteur des places réservées: c'est une fifo PR() contenant des entiers (remplir la fifo signifie ajouter un élément, la vider signifie retirer un élément).
Le compteur de place sur le séquenceur: c'est un couple d'entiers (PP,PD) qui servent à placer le premier paquet en attente de la station, dans les sessions en préparation. PP peut s'interpréter comme la place provisoire du paquet, et PD comme la place définitive.

L'algorithme d'allocation peut se séparer en quatre parties.La première gère l'émission des paquets.La deuxième exploite le feed-back. La troisième maintient la cohérence des compteurs. Enfin la quatrième prépare les messages à émettre.Ces différentes tâches se répètent à chaque slot.

La première partie gère la mise en session du premier message de la file d'attente de la station. Pour cela elle utilise le compteur de place sur le séquenceur. Si l'on peut ouvrir une session (s = 0) et que PP = 1 ou PD = 1 le paquet est émis avec l'ouverture de la nouvelle session. Sinon aucune action n'est exécutée.

Dans la deuxième partie on s'occupe d'exploiter le feed-back:

Si c'est un blanc ou une collision aucune instruction n'est à exécuter.

Si le feed-back est le succès d'un paquet d'une autre station le compteur de session en préparation est incrémenté de 1.

Si le feed-back est le succès d'un paquet émanant de la station le compteur de session en préparation est mis dans la fifo puis il est incrémenté de 1.

Ensuite dans la troisième on gère la cohérence des compteurs si s = 0 une session a été ouverte, il faudra donc décrémenter de 1 tous les compteurs ainsi que les éléments de la fifo des places réservées. (Pour éviter les effets de bord on ne décrémentera pas les valeurs nulles)

Enfin dans la quatrième il s'agit de préparer l'émission du premier paquet en file d'attente lui attribuant une place provisoire ou une définitive.

Si PD>0 le message a déjà été placé dans ce cas aucune instruction n'est à exécuter.

Si PD = 0 et PP>0 le paquet a reçu une place provisoire on va chercher si la fifo des places réservées contient au moins un élément dans ce cas il sera vidé dans PD.

Si PP = PD = 0 le paquet en question n'a pas encore reçu de place on va lui donner la place provisoire corresepondant à l'indice de la session courante PP = No.

Ecrivons la tâche "Allocation pour S_i "

Tâche "Allocation pour S_i "

Faire à chaque slot

```
Tâche "Mise en session"
Début
Si (s = 0 et (PP = 1 ou PD = 1))
        Alors
                Mise en session du premier message de la file d'attente et
                activation d'une tâche "Session".
Fin
```

```
Tâche"Exploitation du feed back"
Début
Si (feed-back = succès pour Sj avec j≠ i)
        Alors
                No = No + 1.
Si (feed-back = succès pour Si)
        Alors
                Début
                        No est mis dans la fifo PR().
                        No = No + 1.
                Fin
Fin
```

```
Tâche "Cohérence des compteurs"
Début
Si (s = 0)
        Alors
                Début
                        PD = sup(PD-1,0).
                        PP = sup(PP-1,0).
                        No = sup(No-1,1).
                        PR() = PR()-1.(signifie que si la fifo est non vide on
                        décrémente ses éléments de 1)
                Fin
Fin
Tâche "Placement du paquet sur le séquenceur"
Début
Si (PD = 0 et PP > 0 et un message au moins est en attente)
        Alors
                Début
                        Si (la fifo PR() est non vide)
                        Le premier élément de PR() est vidé dans PD.
                Sinon
                        PP = No.
        Fin

Si (PP = PD = 0)
        Alors
                PP = No.
Fin
```

c .Synthèse et déroulement glogal de l'algorithme

Il nous semble important d'insister sur le déroulement global du protocole et plus particulièrement sur les tâches qui peuvent s'exécuter en parrallèle ainsi que sur les contraintes de timing.

On peut distinguer trois tâches parrallèles essentielles, pour expliciter le fait que ces tâches s'effectue en parrallèle nous utiliserons le constructeur PAR pour décrire des tâches s'exécutant en séquence on prendra le constructeur SEQ. (constructeurs empruntés à OCCAM [7])
Avant de pouvoir écrire l'algorithme, il est nécessaire d'introduire un tâche "Compteur de session" que l'on écrit comme suit:

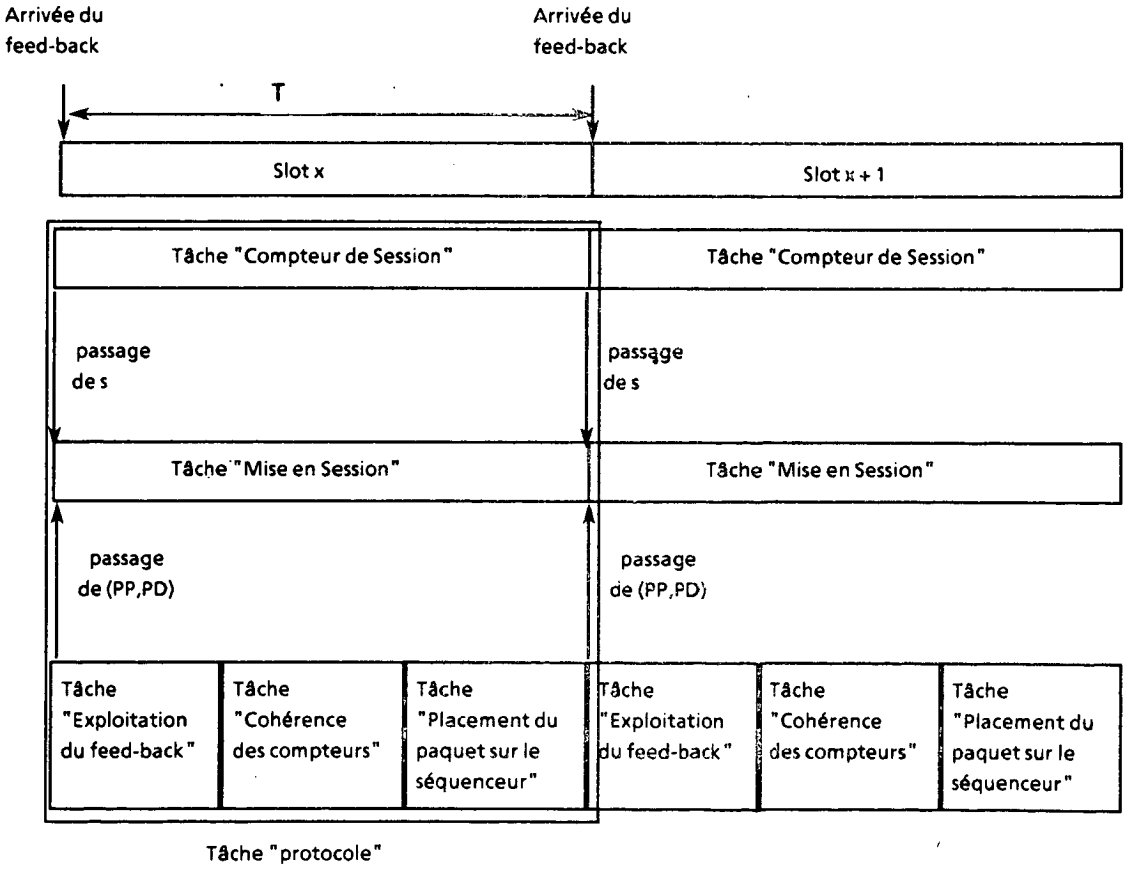
```
Tâche "Compteur de session"
    Début
    Si réception d'une réinitialisation s = 0.
    Si (feed-back = collision)
            Alors
                    Si (s = 0)
                            Alors
                                    s = 2.
                            Sinon
                                    s = s + 1.
            Sinon
                    s = max(0,s-1).
    Fin
```

Le protocole sécrit alors:

Faire à chaque slot

```
Début
PAR
        Début
        Tâche "Compteur de session"
        Tâche "Mise en session"
        SEQ
                Début
                Tâche "Exploitation du feed-back"
                Tâche "Cohérence des compteurs"
                Tâche "Placement du paquet sur le séquenceur"
                Fin
        Fin
Fin
```

Le schéma suivant donne le mécanisme du protocole avec les différents échanges entre les tâches.



Les données calculées par les différentes tâches décrites sont transmises à la fin de chaque slot. La tâche "Compteur de session" génère des tâches "Session" qui vont jusqu'à leur désactivation être indépendantes de l'algorithme général, on considère qu'elles se déroulent en parrallèle.

Notons T1 = temps d'exécution de la tâche "Compteur de session".
T2 = temps d'exécution de la tâche "Mise en session".
T3 = temps d'exécution de la tâche "Exploitation du feed-back".
T4 = temps d'exécution de la tâche "Cohérence des compteurs".
T5 = temps d'exécution de la tâche "Placement du paquet sur le séquenceur".
T6 = temps d'exécution à chaque slot de la tâche "Session".
T = la durée du slot.

Si l'on tient compte du parrallélisme indiqué la contrainte de timing s'écrit:

$$Sup(T1,T2,T3 + T4 + T5,T6) < T$$

Si l'on ne tient compte que de l'exécution en parrallèle que des tâches "Session":

$$Sup(T1 + T2 + T3 + T4 + T5,T6) < T$$

Si aucune des tâches précédentes ne se déroulent de façon parrallèle et étant donné qu'une station n'a jamais à gérer plus de b + 1 sessions ouvertes à un même instant, la contrainte de timing devient:

$$T1 + T2 + T3 + T4 + T5 + (b + 1)T6 < T$$

III. Propriétés

a.Le protocole de Machnet est déterministe.

La démonstation repose sur les paragraphes a,b,c,d,e,f de l'appendice. On arrive à la conclusion suivante: un message doit atttendre un nombre de slots inférieur à $4(b + 2)(b + 1)n^2 + E(\log_2(n)) + 1)(2n + 1)(b + 1) = O(b^2n^2)$ pour passer avec succès sur le canal.

b.Le protocole de Machnet permet d'assurer un transparence canal de 40%.

Voir paragraphe g de l'appendice. Ceci signifie que jusqu'à concurrence d'une charge d'entrée de 40% les délais d'acheminement restent fini indépendament du nombre de stations actives. Et en conséquence les phénomènes de file d'attente disparaissent à mesure quue l'on considère des populations actives de plus en plus importantes.

c.Le protocole de Machnet peut atteindre une capacité maximale indépendament voisine de 100 %
Voir paragraphe h de l'appendice.

d.Les propriétés de transparence canal font que tout trafic est débité à concurrence de la transparence canal: ie 40%. Le réseau est équivalent d'une certaine manière à un réseau complet point à point (avec des arètes de taille variable)

e.Les délais

Pour un paquet donné on va distinguer pour celui-ci trois types de délai:

-le délai en processing qui sépare la première émssion du paquet (de l'émission finale(donc avec succès)

-le délai en séquenceur qui sépare pour un paquet son attribution d'une place de sa mise en session

-le délai en file qui sépare pour un paquet son arrivée en station de son attribution d'une place sur le séquenceur.

Pour un paquet le délai d'attente est la somme de ces délais.

Nous avons testé Machnet par simulation. Le programme est écrit en C + + et utilise les facilités de Sphinx[5].

La première série de simulations correspond au cas où 50 stations sont connectées et le traffic est poissonien uniformément réparti sur toutes les stations. On étudie l'influence du délai de propagation b, les délais moyens ainsi que la façon dont la charge d'entrée sur le canal est écoulée.

On donne sur les *FIG.*4 à 7 des résultats de simulation avec différentes valeurs pour b: 2, 10, 20, 30, 40.
On note que le délai d'accès moyen d'accès est très bon à faible charge(inférieure à 30 %). Celui-ci augmente avec b; à faible charge l'augmentation est régulière avec b. Au voisinage de la saturation le phénomène est moins prononcé.

Le délai moyen en processing passe par un maximum pour une charge d'environ 40% ceci de façon indépendante de b. La délai moyen en processing dépend homothétiquement de b; c'est bien ce que l'on constate avec une bonne approximation.
Le délai moyen en séquenceur reste négligeable jusqu'à 30%, ensuite on tend par valeurs supérieures vers un TDMA sur les stations actives (toutes en l'occurence).
Comme le laissait prévoir le modèle analytique il n'y a pas d'attente en file pour une charge inférieure à 40 %.
La *FIG.*8 illustre le fait que c'est l'attente en séquenceur qui engendre la file d'attente.

Le modèle analytique permet aussi de prévoir le pourcentage de paquets passant sur places réservées. Les hypothèses faites dans le modèle conduisent à ne pas prendre en compte le délai de propagation. *FIG.*9 on vérifie que c'est une assez bonne approximation puisque seule la courbe b = 2 se distingue des quatre autres qui correspondent au cas où b = 10,20,30,40. *FIG.* 10 on compare les simulations avec les résultats du modèle. La concordance est bonne.

On se propose dans ce qui suit de montrer que le protocole de Machnet permet de gérer tout type de trafic.

On présente d'abord un réseau où 50 stations sont connectées; On choisit de répartir une charge totale poissonienne de 100% entre deux ensembles de stations. On considèrera les cas suivants:
-1er cas les ensembles sont:{1,2} et {3,4,5,...,50},
-2ème cas les ensembles sont:{1,26} et {2,3,4,...,25}U{26,27,28,...,50},
-3ème cas les ensembles sont:{1,2,3,...,25} et {26,27,28,...,50}
Sur ces deux ensembles de stations la charge est uniformément répartie. Les FIG 11 et 12 donnent le pourcentage de bande passante effectivement utilisée dans les différents cas suivant la charge supportée par le premier ensemble. Les résultats sont comparés avec les performances d'un protocole TDMA.
Le même type d'étude est présenté FIG 13 pour le cas d'un réseau à huit stations. On prendra les cas suivants:
-1er cas les ensembles sont:{1,2} et {3,4,5,...,8},
-2ème cas les ensembles sont:{1,5} et {2,3,4}U{6,7,8}.

f.Tolérances aux fautes.

Si la cohérence des compteurs de session ou de session en préparation n'est pas assurée, les performances du protocole sont altérées. La non cohérence du compteur de session en préparation ne supprime pas le déterminisme du protocole en arbre. Par contre pour le compteur de session cela peut entrainer des collisions aux feuilles qui devront être signalées et traitées par des couches de logiciel plus hautes.
Pour palier à de telles défaillances on peut utiliser un champs dans chaque paquet envoyé qui indiqueront la valeur des comptuers. Une tache "Cohérence compteurs" qui suit le canal peut en lisant ce champs et en prenant en compte les b slots précédents en déduire la valeur actuelle des compteurs de session. Si trop "souvent" une station constate des écarts avec ces valeurs loales elle pourra réinitialiser ses compteurs et si ces écarts persistent se déconnecter.

g.Insertion d'une station.

Une station inactivepeut devenir active en initialisant ses compteurs grâce à la tâche "Cohérence compteurs". Il lui faudra au moins b slots pour ce faire. Une fois les compteurs de session initialisés, la tâche "Protocole" peut être activée; l'insertion n'aura pas génée les autres stations.Pour le TDMA une nouvellle insertion consiste à redimensionner la trame périodique d'allocation des slots.

IV Conclusion

Machnet se présente comme un protocole déterministe possédant toutes les bonnes propriétés d'un protocole probabiliste sans en avoir les inconvénients. Il permet de très bon délai moyen d'accès jusqu'à 40 %, autorise une utilisation quasi optimale de la bande passante à forte charge.Machnet peut en outre accueillir tout type de trafic et s'adapter à des topologies variées.
Machnet est le prototype de toute une panoplie de protocoles dérivés. En effet tous les variantes du protocole en arbre pourraient être utilisées et fourniraient autant de protocoles dérivés de Machnet. Le système d'allocation pourrait lui aussi être modifié, en utilisant par exemple un protocole à fenêtre ou encore un méchanisme de réservation avec déclaration. A noter pour finir que le méchanisme d'allocation décrit dans cet article et qui est à notre connaissance original pourrait être utilisé sans l'hypothèse du haut-débit en fournissant des résultats analogues.
On constate une assez bonne concordance entre les résultats des simulations et du modèle analytique.

Références:

[1] F Tobagi, F Borgonovo, L Fratta, "Expressnet: a high-performance integrated-services local area network" IEEE Journal on selected areas in communications,vol.sac-1,No 5, November 1983, pp. 898-913 .
[2] John I. Capetanakis. Generalized TDMA: The multi-Accessing Tree Protocol. IEEE Transactions on communications,vol.Com-27,No 10, October 1979.
[3]Philippe Jacquet. The part and try algorithm adapted to free channel access. Rapport de recherche INRIA No 436. Aout 1985
[4] Dimitri Bertsekas/ Robert Gallager. Data Networks pp 429 439.
[5] L Kleinrock " Queueing Systems "Vol 1 Wiley New-York 1975.
[6] T Billoir, P Sabarthes, "A highly flexible performance evaluation tool for local area networks" Advanced Seminar on Real-Time Local Area Networks .Bandol, France organized by INRIA and sponsored by the US Army European Research Office. April 16-18, 1986 pp. 207-223.
[7] The OCCAM programming manual Inmos. 1982

Appendix

a.Démonstration du théorème qui affirme qu'au plus b + 1 sessions sont en cours.

Remarquons pour commencer qu'une session se termine lorsque le nombre de succès ou de blancs pour l'émission des paquets de cette session est égal au nombre de collisions plus 1.
Supposons qu'avant le slot 0 aucun paquet n'ait été émis sur le réseau. Par conséquent sur les b + 1 slots du slot 0 au slot b, b + 1 sessions peuvent être ouvertes. Pour qu'une session supplémentaire soit ouverte, il est nécessaire que la pile de réémission se vide. Supposons que celle-ci reste vide pendant a slot. Le nombre total des collisions doit être alors inférieur de a unités au nombre des succès ou blancs. Ceci implique que pour au moins a sessions en cours de résolution le nombre des succès ou blancs est supérieur d'une unité au nombre des collisions, sinon un raisonnement par l'absurde amènerait une contradiction. Par suite au moins a sessions sont fermées et pendant les a slots où la pile reste vide au plus a sessions peuvent être réouvertes. Donc a chaque instant b + 1 sessions au maximum sont en cours.

b.Démonstration du théorème qui affirme qu'une station a au plus b + 1 places réservées.

Pour une station et un slot t donnés on appelle EPR(t) le nombre de paquets de la station en question en processing à l'instant t envoyés sur places réservées,EHR(t) le nombre de paquets de la station en question en processing à l'instant t envoyés hors places réservées On appelle PR(t) le nombre de places réservées pour la station donnée à l'instant t. On pose PPR(t) = EPR(t) + PR(t). PPR(t) n'augmente que si un paquet passe avec succès ayant été envoyé hors place réservée. Considérons le dernier envoi d'un paquet par notre station hors place réservée. A ce slot t-b nécessairement il n'y a pas de places réservées ie PR(t-b) = 0 et PPR(t-b) = EPR(t-b) PPR augmente jusqu'au slot t au maximum du nombre de paquets envoyés hors place réservée dont le feed-back du succès a été reçu entre t-b et t. Hors ceux ci sont nécessairement en processing au slot t-b par conséquent $PPR(t) \leq PPR(t-b) + EHR(t-b)$ or PPR(t-b) = EPR(t-b) par suite $PPR(t) \leq EPR(t-b) + EHR(t-b) \leq b + 1$. On obtient finallement $PPR(t) = EPR(t) + PR(t) \leq b + 1$ soit $PR(t) \leq b + 1$, c'est ce qu'on voulait démontrer.

c.Démonstration du théorème qui donne la longueur maximum du séquenceur.

On appelle séquenceur l'ensemble des sessions en préparation. On se propose de démontrer que si l'on prend un séquenceur quelconque que l'on photographie à l'instant t le séquenceur issu lorsque le précédent aura été complètement consommé est de taille bornée par une borne indépendante du séquenceur initial.
Chaque station a au plus b + 1 places réservées par suite il y aura au plus (b + 1)n paquets passant sur places réservées lors de la consommation du séquenceur. Pendant le même temps chaque station envoie au plus 2 paquets hors places réservées.(en général celle-ci n'en enverra qu'un seul sauf dans le cas tous les paquets envoyés avant celui-ci n'ont engendrés aucun succès dans ce cas le prochain paquet envoyé hors place réservée le sera sur la dernière session du séquenceur initial). Par conséquent il y a au plus 2n paquets passant hors places réservées. Il y a en processing au plus (b + 1)n paquets. Le nombre total de paquets envoyés pendant le déroulement du séquenceur initial est donc inférieur à (b + 1)n + 2n + (b + 1)n = (2b + 4)n par suite la longueur du séquenceur issu est inférieure à (2b + 4)n.

d.Démonstration du théorème qui donne le nombre maximum de slots entre deux sessions successives.

Soit au slot t l'ouverture d'une session au slot t + 1, il y a en processing au plus b + 1 sessions par conséquent l'ouverture d'une nouvelle session se fera au plus tard (b + 1)(2n + 1) slots après. L'intersession est donc inférieure à (b + 1)(2n + 1) slots

e.Théorème qui donne la durée maximum en processing d'un paquet.

Un paquet envoyé pour la première fois au slot t subit au plus $E(\log_2(n)) + 1$ collisions. Deux tentatives successives sont séparées au plus par une intersession. Par suite la durée maximum en processing d'un paquet est: $E(\log_2(n)) + 1)(2n + 1)(b + 1)$.

f.Démonstration du déterminisme du protocole.

Supposons qu'au slot t un paquet arrive dans une station. D'après le fonctionnement du système de réservation au pire ce paquet passera en processing à la fin du séquenceur issu du séquenceur photographié à l'instant t. Au pire il y aura $(4b + 8)n$ sessions ouvertes avant sa première émission. Par conséquent celui-ci peut attendre au maximum $4(b + 2)(b + 1)n^2$ slots plus le nombre de slots nécessaires avant d'être émis pour la première fois. Pour passer il lui faudra encore attendre au pire le temps d'un processing paquet.
Le nombre de slots à attendre pour ce message est inférieur à: $4(b + 2)(b + 1)n^2 + E(\log_2(n)) + 1)(2n + 1)(b + 1)$.

g. Modèle analytique, transparence canal.

α.Rappel sur les protocoles en arbre.
Le protocole de Machnet utilise le protocole de résolution en arbre de Capetanakis. Soit L_n le nombre moyen de slots constituant une session contenant n messsages. On sait qu'on a les évaluations suivantes:

n	L_n
0	1
1	1
2	5
3	7,666
4	10,523
5	13,419
6	16,313
7	19,2
8	22,085
9	24,969
10	27,853
11	30,738
12	33,623
15	42,281
20	56,707

On pose:

$$L(z) = \sum_{n \geq 0} L_n \frac{z^n}{n!} e^{-z}.$$

L (z) est la série génératrice exponentielle de L_n. C'est aussi la longueur moyenne des sessions (en nombre de slots) lorsque l'arrivée des messages suit une loi de Poisson de paramètre z.

β.Evaluation de processus de réservation.

Soit y la probabilité pour un paquet de passer sur une place réservée. Comme il y a autant de paquets passés que de session en préparation, y est aussi la probabilité pour qu'une station contienne une place réservée. Soit x + y = 1 x étant la probabilité pour qu'un paquet passe hors place réservée.
Supposons maintenant que les succès soient distribués suivant une loi géométrique de raison 1-λ; la probabilité pour qu'un slot sépare deux succès est λ; que deux slots séparent deux succès λ(1-λ) de façon générale k slots séparent deux succès avec une probabilité λ(1-λ) $^{k-1}$. La moyenne des intersuccès ou inter-arrivée est par conséquent 1/λ.
Une session aléatoire contient avec une probabilité y une place réservée effectivement occupée et avec une probabilité 1-y une place réservée non occupée. D'autre part les arivéeshors place reservées sont des mélanges de lois poissoniennes de paramètre kλx et de facteur de pondération λ(1-λ) $^{k-1}$ puisque le taux d'arrivée hors place réservée est de kλ paquets par slot.
Donc la longueur moyenne d'une session aléatoire est:

$$\Lambda(\lambda,y)=\sum_{k\geq 1}\left(xL(k\lambda x)+y(L(k\lambda x)+L'(k\lambda x))\right)\lambda(1-\lambda)^{k-1},$$

en sachant que

$$L(z)+L'(z)=\sum_{n\geq 0}L_{n+1}\frac{z^n}{n!}e^{-z},$$

il vient:

$$\Lambda(\lambda,y)=\sum_{k\geq 1}\left(L(k\lambda x)+yL'(k\lambda x)\right)\lambda(1-\lambda)^{k-1}.$$

L'équaton L(λ,z) = 1/λ exprime le fait que la longueur moyenne d'une session est égale à la durée moyenne interarrivée compte tenu du système de controle des arrivées.
La résolution numérique de cette équation d'inconnue y et de paramètre p donne la courbe "y en fonction de λ" que nous donnons FIG 10. Notons le fait que lorsque y tend vers 0, y tend vers 0,4 (exactement 0,392) ce qui donne le seuil de transparence canal.

Application: le nombre de collisions et de délai en processing. Soit C_n le nombre moyen de collisions.messages d'une session contenant n paquets. On a les évaluations

Dans le cadre du processus de réservation, le cumul pour une session du nombre de collisions pour tous les paquets: Ξ (λ) est:

n	C_n
0	0
1	0
2	4
3	8
4	12,571
5	17,524
6	22,765
7	28,244
8	33,927
9	39,790
10	45,813
15	77,865

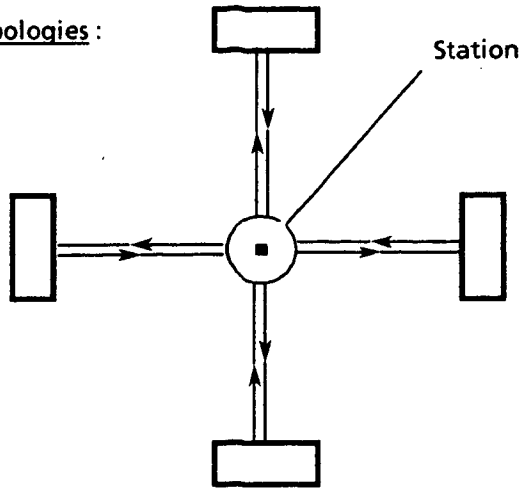
$$\Xi(\lambda) = \sum_{k \geq 1} (C(kx) + yC'(k\lambda x))\,\lambda\,(1-\lambda)^{k-1}.$$

Comme il y a en moyenne un paquet par session, $\Xi(\lambda)$ est donc le nombre moyen de collisions par paquet; $(b+1) \times \Xi(\lambda)$ donne une évaluation du délai en processing quand $0{,}4 \leq \lambda \leq 1$. Voir *FIG* 14 pour une comparaison entre les résultats des simulations et du modèle anaytique.

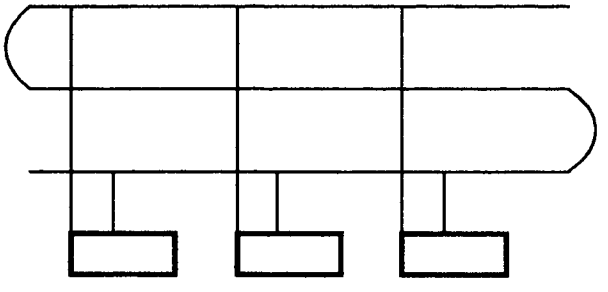
h.Débit à saturation.

Dans une station saturée, tous les messages sauf un nombre fini, passent sur place réservée.Si on pose NR le nombre de places réservées, MP le nombre de messages passés avec succès, MPR le nombre de messages passés sur place réservée. On a NR = MP du fait du processus de réservation. La saturation fait que toutes les places réservées sont utilisées. NR = MPR + le nombre de paquets en processing placés en place réservée. Donc $NR \leq MPR + b + 1$, finallement on obtient $MPR \geq MPR-(b+1)$. Donc au plus b + 1 messages passent hors places réservées.
Si chaque station est saturée tous les paquets sauf un nombre fini passent sur places réservées, donc on tend vers une bande passante effectivement utilisée de 100 %.

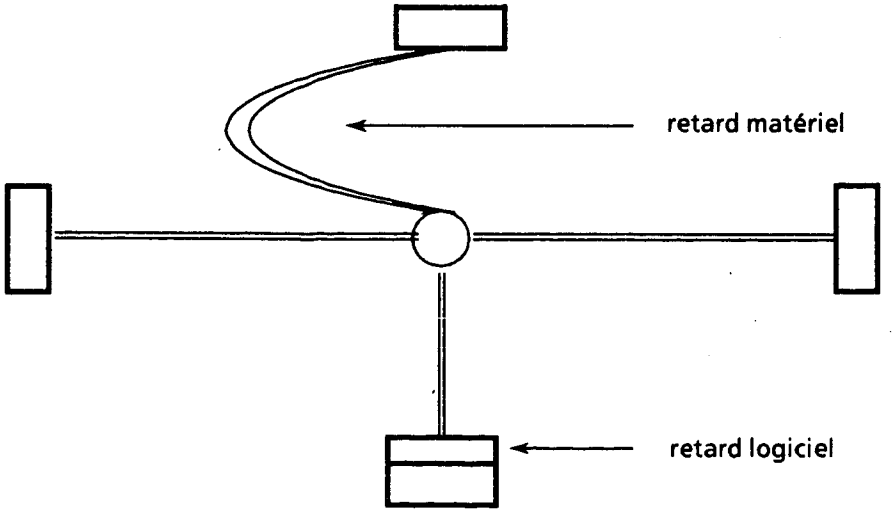
Les topologies :



- en "étoile" avec branches de même longueur



- expressnet en S



On peut se ramener au cas ou toutes les stations reçoivent le feed back sur le même slot.

FIG 1 Différentes topologies vérifiant les hypothèses du I b

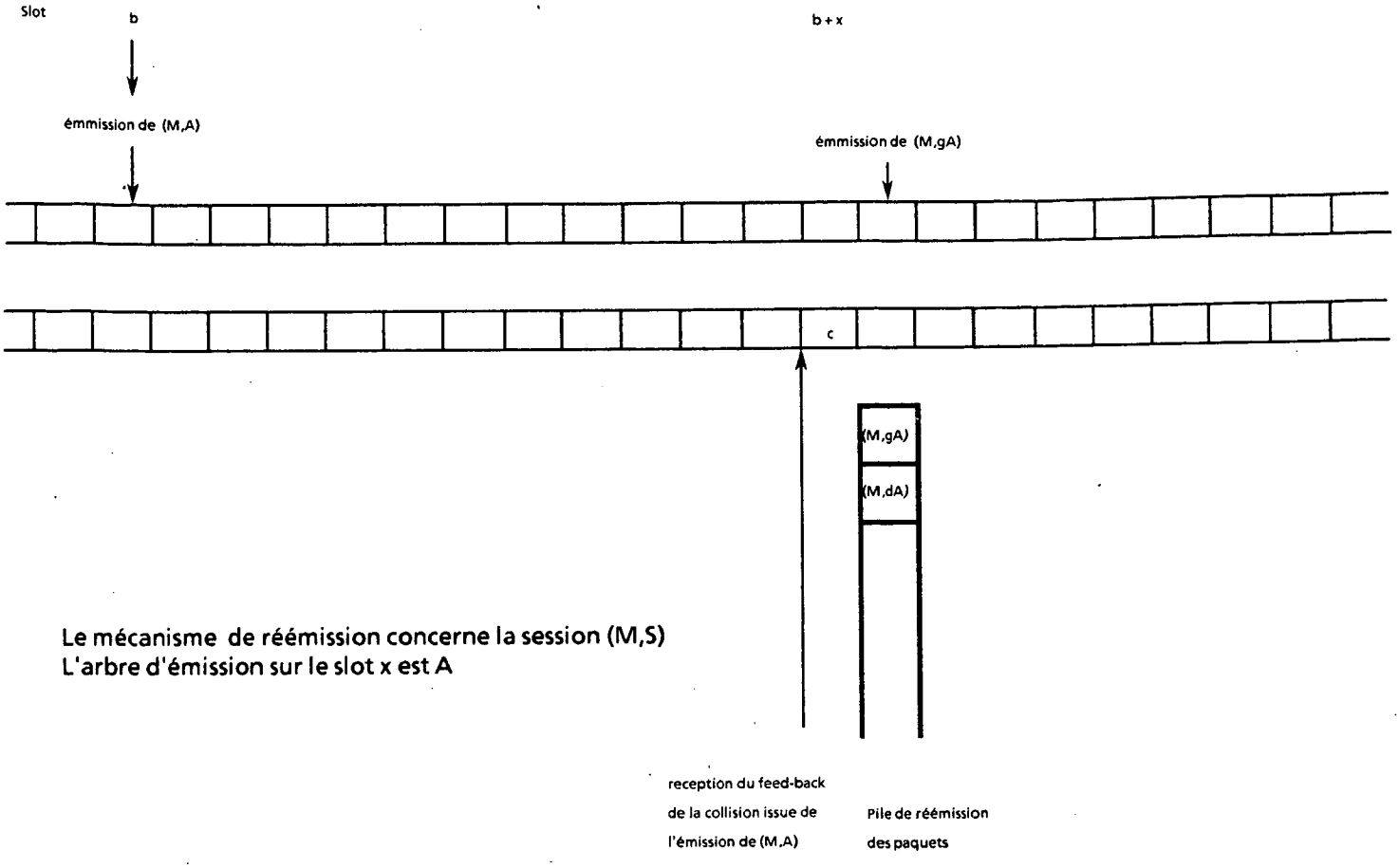
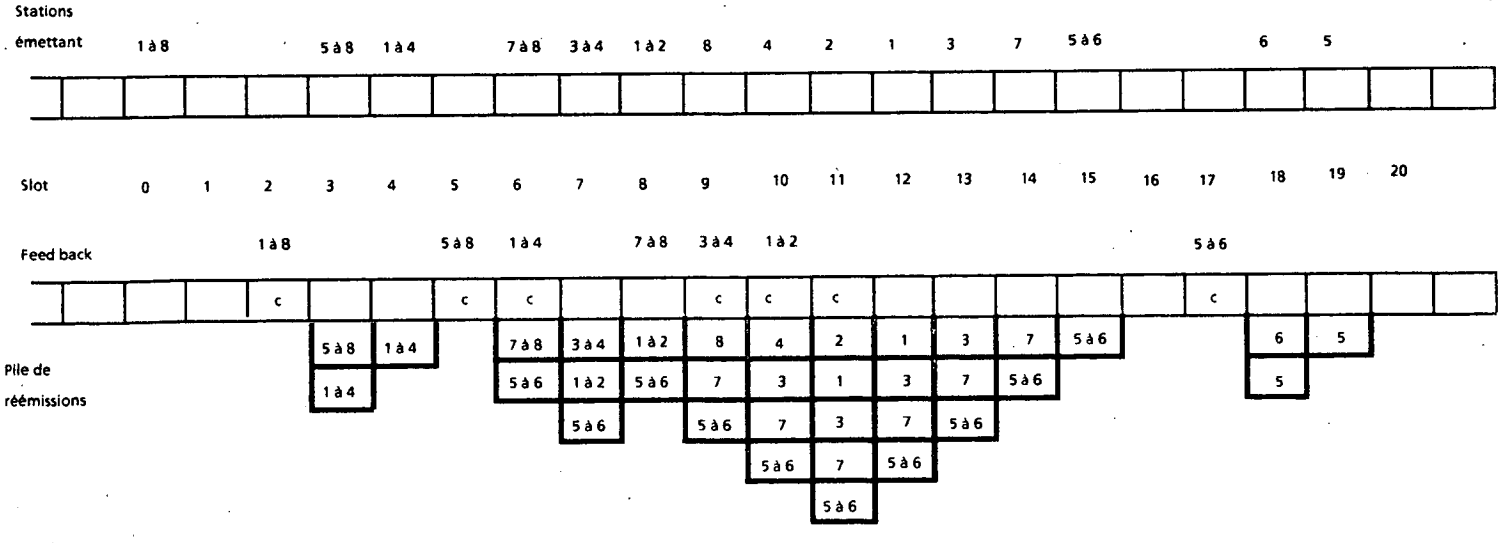


FIG 2. Gestion des réémissions par la pile

Session où toutes les stations connectées, S1, ... ,S8, sont impliquées.
b = 2



On a figuré : - le canal slotté, au-dessus de chaque slot est noté le numéro des stations émettrice
- le feed-back : C signifiant collision, on rappelle également les stations entrées en collision.
- la pile qui gère les réémissions sur le canal.

Aux slots 1, 2, 5, 16, 17 on peut ouvrir de nouvelles sessions.

FIG 3. Exemple de déroulement de session

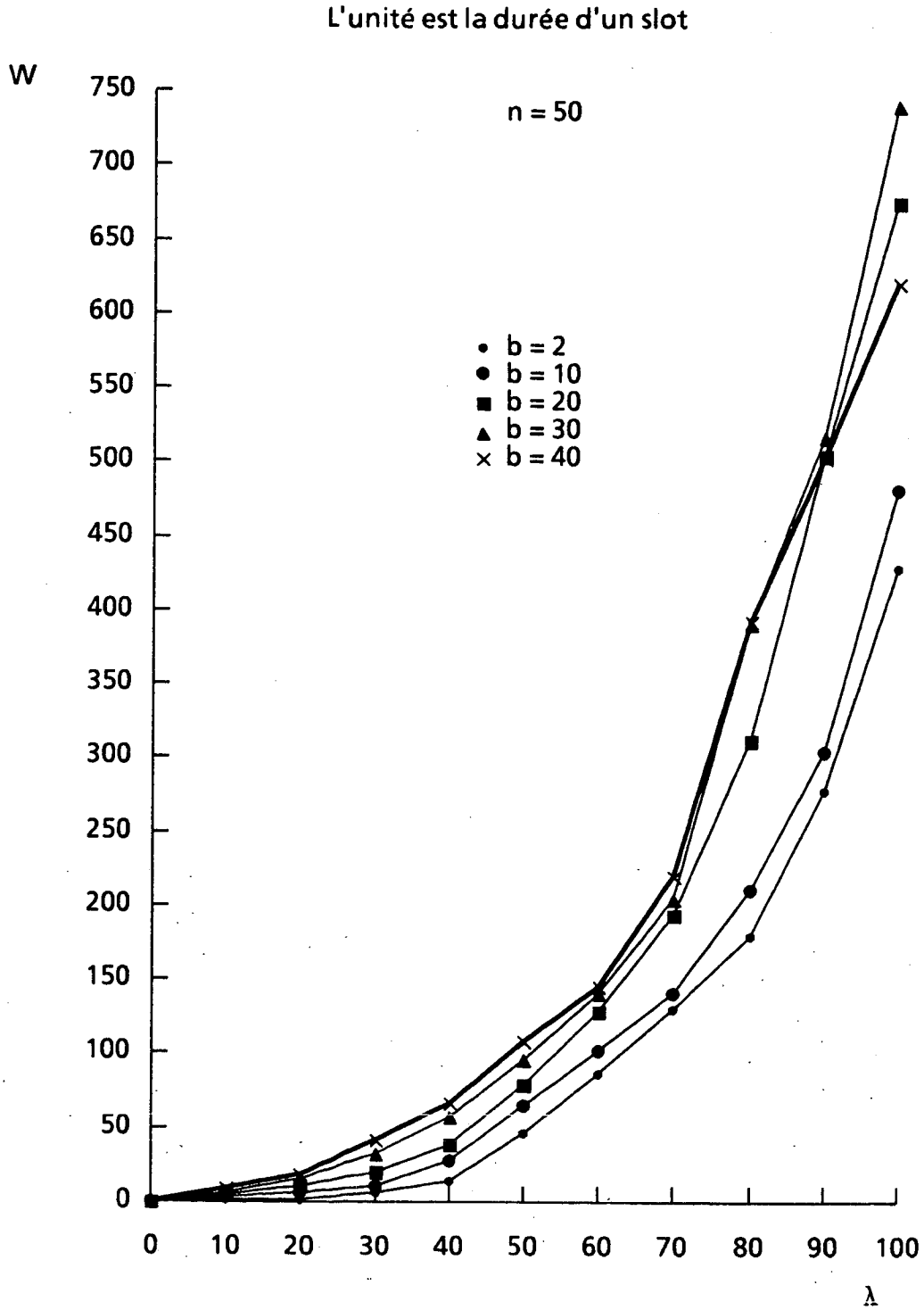


FIG 4 Delai moyen d'accès en fonction de la charge d'entrée λ du canal

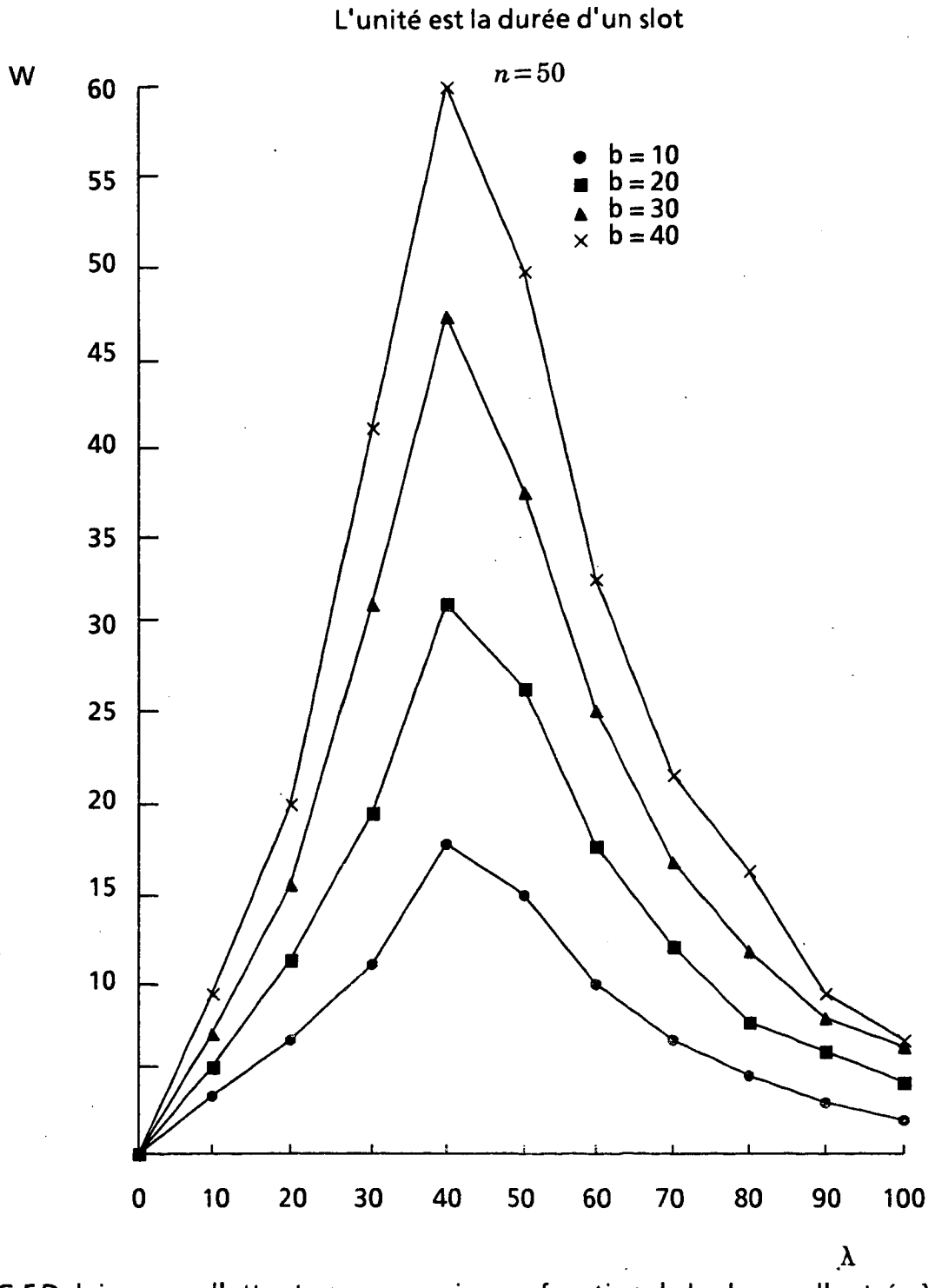


FIG.5 Delai moyen d'attente en processing en fonction de la charge d'entrée λ.

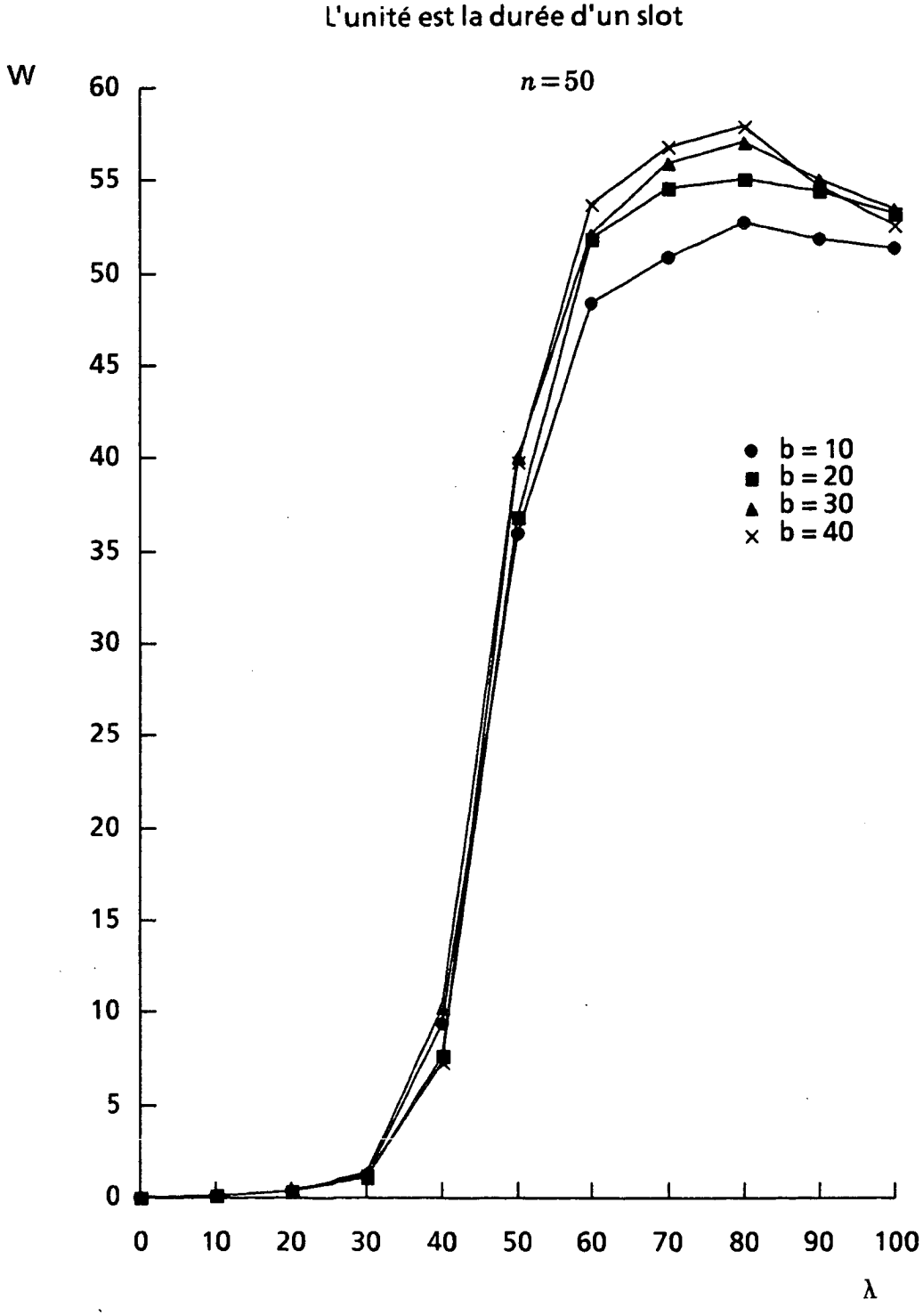


FIG.6. Délai moyen d'attente en séquenceur en fonction de la charge d'entrée λ.

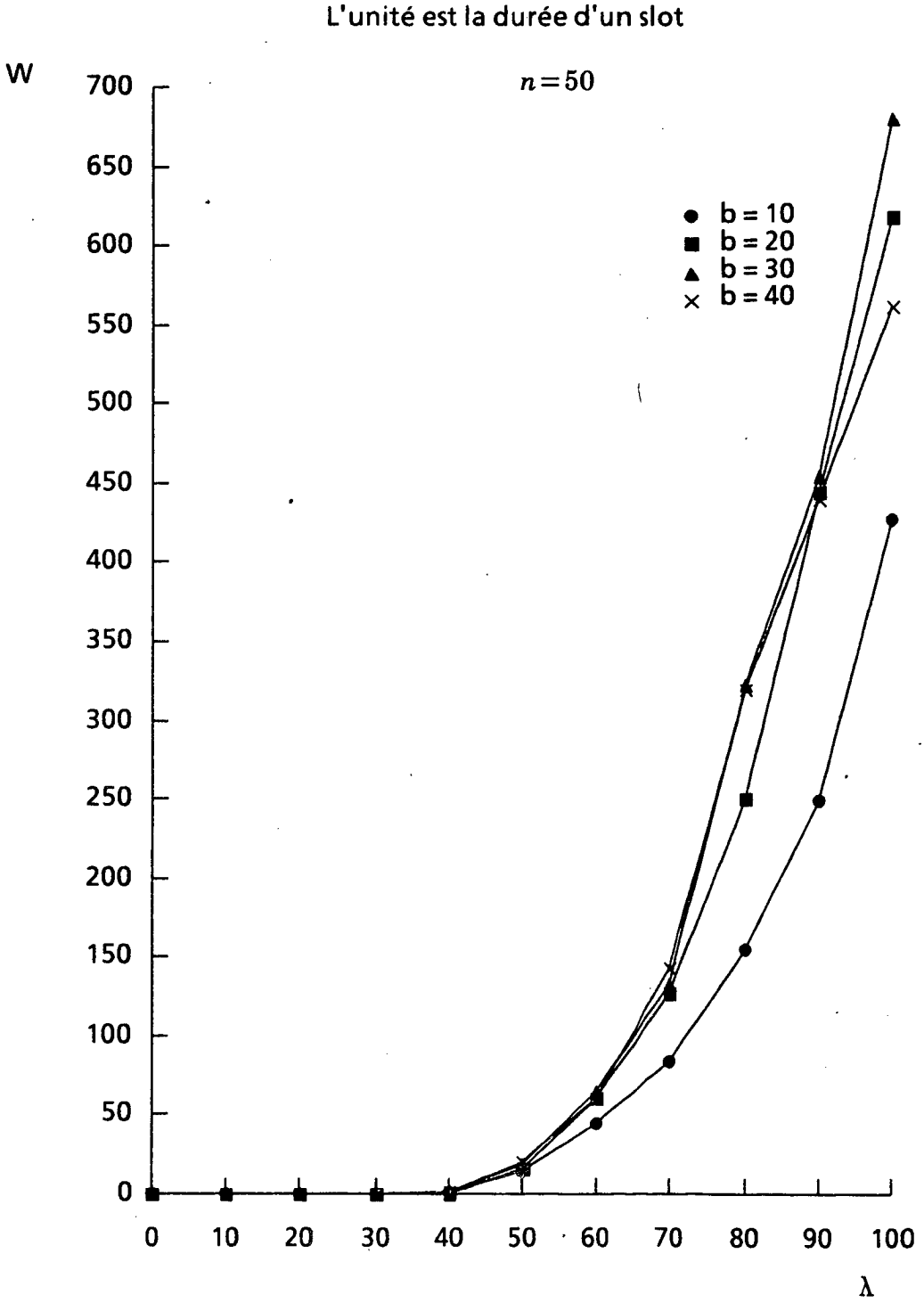


FIG 7.Délai moyen d'attente en file en fonction de la charge d'entrée λ.

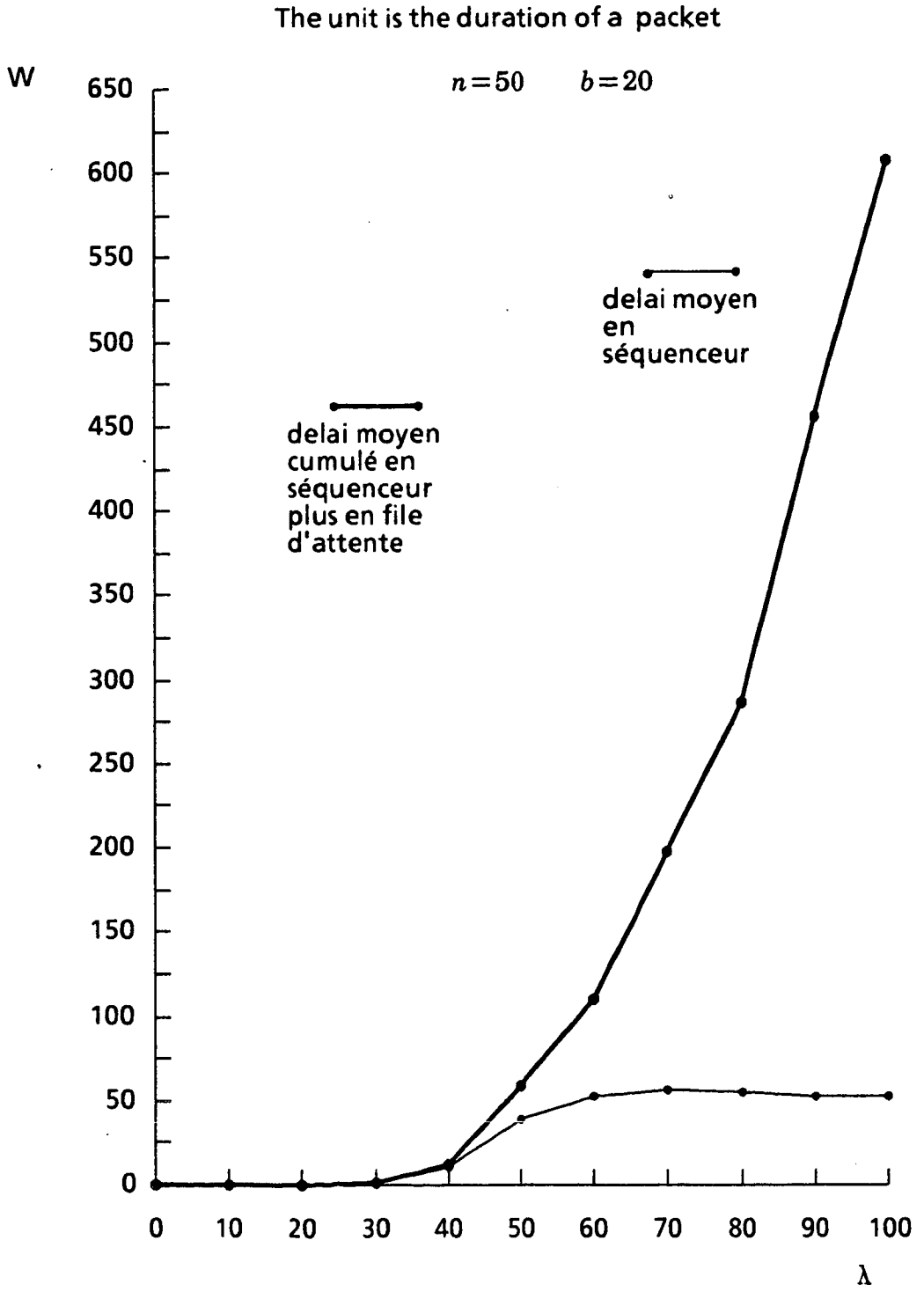


FIG 8. Délai moyen en séquenceur en fonction de la charge d'entrée λ.
Délai moyen cumulé en séquenceur plus en file d'attente en fonction de la charge d'entrée λ.

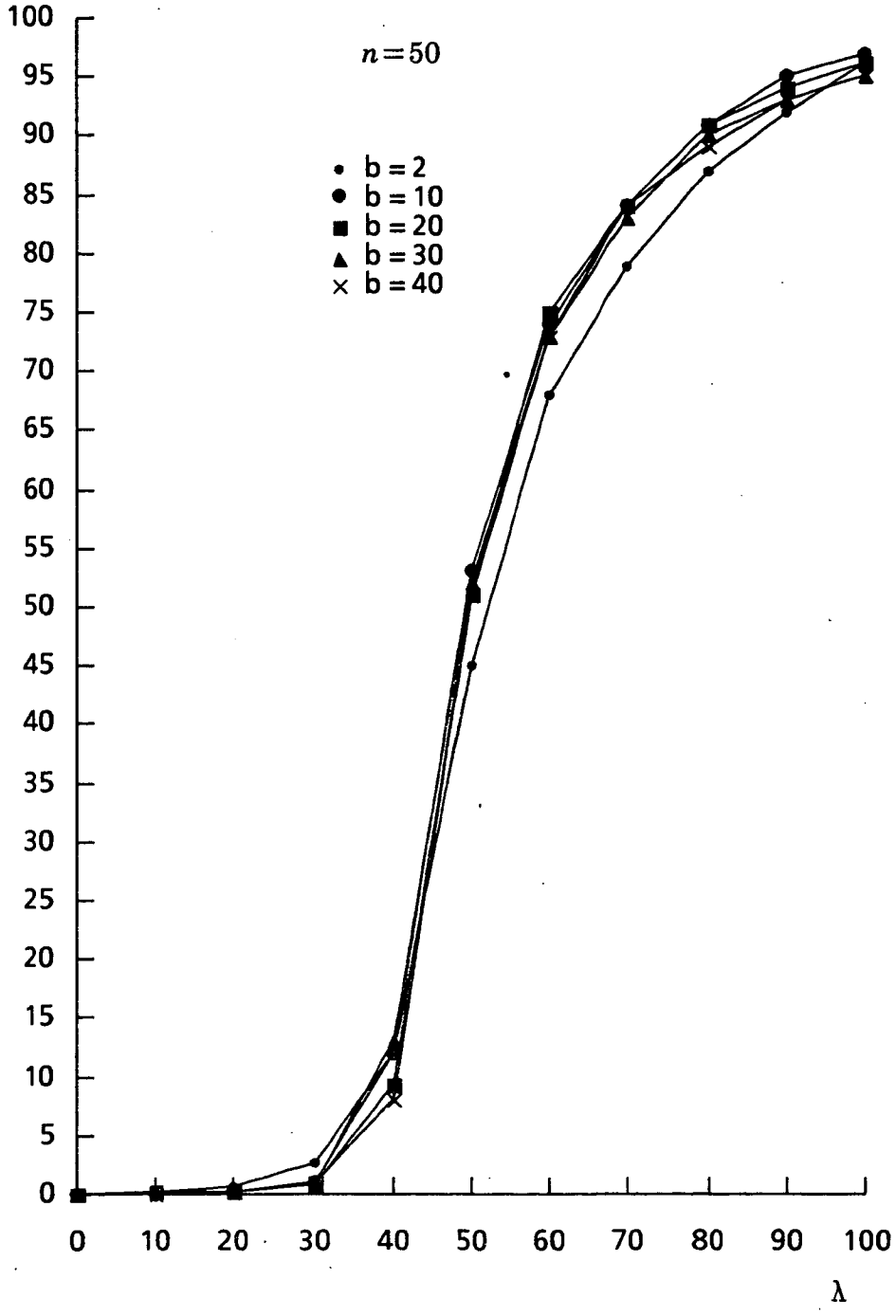


*FIG.*9 Pourcentage de paquets passés sur places réservées en fonction de la charge d'entrée λ.

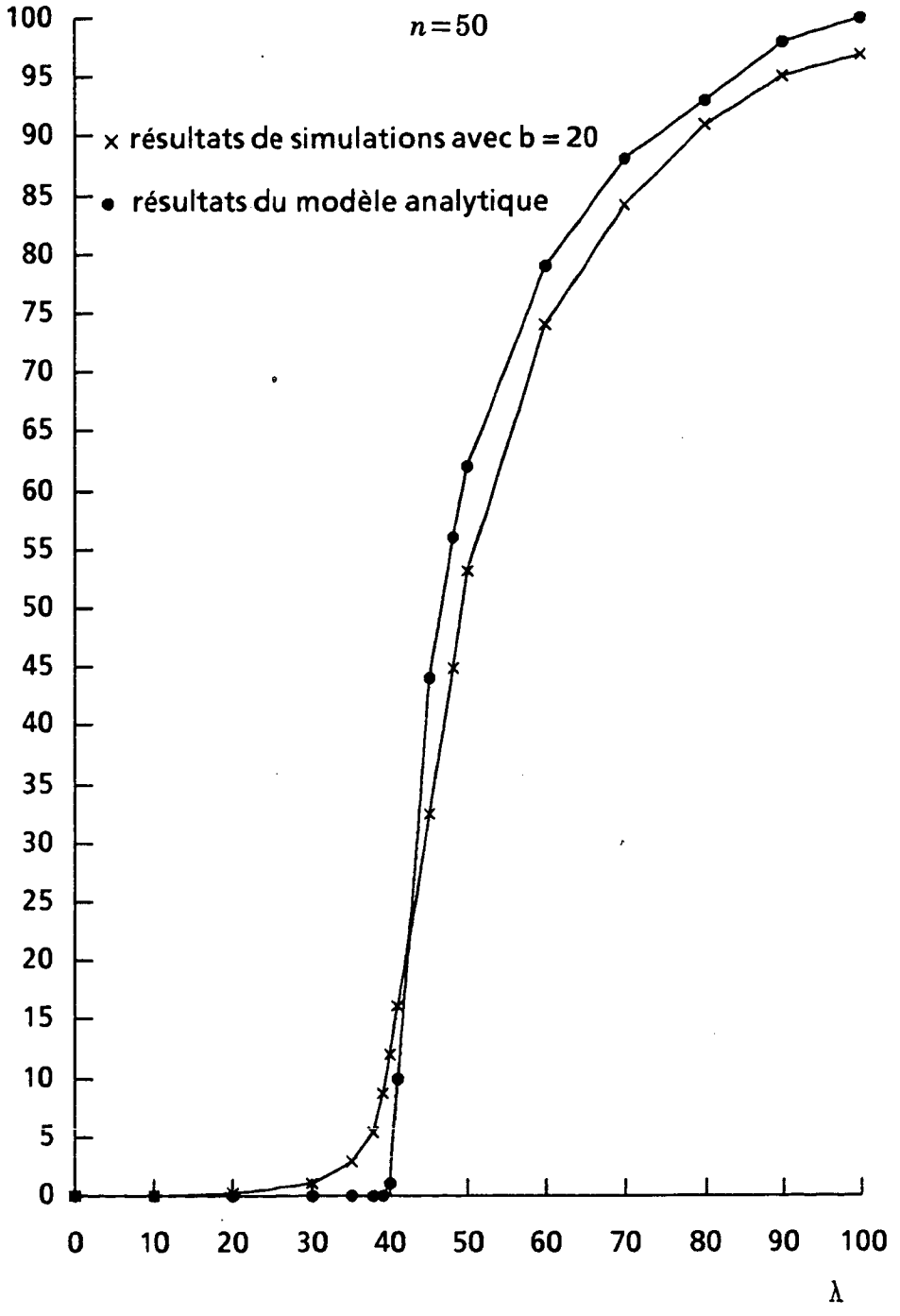


FIG. 10 Pourcentage de paquets passés sur places réservées en fonction de la charge d'entrée λ: comparaison entre le modèle analytique et les simulations.

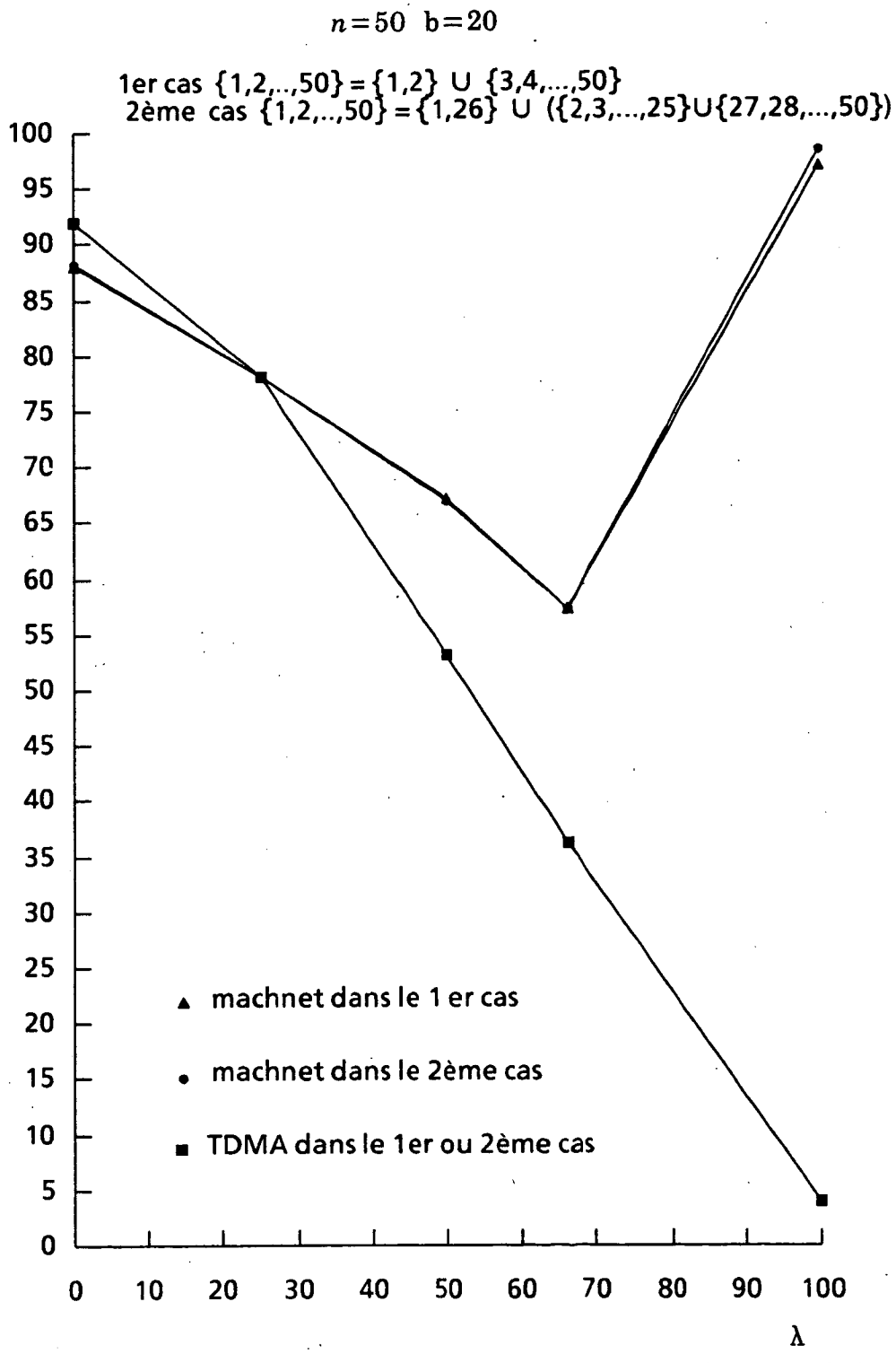


FIG. 11 Pourcentage de bande passante effectivement utilisée en fonction de la charge du premier ensemble de stations λ.

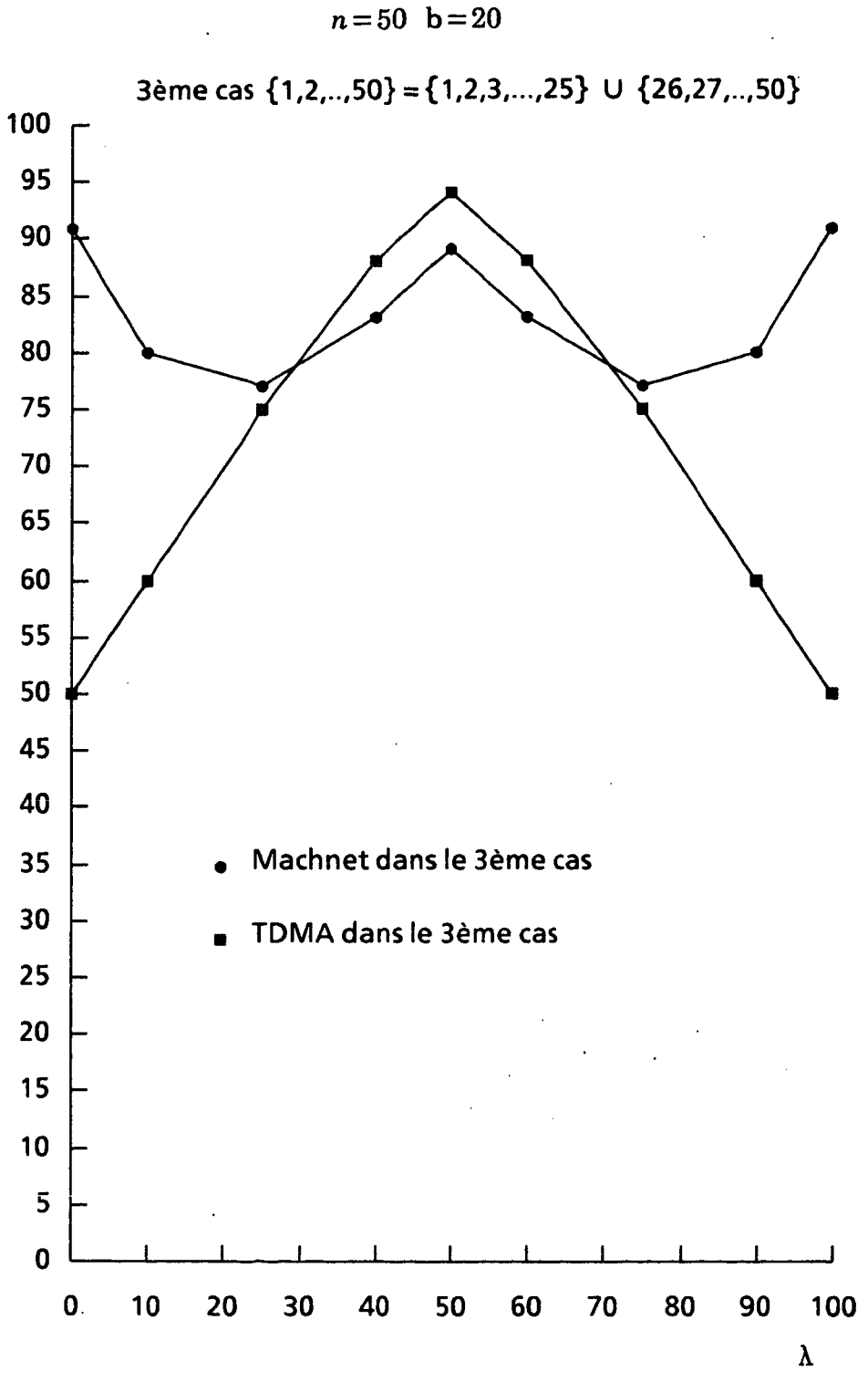


FIG. 12 Pourcentage de bande passante effectivement utilisée en fonction de la charge du premier ensemble de stations λ.

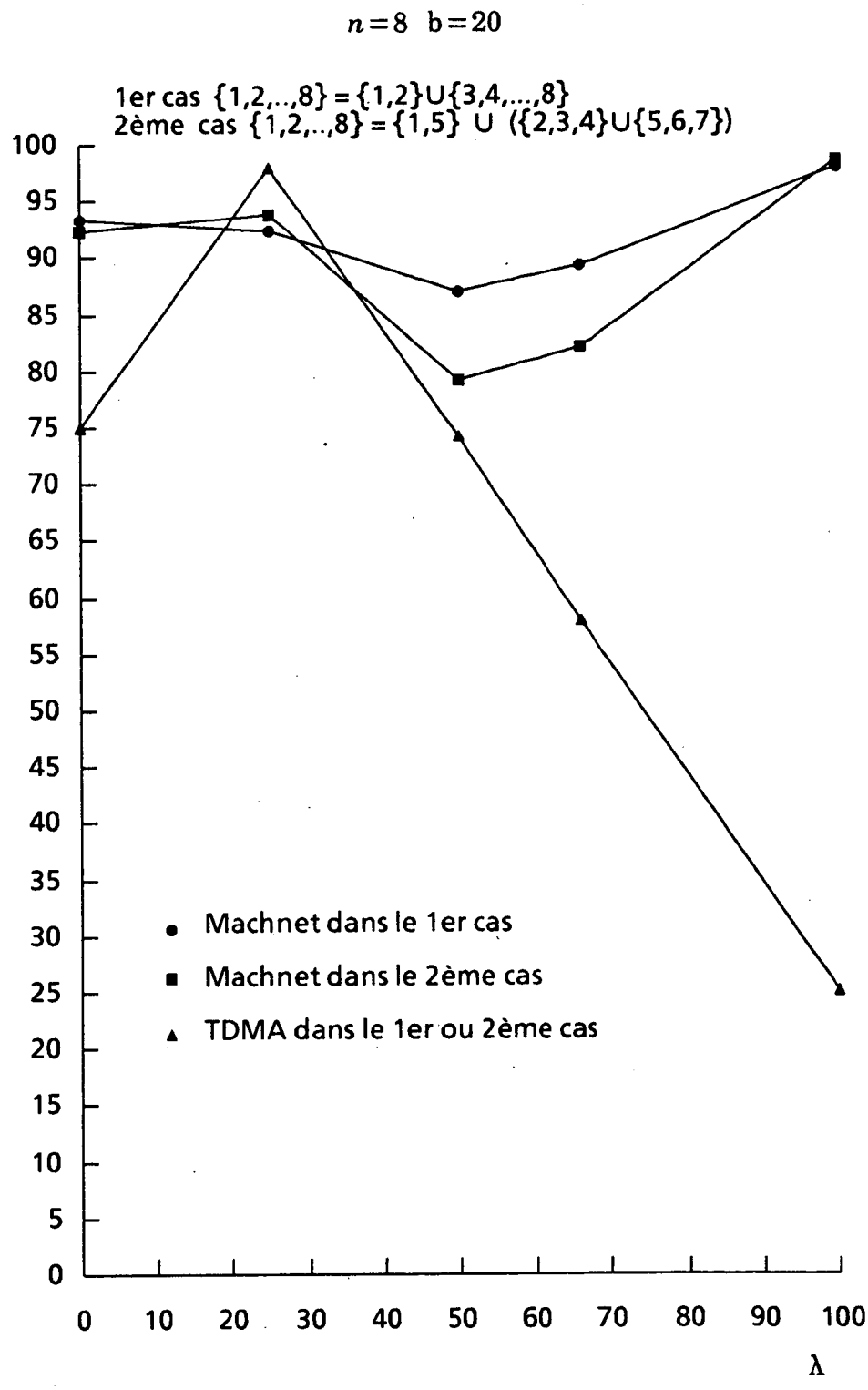


FIG. 13 Pourcentage de bande passante effectivement utilisée en fonction de la charge sur le premier ensemble de stations: λ.

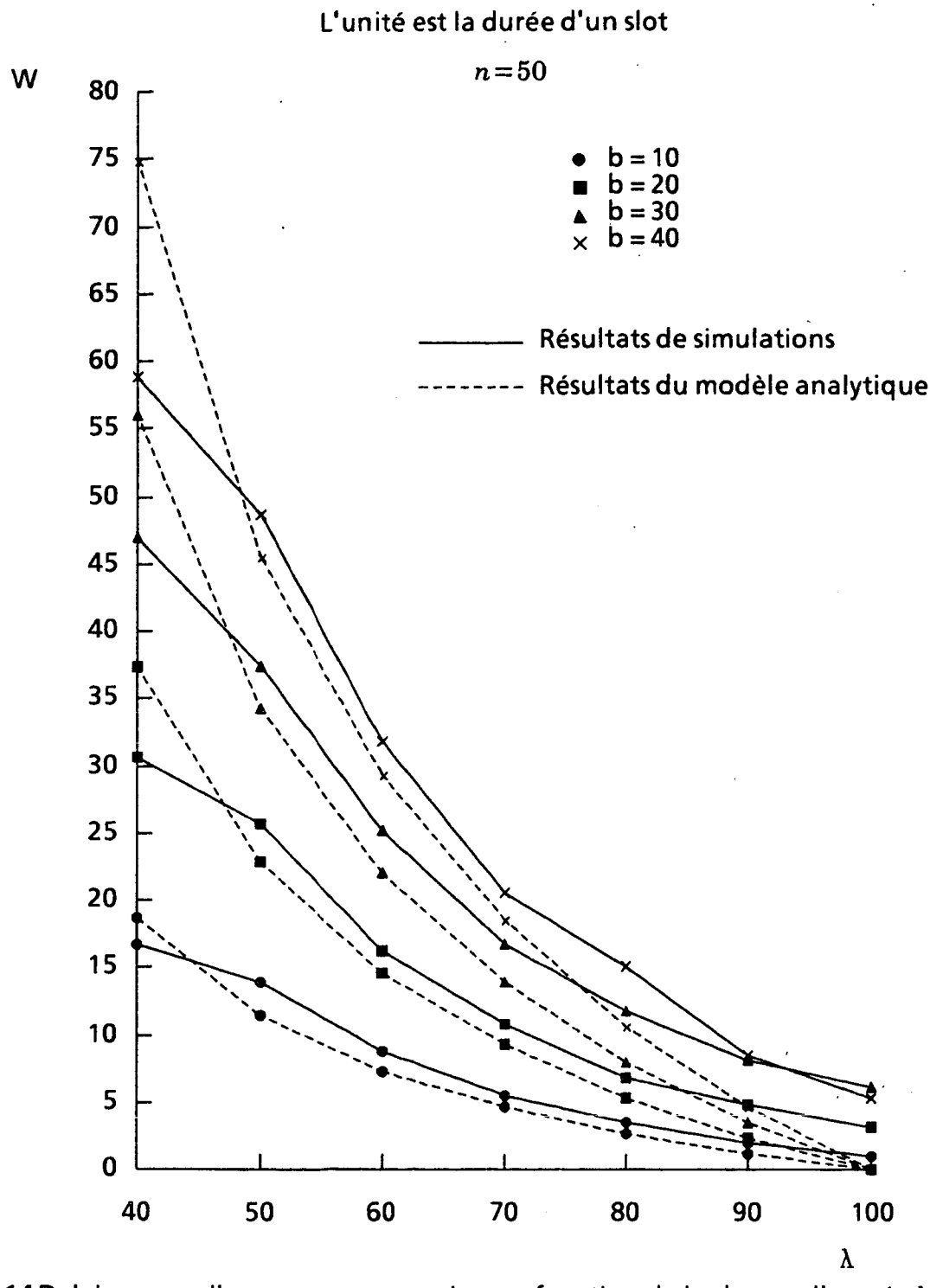


FIG. 14 Delai moyen d'attente en processing en fonction de la charge d'entrée λ, comparaison entre simulations et modèle analytique.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique