PUC

Series: Monografias em Ciência da Computação

Nº 8/79

ON THE PROBLEM OF CAPACITY ASSIGNMENT IN COMPUTER NETWORKS WHICH CARRY MESSAGES WITH DIFFERENT PRIORITIES

by

Daniel A. Menascé
Mário A. Monteiro

~~Departamento de Informática~~

Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225 – CEP-22453
Rio de Janeiro – Brasil

PUC/RJ - DEPARTAMENTO DE INFORMATICA

Series: Monografias em Ciência da Computação

Nº 8/79

Editor : Daniel A. Menascê

May, 1979

ON THE PROBLEM OF CAPACITY ASSIGNMENT IN COMPUTER NETWORKS WHICH CARRY MESSAGES WITH DIFFERENT PRIORITIES*

Daniel A. Menascê

Mário A. Monteiro**

* This research was partially supported by the Conselho Nacional de Desenvolvimento Cientifico e Teconlógico, CNPq, under contract 1885/78.

** CDR. Brazilian Navy, concluding postgraduate Studies in Computer Science with financial support of the Brazilian Navy.

RESUMO

Este trabalho trata do problema de alocação ótima de capacidades em redes de computadores com mensagens de prioridades diferentes. Dado que não é possível obter-se uma solução analítica para este problema, este artigo apresenta um algoritmo, que utiliza técnicas heurísticas, a fim de obter soluções subótimas para o problema. O artigo contém os cálculos necessários para a obtenção do atraso médio, Tp, sofrido por uma mensagem de prioridade p. O trabalho conclui com a apresentação de diversos exemplos numéricos da utilização do algoritmo, bem como uma avaliação das técnicas heurísticas propostas.

Palavras-chave: redes de computadores, mensagens com prioridades, alocação ótima de capacidades, atraso médio, técnicas heurísticas.

ABSTRACT

This paper deals with the problem of optimal capacity assignment in computer networks which carry messages of different priorities. Given that it is not possible to obtain a closed form analytical solution to this problem, the paper presents an algorithm, which uses heuristic techniques, in order to obtain suboptimal solutions to the problem. It is shown here how to derive the average delay, Tp, experienced by a message of priority p. The paper concludes with the presentation of several numerical examples of the utilization of the algorithm, as well as an evaluation of the effectiveness of the proposed heuristic techniques.

Keywords and phrases:

Computer communication networks, messages with priorities, optimal capacity assignment, average delay, heuristic techniques.

1. Introduction

One of the problems that must be faced during the design of computer communication networks is the assignment of capacities to the various communication channels that link the different processors in the network. Two important factors must be taken into account in the selection of channel capacities: the cost of the channels and the average delay experienced by a message or class of messages in order to traverse the network. The cost of communications channels increases with the channel capacity and also with the distance between the two points to be linked by the channel. On the other hand, the average delay is a decreasing function of the channel capacities selected for the network.

In this paper, we shown how to select capacities in order to achieve a minimun cost for the network under the constraint that the average delay for messages of each priority class does not exceed a given pre-determined limit for that class.

That are P priority classes for messages. For the case in which there is only one priority class, an optimal solution has been obtained for the capacity assignment problem, by Kleinrock [KLEI 64 and KLEI 76]. For the case in which there are several priorities classes it is not possible to find a closed form analytical solution to the problem. Therefore, we present here an algorithm which resorts to heuristic techniques in order to find suboptimal solutions to the problem.

The remainder of this paper is organized as follows. Section 2 derives the formula to find the average delay, Tp, for each priority class p, p=1,2,...,P. Section 3 presents an informal description of the algorithm which is formally described in section 4. Finally, in section 5, several numeric results obtained with the algorithm are discussed and the proposed heuristics are compared as to their effectiveness.

2. Average Delay Analysis

This section defines a mathematical model for the network and derives the formula for the average delay experienced by messages of each priority class.

Channel Model

A communications channel, or simply a channel, is going to be modeled as an M/M/1 queueing system with P priority classes of customers, numbered from 1 to P. Prority class P represents the highest priority. It is also assumed that a preemptive resume discipline is in effect. In other words, the arrival of a message of priority $\underline{i}$ interrupts the transmission of messages of priority less than $\underline{i}$ and seizes the channel.

Let us define, initially, some parameters of the model:

$\frac{1}{\mu}$ = average message size (in bits).

C_i = capacity of the channel i (in bps).

$\lambda_{i,p}$ = arrival rate of messages of priority p at the channel i (in messages/sec).

$T_{i,p}$ = average delay suffered by a message of class p at the channel i. This time includes waiting time and transmission time.

The size of a message is assumed to be exponentially distributed with mean $1/\mu$. The independence assumption is used here (see KLEI 64 or KLEI 76). Let $\tilde{x}_i$ be the random variable which represents the transmission time of a message through channel i. The probability density function of $\tilde{x}_i$ is given by

$$f_{\tilde{x}_i}(x) = \mu C_i \; e^{-\mu C_i x} \qquad (1)$$

In this way, the average delay $T_{i,p}$ suffered by a tagged message of class p in channel i is given by its average transmission time $\tilde{x}_i$, plus the waiting time W_1 due to messages of priority not less than p found in the system by our tagged message, plus the average delay W_2 which represents the total transmission time of all the messages of class grater than p that arrive in the system while our tagged message is still in the system.

W_1 can be calculated as the average unfinished work found by our tagged message upon its arrival. For an M/G/1 system this unfinished work is equal to the average waiting time. Therefore,

$$W_1 = \frac{\sum_{j=p}^{p} \lambda_{i,j} \tilde{x}_i^2/2}{1 - \sigma_{i,p}} \tag{2}$$

where

$$\sigma_{i,p} \triangleq \sum_{j=p}^{p} \rho_{i,j} = \sum_{j=p}^{p} \frac{\lambda_{i,j}}{\mu C_i} \tag{3}$$

and

$$\tilde{x}_i^2 = \int_0^\infty x^2 \mu C_i \, e^{-\mu C_i x} \, dx = \frac{2}{(\mu C_i)^2}$$

therefore,

$$W_1 = \frac{\sum_{j=p}^{p} \lambda_{i,j}/(\mu C_i)^2}{1 - \sigma_{i,p}} = \frac{1/\mu C_i \sum_{j=p}^{p} \lambda_{i,j}/(\mu C_i)}{1 - \sigma_{i,p}}$$

Using (3) in the above equation we have that

$$W_1 = \frac{\sigma_{i,p}}{\mu C_i(1-\sigma_{i,p})} \qquad (4)$$

W_2 can be found as follows. Our tagged message of priority $\underline{p}$ is going to be delayed by a time $1/\mu C_i$ by each message of priority $\underline{j}$, strictly greater than $\underline{p}$, which arrives at the channel before our tagged message leaves the system. The average number of messages of priority j $(j > p)$ that arrive in the time interval $T_{i,p}$ is equal to $\lambda_{i,j}T_{i,p}$. Each one of them represents a delay of $1/\mu C_i$ to our tagged message.

Therefore,

$$W_2 = \sum_{j=p+1}^{P} \frac{\lambda_{i,j}}{\mu C_i} T_{i,p} = \sum_{j=p+1}^{P} \rho_{i,j} T_{i,p} \qquad (5)$$

Finally, we have the following equation:

$$T_{i,p} = \frac{1}{\mu C_i} + \frac{\sigma_{i,p}}{\mu C_i(1-\sigma_{i,p})} + \sum_{j=p+1}^{P} \rho_{i,j}\, T_{i,p} \qquad (6)$$

Solving the above equation for $T_{i,p}$ we have

$$T_{i,p} = \frac{1}{C_i\left(1-\frac{\alpha_{i,p}}{C_i}\right)\left(1-\frac{\alpha_{i,p+1}}{C_i}\right)} \qquad (7)$$

$$p = 1,2,\ldots P$$

where

$$\alpha_{i,p} \triangleq \sum_{j=p}^{P} \frac{\lambda_{i,j}}{\mu}$$

and $\alpha_{i,p+1} \triangleq 0$

Let us now calculate the average delay, Tp, suffered by a message of priority p. Let us follow a derivation analogous to the one given by Kleinrock in [KLEI 76] for the one priority case. Let $Z_{j,k;p}$ be the average delay experienced by messages of priority p, which have a node j as a source node and a node k as a destination node. Thus,

$$Z_{j,k;p} = \sum_{i:C_i \in \pi_{jk}} T_{i,p} \qquad (8)$$

Let $\gamma_{jk;p}$ be the external traffic of priority p entering the network at node j and leaving it at node k. Let

$$\gamma_p = \sum_{j=1}^{N} \sum_{k=1}^{N} \gamma_{jk;p} \tag{9}$$

where N is the number of nodes in the network. Therefore we can write

$$T_p = \sum_{j=1}^{N} \sum_{k=1}^{N} \frac{\gamma_{jk;p}}{\gamma_p} Z_{jk;p} \tag{10}$$

Since $\gamma_{jk;p}/\gamma_p$ is the fraction of the j-k traffic that experiences a delay $Z_{jk;p}$. Substituting (8) in (10) we have that

$$T_p = \sum_{j=1}^{N} \sum_{k=1}^{N} \frac{\gamma_{jk;p}}{\gamma_p} \sum_{i:C_i \epsilon \pi_{jk}} T_{i,p} \tag{11}$$

Exchanging the order of the summations in the above equation we get

$$T_p = \sum_{i=1}^{M} \frac{\lambda_{i,p}}{\gamma p} T_{i,p} \tag{12}$$

where M is the number of links in the network.

3. Capacity Assignment Algorithm- An Informal Description

We are now ready to define more precisely the capacity assignment problem in a network which carries messages of different priorities.

Given

1- the network topology

2- the values of the rates $\lambda_{i,p}$

3- a set of available capacities, AVAILCAP, out of which the capacities to be assigned to the various links must be selected.

4- the values for the external traffic rates γ_p

Assign

capacities, selected out of AVAILCAP, to the various communication links in order to

Minimize

the total cost $D = \sum_{i=1}^{M} d_i(C_i)$ where $d_i(C_i)$ is the cost of leasing a communication line of capacity C for the i-th link.

Under the restrictions

$T_1 <= TMAX_1$

$T_2 <= TMAX_2$ --------------

$T_p <= TMAX_p$

Given that it is not possible to find an analytic solution to the above problem, we are going to look for suboptimal solutions, resorting to heuristic techniques.

The capacity assignment algorithm, can be intuitively described as follows. The smallest capacity of the set AVAILCAP is initially assigned to all the channels in the network. Then, the average delays $T_{i,p}$ are calculated and simultaneously the capacities of the some channels are altered according to the two following criteria:

CRITERIUM 1: If one observes the formula for $T_{i,p}$ (formula (7)), it can be seen that both of the terms in parentheses in the denominator have to be positive. Therefore,

$1 - \alpha_{i,p} / C_i > 0$ and

$1 - \alpha_{i,p+1} / C_i > 0$

According the definition of $\alpha_{i,p}$, if

$1 - \alpha_{i,p} / C_i > 0 \quad 1 - \alpha_{i,p+1} / C_i > 0$

The meaning of this condition can be better understood if we notice that

$$\frac{\alpha_{i,p}}{C_i} = \sum_{j=p}^{P} \frac{\lambda_{ij}}{\mu C_i}$$

which is therefore equal to the utilization of channel <u>i</u> by messages of priority equal or grater than <u>p</u>. This utilization must be less than 1 in order for the time $T_{i,p}$ to be finite and in order for the channel <u>i</u> to be in a stable condition.

Therefore, the initial capacity in each channel is incremented until

$1 - \alpha_{i,1}/C_i > 0$. Note that if the above condition is satisfied for messages of priority 1 it will be satisfied for messages of higher priorities.

CRITERIUM 2: According to formula (12) for Tp we have

$$T_p = \sum_{i=1}^{M} \frac{\lambda_{i,p}}{\gamma_p} T_{i,p}$$

but if Tp <= TPMAX then we have that

$$\sum_{i=1}^{M} \frac{\lambda_{i,p}}{\gamma_p} T_{i,p} <= TMAX_p$$

or

$$\sum_{i=1}^{M} \lambda_{i,p} T_{i,p} <= \gamma_p * TMAX_p$$

But if a summation has an upper bound and if all of its terms are positive, then each term individually must be less than or equal the upper bound.

Hence,

$$\lambda_{i,p} T_{i,p} <= \gamma_p * TMAX_p \qquad (13)$$

Therefore, each time that a T is calculated, the condition expressed by equation (13) is checked. If it is not satisfied then the channel capacity is increased until the condition becomes valid. An increase in the capacity of a given channel implies in the recalculation of the values of $T_{i,p}$ already calculated for that channel.

At the end of this process, we have all the values for $T_{i,p}$ and we can, therefore, find all the values of Tp for p=1,2,...,P. If one of the constraints on Tp are violated, then some channels are selected to have their capacities increased to the next available capacity in AVAILCAP according to some heuristic techniques. The values of $T_{i,p}$ for the channel which had their capacities increased are computed again and new values for Tp are obtained. These new values are again checked with the constraints. If necessary more channels will have their capacities increased until all the constraints are satisfied.

In order to complete this discussion, we have to describe the heuristic techniques which are being proposed here. The results obtained with each of them are analyzed in section 5.

HEURISTIC 1: the technique used in this case can be best described by the following steps.

1- Evaluate the mean value, TMEAN, of all $T_{i,p}$ values for each priority.

$$\text{TMEAN}_p = \frac{1}{M} \sum_{i=1}^{M} T_{i,p} \qquad \text{for} \quad p=1,\ldots P$$

2- Increase the capacity of channel <u>i</u> to the next capacity in AVAILCAP if

$T_{i,p} > TMED_p$ for any $p=1,\ldots P$.

HEURISTIC 2: this technique tries to increase the capacity of those channels for which one can obtain a larger reduction in the value of $T_{i,p}$ for the same increase in C . This can be measured by the value of the partial derivative of $T_{i,p}$ relative to C_i evaluated at the current value of C_i.

$$\left.\frac{\partial T_{i,p}}{\partial C_i}\right|_{C_i} = -\frac{1}{\mu}\ \frac{1-\alpha_{i,p}\alpha_{i,p+1}/C_i^2}{C_i^2\ (1-\frac{\alpha_{i,p}}{C_i})^2\ (1-\frac{\alpha_{i,p+1}}{C_i})^2} \qquad (15)$$

The value of this partial derivative is always negative, since the denominator is positive, since the numerator is also positive because $\alpha_{i,p}/C < 1$ for $p=1,\ldots,P$ (see criterium 1) and since the whole expression is preceded of a minus sign. In other words, $T_{i,p}$ decreases as C increases, as it could be expected. Therefore, the greatest is the reduction of $\left.\frac{\partial T_{i,p}}{\partial C_i}\right|_{C_i}$

the greatest is the reduction in $T_{i,p}$ for an increase in C_i. The heuristic can now be described by the following steps:

1- evaluate the quantity h defined below

$$h_{i,p} = \frac{\lambda_{i,p}}{l_i} \left| \frac{\partial T_{i,p}}{\partial C_i} \Big|_{C_i} \right| \qquad \begin{matrix} i=1,\ldots M \\ p=1,\ldots P \end{matrix} \qquad (16)$$

where l is the distance covered by channel i. The greatest the value of l the gratest is the cost of a given channel for the same capacity.

2- Let p1,p2,...,pn be the priorities for which the constraints habe been violated. Let p1,p2,...,pn be ordered in increasing order. Increase the capacity of the k greatest values for $h_{i,p}$. A bigger value of $h_{i,p}$ implies in a greater reduction of $T_{i,p}$ per unit of increase in C_i. This value is multiplied by $\lambda_{i,p}$, since this is the multiplying factor of $T_{i,p}$ in Tp. Moreover, the greater the value of l_i, the smaller the value of $h_{i,p}$ and therefore there is a smaller probability that a long channel will have its capacity increased. The value of k is a parameter that must be adjusted according to the computational effort that one desires to spend in the execution of the algorithm. Best solutions are obtained for values of k=1.

3- reevaluate the values of $T_{i,p}$ for the channels which had their capacities altered in step 2 and reevaluate the values of Tp for p=1,...,P.

4- check whether there is any constraint not yet satisfied. In the affirmative case return to step 2.

4. Capacity Assignment Algorithm- Formal Description

This section describes formally the capacity assignment algorithm. The algorithm is described in ALGOL 60 in the Appendix. The program is composed of several routines which will be described in what follows. In the main program, the values of $T_{i,p}$ are evaluated using criteria 1 and 2. Then the values of Tp are evaluated and one of the heuristic techniques is applied as many times as necessary, as described in the previous section. The main routines of the program are:

1- incrementacap(channel)- this routine increments the capacity of channel 'channel' and of its simplex pair. All the channels are considered full duplex but are represented as a pair of simplex channels in opposite directions.

2- evaltip(i,p)- this procedure evaluates $T_{i,p}$

3- sort(a,b)- this routine sorts the array a in increasing order and places in the array b the original indexes of the elements in a.

4- evaltp(constraint,flag)- this routine evaluates all values of Tp. If any of the constraints

$Tp < TMAXp$

is violated, flag returns the value true. If the constraint

$T_i < TMAX_i$

is violated then the constraint i returns the value <u>true</u>.

5- heuristic1(constraint,modified)- this routine applies heuristic 1 described in the previous section and assigns the value <u>true</u> to modified i if the capacity of channel <u>i</u> was modified.

6- heuristic2(constraint,modified)- similar to the previous routine except that heuristic2 described in the previous section is applied.

Next, we give a list of the relationships between the main variables of the program and the mathematical symbols defined in section 2.

M----	nlinks
P----	maxp
$\alpha_{i,p}$-	alfa[i,p]
$T_{i,p}$-	tip[i,p]
$\lambda_{i,p}$-	lambda[i,p]
γ_p---	gama[p]
T_p---	tp[p]
l_i---	dist[i]
C_i---	availcap[cap[i]]
μ----	mu

In the algorithm, all the indexes in arrays of dimension equal to <u>nlinks</u> are such that the pair of indexes that identify a full-duplex channel is (j,j+nlinks/2) for j=1,2,...,nlinks/2.

5. <u>Examples and results obtained with the use of the Algorithm</u>

In order to test the effectiveness of the two heuristic techniques and in order to illustrate the behavior of the algorithm, we ran three numerical examples using three different topologies, called A,B,C heretofore. The topologies are illustrated in figures 1,2 and 3 respectively.

In every case, the value of $1/\mu$ was made equal to 3200 bits. We considered the case of four priorities. For the examples, we made the assumption that γ_p is a percentage f_p of the value of γ.

The percentages used are:
f1=29%; f2=45%; f3=25%; f4=1%

The value of γ for the three cases is 7.5Kbps. The time constraints for each priority are
TMAX1=90s; TMAX2=45s; TMAX3=30s; TMAX4=15s

The values of $\lambda_{i,p}$ are by assumption a fraction f_p of λ_i. The values of λ_i are given in figures 1,2 and 3 for each case.

The results obtained are summarized in the table 1.

Cost in the table 1 is given in thousands of dollars per year of operation. By inspection of the above table we can see that heuristic 2 is better than heuristic 1 in the sence that it yields cheaper networks, while preserving the times within the limits. This result was somehow expected since heuristic 2 is based on a more consistent argument that heuristic 1. In the tests described below, the value k=1 was u-

	TOPOLOGY					
	A		B		C	
	HEURISTIC		HEURISTIC		HEURISTIC	
	1	2	1	2	1	2
T1	30.6	82.4	30.6	76.8	26.4	86.2
T2	13.3	19.4	12.9	20.0	12.8	22.5
T3	8.0	9.8	7.7	9.9	8.1	11.3
T4	6.9	8.1	6.6	8.2	7.0	9.3
Cost	127.7	110.1	110.0	104.7	120.7	105.1

sed in executing heuristic 2. In other words, one channel capacity was incremented at a time.

The costs obtained in the table 1 were computed taking into account the tariffs provided by EMBRATEL, which is the single brazilian common carrier, as of 1978.

Conclusions

This paper presents a formula which allows us to evaluate the average delay spent by a message of a given priority in a given network. An algorithm to assign capacities to the various channels of the network was also presented here. The algorithm resorts to heuristic techniques in order to obtain suboptimal solutions to the problem. One of the two proposed heuristics has an excellent behavior as it can be seen in the table 1. For the case, the time constraint for one of the priority classes is matched very closely by the actual time obtained.

7. References

-KLEI 64 - KLEINROCK, l. - "Communication Nets; Stochastic Message Flow and Delay", McGraw Hill, New York, 1964

-KLEI 76 - KLEINROCK, L. - "Queueing Systems: Computer Applications", Vol. 2 John Wiley and Sons, New York, 1976.

APPENDIX: Algorithm in Algol60

```
PROCEDURE Alocap;
  BEGIN

    COMMENT
      nlinks=# of channels in the net
      availcap[1:n]=array of the n available capacities;
      cap[1:nlinks]=array of channel capacities (contains
                    indexed into availcap);

    INTEGER i,p,j,nlinks,pmax,n;
    REAL ARRAY alfa[nlinks,pmax+1],tip[nlinks,pmax],
               lambda[nlinks,pmax],gama[pmax],
               tp[pmax],tmax[pmax],availcap[n],
               dist[nlinks];
    INTEGER ARRAY cap[nlinks];
    REAL c,t,mu;
    BOOLEAN flag;
    BOOLEAN ARRAY constraint[pmax],modified[nlinks];

    PROCEDURE incrementcap(channel);
      BEGIN

        COMMENT this procedure increments the capacity
                of channel 'channel' and of its simplex pair;

        INTEGER channel;

        cap[channel]:=cap[channel] + 1;
        IF channel > nlinks/2
        THEN cap[channel - nlinks/2]:=cap[channel];
        ELSE cap[channel + nlinks/2]:=cap[channel];
      END;
    END incrementcap;

    PROCEDURE evaltip(i,p);
      BEGIN

        COMMENT this procedure evaluates tip[i,p];

        REAL templ,temp2;
        INTEGER i,p;

        templ:=1 - (alfa[i,p]/availcap[cap[i]]);
        temp2:=1 - (alfa[i,p+1]/availcap[cap[i]]);
        tip[i,p]:=1/(mu*availcap[cap[i]]*templ*temp2);
      END;
    END evaltip;
```

```
PROCEDURE sort(a,b);

  BEGIN

    COMMENT this procedure sorts in decreasing order
            the array a and places in b the original
            indexes of the elements in a;

    ----------------
  END
END sort;

PROCEDURE evaltp(constraint,flag);

  BEGIN

    COMMENT this procedure evaluates Tp for all
            values of p, checks the constraints
            and returns the value true in flag if
            any of them was violated. It returns
            the value true in constraints[p]
            if the constraint for priority p was
            violated;

    BOOLEAN flag;
    BOOLEAN ARRAY constraint[pmax];

    flag:=false;
    FOR p:=1 STEP 1 UNTIL pmax DO
       BEGIN

         COMMENT evaluate Tp's;

         tp[p]:=0;
         FOR i:= 1 STEP 1 UNTIL nlinks DO
            tp[p]:=tp[p]+(lambda[i,p]*tip[i,p]);
         tp[p]:=tp[p]/gama[p];

         COMMENT check the constraints;

         constraint[p]:=false;
         IF tp[p] > tmax[p]
         THEN BEGIN
                flag:=true;
                constraint[p]:=true;
              END;
      END;
  END;
END evaltp;
```

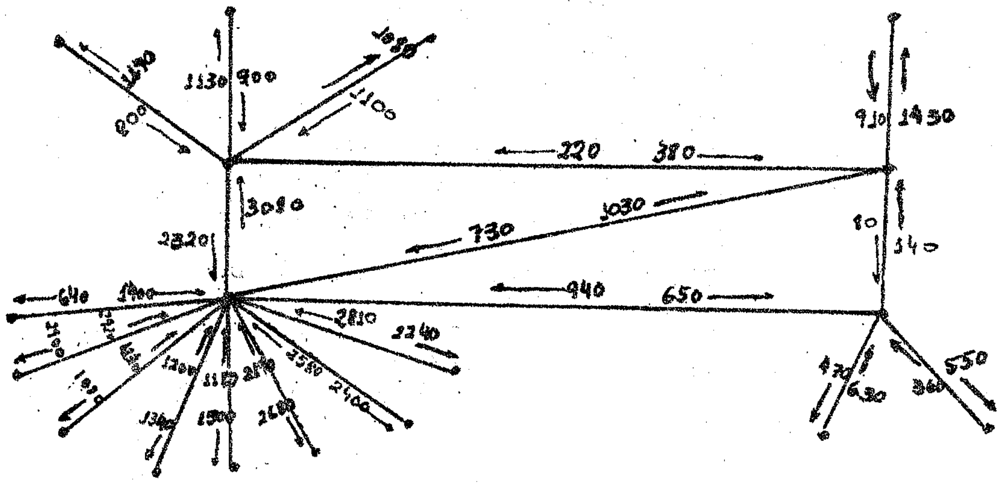
```
PROCEDURE heuristicl(constraint,modified);
  BEGIN

    COMMENT this procedure increments the capacity
            of those channels for which Tip exceeds
            the mean value;

    INTEGER i,p,j;
    BOOLEAN ARRAY constraint[pmax],modified[nlinks];
    BOOLEAN flag;
    REAL tmean;

    FOR i:=1 STEP 1 UNTIL nlinks DO
       modified[i]:=false;
    flag:=true; j:=1;
    FOR p:=j WHILE flag DO
       IF constraint[p]
       THEN BEGIN
              flag:=false;

              COMMENT mean value evaluation;

              tmean:=0;
              FOR i:=1 STEP 1 UNTIL nlinks DO
                 tmean:=tmean+tip[i,p]:
              tmean:=tmean/nlinks;

              COMMENT comparision with the mean;

              FOR i:=1 STEP 1 UNTIL nlinks DO
                IF tip[i,p] >= tmean
                THEN BEGIN
                       incrementcap(i);
                       modified[i]:=true:
                       IF i>nlinks/2
                       THEN modified[i-nlinks/2]:=true;
                       ELSE modified[i+nlinks/2]:=true;
                     END;
            END;
       ELSE j:=j + 1;
  END;
END heuristicl;
```

```
PROCEDURE heuristic2(constraint,modified);
  BEGIN

    COMMENT this procedure applies heuristic 2 and
            indicates in the array modified whether
            the capacity of a given channel was altered;

    INTEGER i,p,j,k;
    BOOLEAN ARRAY constraint[pmax],
                  modified[nlinks];
    BOOLEAN flag;
    REAL h[nlinks],temp1,temp2,temp3
    INTEGER ARRAY channels[nlinks];

    FOR i:=1 STEP 1 UNTIL nlinks DO
       modified[i]:=false;
    flag:=true; j:=1;
    FOR p:=j WHILE flag DO
       IF constraint[p]
       THEN BEGIN
              flag:=false;

              COMMENT evaluation of h;

              FOR i:=1 STEP 1 UNTIL nlinks DO
                BEGIN
                 temp1:=alfa[i,p]*alfa[i,p+1];
                 temp1:=temp1/(availcap[cap[i]])**2);
                 temp2:=1-(alfa[i,p]/availcap[cap[i]]);
                 temp3:=1-(alfa[i,p+1]/availcap[cap[i]]);
                 temp2:=temp2*temp3*availcap[cap[i]];
                 h[i]:=(1-temp1)/temp2;
                 h[i]:=(lambda[i,p]+h[i])/dist[i];
                END;

              COMMENT sort the array h;

                sort(h,channels);

              COMMENT increment the capacity of the k
                      channels with the greatest h;

              FOR j:=1 STEP 1 UNTIL k DO
                 BEGIN
                  IF modified[channels[j]]:=false
                  THEN BEGIN
                         incrementcap(channels[j];
                        modified[channels[j]]:=true;
                       END;
                   IF channel[j] > nlinks/2
                   THEN modified[channel[j]-nlinks/2]:=true;
                   ELSE modified[channel[j]+nlinks/2]:=true;
                 END;
              END;
       ELSE j:=j + 1;
  END;
 END heuristic2;
```

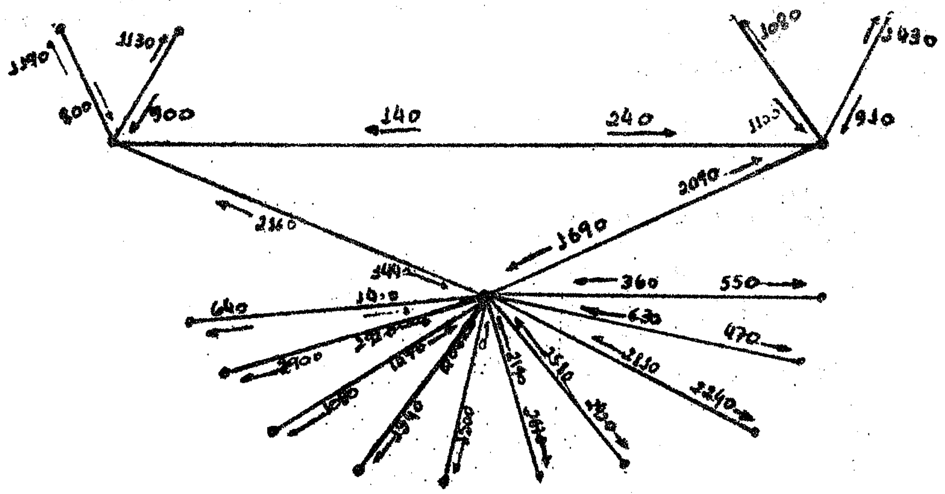
```
COMMENT initialize capacities with the smallest
        capacity in AVAILCAP;

FOR i:=1 STEP 1 UNTIL nlinks DO cap[i]:=1;

COMMENT evaluation of alfa[i,p] for p<pmax;

FOR i:=1 STEP 1 UNTIL nlinks DO
   FOR p:=1 STEP 1 UNTIL pmax DO
      BEGIN
        alfa[i,p]:=0;
        FOR j:=p STEP 1 UNTIL pmax DO
           alfa[i,p]:=alfa[i,p]+lambda[i,j];
        alfa[i,p]:=alfa[i,p]/mu;
      END;

COMMENT evaluation of alfa[i,pmax+1];

FOR i:=1 STEP 1 UNTIL nlinks DO
   alfa[i,pmax+1]:=0;

COMMENT evaluation of tip[i,p];

For i:=1 STEP 1 UNTIL nlinks DO
   BEGIN

     COMMENT evaluation of initial capacity
            (criterium 1);

     IF i <= nlinks/2
     THEN FOR c:=availcap[cap[i]]
                WHILE(1-alfa[i,1]/c < 0
                     | 1-alfa[i+nlinks/2,1]/c<0) DO
            incrementcap(i);
     FOR p:=1 STEP 1 UNTIL pmax DO
        BEGIN
          evaltip(i,p);
          flag:=false;

          COMMENT check condition given by criterium 2;

          FOR t:=tip[i,p] WHILE(lambda[i,p]*t
                         >=gama[p]*tmax[p]) DO
             BEGIN

               COMMENT increment capacity and reevaluate
                       tip;
               incrementcap(i);
               evaltip(i,p); flag:=false;
             END;
           IF flag
           THEN BEGIN
```

```
                     COMMENT reevaluate tip for the
                             priorities < p and for the
                             simplex pair;

                     FOR j:=1 STEP 1 UNTIL p-1 DO
                        evaltip(i,j);
                     IF i > nlinks/2
                     THEN FOR j:=1 STEP 1 UNTIL pmax DO
                             evaltip(i-nlinks/2,j);
                   END;
            END;
      END;

      COMMENT evaluation of tp, constraint checking and
              use of one of the heuristics to increase the
              capacity on some channels;

      evaltip(constraint,flag);
      FOR j:=1 WHILE flag DO
         BEGIN
           heuristic(constraint,modified);
           FOR i:=1 STEP 1 UNTIL nlinks DO
              IF modified[i]
              THEN FOR p:=1 STEP 1 UNTIL pmax DO
                         evaltip(i,p);
           evaltp(constraint,flag);
         END;
      END;
END alocap;
```

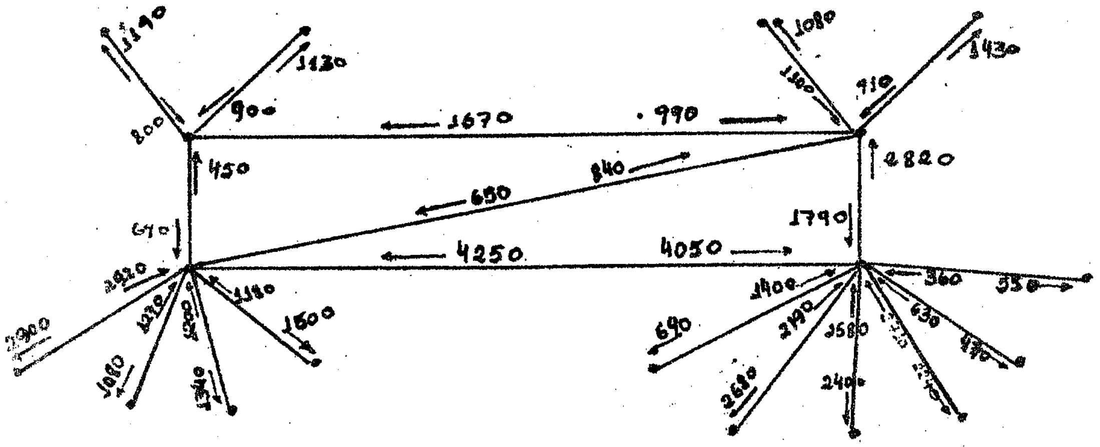
#7 A



TOPOLOGY A



TOPOLOGY B



TOPOLOGY C

All lines represent full-duplex channels.
Values are indicated in msg/day.