Robust and implicitly ”bimodal” collision resolution protocols

Philippe Jacquet

INRIA

UNITE 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° 717

ROBUST AND IMPLICITLY "BIMODAL" COLLISION RESOLUTION PROTOCOLS

Philippe JACQUET

AOUT 1987

ROBUST AND IMPLICITLY"BIMODAL" COLLISION RESOLUTION PROTOCOLS

Philippe Jacquet
INRIA

Abstract

We introduce CSMA/CD protocols which behave like pure packet contention protocols when the input load is under a critical value λ_c. When the load is above λ_c, the design of the protocol lets the system naturally behave like a collision-free mode which allows the high throughputs of TDMA or Token Bus schemes. There is no explicit switch between CSMA/CD mode and collision-free mode : when the load increases, the protocol reduces the collision occurrences but does not suppress them. Thus the protocol is equivalent to a collision-free protocol in terms of high throughputs, but essentially remains a CSMA/CD protocol in terms of high robustness and flexibility.

PROTOCOLES DE COMMUNICATION ROBUSTES A COMPORTEMENT "BIMODAL"

Résumé

Nous présentons des protocoles de communication qui se comportent comme les protocoles à résolution de collision quand la charge du canal n'excède pas une certaine quantité λ_c. Quand la charge est supérieure à λ_c alors la construction de l'algorithme fait que le système tend naturellement à se comporter comme un protocole à évitement de collision ce qui lui permet d'atteindre les hauts débits qu'offrent générallement ce type d'architecture. Il n'y a pas de transition explicite entre le mode résolution de collision et le mode évitement de collision ; en fait, quand la charge augmente, le protocole réduit l'occurence des collisions mais ne la supprime pas. En conséquence le protocole devient équivalent à un protocole à évitement de collision en terme de débits élevés, mais reste essentiellement un protocole de résolution de collision notamment sur le plan de la robustesse et de la flexibilité.

This note introduces the bimodal protocols which resolve most of the problems that we briefly pointed out in the above discussion. In the first section we give a general description of a bimodal protocol. In the second one we outline some performance evaluations. In the third section we separately introduce a bimodal protocol build from the Part & Try Resolution Kernel. In the last section we summarize some numerical results.

II GENERAL DESCRIPTION OF A BIMODAL PROTOCOL

II-1 General Collision Resolution Algorithm

Assume that we have a collision resolution algorithm which proceeds in successive separate sessions, *i.e.* if session i start with the collision of n stations (thus $n \geq 2$), then session i is the time ellapsed for separating this n colliders and nobody else can participate to this session i. A good example of such algorithm is the blocked access tree algorithm [7]. Let L_n be the average session length with n colliding packets. We assume that $L_0 = L_1 = 1$ and the session is degenerate in this case. It will be convenient to introduce the Poisson generating function

$$L(z) = \sum_n L_n \frac{z^n}{n!} e^{-z} ,$$

which is the expected length of session when the number of involved packets is Poisson with mean z.

With this collision resolution algorithm we build a bimodal protocols as following. The collision resolution algorithm is called the ***Resolution Kernel*** of the bimodal protocol.

II-2 Description of the Station Automata

All stations deal with a common counter s, so-called session counter. This counter evoluates according to the feedback of the channel as described below.

1 If a success occurs on the channel, then

$$s \leftarrow s + 1 .$$

2 If a session terminates, then

$$s \leftarrow \sup\{s - 1,\ 1\} .$$

Thus each station needs to monitor the channel from its initialization; but, in case of failure or insertion of a new station, it is easy to broadcast this single integer to new stations. At every time the interval $[0, s)$ is called the *scale* of sessions ***in preparation*** in opposition with the session which is in current resolution. Stations put and drive their packets on the scale before processing the first transmision on the channel.

Every station deals with a local integer parameter, say r_i for the station number i, which is a sort of ***implicit*** reservation. This parameter evoluates as described below.

3 If a successs occurs from station i, then

$$r_i \leftarrow s .$$

(this is possible without reading the packet, because station i *knows* that the current packet is from itself)

4 If a session terminates, then

$$r_i \leftarrow \sup\{r_i - 1,\ 0\} .$$

If station i has a packet in its buffer, the service starts by creating an integer t_i especially for this packet and which is the *allocated place* on the scale.

5 If $r_i > 0$ then

$$t_i \leftarrow r_i ,$$

6 If $r_i = 0$ then t_i is randomly set between 1 and s

$$t_i \leftarrow \text{random}(s) .$$

7 When a session terminates then

$$t_i \leftarrow \sup\{t_i - 1,\ 0\} ,$$

and the packet participates to the new session if $t_i = 0$.

II-3 General Comments and Notations

The scale can be seen as a set of ordered session in preparation, numbered from 1 to s. At every success on the channel a new session is added at the end of the scale. At every termination of session resolution, the session 1 constitutes the new session to be resolved and the other are shifted foreward (the sessions $2, \ldots, s$ respectively become sessions $1, \ldots, s-1$).

For a given scale, exhibited at a given time θ_1), we define its *resolution interval* as the time interval $[\theta_1, \theta_2]$, where θ_2 is the time when the session, which has been pointed out as last session at time θ_1, will enter resolution. Similarily we define the *next* scale – of a given scale – by being the scale at the time θ_2 when the first scale has just been resolved. The session in preparations of this next scale are those which have been created by the feedback of the channel during the resolution interval of the first scale. Furthermore the length of the new scale is exactly the number of packets (1 by default) which have been transmitted during the resolution interval of the former scale.

The interest of this *implicit* reservation scheme [8] (there is no declaration at all) consists in the fact that two different stations i and j ($i \neq j$) which respectively have nonzero reservation places on the scale, r_i and r_j necessarily satisfy $r_i \neq r_j$ (the packets which initiated the reservations were succesfully transmitted on different slots). Thus these stations cannot collide if they both use their reservations. The reservation scheme reduces the probability of collision when the load increases. It is easy to see that, in *saturation* state of the system, each packets finds a predecessor in its queue and thus uses a reservation place. The obvious consequence is the fact that the collision phenomena disappear: the network behaves like a TDMA on active (and saturated) stations and the output load of the channel is exactly 1 packet per slot.

III PERFORMANCE EVALUATION

III-1 Stability in infinite population

Before analyzing the queueing phenomena, we will analyze the behaviour of the system in infinite population of stations ($N = \infty$), namely we wil determine λ_c. Let us first introduce the real function $f(x)$, defined for $x > 0$:

$$f(x) = \frac{L(x)}{x} .$$

Theorem: *If for every λ the equation*

$$\lambda f(x) = x$$

has a single positive root x_λ (the fixed point) and for any positive sequence $\{x_n\}_{n \in N}$, such that

$$x_{n+1} = \lambda f(x_n) ,$$

converges to the fixed point, then

$$\lambda_c = \frac{1}{L(1)} .$$

sketch of proof: It is obvious that the reservation scheme will be of no use when $N = \infty$. Thus the protocol description can be reduced to statements **1**, **2**, **6** and **7**. Thus only the random allocation is considered here. It is also pretty obvious that the stability of the protocol is equivalent to the non divergence of the length s of the scale.

Let us assume that at a certain moment the scale is of length s, s being large. In order to simplify the proof we will assume that

$$\text{random}(s) = s - \lfloor \chi \sqrt{s} \rfloor$$

χ being a random real between 0 and 1. $\sqrt{s}$ is large too.

Let $n_1, \ldots, n_s$ be the number of packets in the respective sessions $1, \ldots, s$ in preparation on the scale. This number is taken when the session will be just to be resolved. Let $l_1, \ldots, l_s$ be the respective lengths of the sessions in preparation when they will be resolved. The number of packet in session i is Poisson of parameter x_i which is the fraction of the input load the session received by the random allocation during its preparation. Thus

$$\text{E}[n_i] = x_i$$

and

$$\text{E}[l_i] = L(x_i) .$$

Let s_i be the length of the scale when session i will enter resolution. Thus

$$x_s = \frac{\lambda l_1}{\sqrt{s_1}} + \cdots + \frac{\lambda l_k}{\sqrt{s_k}}$$

with k being the last integer such that

$$n_1 + \cdots + n_k \leq \sqrt{s_k} .$$

This equations mean that, at every session i, n_i new sessions are created, the distance between former session s (now session $s - i$) increases of n_i and the input load is λl_i which has to be divided among the $\sqrt{s_i}$ last sessions in preparation. We deliberately omitted the fraction of the first current session in resolution and approximated continuous input as geometric input at the termination of every session (the larger s is, more accurate is the approximation)

s being large we can assume that $\sqrt{s}$ does not change very much during these k first sessions ($\sqrt{s}_i \sim \sqrt{s}$). $\sqrt{s}$ being large we can assume that x_i constitutes a sequence of random variable and, according to the Law of Large Numbers

$$k \sim \frac{\sqrt{s}}{\mathrm{E}[x]}$$

and finally

$$x_s = \frac{\lambda \mathrm{E}[L(x)]}{\mathrm{E}[x]} .$$

The planar vector $\Big(\mathrm{E}[x]\, ,\, \lambda\mathrm{E}[L(x)]\Big)$ is included in the convex cone generated by the vectors $\Big(x\, ,\, \lambda L(x)\Big)$, x being all possible values of x_i. The ratio $\lambda\mathrm{E}[L(x)]/\mathrm{E}[x]$ is member of the interval (convex set) generated by the scalars $\lambda L(x)/x$. Let us suppose that $\forall i < s$, $x_i \in [x_{\min}, x_{\max}]$, then the coefficients x of the new created sessions will be member of the new interval $\lambda f\big([x_{\min}, x_{\max}]\big)$. Thus the set of x coefficient of the successive *next* scales will be respectively included in $\lambda f\big([x_{\min}, x_{\max}]\big)$, $\lambda f \circ \lambda f\big([x_{\min}, x_{\max}]\big)$, $\lambda f \circ \cdots \circ \lambda f\Big([x_{\min}, x_{\max}]\Big)$. We know that this sequence of real interval converges to the fixed point $\{x_\lambda\}$.

Now assume that all sessions in the current scale have their x-coefficients identical to x_λ. The average growth of s at every session termination is

$$\mathrm{E}[\Delta s] = x_\lambda - 1 .$$

According to the Lyapunov's function theory, the maximum value of λ for stability must satisfy $\mathrm{E}[\Delta s] = 0$. Thus

$$x_{\lambda_c} = 1$$

which terminates the proof of the theorem.

Remark 1: We can conjecture that the theorem holds for any "regular" procedure random(s), for example

$$\mathrm{random}(s) = \lceil \chi s \rceil$$

but the proof seems harder.

Remark 2: It is interesting to notice that the theorem holds when the input is not Poisson. As a matter of fact, when s is large, the splitting of the input over a large population of sessions, say $\sqrt{s}$, "simulates" a Poisson arrival on each session. this property of the protocol can be put in parallel with [9]

Remark 3: The protocol can deal with packets of different lengths (voice *versus* data). Let $\mathcal{D}$ be the distribution of packet lengths (in slots) and let $\lambda_c(\mathcal{D})$ be the maximum throughput of the protocol in infinite population with the distribution $\mathcal{D}$. It is interesting to notice that $\lambda_c(\mathcal{D})$ is quite insensible to $\mathcal{D}$ and only depends on the mean length T of packet:

$$\lambda_c(\mathcal{D}) = \frac{\lambda_c T}{1 + \lambda_c(T-1)} .$$

Numerical computations provide $\lambda_c = 0.4277$ when the resolution kernel is the blocked access tree collision resolution algorithm, and $\lambda_c = 0.4572$ when the resolution kernel is the "modified" blocked access tree algorithm (0.4624 with an optimally biased splitting probability). It is interesting to notice that the this maximum throughputs are close to the throughputs of the correspondant Interval Searching algorithms [1]: 0.429 for the Tree Interval Searching Algorithm and 0.462 for the Modified Tree Interval Searching Algorithm. Interval Searching algorithms are fairly better than the corresponding bimodal protocols in infinite population, but they do not satisfy remark **2** and **3**, and their bimodal stability is doubtfull.

III-2 The Bimodal Behaviour of the Protocol.

III-2-1 The Equilibrium Equation

In this section we briefly outline the behaviour of the protocol when $\lambda_c \leq \lambda < 1.0$. In such area, queueing phenomena take place and the delays are $O(N)$, N being the number of identical active stations. We consider N large.

The input load being above λ_c the stability of the system is possible iff the reservation scheme is consistently used. Thus s has to be $O(N)$. Therefore s is large too. At every time the total number of sessions (in preparation and already resolved) is obviously greater than the number of successfully transmitted packets. Thus the average number of packets per session is less than 1. In bimodal behaviour the probability of getting $s = O(1)$ is obviously inconsistent, thus we can precise that the average number of packets per session is exactly 1.

Let y be the probability that a random packet use a reserved place. Let x be the probability that a random packet use random allocation; of course $x + y = 1$.

Theorem *: x and y satisfy the Equilibrium Equation:*

$$\frac{1}{\lambda} = L(x) + yL'(x) \ .$$

Proof : According to the relation between number of sessions and number of packets, y is also the probability that a random session involve a packet in reserved place (we know that this packet is unique). The scalar x is also the average number of packet in random allocation in this session. Let us suppose that the line of reasoning we introduced in section **III** can be applied in the bimodal behaviour, namely:

(i) The length of the scale is approximately constant (*i.e.* s/N does not change if N is large).

(ii) the number of randomly allocated packets in any session is *exactly* a Poisson distribution of mean x; independently this session has a probability y for involving a packet on reserved place.

Thus the expected length of a random session is exactly

$$(1-y)\sum_n L_n \frac{x^n}{n!} e^{-x} + y \sum_n L_{n+1} \frac{x^n}{n!} e^{-x} \ ,$$

which can be written in the compacter form

$$L(x) + yL'(x) \ .$$

With an average total number of involved packets of 1, the expected length of a random session is necessarily $1/\lambda$, which terminates the proof of the Theorem.

Remark 4: Thus, for a given λ, there is only one y available, which is largely independent of the random allocation procedure.

Remark 5: if the random procedure splits the input load among a large population of session in preparation, say $\sqrt{s}$ or $O(s)$, then it simulates a Poisson arrival rate on each session in preparation and statements (i) and (ii) hold. Thus the equilibrium equation is also independent of the input process (it depends only on the average rate λ). Thus this equation holds in the case of an asymmetric distribution of traffic on active stations.

Remark 6: The equation is also insensible to the distribution of packet length. In the case of variable length of packet the equation becomes:

$$\frac{T}{\lambda} = T - 1 + L(x) + yL'(x) .$$

III-2-2 The Evaluation of Packet Delay

Sessions are finite, thus the service time of any random packet is mostly the time spent on the scale. According to the fact that $\sqrt{s}/s \ll 1$ we can assume that random$(s) \sim s$ and the service time in random allocation is s/λ (number of sessions $\times$ average length of session).

Let us adjustate the time scaling in such way that s/λ be the time unit. According to a single station the system is equivalent to a queueing with single server submitted to a Poisson load μ. μ satisfyes

$$\mu = \frac{\lambda}{N}\frac{s}{\lambda} = \frac{s}{N} .$$

The service time S is the following

1 If the packet found a predecessor when it arrived in file, then

$$S = 1 \quad (\textit{reserved place}) .$$

2 Otherwise, if the packet arrived t unit of time after the last termination of service then

$$\begin{cases} S = 1 - t \;\; if\; t \leq 1 & (\textit{reserved place}) , \\ S = 1 \;\; if\; t > 1 & (\textit{random allocation}) . \end{cases}$$

The classical analysis of this queueing ([10]) entails the probability that a random packet find an empty queue:

$$\frac{1-\mu}{1+\mu} e^{\mu} .$$

Because of reversibility of the stochastic process [10], this probability is also the probability that a random packet is the last one in its queue. Multiplying this quantity by $e^{-\mu}$, we obtain the probability that a reserved place may be not used, which also is similarily the probability that a packet get into random allocation, namely :

$$x = \frac{1-\mu}{1+\mu} .$$

This last equation provides μ in function of λ and thus s/N.

The mean delay for a random packet is

$$\frac{\mathrm{E}[W]}{N} = \frac{1}{\lambda}\frac{\mu}{1-\mu^2} .$$

When $\mu \sim 1$, *i.e.* when we are near saturation,

$$\frac{\mathrm{E}[W]}{N} \sim \frac{1}{2}\frac{1}{1-\mu} .$$

This has to be compared with the mean delay with an hypothetical TDMA on the active stations, which entails:

$$\frac{\mathrm{E}[W_{\mathrm{TDMA}}]}{N} = \frac{1}{2}\frac{1}{1-\lambda} .$$

According to Remark **2**, we can analyze the simpler case where stations are without buffers (the station cannot store data and must disgard new generated packets which occur during a service). In this case the λ in the equilibrium equation is the output load, λ_{out}, which is slighty different from the input load λ_{in}. The probability that a packet get into random allocation, now is :

$$x = e^{-\mu} ,$$

and the mean delay

$$\frac{\mathrm{E}[W]}{N} = \frac{1}{\lambda_{\mathrm{in}}}\Big((1+\mu)e^{-\mu} - (1-\mu)\Big) .$$

We complete the analysis with the Little formula

$$\frac{N}{\lambda_{\mathrm{out}}} = \mathrm{E}[W] + \frac{N}{\lambda_{\mathrm{in}}} ,$$

λ_{in}/N and λ_{out}/N being respectively the input and output load per station.

III-3 Some General Properties of Bimodal Protocols

1 Determinism: if the collision resolution algorithm is determinist (for example the determinist tree algorithm [7]), then the bimodal protocol also is determinist
Proof: It is sufficient to prove that the length of the scale is upper bounded by a finite scalar $s_{\max}$. In the case where random(s) $= s - \lfloor\chi\sqrt{s}\rfloor$. We can prove that the maximum number of times a station can be allocated on different sessions of a given scale of length s is $\log_2 \log s$. Thus the next scale is upper bounded by $N \log_2 \log s$. Thus

$$N \sim \frac{s_{\max}}{\log_2 \log s_{\max}} .$$

2 High priority for emergency traffic: when the network is submitted to real time constraints [7], we can add a procedure of *emergency allocation* dedicated to emergency packets. In case of emergency the station i set $t_i \leftarrow 1$. Thus its packet will be sent in the next session. If the emergency traffic is so occasionnal that it does not consistently pertube the steady state of the communication process, then the delay of a random emergency packet remains $O(1)$ whatever be the utilization of the channel.

3 the optimal utilization of the channel is available whatever be the distribution of active population. This can be put in opposition with TDMA and Token Bus schemes which provides maximum throughput such as

$$\lambda_{\max} = \frac{N}{N_T}$$

N_T being the total population of stations and N is the active population.

4 High Adaptability to various types or mixture of traffics: it is a *resumé* of the remarks **2**, **3**, **5** and **6** related to the robustness of performance within traffics. For example, it can be insured the stability of the communication process for any traffic whose cumulated load is less than λ_c (or $\lambda_c T/(1+\lambda_c(T-1))$ with average length of packet T), and delays remain finite whatever be the size of the active population.

IV BIMODAL PROTOCOL BUILD FROM PART & TRY ALGORITHM

IV-1 Description

The Part & Try algorithm [11] is a very powerful collision resolution algorithm. The reason why we did not immediately mention it as resolution kernel of a bimodal protocol is the fact that Part & Try algorithm uncompletely resolves sessions. We mean that if a session starts with n initial colliders, then only a fraction of them, say in average w_n, will successfully transmit their packet; the others will be disgarded and must be allocated again in the next sessions. The Poisson generating function

$$w(z) = \sum_n w_n \frac{z^n}{n!} e^{-z}$$

will be of some importance here.

Thus we must add a new statement to the protocol, say the statement number **8**

8 When the packet is disgarded during the current resolution of session, we reallocate it

$$t_i = \text{random2}(s) \ .$$

The procedure random2(s) is not necessary the same than the first one, random(s).

IV-2 Evaluation

The fact that the unstability of this protocol is equivalent to the unstability of the length s of the scale remains true. In fact, if the sessions are saturated, *i.e.* $n \to \infty$, then $w_n \sim 2.505$ [12] (obviously $\lim_{\inf} w_n \geq 2$) and $\mathrm{E}[\Delta s] \sim 2.505 - 1 > 1$. Thus saturation of sessions implyes divergence of the scale.

We can apply the line of reasoning of the previous section, with the difference that

$$\lambda f(x) = \frac{\lambda L(x) + x - w(x)}{w(x)}$$

and

$$\mathrm{E}[\Delta s] = w(x_\lambda) - 1 \ .$$

Consequently

$$w(x_{\lambda_c}) = 1 \ \text{ and } \ \lambda_c = \frac{1}{L(x_{\lambda_c})} \ .$$

Numerical applications give $x_{\lambda_c} = 1.076$ and $\lambda_c = 0.4839$, which favorabily compares with the maximum throughput 0.487 of the ultimate Interval Searching Part & Try Algorithm.

We similarily drive the bimodal evaluation as in the former section. y is the probability that a random packet use a reserved place. Now we have two Equilibrium Equations:

$$\begin{cases} w(x) + y w'(x) = 1 \\ L(x) + y L'(x) = \dfrac{1}{\lambda} \end{cases} .$$

Note that $x + y \neq 1$. In general $x + y$ is greater than 1 and denotes the average number of sessions that a random packet uses before being successfully transmitted instead of being disgarded.

In order to simplify the evaluation of packet delay we will set

$$\text{random2}(s) = \lceil \chi\sqrt{s} \rceil$$

in order to maintain the fact the delay of a packet is mostly the time spent during its *first* sojourn on the scale. The results of the previous section hold.

IV-3 General properties of the Bimodal Part & Try Protocol

1 Determinism: it is possible to get determinism if one insures an upper bound of the number of retry for any packet.

2 High priority for emergency traffic.

3 The optimal utilization is available whatever be the distribution of active population.

4 High Adaptability to various types of traffic mixes.

5 High robustness to channel error: *a priori* the Part & Try algorithm, as the modified tree algorithm, deadlocks if an empty slot is misinterpreted as a collision. In basic tree algorithm, if a collision occurs, the initial colliding population is split into two sub-populations. If a blank occurs after the collision, that means that the first sub-population is empty and the initial population is in totality in the second sub-population. Thus the second sub-population will *necessarily* experience a collision (a collision which follows the blank). The Modified Tree [1] and the Part & Try avoid this collision by skipping the slot by immediately splitting the second sub-population. This "trick" allows high throughput but drastically reduces reliability. In order to avoid such deadlocks there are simple versions of the Part & Try, called b-Part & Try [13] Algorithms, in which, only b skipped slot are allowed in each session, b being an integer. When b increases, the performances are asymptotically thoses of Part & Try ones (which formally corresponds to $b = \infty$) and the robustness remains optimal (the algorithms can endur a large spectrum of random noises). The bimodaal protocols build from $b = 0, 1, 2, 3, 4$ and 5 respectively entail $\lambda_c = 0.4487, 0.4712, 0.4795, 0.4824, 0.4834$ and 0.4837.

6 Easy insertion of new station or reinsertion of stations after failure: the b-Part & Try algorithm described above, when $b < \infty$ have the additionnal property [13, 14] that a station can detect the end of the current session without any knowledge of the past events on the channel. Thus the insertion of new stations simply reduces to the query of s which can be broadcasted by any active station (reinitiation). This property is fairly shared with Tree and Modified Tree algorithms.

V NUMERICAL RESULTS

The first figure is a direct illustration of the equilibrium equation. It shows the probability y for a random packet to issue on a reserved place as a function of the output load λ. The three curves are respectively plotted from the respective bimodal protocols build with (from top to bottom) Basic Tree, Modified Tree and Part & Try algorithms. Note that $y = 0$ when $\lambda \leq \lambda_c$.

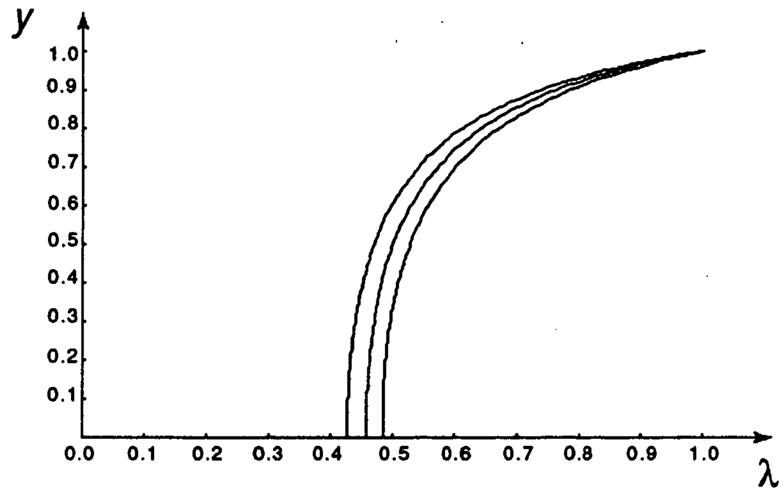

fig. 1: reservation *vs* output load

The second figure shows the mean delay of a random packet divided by the mean delay it should have experienced with a TDMA on active stations, both as function of λ. Note that if the active population is not the total population, but only a fraction of it, this virtual TDMA is quite hypothetical. From top to bottom: Basic, Modified Tree and Part & Try. The dotted line is the TDMA. Note that the limiting value attained by $E[W]/E[W_{TDMA}]$, when $\lambda \rightarrow 1.0$, is exactly $(L_2 - 1)/2$, which can be proven through asymptotic expansion of the Equilibrium Equation near $\lambda = 1.0$.

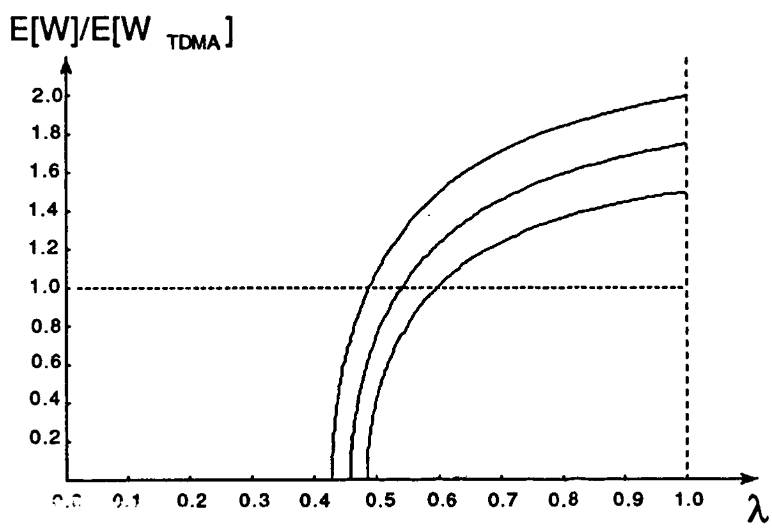

fig. 2: delay vs load

The third figure shows the average delay divided by the number of active stations in the particular case where all stations are without buffer. Note that $\lambda_{\text{in}} = \lambda_{\text{out}}/(1 + \lambda_{\text{out}} E[W]/N)$. From top to bottom: Basic, Modified Tree and Part & Try.

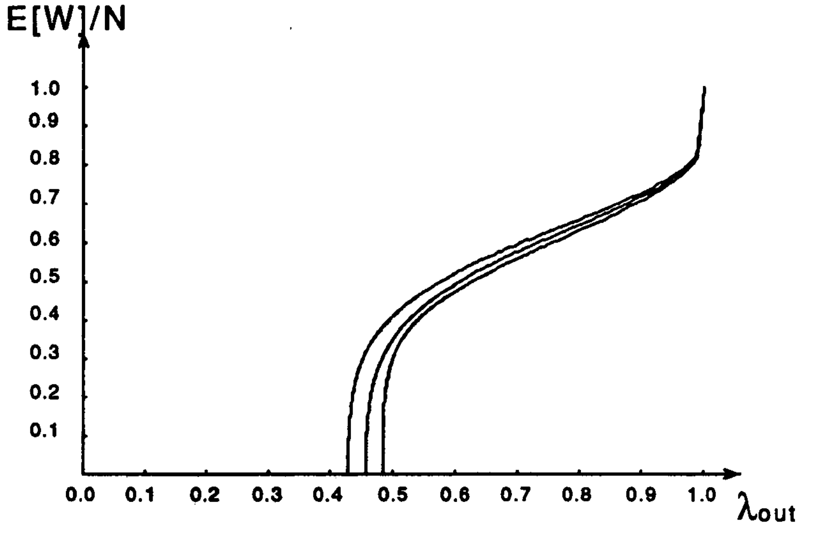

fig 3: no buffer case, delay *vs* output load

The next three figures are not directly derived from the analysis of the paper *in stricto situ*. They give for the three protocols the set of attainable throughputs when the population of active stations is divided into two identical sub-populations respectively submitted to uniform loads λ_1 and λ_2. It is a derivation of a work in [15] on asymmetric load. We can generally assume that the set of active population is divided in m sub-population, of respective size $N_1, \cdots, N_m$ (with $N_1 + \cdots + N_m = N$) such that each station in sub-population i is submitted to a λ_i/N Poisson load. Lets note $\sigma_i = N_i/N$. Thus

$$\lambda = \sum_{i=1}^{m} \sigma_i \lambda_i .$$

We can drive the analysis by introducing x_i being the probability for a packet issued from population i to get in random allocation, and $y_i = 1 - x_i$. The Equilibrium Equation holds with λ, x and y such that

$$x = \frac{\sum_{i=1}^{m} \sigma_i \lambda_i x_i}{\lambda}$$

and

$$y = \frac{\sum_{i=1}^{m} \sigma_i \lambda_i y_i}{\lambda} .$$

The analysis of each queue holds with

$$\mu_i = \frac{s\,\lambda_i}{\lambda\,N}$$

and entails

$$x_i = \frac{1 - \mu_i}{1 + \mu_i} .$$

There are $2m + 1$ equations for $2m + 1$ unknown variables (x_i, μ_i and s). The figures below are particular case where $m = 2$ and $\sigma_1 = \sigma_2$.

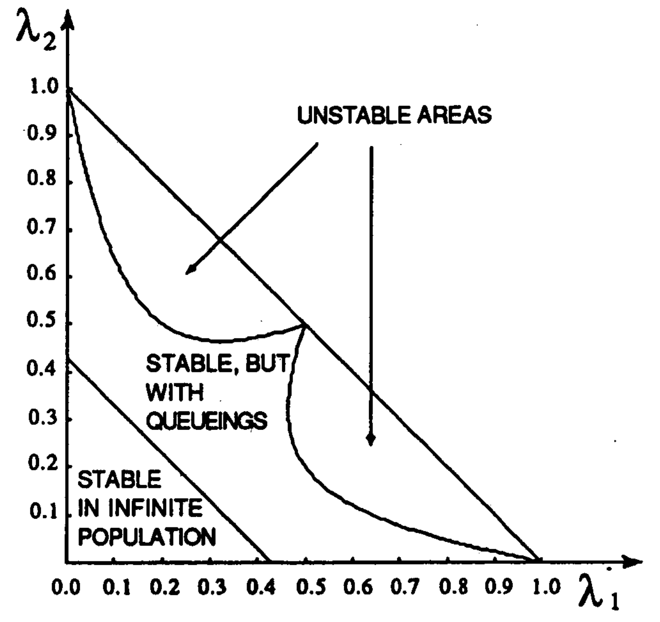

fig. 4: attainable throughputs for Basic Tree Kernel Protocol

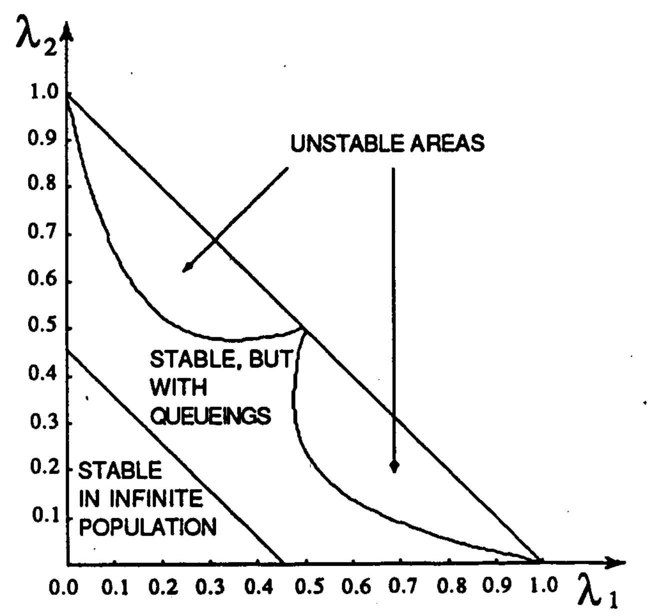

fig. 5: attainable throughputs for Modified Tree Kernel Protocol

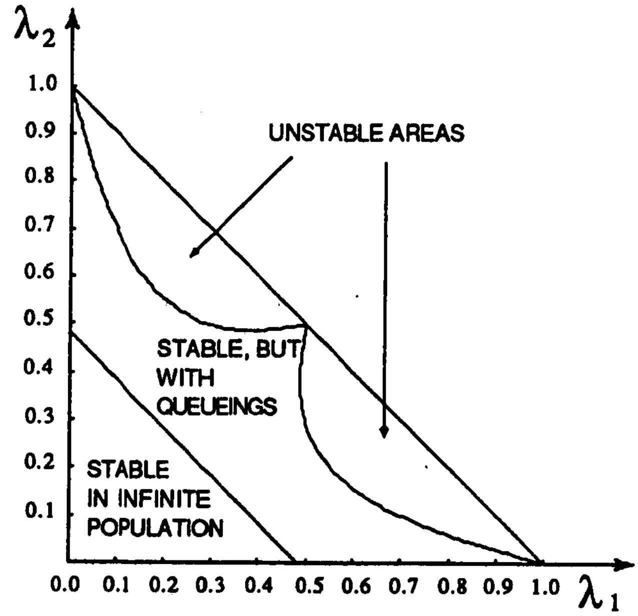

fig. 5: attainable throughputs for Part & Try Kernel Protocol

Note that optimal throughputs slightly degrade in assymetric cases but remain largely above λ_c. The attainable set of throughputs from the hypothetical TDMA on the active stations should be the square $\lambda_1 < 0.5$ and $\lambda_2 < 0.5$, and without stability in infinite population.

References

[1] J.L. Massey, "Collision-resolution algorithm and random-access communications," in *Multi-User Communication System*, G. Longo Ed., CISM Courses and Lectures, No. 265. New York: Springer, 1981, pp. 73-137.

[2] P.M. Gopal and J.W. Wong, "Analysis of a Hybrid Token-CSMA/CD Protocol for Bus Network," *Performance of Computer-Communication Systems*, H. Rudin and W. Bux Ed., pp. 65-78, North Holland, Amsterdam, 1984.

[3] M. Rios and N.D. Georganas, "A Hybrid Multiple-Access Protocol for Data and Voice-Packet over Local Area Networks," *IEEE Trans. on Computers*, Vol C-34, No 1, pp. 90-94, 1985.

[4] G. Fayolle, E. Gelembe and J. Labetoulle, "Stability and optimal control of the packet switching broad-

cast channel," *JACM*, Vol. 24, pp. 375-386, 1977.

[5] J. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inform. Theory*, Vol. IT-25, No. 5, pp. 505-515, 1979.

[6] J. Capetanakis, "Generalized TDMA: The multi-accessing tree protocol," *IEEE Trans. Commun.*, Vol. COM-27, No. 10, pp. 1476-1484, 1979.

[7] P. Flajolet and P. Jacquet, "Analytic Model for Tree Communication Protocols, *Flow Control of Congested Networks*, NATO Advanced Study Institute, Capri, Proceedings published by Springer, New York, 1987.

[8] P. Jacquet and P. Muehlethaler, " Machnet: protocole pour réseaux hauts débits," *INRIA*, Res. Rep., 1987.

[9] A. Greenberg, P. Flajolet and R. Ladner, "Estimating the multiplicities of conflict in multiple access channels," *INRIA*, Res. Rep., 1984.

[10] F.P. Kelly, "Reversibility and Stochastic Networks," John Wiley, New York, 1979.

[11] V.A. Mikhailov and B.S. Tsybakov, "Random Multiple Access: Part and Try Algorithm," *Problemy Peredachi Informatsii*, Vol. 16, No. 4, pp. 65-79, 1980.

[12] P. Jacquet and W. Szpankowski, "Ultimate Characterizations of the Burst response of the Interval Searching Algorithm," *To be published*, 1987.

[13] P. Jacquet, "The Part and Try Algorithm adapted to free channel access," *INRIA* Res. Rep., No. 436, 1985.

[14] P. Humblet, "On the throughput of Channel Access Algorithms with Limited Sensing," *IEEE Trans. Commun*, COM-34, pp. 345-347, 1986.

[15] P. Jacquet, "Evaluation des files d'attente pour le protocole ALOHA," *INRIA*, Res. Rep., 1987.

[16] R.G. Gallager, "Conflict resolution in random access broadcast networks," in *Proc. AFORS Workshop Communications Theory and Applications*, Provincetown, MA, pp. 74-76, 1978.

[17] G. Ruget, "Some tools for the study of channel-sharing algorithms," in *Multi-User Communication System*, G. Longo Ed., CISM Courses and Lectures, No. 265. New York: Springer, 1981, pp. 201-231.

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