# The part and try algorithm adapted to free channel access

Philippe Jacquet

INRIA

CENTRE DE ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél. (3) 954 90 20

Rapports de Recherche

N° 436

# THE PART AND TRY ALGORITHM ADAPTED TO FREE CHANNEL ACCESS

Philippe JACQUET

Août 1985

RESUME

Les algorithmes de résolution de collisions les plus performants sont très vulnérables aux erreurs du canal et aux pannes de stations. Nous présentons des versions plus robustes du protocole Part and Try de Gallager, Mikhailov, Tsybakov. Ces algorithmes sont adaptés de telle manière que les stations n'ont à écouter le canal que lorsque elles ont un paquet à transmettre. En outre les protocoles décrits peuvent se rapprocher arbitrairement près d'une capacité maximum de 0,487 paquets par slot.

ABSTRACT

High throughput collision resolution algorithms (CRA) are disadvantaged by a high sensitivity to channel or station failure. In this paper we introduce robuster versions of the Gallager Mikhailov Tsybakov Part and Try algorithm adapted in such a way that stations need to monitor the channel only when they are active. These algorithms can be tuned to attain throughput arbitrarily close to the upper value. 487.

# THE PART AND TRY ALGORITHM ADAPTED TO FREE CHANNEL ACCESS

*Philippe Jacquet*

INRIA
Domaine de Voluceau, Rocquencourt
78150-Le Chesnay (France)

High throughput collision resolution algorithms (CRA) are disadvantaged by a high sensitivity to channel or station failure. In this paper we introduce robuster versions of the Gallager Mikhailov Tsybakov Part and Try algorithm adapted in such a way that stations need to monitor the channel only when they are active. These algorithms can be tuned to attain throughput arbitrarily close to the upper value .487.

## I INTRODUCTION

We consider the problem of organizing communication on a *random access channel*. A population of transmitters (geographically distributed) shares a common *slotted-time* channel on which they are synchronized. Transmitters are constrained to transmit data in form of *packets* whose length is one time slot, and the beginning of every transmission must coincide with the beginning of a slot. When more than one transmitter sends a packet in the same slot a *collision* occurs and all packets are destroyed. Thus any station is aware of the channel feedback at the end of each slot, which may be:

| | | |
|---|---|---|
| blank | : | no transmission |
| success | : | only one transmitted |
| collision | : | two or more stations just intended to transmit |

A collision resolution algorithm (CRA) is an algorithm followed by any (colliding) transmitter which ensures a successful transmission of its packet. The input of this algorithm is only the ternary feedback of the channel which is supposed exempt from errors.

In practice, different stations will maintain a queue of waiting messages which have to be sent in whatever priority order. But the analysis of CRA is classically made under the assumption of an infinite population of identical transmitters and assuming that the packet generation be a Poisson process. Let $\lambda$ be the Poisson parameter. The queuing problems in each station are negligible: a station is assumed to have at most only one packet to transmit at every time: so the generation of $k$ packets is equivalent to the activation of the same number $k$ of stations.

The first algorithm proposed was ALOHA by Abramson (1970) which later appeared unstable without an external control. In general the *stability* of a CRA is the crucial notion: a CRA is stable if it ensures to every colliding packet a finite delay - in the meaning of the Theory of Probability - until a successful transmission. The Capetanakis - Tsybakov - Mikhailov tree algorithm opened in 1978 a large class of CRA whose stability was ensured when the Poisson arrival rate $\lambda$ is below some critical value. Such a critical value is called the maximum throughput of the CRA (see [Ma81] and [Ma85], for general references).

At the time of this writing the most efficient algorithm is the *part and try* algorithm or so-called .487 algorithm. It has been independently introduced by

Tsybakov, Mikhailov and by Gallager in 1979. This protocol is a member of the class of *window* tree algorithms which is of course a subclass of tree algorithms.

All known CRA's with throughput higher than .456 ([GK85]), the part and try algorithm and all window algorithms, require every station to continuously monitor the channel even when they have no packet to transmit. This fact presents the disadvantage that it is difficult to restore a station after either a failure or a disconnection. A more robust class of algorithm has been introduced as so-called limited feedback sensing algorithms or *free access* algorithms in order to overcome this difficulty. In a free access algorithm a station has to monitor the channel only when active and can switch off after its packet has been successfully transmitted. Uncontrolled ALOHA, and Ethernet are free access algorithms. We know the former to be unstable and the latter poses difficulty as to analysis and ergodicity. The only completely analysed algorithms of this class (maximum throughput, mean and standard deviation or higher moment of session-length, mean and standard deviation or higher moment of delay transmission, state of channel, state of the stack...) are the free access versions of the Capetanakis - Tsybakov - Mikhailov tree algorithms with binary or Q-ary branching ([J83], [FFHJ85], [MF85]). None of this algorithms however has throughput higher than .425 ([GK85]).

In this paper we describe a new set of free access algorithms directly adapted from the part and try algorithm. These algorithms depend on a design parameter that can be chosen so as to attain a maximum throughput arbitrarily close to .487.

In part II we define and analyze the general class of the window tree protocols as described in [Ma81]; we focus on the part and try algorithm (MT80]). In part III we show how this algorithm (in fact its basic version) can be adapted to the free access context. In part IV we introduce the new algorithms *e-part and try*. We analyse the ergodicity of these algorithms.

## II THE PART AND TRY ALGORITHM

### I 1. The window tree algorithms

Every station maintains a *window*, representing a time interval, which is sliding from the past to the present; all stations, having monitored the channel continuously since its initialization, will at any given time have the same window. The transmission proceeds by stages, called here *sessions*.

At session $i$, starting at the (current) time $s_i$, all messages arrived between time $w_i$ and time $w'_i$ ($w_i \le w'_i \le s_i$), will contend for transmission on the channel using a blocked access protocol (here the Capetanakis tree protocol). Quantity $w'_i$ is defined as

$$w'_i = \inf\{w_i+\tau, s_i\} .$$

This means that during the $i$-th session all messages in the window of length $\tau$ (possibly truncated at the current time) starting at $w_i$ will transmit. Let $W$ be the name of the current window. Once all collisions have been resolved, session $(i+1)$ can start; one then takes $w_{i+1} = w'_i$ and the process is repeated. The process can be arbitrarily initialized, but the more natural way is to take $w_0 = 0$ and $s_0 = w_0$, or $s_0 = w_0+1$, or $s_0 = w_0+\tau$. In the following illustration (fig 1) we describe the evolution of the *non resolved* time interval $[w_i, s_i]$ during an arbitrary $i$-th session (the two axis show the situation respectively at the beginning and at the end of the session of rank $i$).

A session will be degenerate if no packet or just one packet is contained in $W$. The length of such a session is of course exactly one slot. Note $W^{\#}$ the number of packets contained by the window $W$. In case $W^{\#} \geq 2$, the session will begin with a collision and the session will be non degenerate. Note that $\tau$ is a real parameter to be optimized and is not necessarily an integer number of slots. The duration of a slot is taken as the time unit.

The resolution of packets is the tree algorithm with *deterministic* splitting according to the precise arrival dates as following.

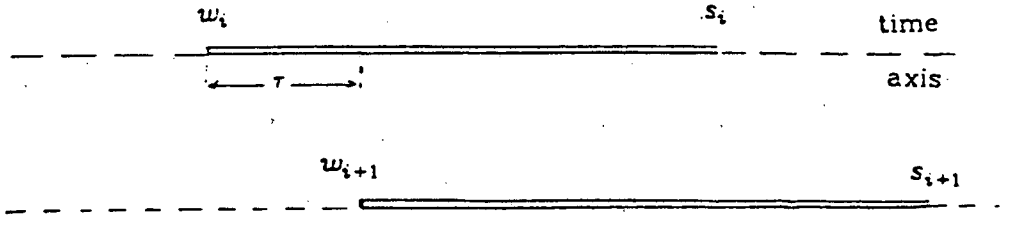
```
procedure Transmission(W:time interval);
(*all packet with arrival date in W transmit in this slot*)

procedure Resolution(W);
begin
    Transmission(W);
    if feedback=blank or success then return degenerate;
    if feedback=collision then Splitting(W)
end;

procedure Splitting(W);
begin
    Resolution(head(p,W));
    Resolution(tail(q,W));
end;
```

In the above specification, we have used the functions head and tail as defined below.

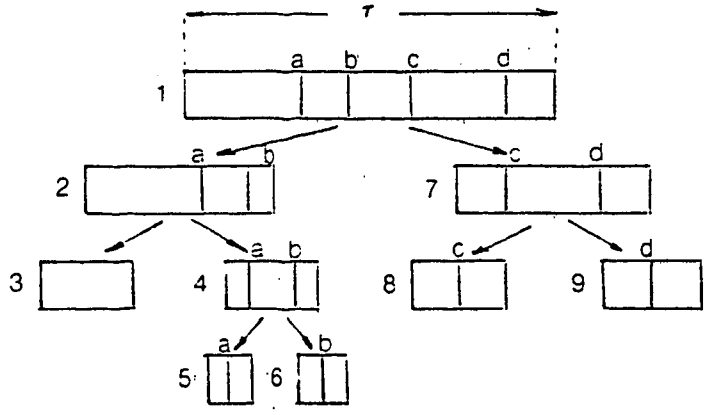
$$W = [a,b] \qquad \text{head}(p,W) = [a, a+p(b-a)]$$
$$\text{tail}(q,W) = [b+q(a-b), b],$$

where $p$ and $q$ are positive real numbers satisfying $p+q = 1$. These are design parameters called the *bias* of the tree algorithm. The length of the session is called the *Collision Resolution Interval* (CRI).

We give an illustration of the window tree CRA (fig 2): a, b, c and d are the packets contained in the window; The scheme shows the successive different splittings of the window according to the tree protocol. We first give the tree representation followed by the stack representation with a table containing the status of successive slots.




(fig 1)




| slots | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| channel | × | × | | × | ↑ | ↑ | × | ↑ | ↑ |
| stack | abcd | ab | ∅ | ab | a | b | cd | c | d |
| | - | cd | ab | cd | b | cd | - | d | - |
| | - | - | cd | - | cd | - | - | - | - |

(fig 2)

We have used the conventional notations:

" ": blank
"↑": success
"×": collision

The chronological order of the successive slots of the CRI can be obtained by the reading in prefix order of the tree.

### I 2. Analysis of tree algorithm

If we condition by $W^{\#} = n$, the distribution of the packets arrival times in a random window is uniform and independent of the past and the future because of the *markovian* memoryless property of the process arrival. So the deterministic splitting according to the arrival date in the window is statistically equivalent to a *Bernoulli random splitting* with conjugate probability $p$ and $q$ to which every packet of the window would be submitted.

So, still with $W^{\#} = n$

$$\text{Proba}(\text{head}(p,W)^{\#} = k \;/\; W^{\#} = n) = \binom{n}{k} p^k q^{n-k}$$

$$= \text{Proba}(\text{tail}(q,W)^{\#} = n-k \;/\; W^{\#} = n)\,.$$

Note $X_n = E[CRI \;/\; W^{\#} = n]$ the conditional mean of CRI length; we have, according to FH ]].

$$\begin{cases} X_n = 1 + \sum_k \binom{n}{k} p^k q^{n-k} (X_k + X_{n-k}) & n \geq 2 \\ X_0 = X_1 = 1 \end{cases}$$

We could directly resolve this recursion, but we can use a more compact form by introducing the so-called Poisson generating function:

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

and we obtain the functional equation:

$$X(z) = X(pz) + X(qz) + 1 - 2(1+z)e^{-z} ,$$

or equivalently:

$$X(z) - 1 = X(pz) - 1 + X(qz) - 1 + 2 - 2(1+z)e^{-z} ,$$

which can be resolved by iteration:

$$X(z) = 1 + \sum_{n \in N} S^n \left(2 - 2(1+z)e^{-z}\right) ,$$

with $Sf(z) = f(pz) + f(qz)$ being a linear operator on the space of the holomorph function $f(z)$ in the complex plan. See the appendix for convergence of iterations. The next table shows the numerical results computed for $p = q = 0.5$.

| $n$ | $X_n$ |
|---|---|
| 0 | 1.00000 |
| 1 | 1.00000 |
| 2 | 5.00000 |
| 3 | 7.66667 |
| 4 | 10.52381 |
| 5 | 13.41905 |
| 6 | 16.31306 |
| 7 | 19.20092 |
| 8 | 22.08535 |
| 9 | 24.96901 |
| 10 | 27.85318 |

### I 3. Stability of the window tree algorithms

We know the distributions of packets in each window to be mutually independent also $W^{\#}$ obey the law of Poisson of parameter:

$$\lambda \times \text{size of the window} .$$

So the discrete time process of the interval lengths $s_i - w_i$ is a continuous state Markov process with the (irreducible) transitions:

* If $s_i - w_i > \tau$

  The CRI, $s_{i+1} - s_i$ is conditioned by the distribution of the number of packets arrived in the window, a quantity itself dependent on the parameter $\lambda$ of the law of arrivals and by the size of the window, namely a Poisson law of parameter $\lambda\tau$.

$$s_{i+1} - w_{i+1} = s_i - w_i + CRI(\lambda\tau) - \tau ,$$

  where $CRI(x)$ is the random variable of the CRI length when the number of colliding packets obeys to the Poisson law of parameter $x$.

* If $s_i - w_i < \tau$

$$s_{i+1} - w_{i+1} = CRI(\lambda(s_i - w_i)) .$$

A sufficient condition for instability of the process is:

$$\lim_{l\to\infty} \inf E[s_{i+1}-w_{i+1} \,/\, s_i-w_i=l]-l > 0$$

(all states are transient, the window statistically cannot reach the present if we start the process with $w_0$ arbitrarily far from $s_0$)

A sufficient condition for stability is also:

$$\lim_{l\to\infty} \sup E[s_{i+1}-w_{i+1} \,/\, s_i-w_i=l]-l < 0 \ .$$

This last condition can be expressed, according to the transition conditions mentioned above, namely:

$$E[s_{i+1}-w_{i+1} \,/\, s_i-w_i>\tau] = X(\tau\lambda)-\tau+s_i-w_i \ .$$

Thus we get stability if $\tau > X(\tau\lambda)$, instability otherwise. Thus the maximum throughput achieved by the window tree algorithm is

$$\lambda_{max} = \max_{x\in R^+}\{\frac{x}{X(x)}\} \ .$$

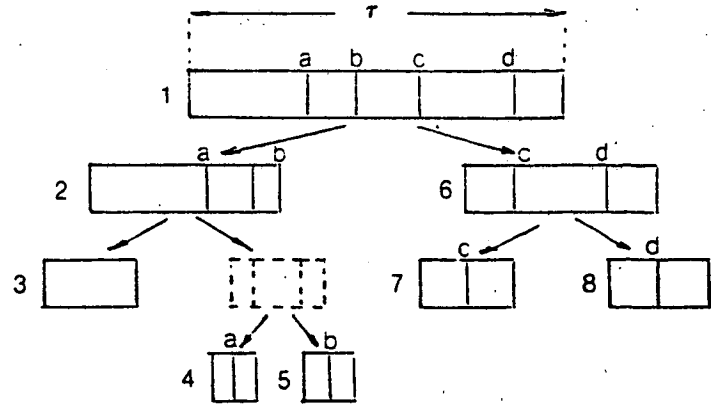
Numerically we obtain $\lambda_{max}$ = .429, with $x$ = 1.15: the choice of the optimal $\tau$ is 2.7 (see [Ma81]).

**The modified algorithm**

We can improve the CRI by suppressing *certain to occur* collisions: if a blank occurs after a collision that means that all packets are included in the tail part of $W$ and we split immediately this tail before the next slot.

```
procedure Modified Resolution(W);
begin
    Transmission(W);
    if feedback=blank then return degenerate-or-c-o-collision;
    if feedback=success then return degenerate;
    if feedback=collision then Splitting(W)
end;

procedure Modified Splitting(W);
begin
    Modified Resolution(head(p,W));
    if degenerate-or-c-o-collision then Modified Splitting(tail(q,W))
    else
    Modified Resolution(tail(q,W))
end;
```

For example, in the illustration (fig 3) we avoid the certain to occur collision "ab" just after $\emptyset$ and we improve the value of the CRI length to 8 instead of 9 previously.




| slots | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| channel | × | × | | ↑ | ↑ | × | ↑ | ↑ |
| stack | abcd | ab | ∅ | a | b | cd | c | d |
| | - | cd | ab | b | cd | - | d | - |
| | - | - | cd | cd | - | - | - | - |

(fig 3)

The modification results in a slight modification in the analysis of the conditional mean of the CRI:

$$\begin{cases} X_n = 1+\sum_k \binom{n}{k} p^k q^{n-k}(X_k+X_{n-k}) - q^n & n\geq 2 \\ X_0 = X_1 = 1 \end{cases}$$

whence a new equation:

$$X(z) = X(pz) + X(qz) + 1 - 2(1+z)e^{-z} - e^{-pz} + (1+qz)e^{-z},$$

which can again be resolved by iteration

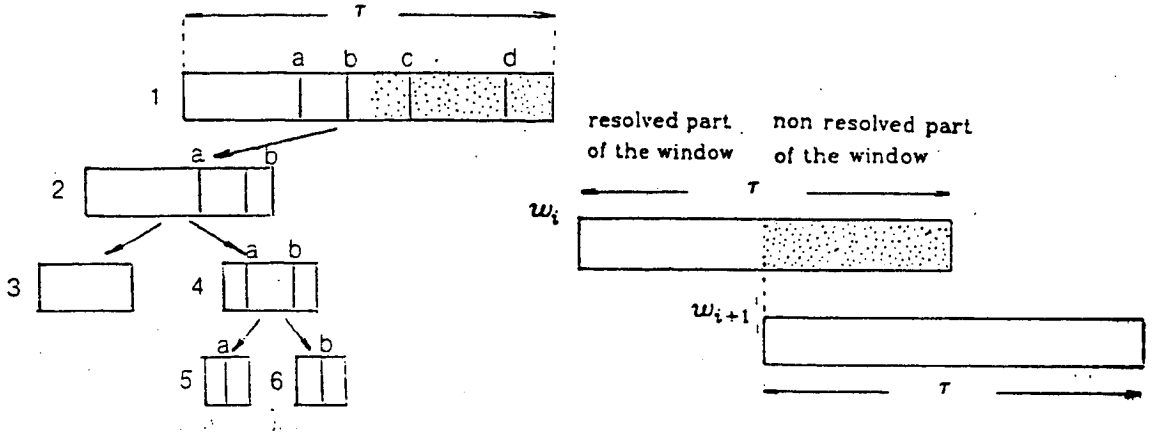
$$X(z) = 1 + \sum_{n\in N} S^n(2 - 2(1+z)e^{-z} - e^{-pz} + (1+qz)e^{-z}).$$

Numerically we find now $\lambda_{max}$= .462, and the computations gives the following table with $p = q = 0.5$.

| $n$ | $X_n$ |
|---|---|
| 0 | 1.00000 |
| 1 | 1.00000 |
| 2 | 4.50000 |
| 3 | 7.00000 |
| 4 | 9.64286 |
| 5 | 12.31429 |
| 6 | 14.98479 |
| 7 | 17.65069 |
| 8 | 20.31407 |
| 9 | 22.97676 |
| 10 | 25.63990 |

II 4. **The part and try algorithm**

The part and try algorithm is a full member of the class of window tree algorithms. It is based on the following "trick" (see [Ma80]). If a collision is followed immediately by another collision, then one has obtained no information about the number of packets in the tail of the current window. Thus, this tail can be merged into the non resolved interval of the time axis rather than explored as determined in the previous tree algorithm. Put in an other way we develop the resolution tree in prefix order until all encountered collisions have been justified by combination a of the first encountered leave of the tree.




(fig 4)

For example (see fig 4), the isolation of a and b in respectively head($\frac{1}{2}$, tail($\frac{1}{2}$, head($\frac{1}{2}$, W))) and tail($\frac{1}{2}$, tail($\frac{1}{2}$, head($\frac{1}{2}$, W))) is sufficient to justify collisions in slot (1),(2) and (4) without c and d.

This means that the distribution of packets in the *non resolved part* in the window is independent of the already *resolved part* and can be included in the following window. Because of the memoryless markovian property of the arrival process, the distribution of packets in the next new window remains independent of the past and is only conditioned by the Poisson law of arrivals. The resolution algorithm is:

```
procedure Part-and-Try-Resolution(W);
begin
    Transmission(W);
    if feedback=blank or success then return degenerate;
    if feedback=collision then Part-and-Try-Splitting(W)
end;

procedure Part-and-Try-Splitting(W);
begin
    Part-and-Try-Resolution(head(p,W));
    if degenerate then Part-and-Try-Resolution(tail(q,W))
end;
```

If it is non-degenerate, the session ends after two consecutive successes.

II 5. **Analysis of the part and try algorithm**

Let $\tau W_n$ be the mean of the size of the resolved part of the window conditioned by $W^{\#} = n$:

$$\tau W_n = \mathrm{E}[w_{i+1} - w_i \;/\; W^{\#} = n] .$$

We have the recurrence equation:

$$\begin{cases} W_n = \sum_k \binom{n}{k} p^k q^{n-k} p W_k + q^n q W_n + npq^{n-1} q W_{n-1} & n \geq 2 \\ W_0 = W_1 = 1 \end{cases}$$

So, with $W(z) = \sum_n W_n \frac{z^n}{n!} e^{-z}$:

$$W(z) = pW(pz) + qW(qz)(1+pz)e^{-pz} .$$

And, for the CRI, the new equation:

$$\begin{cases} X_n = 1 + \sum_k \binom{n}{k} p^k q^{n-k} X_k + q^n X_n + npq^{n-1} X_{n-1} & n \geq 2 \\ X_0 = X_1 = 1 \end{cases}$$

$$X(z) = X(pz) + X(qz)(1+pz)e^{-pz} + 1 - 2(1+z)e^{-z} .$$

**Resolution** : we write

$$(X(z)-1) = (X(pz)-1) + (X(qz)-1)(1+pz)e^{-pz} + 1 + (1+pz)e^{-pz} - 2(1+z)e^{-z} ,$$

and note $w(z) = zW(z)$,

$$(w(z)-z) = (w(pz)-pz) + (w(qz)-qz)(1+pz)e^{-pz} - qz(1-(1+pz)e^{-pz}) .$$

Then

$$X(z) = 1 + \sum_{n \in N} T^n (1 + (1+pz)e^{-pz} - 2(1+z)e^{-z}) ,$$

$$w(z) = z + \sum_{n \in N} T^n (-qz(1-(1+pz)e^{-pz})) .$$

With $T$ linear operator, $Tf(z) = f(pz) + f(qz)(1+pz)e^{-pz}$, - see again the appendix for convergence considerations. The following table summarizes the results for $p = q = 0.5$.

| $n$ | $X_n$ | $nW_{n-1}$ |
|---|---|---|
| 0 | 1.00000 | 0.00000 |
| 1 | 1.00000 | 1.00000 |
| 2 | 4.50000 | 2.00000 |
| 3 | 6.50000 | 2.50000 |
| 4 | 7.14286 | 2.57143 |
| 5 | 7.31429 | 2.52381 |
| 6 | 7.47442 | 2.49770 |
| 7 | 7.66859 | 2.49578 |
| 8 | 7.86056 | 2.50079 |
| 9 | 8.03335 | 2.50523 |
| 10 | 8.18488 | 2.50748 |

**Stability**

We appeal to the general condition for stability of window algorithms:

instability:

$$\liminf_{l\to\infty} \mathrm{E}[s_{i+1}-w_{i+1} \,/\, s_i-w_i=l] - l > 0\,,$$

stability:

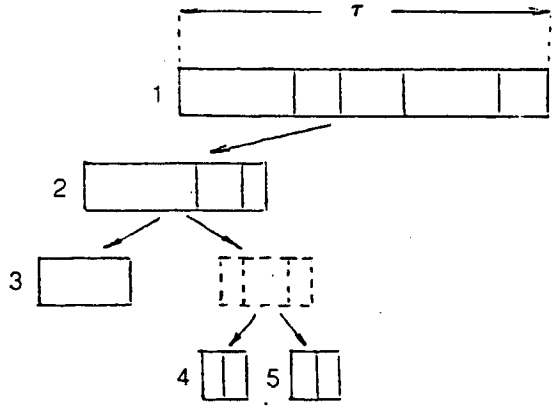
$$\limsup_{l\to\infty} \mathrm{E}[s_{i+1}-w_{i+1} \,/\, s_i-w_i=l] - l < 0\,.$$

Thus we have stability if and only if $\tau W(\lambda\tau) > X(\lambda\tau)$. Thus the maximum throughput is:

$$\lambda_{\max} = \max_{x\in R^+}\{\frac{w(x)}{X(x)}\}\,.$$

In fact the algorithm described is the *basic* part and try algorithm which gives only (!) $\lambda_{max}$ = .453 (with a bias: $p$ = .55). The *real* .487 algorithm is the improvement of the basic part and try algorithm by avoidance of the certain to occur collisions:

```
procedure Resolution-.487(W);
begin
    Transmission(W);
    if feedback=blank then return degenerate-or-c-o-collision;
    if feedback=success then return degenerate;
    if feedback=collision then Splitting-.487(W)
end;

procedure Splitting-.487(W);
begin
    Resolution-.487(head(p,W));
    if degenerate-or-c-o-collision then Splitting-.487 (tail(q,W));
    if degenerate then Resolution-.487(tail(q,W))
end;
```




(fig 5)

The CRI length leads to a new equation

$$\begin{cases} X_n = 1+\sum_k \binom{n}{k} p^k q^{n-k} X_k + q^n (X_n - 1) + npq^{n-1} X_{n-1} & n \geq 2 \\ X_0 = X_1 = 1 \end{cases}$$

and

$$X(z) = X(pz) + X(qz)(1+pz)e^{-pz} - e^{-pz} + 1 - 2(1+z)e^{-z} + (1+qz)e^{-z} .$$

$$X(z) = 1 + \sum_{n \in N} T^n (1 + (1+qz)e^{-z} - 2(1+z)e^{-z} + pze^{-pz}) .$$

By numerical computations we obtain the new table($p = q = 0.5$):

| $n$ | $X_n$ |
|---|---|
| 0 | 1.00000 |
| 1 | 1.00000 |
| 2 | 4.00000 |
| 3 | 5.83333 |
| 4 | 6.47619 |
| 5 | 6.66984 |
| 6 | 6.83625 |
| 7 | 7.02863 |
| 8 | 7.21804 |
| 9 | 7.38911 |
| 10 | 7.54055 |

Numerically we obtain $\lambda_{max}$ = .487 (.48778 with bias) at $x$ = 1.26, and the optimal $\tau$ is 2.3.

## III THE PART AND TRY ALGORITHM ADAPTED TO FREE ACCESS

### III 1. The problem of the session end.

Modifying the basic part and try algorithm so as to make it a free access protocol where stations only need monitor the channel when they have a packet to transmit is made possible by the following observation: the end of a non degenerate session simply occurs when two consecutive packets are successfully transmitted. So a new arrival packet needs only two slots to recognize whether it is in a degenerate session or not, and then to be ready for the next session.

We call these two slots the *witness* slots. The table below describes the nine possibilities for the status of these two slots according to the ternary feedback and shows how a station can deduce the type of the current session.

| | | |
|---|---|---|
| blank | blank | degenerate: in a non degenerate session a blank is always followed by a certain to occur collision |
| blank | success | degenerate session (the same reason) |
| success | blank | degenerate session: the success may be is the end of a non degenerate session but the blank is degenerate |
| success | success | end of non degenerate session or two degenerate sessions |
| success | collision | non degenerate: the collision has to be resolved |
| blank | collision | *idem* |
| collision | collision | *idem* |
| collision | blank | non degenerate session: the next slot is a certain to occur collision |
| collision | success | non degenerate: the collision is not yet justified |

In summary, if a collision occurs during the witness slots there is a non degenerate session currently being resolved; otherwise, if no collision, the session just ends and a new session begins in the next slot.

III 2. **The problem of the localisation of the window**

Another problem is how to know the location of the window in the context of a free access protocol where every packet has no knowledge of the succession of events before its birth?

The solution consists in permuting the role of non resolved interval and window. Instead of sliding the window from the past to the present, keep the window two slots below present and slide through the non resolved interval from the past to the present.

**The adapted algorithm**

The $i^{th}$ session is the part and try resolution of the packets whose arrival date is in the interval $[s_i-2-\tau,s_i-2] = W$ (the packets which arrived in $[s_i-2,s_i]$ are not able to recognize that we begin a new session and have to wait until the next one).

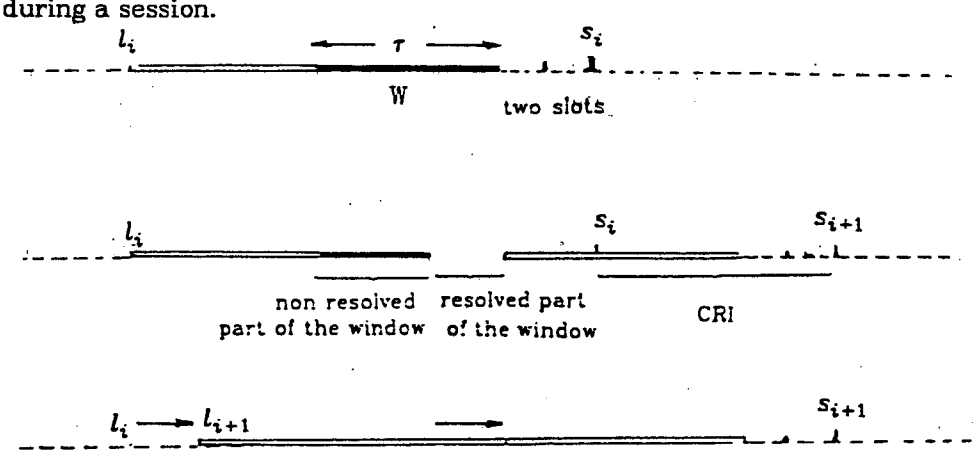
The part and try free access resolution is the basic part and try resolution but with a time symmetry:

```
procedure Free-Access Resolution(W);
begin
    Transmission(W);
    if feedback=blank or success then return degenerate;
    if feedback=collision then Free-Access Splitting(W)
end;

procedure Free-Access Splitting(W);
begin
    Free-Access Resolution(tail p W);
    if degenerate then Free-Access Resolution(head q W)
end;
```

The non resolved part of the window is incorporated to the remaining of the old non resolved interval which will be slid and merged with the new non resolved interval $[s_i-2, s_{i+1}-2]$ by the operation:

```
procedure Adjust;
begin
    if arrival date < s_i −2 then
    arrival date := arrival date + size of the window resolved part
end;
```

Let $l_i$ be the left end of the non resolved interval at the beginning of the $i^{th}$ session (this general-system parameter cannot be evaluated by a station). The following scheme gives an illustration of the evolution of the non resolved interval during a session.




(fig 6)

At the end of the session, after execution of procedure Adjust, $l_{i+1} = l_i$+size of the window resolved part. And we can begin a new session.

III 3. **Conditions for stability**

The behaviour of $s_i - l_i$ is a discrete time continuous state Markov process and, when $s_i - l_i > \tau$, it behaves like $s_i - w_i$ of the previous window algorithms. When $s_i - l_i < \tau$ there is slight difference because no station can be aware of this event and the window is always assumed to be of size $\tau$. But, the markovian

properties being preserved, the ergodicity conditions deal with $E[s_{i+1}-l_{i+1} \,/\, s_i-l_i=l] - l$ when $l \to \infty$. So the throughput is the same of the basic part and try one: $\lambda_{max} = .453$.

The reason that we cannot immediately adapt the real .487 algorithm resides in the fact that a packet cannot decide in a deterministic finite number of slots whether it is in a degenerate session or not. In a non degenerate session a collision can be followed by an arbitrary number of consecutive empty slots if the certain to occur collisions are eliminated (this event is always possible but, of course, its probability drastically decreases with respect to the length of the tail of empty slots).

## IV THE $e$ -PART AND TRY ALGORITHMS

We can build and analyse the class of so called *e-(free access)-part and try* algorithms, $e$ being a design parameter with integer value. These protocols allow free access and, as $e$ gets large, their throughputs tend to the upper limiting value .487. However delay will increase with $e$.

### IV 1. Principle of the e-part and try algorithm

The $e$-part and try algorithm is the part and try algorithm in which only $e$ eliminations of certain to occur collision are allowed in each session. So the 0-part and try algorithm is the basic part and try algorithm and the $\infty$-part and try algorithm is the .487 algorithm. The $e$-part and try algorithm, $e$ being a fixed integer, is adapted to free access. It has the property that a packet needs only $e+2$ slots to recognize whether it is in a degenerate session or not and to be ready for the next session.

In order to illustrate the $e$-part and try algorithm take fig. 8 and replace the two slots below every session by $e+2$ slots (only packets which have yet monitored the channel more than $e+2$ slots can participate to a session). The $e$-part and try resolution protocol follows.

```
procedure Free-Access Resolution(W:time interval,e:integer);
begin
    Transmission(W);
    if feedback=blank then return degenerate-or-c-o-collision;
    if feedback=success then return degenerate;
    if feedback=collision then Free-Access Splitting(W,e)
end;

procedure Free-Access Splitting(W,e);
begin
    Free-Access Resolution(W,e);
    if degenerate-or-c-o-collision then
    begin
        if e>0 then Free-Access Splitting(head q W,e-1);
        if e=0 then Free-Access Resolution(head q W,0)
    end;
    if degenerate then Free-Access Resolution(head q W,e)
end;
```

IV 2. **The analysis of the e-part and try algorithm**

Note $X_n^e = \mathrm{E}[e\text{-part and try CRI} \;/\; W^{\#} = n]$. We have the recurrent equation for $e > 0$.

$$\begin{cases} X_n^e = 1 + \sum_k \binom{n}{k} p^k q^{n-k} X_k^e + q^n (X_n^{e-1} - 1) + npq^{n-1} X_{n-1}^e & n \geq 2 \\ X_0^e = X_1^e = 1 \end{cases}$$

Note $X^e(z) = \sum_n X_n^e \frac{z^n}{n!} e^{-z}$. We reach the functional equation.

$$X^e(z) = X^e(pz) + X^{e-1}(qz)e^{-pz} + X^e(qz)pze^{-pz} - e^{-pz} + 1 - 2(1+z)e^{-z} + (1+qz)e^{-z}.$$

$$X^e(z) = 1 + \sum_{n \in N} R^n((X^{e-1}(qz) - 1) + 1 - 2(1+z)e^{-z} + (1+qz)e^{-z} + pze^{-pz}),$$

with $Rf(z) = f(pz) + pze^{-pz} f(qz)$ (see the appendix for convergence).

With the knowledge of $X^0(z)$ (basic part and try algorithm) we can compute $X^1(z)$, then with $X^1(z)$ we compute $X^2(z)$. The next tables summarize the numerical results for $e = 1,2,3,4,5$ and 10, and the respective maximum throughputs. Note the monotonic increase of the maximum throughput with respect to $e$, and its convergence to $\lambda_{max}^{\infty} = .48711$.

| $n$ | $X_n^0$ | $X_n^1$ | $X_n^2$ | $X_n^3$ |
|---|---|---|---|---|
| 0 | 1.00000 | 1.00000 | 1.00000 | 1.00000 |
| 1 | 1.00000 | 1.00000 | 1.00000 | 1.00000 |
| 2 | 4.50000 | 4.16667 | 4.05556 | 4.01852 |
| 3 | 6.50000 | 6.07143 | 5.91497 | 5.86087 |
| 4 | 7.14286 | 6.71429 | 6.55782 | 6.50373 |
| 5 | 7.31429 | 6.89800 | 6.74779 | 6.69609 |
| 6 | 7.47442 | 7.06180 | 6.91326 | 6.86218 |
| 7 | 7.66859 | 7.25505 | 7.10597 | 7.05468 |
| 8 | 7.86056 | 7.44559 | 7.29580 | 7.24424 |
| 9 | 8.03335 | 7.61756 | 7.46739 | 7.41568 |
| 10 | 8.18488 | 7.76886 | 7.61858 | 7.56684 |
| $\lambda_{max}$ | .44935 | .47324 | .48231 | .48548 |
| $x_{max}$ | 1.16 | 1.22 | 1.25 | 1.25 |

| $n$ | $X_n^4$ | $X_n^5$ | $X_n^{10}$ | $X_n^\infty$ |
|---|---|---|---|---|
| 0 | 1.00000 | 1.00000 | 1.00000 | 1.00000 |
| 1 | 1.00000 | 1.00000 | 1.00000 | 1.00000 |
| 2 | 4.00617 | 4.00206 | 4.00001 | 4.00000 |
| 3 | 5.84256 | 5.83641 | 5.83335 | 5.83333 |
| 4 | 6.48542 | 6.47927 | 6.47620 | 6.47619 |
| 5 | 6.67863 | 6.67278 | 6.66985 | 6.66984 |
| 6 | 6.84493 | 6.83915 | 6.83626 | 6.83625 |
| 7 | 7.03735 | 7.03154 | 7.02864 | 7.02863 |
| 8 | 7.22682 | 7.22098 | 7.21806 | 7.21804 |
| 9 | 7.39821 | 7.39235 | 7.38943 | 7.38911 |
| 10 | 7.54936 | 7.54350 | 7.54057 | 7.54055 |
| $\lambda_{max}$ | .48657 | .48693 | .48711 | .48711 |
| $x_{max}$ | 1.26 | 1.27 | 1.26 | 1.26 |

## VI CONCLUSION

We showed that free access be possible with high throughput. We can build such algorithms with maximum throughput arbitrarily close to the highest throughput reached by the current versions of the part and try algorithm.

Of course, the delay for successful transmission is slightly affected by the incompressible $e+2$ witness slots and by the loss of the *first in first out* strategy. However $e = 2$ already gives $\lambda_{max} = .4823$.

$e$-part and try algorithms show robustness to channel errors. We know that free access protocol can be easily restored *after* general or local failures, but it is more difficult to insure relatively good behaviour during the critical time interval when the failure occurs but has not already been detected. Massey M80 ]] showed that tree algorithms which automatically eliminate certain to occur collision can suffer deadlock due to channel errors (a degenerate session assumed as collision should forbid any further transmission). Our resolution algorithms, by allowing only a finite number of suppression of certain to occur collisions, avoid such deadlocks.

**References**

Ma81:

J L Massey. Collision Resolution Algorithms and Random Access Communication. 1981. CISM course and lecture: Multi User Communication Systems. pp 73-137. Springer Verlag.

Ma85:

J L Massey. Special Issue on Random-Access Communications. March 1985. IEEE Transaction on Information Theory. Number 2.

FFHJ85:

G Fayolle, P Flajolet, M Hofri, P Jacquet. Analysis of a Stack Algorithm for Random Multiple-Access Communication. Same issue. pp 244-254.

MF85:

P Mathys, P Flajolet. Q-ary Collision Resolution Algorithms in Random-Access Systems with Free or Blocked Channel Access. Same issue. pp 217-243.

GK85:

L Georgiadis, P Papantoni-Kazakos. Limited Feedback Sensing Algorithms for the Packet Broadcast Channel. Same issue. pp 280-294.

FH83:

G Fayolle, M Hofri. On the Capacity of a Collision Resolution Channel under Stack based Collision Resolution Algorithm. Oct 1983. Technion Research Report. number 237. Haifa Israel.

MT80:

V A Mikhailolv, B S Tsybakov. Random Multiple Packet Access: Part and Try Algorithm. 1980. Problemy Peredachi Informatsii. Volume 16, number 4. pp 65-79.

J83:

P Jacquet. Proprietes moyennes du protocole capetanakis a arrivees continues. 1983. INRIA Research Report. Number 240. Le Chesnay, France.

## APPENDIX

### A- The Window Tree Algorithm: convergence of the sum $\sum_n S^n f(z)$.

The use of the generating functions gives a more compact view of the recursion to be solved. Also the accuracy of the computation is easier to control.

$S$ is a linear operator on the space $\{f / f(z)=z^2 g(z),\ g$ analytic$\}$ with the norm:

$$\|f\|_K = \max_{x\in K}\left\{\left|\frac{f(x)}{x^2}\right|\right\},$$

where $K$ is an arbitrary convex compact neighbourhood of 0. So:

$$\|Sf\|_K = \max_{x\in K}\left\{\left|p^2\frac{f(px)}{(px)^2} + q^2\frac{f(qx)}{(qx)^2}\right|\right\}$$

$$< (p^2+q^2)\|f\|_K .$$

So the norm of the operator, namely

$$\|S\|_K = \max\left\{\frac{\|Sf\|_K}{\|f\|_K}\right\},$$

is not greater than $p^2+q^2 < 1$. Thus the infinite summation $\sum_n S^n$ converges and we can extract the Taylor coefficients and the numerical values of $\sum_n S^n f(z)$.

The accuracy is controlled by:

iteration, we have: $\|\sum_{n=N}^{\infty} S^n f\|_K \le \frac{(p^2+q^2)^{N+1}}{1-p^2-q^2}\|f\|_K$,

truncation, if we note: $X_M(z) = X(0)+\cdots+X^{(M)}(0)\frac{z^M}{M!}$, then

$$\|X_M - X\|_K \le \frac{1}{2^{M+1}}\|X\|_{2K} .$$

In the numerical application we have only to test the accuracy with the formula above applied to the functions $f(z) = 2-2(1+z)e^{-z}$ and $f(z) = 2-2(1+z)e^{-z}-e^{-pz}+(1+qz)e^{-z}$.

### B- The part and try algorithm: convergence of the sum $\sum_n T^n f(z)$

We use the same line of reasoning as above except that $K$ is no longer an arbitrary convex compact neighbourhood of 0. Let $K = O \cup C_{\vartheta,r}$ where

(i) $O$ is a convex neighbourhood of 0 such that

$$(p^2+q^2\max_{x\in O}\{|(1+x)e^{-x}|\}) = \rho < 1$$

(ii) $C_{\vartheta,r} = \{z / -\vartheta<\mathrm{Arg}\,z<\vartheta,\ 0<\mathrm{Re}(z)<r\}$ with $\vartheta\in[0, \frac{\pi}{2}[$ and $r$ arbitrarily chosen.

We need $O$ in order to justify the extraction of Taylor coefficients of $\sum_n T^n f(z)$ and we need $C_{\vartheta,r}$ in order to compute with suitable accuracy the value of this summation for every non negative real $z$. So

$$\|Sf\|_K = \max_{x\in K}\{|p^2\frac{f(px)}{(px)^2}+q^2(1+px)e^{-px}\frac{f(qx)}{(qx)^2}|\}$$

$$= \max\{\max_{x\in O}\{|p^2\frac{f(px)}{(px)^2}+q^2(1+px)e^{-px}\frac{f(qx)}{(qx)^2}|\};$$

$$\max_{x\in C_{\phi,r}}\{|p^2\frac{f(px)}{(px)^2}+q^2(1+px)e^{-px}\frac{f(qx)}{(qx)^2}|\}\}.$$

With $\max_{Re(x)>0}\{(1+x)e^{-x}\} = 1$, we get $\|Sf\|_K \le \rho\|f\|_K$. And we conclude like in A. But we must take into account the fact that

$$\|X\|_{C_{2\pi,r}} \le e^r\|X\|_K$$

since $X(z)e^z$ has non negative real Taylor coefficients.

**C- The $e$-part and try algorithm convergence of the sum $\sum_n R^n f(z)$**

The same proof as above works: instead of $(1+x)e^{-x}$ we study the simpler function $xe^{-x}$ which attains its maximum norm on the right half plane at $x=1$.

**Imprimé en France**

**par**

**l'Institut National de Recherche en Informatique et en Automatique**