# NECESSARY AND SUFFICIENT CONDITIONS TO DETECT MESSAGE DUPLICATION IN PROTOCOL HIERARCHIES

M. G. Gouda and B. N. Jain*

Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

TR-192 February 1982

*On leave from Dept. of Electrical Eng., Indian Institute of Technology, Delhi 11016, India

put on cover page


## ABSTRACT

We discuss the phenomena of message duplication in a protocol layer that is caused by the lower layer in a protocol hierarchy. For example, packet duplication in the packet layer of an X.25 protocol can be caused by the reset feature in the lower frame layer of the same protocol. We propose two techniques to detect duplicate messages. One technique requires that serial numbers be attached to all sent messages. The other technique requires that each process in the protocol layer distinguishes between the different types of received messages. Necessary and sufficient conditions for the correctness of each technique are developed and verified.



**Keywords:** Communication protocols, duplication detection, message duplication, protocol hierarchy, X.25 protocol.

# I. INTRODUCTION

Message duplication between communicating processes in a protocol layer can be caused either by the processes themselves or by the lower protocol layer. An example of message duplication caused by the processes is when one process re-sends a previously sent message because it has not received a reply for the first message in a reasonable amount of time [3]. An example of duplication caused by the lower layer is when the lower layer transmits a message from one process to another, then "forgets" that it has delivered the message (because of an internal "reset") and redelivers the same message again [5]. Message duplication caused by the processes themselves is a well understood phenomena; see [3] and [7]. Duplication caused by the lower layer is the subject of this paper.

In Section II, we discuss a simple model for message duplication in a protocol hierarchy. A technique to detect duplicate messages by attaching serial numbers to all sent messages is discussed in Section III. In Section IV, another technique to detect duplication by distinguishing the different types of received messages is discussed. Concluding remarks are in Section V.

# II. THE MODEL

Consider two adjacent layers in a protocol hierarchy. The upper layer consists of two processes $P_0$ and $P_1$ which communicate by exchanging messages via the lower layer; i.e., every sent message by $P_0$ (or $P_1$) is first received by the lower layer then it is delivered to $P_0$ (or $P_1$ respectively).

As shown in Figure 1, the two layers are connected by four FIFO queues $S_0$, $S_1$, $R_0$, $R_1$. Queue $S_0$ (or $S_1$) holds the messages sent by $P_0$ (or $P_1$ respectively) and not yet delivered by the lower layer. Queue $R_0$ (or $R_1$) holds the messages delivered by the lower layer and not yet received by $P_0$ (or $P_1$ respectively).

The following notation is adopted:

| | |
|---|---|
| $P_i$ | denotes a process, either $P_0$ or $P_1$. |
| $S_i$ and $R_i$ | denote the two buffers between process $P_i$ and the lower layer. |
| $P_{i\oplus 1}$ | denotes the other process; thus $P_i$ denotes $P_0$ iff $P_{i\oplus 1}$ denotes $P_1$, and vice versa. |
| $S_{i\oplus 1}$ and $R_{i\oplus 1}$ | denote the two buffers between process $P_{i\oplus 1}$ and the lower layer. |

$m_{i,j}$ denotes the jth message sent by process $P_i$ where j=0,1,2,...; index j is called the *order* of message $m_{i,j}$.

The lower layer can be viewed as executing an infinite sequence of operations; each operation is either a delivery operation or a duplicate-delivery operation defined as follows.

A Delivery Operation:

if $S_i$ has at least one message
then the lower layer removes the head message from $S_i$ and adds it to the tail of $R_{i\oplus 1}$.

A Duplicate-Delivery Operation:

If k is the order of the highest order message which has already been delivered by the lower layer from $P_i$ to $P_{i\oplus 1}$
then the lower layer adds a previously delivered message $m_{i,j}$ where $j \leq k$ to the tail of $R_{i\oplus 1}$, provided

$$j > k - D_i \qquad (1)$$

where $D_i$ is some non-negative integer called the duplication bound for the lower layer from $P_i$ to $P_{i\oplus 1}$.

Notice that $D_i \geq 0$. If $D_i > 0$ then duplicate copies of previously delivered messages can be redelivered by the lower layer to $P_{i\oplus 1}$. If $D_i = 0$ then no duplication from $P_1$ to $P_{i\oplus 1}$ is possible.

If $D_i \rightarrow \infty$ then the duplication from $P_i$ to $P_{i\oplus 1}$ caused by the lower layer is said to be *unbounded*; otherwise it is *bounded*. Many existing protocol layers cause only bounded duplication. For instance, the duplication caused by the reset feature in the frame layer of X.25 [2] is bounded by the window size of that layer [5]. Similarly, the duplication caused by the packet layer of that same protocol is also bounded [5].

Next, we discuss two techniques for the upper layer to detect message duplication caused by the lower layer. One technique requires the processes in the upper layer to attach serial numbers to all sent messages. The second technique requires the processes to exchange distinguishable types of messages.

## III. DUPLICATION DETECTION BY ATTACHING SERIAL NUMBERS TO MESSAGES

Assume that for j=0,1,2,..., a number $n_{i,j}$ is attached to the message $m_{i,j}$ sent by $P_i$; and assume that the attached numbers are defined as follows:

$$n_{i,0} = 0 \tag{2}$$

$$n_{i,j} = n_{i,j-1} + 1 \bmod N_i \tag{3}$$

where $N_i$ is called the *numbering bound* for the messages sent by $P_i$.

**Lemma 1:** [For all j=0,1,2,...] [$n_{i,j} = j \bmod N_i$]

**Proof:** (by induction on j)

Initial step (j=0): From (2), $n_{i,0} = 0$
Induction hypothesis (j=k): $n_{i,k} = k \bmod N_i$
Induction step (j=k+1): From (3), $n_{i,k+1} = n_{i,k+1} \bmod N_i$;
then from the induction hypothesis $n_{i,k+1} = k+1 \bmod N_i$ []

Process $P_{i \oplus 1}$ can detect duplicate messages by using the following program to receive messages.

```
var x : [0,...,N_i-1];
  {x contains the number attached to the next nonduplicate message}
begin x := 0;
      while true do
         receive a message m and the number
         n attached to it;
         if x = n
            then {m is a nonduplicate message}
                 x := x+1 mod N_i;
                 store m;
            else {m is a duplicate message}
                 discard m
         endif;
      endwhile;
end
```

In this program, a message m is a nonduplicate iff its number equals the current value of variable x. Thus, the program can be verified by proving the following two assertions:

A. For j=0,1,2,..., the jth value $x_j$ of variable x equals the number attached to the jth message $m_{i,j}$ sent by $P_i$.

B. For j=0,1,2,..., the jth value $x_j$ of variable x does not equal the number attached to any duplicate message received by $P_{i \oplus 1}$ before $m_{i,j}$.

Assertion A holds; see Theorem 1 below. For assertion B to hold, a necessary and sufficient condition, namely that $N_i > D_i$, must hold; see Theorem 2 below.

**Theorem 1:** Let $x_j$ be the jth value (j=0,1,2,...) assigned to variable x in the above program.
[for all j=0,1,2,...] [$x_j = n_{i,j}$]

**Proof:** (by induction on j)

Initial step (j=0): From the above program,
$x_0 = 0 = n_{i,0}$
Induction hypothesis (j=k): $x_k = n_{i,k}$
Induction step (j=k+1): From the above program,

$$\begin{aligned} x_{k+1} &= x_k + 1 \bmod N_i \\ &= n_{i,k} + 1 \bmod N_i \\ &= n_{i,k+1} \quad \text{(from (3))}. \end{aligned}$$

[]

**Theorem 2:** Let $x_k$ be the kth value (k=0,1,2,...) assigned to variable x in the above program.
[$N_i > D_i$] iff
[for all k=0,1,2,...] [$x_k \neq n_{i,j}$ where j<k]

**Proof:**

If Part: Assume that $N_i > D_i$ and that there exists a k such that $x_k = n_{i,j}$, where j<k. Then, from Theorem 1, we have $n_{i,k} = n_{i,j}$ where j<k. From Lemma 1, we have $k = j \bmod N_i$, where j<k. Therefore, $k = s.N_i + j$ for some s in {1,2,3,...}. Since $j \leq k-1$, then from (1) $j > k-1-D_i$. Thus, we get $D_i + 1 > s.N_i$ for some s in {1,2,...}. This contradicts $N_i > D_i$.

Only If Part: Assume that $N_i \leq D_i$; we show that there exists a k such that $x_k = n_{i,j}$, where j<k. Let the value of k be $k = j + N_i$. This does not violate (1) since $j = k - N_i > k - D_i - 1$.

From Lemma 1,

$$\begin{aligned} n_{i,k} &= k \bmod N_i \\ &= j \bmod N_i \\ &= n_{i,j} \end{aligned}$$

From Theorem 1,

$$\begin{aligned} x_k &= n_{i,j} \\ &= n_{i,j} \end{aligned}$$

[]

**Corollary 1:** Assume that the duplication from $P_i$ to $P_{i \oplus 1}$ is unbounded, i.e., $D_i \to \infty$. Then, the above can correctly detect duplicate messages iff $N_i \to \infty$ (i.e., the numbers $n_{i,j}$ must range over all positive integers).

**Proof:** From Theorems 1 and 2, the above program can correctly detect duplicate messages iff $N_i > D_i$. If $D_i \dashrightarrow$ then $N_i \dashrightarrow$. []

## IV. DUPLICATION DETECTION BY DISTINGUISHING DIFFERENT MESSAGE TYPES

Assume that processes $P_i$ and $P_{i\oplus 1}$ are defined as two communicating finite state machines [1], [4], [6], [8] $M_i$ and $M_{i\oplus 1}$ respectively. The following definition of a communicating finite state machine M applies to both $M_i$ and $M_{i\oplus 1}$.

A *communicating finite state machine* M is a directed labelled graph where each edge is labelled with a *message type* such that no two outputs of the same node have identical labels. M has two types of nodes *sending nodes* and *receiving nodes*. Any output edge of a sending (or receiving) node is called a *sending* (or *receiving* respectively) *edge*. The labels of the output edges of a receiving node n in M are called the *expected message types* at n. One of the nodes in M is called its *initial node*; and all the nodes are reachable by directed paths from the initial node.

Associated with M are one queue S to hold the sent messages, by M and one queue R to hold the messages to be received by M. Also associated with M is a *control token* which at any instant resides at exactly one node in M. Initially, the control token of M is at its initial node.

When the control token of M is at a sending node n, then

(i) one of the output edges of n, say edge e, is selected (at random) to be traversed by the control token in zero time to the next node in M, and

(ii) a message m of type t, where t is the label of e, is added to the tail of queue S. m is said to be sent at node n.

When the control token of M is at a receiving node n and the head message in queue R is of a type t expected at node n, then

(i) the output edge e of n, whose label is t, is selected to be traversed by the control token in zero time to the next node in M, and

(ii) the head message m is removed from R for processing. m is said to be *received at* node n.

The two queues associated with $M_i$ are $S_i$ and $R_i$; and the two queues associated with $M_{i\oplus 1}$ are $S_{i\oplus 1}$ and $R_{i+1}$. As discussed in Section II, the function of the lower layer

is to transmit, with possible duplication, the messages from $S_i$ to $R_{i\oplus 1}$ and from $S_{i\oplus 1}$ to $R_i$.

We assume that $M_i$ and $M_{i\oplus 1}$ satisfy the following two conditions provided that no duplication occurs during their communication.

A. No Unspecified Receptions: If the control token of $M_i$ (or $M_{i\oplus 1}$) reaches a receiving node n and if at this state, the head message in $R_i$ (or $R_{i\oplus 1}$ respectively) is of type t, then t is an expected type at n.

B. No Nonexecutable Paths: Given a directed path p in $M_i$ (or $M_{i\oplus 1}$) it is possible to start from the initial nodes of $M_i$ and $M_{i+1}$ and select control token movements in $M_i$ and $M_{i\oplus 1}$ such that the control token of $M_i$ (or $M_{i\oplus 1}$ respectively) traverses p.

Condition A is essential and needs to be satisfied [4] and [8] regardless of our interest in duplication detection. Later, we argue that condition B is not needed in almost all practical applications.

$M_{i\oplus 1}$ can detect duplicate messages by applying the following rule.

Duplication-Detection Rule for $M_{i\oplus 1}$:

Let the control token of $M_{i\oplus 1}$ be at a receiving node n while m is the head message in $R_{i\oplus 1}$.

```
If    the type of m is expected at n
then {m is a nonduplicate message}
      m is removed from R_{i⊕1} and stored {receive m}; and
      the control token of M_{i⊕1} moves to the next node in M_{i⊕1}

else {m is a duplicate message}
      m is removed from R_{i⊕1} and discarded; and
      the control token of M_{i⊕1} remains at n
```

For $M_{i\oplus 1}$ to correctly detect duplicate messages using this rule, a necessary and sufficient condition must be satisfied; to state this condition some definitions are in order. A directed path p in a communicating finite state machine M is called *nonduplicate* if it starts from a receiving node and ends at a receiving node and the label of its first edge is the same as that for an output edge of its last node. Define R(M) as follows:

$$R(M) = \min_{p \text{ is a nonduplicate}} [\text{Number of receiving edges in } p]$$

path in M

Notice that in general, M can have an infinite number of nonduplicate paths; however to compute R(M) we need only to consider a finite number of these paths, called basic paths. In the appendix we define basic paths and show how to compute R(M) from them.

As shown in the next theorem, a necessary and sufficient condition for $M_{i\oplus 1}$ to correctly detect duplicate messages using the above rule is that $R(M_{i\oplus 1}) > D_i$.

**Theorem 3:** Let $n_r$ denote the rth receiving node (r=0,1,...) reached by the control token of $M_{i\oplus 1}$.

$[R(M_{i\oplus 1}) > D_i]$ iff
[for all r = 0,1,...][
    If a message m of an expected type at $n_r$ is received at $n_r$
    then $m = m_{i,r}$
    else $m = m_{i,j}$ where $j < r$]

**Proof:**

If Part: Assume that $R(M_{i\oplus 1}) > D_i$; we use induction on r to prove the above property.

Initial step (r=0): The first message received at $n_0$ must be $m_{i,0}$. Also since $M_{i\oplus 1}$ has no unspecified receptions, the type of $m_{i,0}$ must be expected at $n_0$.

Induction hypothesis (r=k): Assume that if a message m of an expected type at $n_k$ is received at $n_k$ then $m=m_{i,k}$ else $m=m_{i,j}$ where $j<k$.

Induction step (r=k+1): From the induction hypothesis and since $M_{i\oplus 1}$ applies the above duplication-detection rule, the control token of $M_{i\oplus 1}$ reaches $n_{k+1}$ only after $m_{i,k}$ is received. Thus any message received at $n_{k+1}$ is either $m_{i,k+1}$ or any message $m_{i,j}$ where $j\leq k$ and $j>k-D_i$ (from (1)). Since $M_{i\oplus 1}$ has no unspecified receptions, the type of $m_{i,k+1}$ must be expected at $n_{k+1}$. It remains to prove that the type of any $m_{i,j}$ where $j\leq k$ and $j>k-D_i$ is not expected at $n_{k+1}$; the proof is by contradiction. Assume that the type t of a message $m_{i,j}$ where $j\leq k$ and $j>k-D_i$ is expected at $n_{k+1}$; i.e., node $n_{k+1}$ must have an output labelled t. By the induction hypothesis and since $M_{i\oplus 1}$ applies the duplication-detection rule, $m_{i,j}$ must have already been received as a nonduplicate message at $n_j$. Thus $n_j$ must have an output edge labelled t. Therefore, the directed path traversed by the control token of $M_{i\oplus 1}$ from $n_j$ to $n_k$ is a nonduplicate path with k-j

receiving edges. Thus $k-j \geq R(M_{i\oplus 1})$. Since $j > k-D_i$, we get $D_i > k-j \geq R(M_{i\oplus 1})$ contradicting $R(M_{i\oplus 1}) > D_i$.

Only If Part: Assume that $D_i \geq R(M_{i\oplus 1})$; we show that the control token of $M_{i\oplus 1}$ can reach a receiving node n at which a duplicate message, whose type is expected at n, can be received. Since $D_i \geq R(M_{i\oplus 1})$, $M_{i\oplus 1}$ must have a nonduplicate path p with h receiving edges where $D_i \geq h$. Let $n_1$ and $n_2$ be the first and last (receiving) nodes in p. Also let t be the label of the first edge in p; hence, both $n_1$ and $n_2$ must have outputs labelled t. Because, every path in $M_{i\oplus 1}$ is executable, the control token movements in $M_i$ and $M_{i\oplus 1}$ can be selected such that the control token traverses path p and reaches node $n_2$. Assume that at this state the head message in $R_{i\oplus 1}$ is a duplicate copy of the nonduplicate message received at $n_1$. This is possible since $D_i \geq h$. The type of this duplicate message is t which is an expected type at $n_2$. []

Notice that in this proof the requirement that 'every path in $M_{i\oplus 1}$ be executable' is only needed to establish the necessity (rather than the sufficiency) of the condition "$R(M_{i\oplus 1}) > D_i$". Therefore, even if some paths in $M_{i\oplus 1}$ are nonexecutable, $M_{i\oplus 1}$ can still use the above rule to correctly detect duplicate messages provided that "$R(M_{i\oplus 1}) > D_i$".

**Corollary 2:** If $D_i \dashrightarrow$ , then $M_{i\oplus 1}$ cannot detect duplicate messages using the above duplication-detection rule.

**Proof:** From Theorem 3, $M_{i\oplus 1}$ can correctly detect duplicate messages using the above rule iff $R(M_{i\oplus 1}) > D_i$. If $D_i \dashrightarrow$ then $R(M_{i\oplus 1}) \dashrightarrow$ This means that $M_{i\oplus 1}$ must have an infinite basic path (see the appendix) of distinct nodes contradicting that $M_{i\oplus 1}$ is a *finite* state machine. []

## V. CONCLUDING REMARKS

The two proposed techniques to detect message duplication can be compared as follows. The first technique is less efficient since it requires to attach an extra number to each sent message. The number of bits required to encode each of these numbers is $\log_2 N_i$ where $N_i$ is bounded by $N_i > D_i$. On the other hand, the second technique is less general since it can be applied only if the upper layer satisfies some condition, namely $R(M_{i\oplus 1}) > D_i$.

This suggests that in most cases, the second technique should be considered first. Only when it is clear that the second technique cannot be used (i.e., $R(M_{i\oplus 1}) \leq D_i$), the designer resorts to the first, more general technique.

**ACKNOWLEDGEMENT:** The authors are thankful to K. F. Carbone for her careful typing.

## REFERENCES

[1] G. V. Bochmann, "Finite state description of communication protocols," Computer Networks, Vol. 2, 1978, pp. 361-371.

[2] G. V. Bochmann and T. Joachim, "Development and structure of an X.24 implementation," IEEE Trans. on Software Engineering, Vol. SE-5, No. 5, Sept. 1979, pp. 429-439.

[3] D. W. Davies, et al, Computer networks and their protocols, New York, John Wiley, 1979.

[4] M. G. Gouda and Y. Yu, "Designing deadlock-free and bounded communication protocols," Tech. Rep. 179, Dept. of Computer Sciences, Univ. of Texas at Austin, June 1981.

[5] B. N. Jain, "Duplication of packets and their detection in X.25 communication protocols," 9th International Symp. on Computer Architecture, Austin, TX, May 1982.

[6] C. A. Sunshine, "Formal modeling of communication protocols," USC/Inform. Sc. Institute, Research Report 81-89, March 1981.

[7] A. S. Tanenbaum, Computer networks, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[8] P. Zafiropulo, et. al., "Towards analyzing and synthesizing protocols," IEEE Trans. Comm., Vol. COM-28, No. 4, April 1980, pp. 651-661.

## APPENDIX: COMPUTATION OF R(M)

A nonduplicate path p is called *basic* if it satisfies the following condition. All nodes in p are distinct except possibly its first and last nodes.

**Lemma:**

$$R(M) = \min_{\substack{p \text{ is a basic} \\ \text{path in } M}} [\text{Number of receiving edges in } p]$$

**Proof:** We show that for every nonduplicate path p which is not basic there exists a basic path q such that the number of receiving edges in p $\geq$ the number of receiving edges in q. Let p be a nonduplicate path which is not basic; i.e., p can be in any one of the following three forms:

$a, x_1, b, x_2, b, x_3, c$
$a, x_1, a, x_2, c$
$a, x_1, c, x_2, c$

where a is the first node in p,
b is an internal node in p,
c is the last node in p (not necessarily distinct from a),
and $x_1$, $x_2$, and $x_3$ represent the other node occurrences in p.

For each of these forms of p, we can find a path q in M such that q is nonduplicate and the number of receiving edges in p $\geq$ the number of receiving edges in q:

For $p = a, x_1, b, x_2, b, x_3, c$, then $q = a, x_1, b, x_3, c$
For $p = a, x_1, a, x_2, c$ then $q = a, x_2, c$
For $p = a, x_1, c, x_2, c$ then $q = a, x_1, c$

If the found q is not basic, q must be in any of the above three forms, and the argument repeats to find a smaller nonduplicate path in M. Since each time the found path is smaller than the previous path, we must find at the end a basic path in M where the number of its receiving edges is less than or equal that for the original path p. []

The number of basic paths in a communicating finite state machine M is finite. Therefore, R(M) can be computed in a finite time from the above lemma.