Reliable network protocols and proofs

Victor Yodaiken
Copyright 2008.*
yodaiken@finitestateresearch.com

October 5, 2008

1 Basics

Let S be a set of *sites* and M be a set of *messages*. Using the conventions of an earlier paper [Yod08b, Yod08a] we use sequence or path dependent variables to represent state: each sequence of events w determines a particular state, the state reached by following w from the initial system state. The w state is the state determined by w. Sometimes we discuss the difference between the w state and a following state reached when an event a drives the system from the w state to the wa state. Functions that depend on the state usually have an event sequence as the first parameter - e.g. $f(w,x)$. In this case, let Rx (received) and Tx (transmitted) be boolean functions so that tell us (respectively) if node s received or sent message m.

$$\begin{aligned}
&\textit{Boolean functions} \\
&Rx(w,s,m) \in \{0,1\} && (1)\\
&Tx(w,s,m) \in \{0,1\} && (2)\\
&\textit{Once true, never become false} \\
&Rx(wa,s,m) \geq Rx(w,s,m) && (3)\\
&Tx(wa,s,m) \geq Tx(w,s,m) && (4)\\
& && (5)
\end{aligned}$$

We will assume that transmitting a message means that the transmitting site at least receives the message (this makes it simpler to deal with sites that may be source and destination sites) and we can assume that the network does not spuriously deliver messages

$$Tx(w,s,m) \leq Rx(w,s,m) \quad (6)$$

$$Rx(w,s,m) \leq \Sigma_{x\in S} Tx(w,x,m) \quad (7)$$

Now suppose that that we have two sets D and C of, respectively, data and "ack" messages and that acks are only sent for received data messages.

$$D, C \subset M \quad (8)$$

$$D \cap C = \emptyset \quad (9)$$

$$Seq : D \mapsto \{0,1,2,\dots\} \quad (10)$$

$$Ack : C \mapsto \{0,1,2,\dots\} \quad (11)$$

Abbreviation

$$d_n \text{ is an element of } D \text{ where } Seq(d) = n \tag{12}$$

$$c_n \text{ is an element of } C \text{ where } Ack(c) = n \tag{13}$$

Sent data messages have unique sequence numbers

$$\text{If } Tx(w, s, d_n) \text{ and } Tx(w, s', d'_n) \text{ Then } d_n = d'_n \tag{14}$$

Ack messages are only sent if they match some received data message

$$\text{If } Tx(w, s, c_n) \text{ then for some } d_n, Rx(w, s, d_n) \tag{15}$$

The following trivial theorem follows easily.

Theorem 1.1 *If $Tx(w, s, d_n)$ and $Rx(w, s, c_n)$ and $Tx(w, s, c_n) = 0$ then at least one for some $s' \neq s$ has received d_n — for some $s' \neq s, Rx(w, s', d_n)$.*

Proof.

$$Rx(w, s, c_n) \quad \text{assumption} \tag{16}$$

$$\text{for some } s', Tx(w, s', c_n) \quad 7 \tag{17}$$

$$s' \neq s \quad \text{assumption} \tag{18}$$

$$\text{for some } d'_n, Rx(w, s', d'_n) \quad 15 \tag{19}$$

$$\text{for some } s'', Tx(w, s'', d'_n) \quad 7 \tag{20}$$

$$d'_n = d_n \quad 14 \tag{21}$$

$$Rx(w, s', d_n) \quad 21, 19 \tag{22}$$

QED

2 Chang's algorithm

A more interesting theorem is based on the Chang-Maxemchuk algorithm [CM84] which was the subject of a nice patent[CM88]. The algorithm is subtle, but simple. In the crude version defined below, a set of Destination sites shares responsibility for sending acknowledgments. A map ϕ assigns destinations to sequence numbers and site $\phi(n)$ is responsible for sending the acknowledgment for a data message with sequence number n. But a destination may not send an ack for n unless it has received message n and all previous messages and all previous acknowledgments. The result: if k is greater than or equal to the number of destination sites, then receiving the ack for $n + k$ informs the sender that every destination has received message numbered n.

$$Dest \subset S \tag{23}$$

The function ϕ assigns some destination to every sequence number

$$\phi : 0, \ldots\ldots \to Dest \tag{24}$$

The assignment is "onto"

$$\text{for every } n, \{\phi(n), \ldots \phi(n + |Dest| - 1)\} = Dest \tag{25}$$

Only the designated destination sites originate ack messages

$$\text{If } Rx(w, s, c_n) \text{ then } Tx(w, \phi(n), c_n) \tag{26}$$

Destination sites send ack messages only if there are no gaps

$$\text{If } Tx(w, s, c_n) \text{ then for all } 0 \leq j \leq n, \text{ there is some } d_j \text{ so that } Rx(w, s, d_j)$$

$$\text{and for all } 0 \leq j \leq n \text{ there is some } c_n \text{ so that } Rx(w, s, c_n) \tag{27}$$

Theorem 2.1 *If $Tx(w, s_0, d_n)$ and $Rx(w, s_2, c_{n+|Dests|})$ then for every $s \in Dests$, $Rx(w, s, d_n)$.*

Proof:

$$\text{Suppose } Tx(w, s_0, d_n) \quad (28)$$

$$\text{and } Rx(w, s_1, c_{n+|Dests|}) \quad (29)$$

$$\text{Suppose there is some } s_2 \in Dests, Rx(w, s_2, d_n) = 0 \quad (30)$$

$$Tx(w, \phi(n+|Dests|), c_{n+|Dests|}) \quad 29, 26 \quad (31)$$

$$\forall 0 \le j \le |Dests|, \exists c_{j+n}, d_{j+n}, Rx(w, \phi(n+|Dests|), c_{n+j})$$

$$\text{and } Rx(w, \phi(n+|Dests|), d_{n+j}) \quad 31, 27 \quad (32)$$

$$\exists 0 \le k \le |Dests|, \phi(n+k) = s_2 \quad 25 \quad (33)$$

$$\exists c_{n+k}, Rx(w, \phi(n+|Dests|), c_{n+k}) \quad 33 \quad (34)$$

$$Tx(w, s_2, c_{n+k}) \quad 34, 25 \quad (35)$$

$$\exists d'_n, Rx(w, s_2, d_n) \quad 35, 27 \quad (36)$$

$$d'_n = d_n \quad 28, 14 \quad (37)$$

$$QED$$

3 Notes

The assumption here is that sequence numbers are never repeated. This assumption can be fixed by redefining *Tx* and *Rx* to forget messages when a new sequence is created (a reliable group is reconstituted) and/or by defining conditions by which sequence numbers can be reused (when they are safely committed).

Readers may be interested in a much earlier run at this algorithm in [YR92].

References

[CM84] J.M. Chang and N.F. Maxemchuk. Reliable broadcast protocols. *ACM Trans. Computer Systems*, 2(3), august 1984.

[CM88] Jo-Mei Chang and N. F. Maxemchuk. Reliable broadcast protocol for a token passing bus network, 1988.

[Yod08a] Victor Yodaiken. The meaning of concurrent programs. Technical report, Finite State Research LLC, 2008. http://www.yodaiken.com/papers/code2.pdf".

[Yod08b] Victor Yodaiken. State and history in operating systems. Technical report, Finite State Research LLC, May 2008. http://www.yodaiken.com/papers/h2.pdf".

[YR92] V. Yodaiken and K. Ramamritham. Verification of a reliable broadcast algorithm. In J. Vytopil, editor, *Formal Techniques in Real-Time and Fault-Tolerant Systems*, number 571 in LNCS. Springer-Verlag, 1992.