A THEOREM ON TERMINATION OF DISTRIBUTED SYSTEMS

K. Mani Chandy

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188

TR-87-09 March 1987

A Theorem

On Termination of Distributed Systems

K. Mani Chandy

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

March 16, 1987

ABSTRACT

A theorem on termination of distributed systems is presented, and applications of the theorem are indicated. The theorem appears to play a role in many termination detection algorithms.

1. Distributed Systems

A distributed system is a set of processes and a set of channels. A channel is directed from precisely one process to precisely one process. We use u, v, w for processes, b, c for channels, e for events, and k for natural numbers. Universal quantification is to be assumed unless otherwise indicated. Associated with each v is a boolean variable $v.idle$, and associated with each c are nonnegative integer variables $c.s$ and $c.r$ where $v.idle$ holds for an idle v, and $c.s, c.r$ are the numbers of messages sent and received (respectively) along c, and the system obeys the following rules:

Rule 1

$$\{v.idle \ \wedge \ [\text{for all input channels } c \text{ of } v \ :: \ (c.s = c.r)]\} \ \ e \ \ \{v.idle\} \tag{1}$$

Therefore an idle process with empty incoming channels remains idle.

Rule 2

$$\{v.idle \ \wedge \ (c.s = k)\} \ \ e \ \ \{c.s = k\} \tag{2}$$

An idle process does not send messages.

Furthermore, a process only receives messages that have been sent earlier:

Rule 3

$$c.s \geq c.r \tag{3}$$

Rule 4

$$\{(c.s = k) \ \wedge \ (c.r = j) \ \wedge \ (k \geq j)\} \ \ e \ \ \{k \geq c.r \geq j\} \tag{4}$$

and

Rule 5

$c.s$ is monotone nondecreasing

$$\{c.s = k\} \ \ e \ \ \{c.s \geq k\} \tag{5}$$

At an *arbitrary* time, v records $v.idle$, and $c.s$ for each of its output channels and $c.r$ for each of its input channels. We introduce variables $v.IDLE, c.S, c.R$ where the value of $v.idle$ is recorded in $v.IDLE$, $c.s$ in $c.S$ and $c.r$ in $c.R$. A process records its state at most once. To indicate that v has completed recording, a boolean variable $v.fini$ is employed, where $\neg v.fini$ holds prior to the recording and $v.fini$ holds after the recording. The event of recording by v is carried out by the program segment:

$$\begin{aligned} &\text{if } \neg v.fini \text{ then begin } v.fini \ := \ true \ ; \ v.IDLE \ := \ v.idle \ ; \\ &\qquad\qquad [\text{for all input channels } c \text{ of } v \ :: \ c.R \ := \ c.r] \ ; \\ &\qquad\qquad [\text{for all output channels } c \text{ of } v \ :: \ c.S = c.s] \\ &\qquad \text{end} \ ; \end{aligned} \tag{6}$$

Variables $v.fini, v.IDLE, c.R, c.S$ are modified only in this statement.

We emphasize that each v records its state at an arbitrary time (independent of the times at which other processes record their states).

Let d be the set of processes that have recorded their states, i.e.,

$$d = \{v \mid v.fini\}$$

Initially for all v : $\neg v.fini$ holds, i.e.,

$$initial\text{-}condition \ \Rightarrow \ \neg v.fini$$

2. Properties of the System

Theorem Invariants of the system are

invariant

$$\begin{aligned} &[\text{for all output } c \text{ of processes in } d \ :: \ c.s \geq c.S] \ \wedge \\ &[\text{for all input } c \text{ of processes in } d \ :: \ c.r \geq c.R] \end{aligned} \tag{7}$$

invariant [for all v in d :: $v.IDLE$] $\wedge$ [for all c between processes in d :: $c.S = c.R$]

$\wedge$ [for all b from processes not in d to processes in d :: $b.s = b.R$]

$\Longrightarrow$

[for all v in d :: $v.idle = v.IDLE$] $\wedge$
[for all output c of processes in d :: $c.s = c.S$] $\wedge$

[for all input c of processes in d :: $c.r = c.R$] (8)

Proof: The proof of (7) is trivial from (3)-(6). Here we shall only prove (8). Let I be the expression in (8); to prove that I is an invariant we shall show:

$[initial\text{-}conditions \Rightarrow I]$ and $\{I\}\ e\ \{I\}$

For brevity, define *ante* as the antecedent of I, and define *cons* as the consequent of I; therefore

$$I \equiv \neg ante \vee cons$$

$$\equiv \neg ante \vee (ante \wedge cons) \quad (9)$$

Initially d is empty.

$$(d = empty) \Rightarrow cons \quad (10)$$

$(d = empty) \Rightarrow I$, from (9), (10).

Therefore I holds initially. Next we prove $\{I\}\ e\ \{I\}$. Consider two sets of events: (1) events that do not modify d and (2) events that modify d.

Case 1 Events e that leave d unchanged.

First consider the case where the antecedent does not hold. Consider a channel b from a process not in d to a process in d.

Hence $b.s \geq b.r$ from (3)

$b.r \geq b.R$ from (7)

Hence $b.s \geq b.R$

Hence $(b.s \neq b.R) \equiv (b.s > b.R)$ (11)

$\{b.s > b.R\}\ e\ \{b.s > b.R\}$ from (5) and since $b.R$ is unchanged by e (because b is directed towards a process in d

Hence $\{b.s \neq b.R\}\ e\ \{b.s \neq b.R\}$ from (11) (12)

[for all v in d :: $\{\neg v.IDLE\}$ e $\{\neg v.IDLE\}$] (13)
because $v.IDLE$ is only modified by statement (6) that adds v to d

[for all c between processes in d :: $\{c.S \neq c.R\}$ e $\{c.S \neq c.R\}$] (14)
because $c.S$ or $c.R$ is modified only by statements that add processes to d

$\{\neg ante\}$ e $\{\neg ante\}$ from (12)-(14) and definition of $ante$ (15)

Now consider the case where the antecedent holds.

$ante$ $\Rightarrow$ [for all b from processes not in d to processes in d :: $b.s = b.R$]
, from (8). (16)

$cons$ $\Rightarrow$ [for all input c of processes in d :: $c.r = c.R$] (17)

$ante \wedge cons$ $\Rightarrow$ [for all b from processes not in d to processes in d :: $b.s = b.r = b.R$]
, from (16) and (17) (18)

$ante$ $\Rightarrow$ [for all c between processes in d :: $c.S = c.R$],
definition of $ante$. (19)

$cons$ $\Rightarrow$ [for all output c of processes in d :: $c.s = c.S$],
definition of $cons$. (20)

$ante \wedge cons$ $\Rightarrow$ [for all c between processes in d :: $c.s = c.r = c.R = c.S$]
, from (17), (19), (20) (21)

$ante \wedge cons$ $\Rightarrow$ [for all input c of processes in d :: $c.s = c.r = c.R$]
, from (18), (21) (22)

$ante \wedge cons$ $\Rightarrow$ [for all v in d :: $v.idle$],
definition of $ante$ and $cons$. (23)

$\{ante \wedge cons\}$ e $\{$[for all v in d :: $v.idle$]$\}$
, from (1), (22), (23).

$\{ante \wedge cons\}$

$\{ante \wedge cons\}$ e $\{$[for all output c of processes in d :: $c.s = c.S$]$\}$
, from (2), (20), (23). (25)

$\{ante \wedge cons\}$ e $\{$[for all input c of processes in d :: $c.r = c.R$]$\}$
, from (4), (22). (26)

$\{ante \wedge cons\}$ e $\{cons\}$, from (24), (25), (26) and definition of $cons$ (27)

$\{I\}$ e $\{I\}$, from (9), (15), (27) and Hoare-logic

Case 2 Events e that change d. The only event that changes d is the statement given in (6). Let this statement be t. It is straightforward to show that

$$\{\neg ante\}\ \ t\ \ \{\neg ante\}$$

and

$$\{ante \wedge cons\}\ \ t\ \ \{cons\}$$

Hence $\{I\}\ t\ \{I\}$

This completes the proof of (8).

3. Consequence of the Invariant

If all processes have recorded their states and each process was recorded when it was idle, and for each channel the recorded number of sends along the channel equals the recorded number of receives along the channel, then computation has terminated, i.e., all processes are idle and all channels are empty.

Theorem $[\text{for all } v :: v.fini \wedge v.IDLE] \wedge [\text{for all } c :: c.S = c.R]$

$$\Rightarrow [\text{for all } v :: v.idle] \wedge [\text{for all } c :: c.s = c.r]$$

Proof: Follows directly from (8).

4. Applications

The theorem suggests several ways of detecting termination. For instance, processes periodically record their states and the numbers of messages sent and received along incident channels, and send their recorded values to a centralized process. The central process discards all but the last set of recorded values from each process; if the last set of recorded values satisfies the antecedent of the theorem then computation has terminated.

5. Related Work

A great deal of work has been carried out on termination detection. This theorem was proposed to Dev Kumar for application in distributed simulation, Kumar [1986] and Misra [1986]. Jay Misra, [1987], Wim Hesselink and Ernie Cohen, among others, have developed elegant proofs of this theorem, perhaps superior to that presented here.

This paper is a contribution to calculational proofs of communicating systems [Dijkstra].

6. Acknowledgement

This note was written at the instigation of Jay Misra whose continuing interest and constructive criticism are gratefully acknowledged. The interest and the hard work of the Austin Tuesday Afternoon Club is gratefully acknowledged. Special thanks to Edsger W. Dijkstra for his lemma.

7. References

Dev Kumar, "Efficient Algorithms for Distributed Simulation and Related Problems," 1987, Ph.D. Dissertation, Computer Science Department, University of Texas at Austin, Austin, Texas.

J. Misra, "Distributed Discrete-Event Simulation," *Computing Surveys*, Vol. 18, No. 1, 1986, 39-65.

J. Misra, "A Proof of Chandy's Theorem," February 16, 1987, Unpublished Computer Sciences Report.

C. A. R. Hoare, "An Axiomatic Basis for Computer Programming," *Communications of the A.C.M.*, Vol. 12, 1969.

E. W. Dijkstra, "A correctness proof for communicating processes: A small exercise," EWD 607.

Appendix: Dijkstra's Lemma

$$\langle \forall\ v\ ::\ \{p.v\}\ e\ \{q.v\}\rangle \ \Rightarrow\ \{\langle \exists\ v\ ::\ p.v\rangle\}\ e\ \{\langle \exists\ w\ ::\ q.w\rangle\}$$

where the quantification over v is over a finite (possibly empty) set,
and $p.v, q.v$ are predicates and e is a statement.

Proof: $\langle \forall\ v\ ::\ \{p.v\}\ e\ \{q.v\}\rangle$

$\equiv\ \langle \forall\ v\ ::\ p.v\ \Rightarrow\ wp(e, q.v)\rangle$

$\Rightarrow\ \langle \forall\ v\ ::\ p.v\ \Rightarrow\ wp(e, \langle \exists\ w\ ::\ q.w\rangle)\ \rangle$

because $q.v\ \Rightarrow\ \langle \exists\ w\ ::\ q.w\rangle$ and wp is monotone in its second argument

$\equiv\ \langle \exists\ v\ ::\ p.v\rangle\ \Rightarrow\ wp(e, \langle \exists\ w\ ::\ q.w\rangle)$

$\equiv\ \{\langle \exists\ v\ ::\ p.v\rangle\}\ e\ \{\langle \exists\ w\ ::\ q.w\rangle\}$

□