Information Processing Letters 16 (1983) 217–219
North-Holland

10 June 1983

DERIVATION OF A TERMINATION DETECTION ALGORITHM FOR DISTRIBUTED COMPUTATIONS

Edsger W. DIJKSTRA

Burroughs, Plataanstraat 5, 5671 AL Nuenen, The Netherlands

W.H.J. FEIJEN and A.J.M. van GASTEREN

Department of Mathematics and Computing Science, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands

Communicated by W.M. Turski
Received 22 February 1983

Keywords: Program derivation, program presentation, termination detection, distributed computations, protocols, networks

The purpose of this paper is twofold, viz. to present a new [0] algorithm for the detection of the termination of a distributed computation and to demonstrate how the algorithm can be derived in a number of steps.

We consider N machines, each of which is either active or passive. Only active machines send so-called 'messages' to other machines; message transmission is considered instantaneous. After having received a message, a machine is active; the receipt of a message is the only mechanism that triggers for a passive machine its transition to activity. For each machine, the transition from the active to the passive state may occur 'spontaneously'.

From the above it follows that the state in which all machines are passive is stable: the distributed computation with which the messages are associated is said to have terminated. The purpose of the algorithm to be designed is to enable one of the machines, machine nr.0 say, to detect that this stable state has been reached; it is furthermore required that the detection algorithm can cope with any distribution of the activity at the moment machine nr.0 initiates the detection algorithm. For brevity's sake we shall denote the process by which termination is to be detected by the 'probe'.

The probe obviously has to involve, in some way or another, all the other machines. Two orderly configurations present themselves: an (N − 1)-point star with machine nr.0 at its centre or the N machines arranged in a ring. Since the latter gives rise to less signalling traffic, we adopt the circular arrangement, more precisely, we assume the availability of communication facilities such that

(i) machine nr.0 can initiate the probe by sending a signal to machine nr.N − 1,

(ii) machine nr.i + 1 can propagate the probe around the ring by sending a signal to machine nr.i

These signalling facilities are assumed to be available, irrespective of the facilities for message sending. Note that being passive (with respect to the distributed computation proper) does not prevent a machine from partaking in the above signalling.

The propagation of the probe around the ring allows us to describe that probe as sending a token around the ring. The token being returned to machine nr.0 will be an essential component of the justification of the conclusion that all machines are passive.

As usual, the system state will be captured by an invariant, P say. In the sequel P will be constructed in a number of steps, each step consisting of an extension of the state space considered and an appropriate adjustment of P

In the following, t denotes the number of the machine at which the token resides. The probe ends with $t = 0$.

To begin with we solve the problem in the absence of messages: in that case an active machine can become passive, but a passive machine cannot become active again. The conclusion that all machines are passive has to follow from (i) the invariant, (ii) $t = 0$, and (iii) further information available at machine nr.0. Furthermore the invariant has to hold, independently of the distribution of the activity, when machine nr.0 has initiated the probe, i.e., when $t = N - 1$.

The above requirements are met by P0, given by

P0: $(\mathbf{A}\ i: t < i < N:$ machine nr.i is passive$)$.

We can now design the traffic of the token so as to keep P0 invariant. Initiation of the probe establishes P0; token transmission by one of the remaining machines, however, decreases t by 1, thereby possibly falsifying P0. Invariance of P0 is achieved by adopting

Rule 0. When active, machine nr.i + 1 keeps the token; when passive, it hands over the token to machine nr.i. (End of Rule 0)

When the other machines are passive, the token will in due time be returned to machine nr.0; when in addition machine nr.0 is passive, termination can be concluded. The above concludes the rules for the traffic of the token.

Next to be taken into account is the possibility of messages being sent: P0 is falsified when machine nr.i with $t < i$ becomes active on account of the receipt of a message. Since only active machines send messages, we deduce that the message that falsified P0 has been sent by a machine nr.j with $j \leqslant t$. In order to save the situation we adopt the weaker invariant $P0 \vee P1$, such that any message sending that may falsify P0 establishes P1.

To this end each machine is postulated to be either black or white; for P1 we choose

P1: $(\mathbf{E}\ j: 0 \leqslant j \leqslant t:$ machine nr.j is black$)$

and message sending is preventing from falsifying $P0 \vee P1$ by adoption of

Rule 1. A machine sending a message to a recipient with a number higher than its own makes itself black. (End of Rule 1)

We have to verify, however, that the information available at machine nr.0 combined with the weaker $P0 \vee P1$ can still suffice to conclude termination. Since

$(t = 0 \ \wedge$ machine nr.0 is white$) \Rightarrow \neg P1$,

detection of termination has not been disabled.

Now we have to return to the token, its traffic having only been designed so as not to falsify P0; it may, however, falsify $P0 \vee P1$ by falsifying P1. Probe initiation is safe: it establishes P0 and hence $P0 \vee P1$; probe propagation, decreasing t by 1, may falsify P1 when, while being black, a machine nr.i + 1 hands over the token to machine nr.i. In order to save the situation we adopt the still weaker invariant $P0 \vee P1 \vee P2$, such that any token transmission that may falsify P1 establishes P2.

To this end the token is postulated to be either black or white. For P2 we choose

P2: the token is black

and token transmission is prevented from falsifying $P0 \vee P1 \vee P2$ by adoption of

Rule 2. When machine nr.i + 1 propagates the probe, it hands over a black token to machine nr.i if it is black itself, whereas while being white it leaves the colour of the token unchanged. (End of Rule 2)

Since (token is white) $\Rightarrow \neg P2$, information available at machine nr.0 can still suffice to conclude termination.

With the colour black a new phenomenon has been introduced, viz. that of the unsuccessful

probe: when a black token is returned to machine nr.0 or the token is returned to a black machine nr.0, the conclusion of termination cannot be drawn. In the first instance this problem is tackled by adopting

Rule 3. After the completion of an unsuccessful probe, machine nr.0 initiates a next probe. (End of Rule 3)

Without the possibility of transitions from black to white, such a next probe is, however, guaranteed to be as unsuccessful as its predecessor. Therefore our next task is to investigate which whitenings do not falsify the invariant $P0 \vee P1 \vee P2$.

In view of the fact that initiating a probe establishes P0, we can safely adopt

Rule 4. Machine nr.0 initiates a probe by making itself white and sending a white token to machine nr.$N-1$. (End of Rule 4)

A possibility of whitening the token and machine nr.0 having been provided, we now look for an opportunity of whitening the other machines. Since whitening a machine can falsify only P1, but does not do so when that machine's number exceeds t, we can safely adopt

Rule 5. Upon transmission of the token to machine nr.i, machine nr.$i+1$ becomes white. (Note that its original colour may have influenced the colour of the token.) (End of Rule 5)

The above whitening protocols suffice: a probe initiated after termination will end with all machines white and, hence, a next probe is guaranteed to return a white token to a white machine nr.0.

Two consequences of our whitening protocol deserve to be mentioned. Firstly, not only is our whole detected process valid for any initial distribution of the activity, its validity is also independent of the initial colours of the machines. Secondly, Rule 1 can be safely replaced by the simpler

Rule 1′. A machine sending a message makes itself black. (End of Rule 1′)

In reaction to a preliminary distribution of the above solution, Professor Mohamed G. Gouda, Department of Computer Sciences, University of Texas at Austin, USA, mailed to us his technical report "Distributed State Exploration for Protocol Validation" of October 1981 [1], which contains an earlier solution to the same problem. His solution resembles ours in the sense that the machines are arranged in a ring. In Gouda's solution the token is integer-valued (viz. up to N) and the value of N needs to be available in each machine; the machines, however, are treated on equal footing. It is a pleasure to mention Gouda's result.

References

[0] E.W. Dijkstra and C.S. Scholten, Termination detection for diffusing computations, Inform. Process. Lett. 11 (1) (1980) 1–4.

[1] M.G. Gouda, Distributed state exploration for protocol validation, Tech. Rept. 185, Department of Computer Sciences, University of Texas at Austin, Austin, 1981.