Self-stabilizing systems with distributed control.

by Edsger W.Dijkstra *)

Abstract. In a system with distributed control the local rules of behaviour can guarantee convergence of the system as a whole towards satisfying a global requirement.

Keywords: multiprocessing, networks, self-stabilization, synchronization, mutual exclusion, robustness, starting, error recovery.
CR Category: 4.32

*) Author's address: BURROUGHS
Plataanstraat 5
NUENEN - 4565
The Netherlands

Self-stabilizing systems with distributed control.

We consider a connected graph in which the majority of the possible edges are missing and with a finite state machine placed at each node; machines placed in directly connected nodes are called each other's "neighbours". For each machine one or more so-called "privileges" are defined, i.e. boolean functions of its own state and the states of its neighbours; when such a boolean function is true, we say that the privilege is "present". If a privilege is present, it can be "selected" and the corresponding "move" is then made, i.e. the machine enjoying the selected privilege is brought into a new state that is a function of its old state and the states of its neighbours; if for such a machine more than one privilege has been defined, that function may also depend on the privilege selected.

Furthermore there is a global criterion, telling whether the system as a whole is in a "legitimate" state. We require that

1) in each legitimate state one or more privileges will be present, and
2) in each legitimate state each possible move will bring the system again in a legitimate state, and
3) each privilege must be present in at least one legitimate state, and
4) for any pair of legitimate states there exists a sequence of moves transferring the system from the one into the other

We call the system self-stabilizing if and only if regardless of the initial state and regardless of the privilege selected for the next move, always at least one privilege will be present and the system is guaranteed to find itself in a legitimate state after a finite number of moves. For a long time it has been an open question whether non-trivial (e.g. all states legitimate is considered trivial) self-stabilizing systems could exist. It is not directly obvious whether the local moves can assure convergence towards satisfaction of such a global criterion; the non-determinacy of the privilege selected is an added complication. The question is settled by each of the following three constructs. For brevity's sake the heuristics that lead me to find them the proofs that they satisfy the requirements have been omitted and --to quote Douglas T.Ross's comment on an earlier draft-- "the appreciation is left as an exercise for the reader".

In all three solutions I consider N+1 machines, numbered from 0 through N. In order to avoid avoidable subscripts I shall use for machine nr.i:

L: to refer to the state of its lefthand neighbour, machine nr.$(i+1)\underline{\text{mod}}(N+1)$,

S: to refer to the state of itself, machine nr.i,

R: to refer to the state of its righthand neighbour, machine nr.$(i+1)\underline{\text{mod}}(N+1)$.

In other words, we confine ourselves to machines placed in a ring; machine nr.0 will also be called "the bottom machine", machine nr.N will also be called "the top machine". For the legitimate states we have chosen those states in which exactly one privilege is present. In describing the designs we shall use the format: "if privilege then corresponding move fi".

Solution with K-state machines ($K \geq N$).

Here each machine state is represented by an integer value S, satisfying $0 \leq S < K$. For each machine, one privilege is defined, viz.

for the bottom machine: if $L = S$ then $S := (S + 1) \underline{\text{mod}}\ K$ fi

for the other machines: if $L \neq S$ then $S := L$ fi .

Solution with four-state machines.

Here each machine state is represented by two booleans xS and upS. For the bottom machine upS = true by definition, for the top machine upS = false by definition: these two machines are therefore only two-state machines.

The privileges are defined as follows:

for the bottom machine: if xS = xR and non upR then xS:= non xS fi

for the top machine: if xS ≠ xL then xS:= non xS fi

for the other machines: if xS ≠ xL then xS:= non xS; upS:= true fi;
if xS = xR and upS and non upR then upS:= false fi .

The four-state machines may enjoy two privileges. The neighbour relation between bottom and top machine is not exploited; we may merge them into a single machine which is then also a four-state machine for which also two privileges have been defined.

Solution with three-state machines.

Here each machine state is represented by an integer value S, satisfying $0 \leq S < 3$. The privileges are defined as follows:

for the bottom machine: if (S + 1)mod 3 = R then S:= (S - 1)mod 3 fi

for the top machine: if L = R and (L + 1)mod 3 ≠ S then S:= (L + 1)mod 3 fi

for the other machines: if (S + 1)mod 3 = L then S:= L fi;
if (S + 1)mod 3 = R then S:= R fi .

Again the machines nr.i with $0 < i < N$ may enjoy two privileges, the neighbour relation between bottom and top machine has been exploited.

Acknowledgements are due to C.S.Scholten who unmasked an earlier effort as fallacious and since then has found a nice proof for solutions of the first type and to M.Woodger whose fascination by the first two solutions was an incentive to find the third.

NUENEN, 5th November 1973

prof.dr.Edsger W.Dijkstra
Research Fellow