A systematic way for finding the algorithm ensuring some desired form of co-operation between a set of loosely coupled sequential processes can in general terms be described as follows: the relation “the system is in a legitimate state” is kept invariant. As a consequence, each intended individual process step that could possibly cause violation of that invariant relation has to be preceded by a test that it won’t do so, and depending on the outcome of that test the critical process step is either caused to take place or it — and with it the process of which it is a part — is delayed until a more favourable system state has been reached. With a suitable choice of the set of legitimate states one can indeed introduce the rule that a critical process step will be delayed only as long as its execution would lead to violation of the corresponding invariant relation.

The resulting design is readily implemented if the different sequential processes can be granted mutually exclusive access to a common store in which the current system state is recorded. Then a relation between (the values of) the variables in that commonly accessible store is the core of what we could call “the centralized control”.

A complication arises when there is no such commonly accessible store and “the system state” must be recorded in variables distributed over the various processes, and furthermore the communication facilities are limited in the sense that each process can only exchange information with “its neighbours”, a (possibly small) subset of the total set of processes. (We can view the processes as nodes of a connected graph in which each of the (sparse) set of edges denotes the neighbour relation.) The complication is that a node’s behaviour can only be influenced by the part of the total system state description that is available in that node: local actions taken on account of local information must accomplish a global objective. Such
systems (with what is quite aptly called “distributed control”)) have been
designed, but all such designs I am familiar with are unstable in the sense
that, when once in an illegitimate state, they could remain so forever. I call a
system “self-stabilizing” when, regardless of its initial state, it is guaranteed
to arrive at a legitimate state in a finite number of steps. (Whether the
property of self-stabilization is interesting as a start procedure, for the sake
of system robustness, or merely as an intriguing problem, is a question that
falls outside the scope of this article.)

Unable to decide on theoretical grounds whether non-trivial self-stabiliz-
ing systems with distributed control could exist at all, I decided to try to
design one under the following constraints and objectives.

We consider a system built from $N + 1$ finite state machines numbered
from 0 through $N$. (The state space for the total system is then the Cartesian
product of the $N + 1$ individual state spaces of the respective machines.)
The machines are arranged in a ring, i.e. for $0 \leq i < N$, machine $nr.i$ has
machine $i + 1$ as its right-hand neighbour, and machine $N$ has machine 0 as
its right-hand neighbour.

In the middle of the ring stands a demon, each time giving, in “fair
random order”, one of the machines the command “to adjust itself”. (In
“fair random order” means that in each infinite sequence of successive
commands issued by the demon, each machine receives the command to
adjust itself infinitely often.) Upon “adjustment” a machine goes into a
(new) state, which must be a function of its own (old) state and the current
states of its (two) neighbours.

Furthermore, as a function of its own state (and possibly of the states of
its neighbours) a machine may be “privileged”. The legitimate states are
defined as those states in which exactly one machine is privileged and for
which all possible successor states are legitimate as well; furthermore it is
required that then the privilege will rotate around the ring.

SIDE REMARK. I was hoping for an existence proof of self-stabilizing systems
with distributed control: a ring is then one of the most natural, simple
connection graphs. My choice of legitimate states, viz. requiring conver-
gence towards a solution of the mutual exclusion problem, is understand-
able for historical reasons [1], [2], [3], [4], it is also justified by its central
position in the whole field of controlling co-operation between loosely
coupled processes. Finally, the choice of the demon was suggested by a
recent experience with a cyclic relaxation problem in which “fair random
relaxation” would converge to a limit, while simultaneous relaxation could
lead to oscillation (EWD386, unpublished). So much for the justification of
the problem choice.

Again I beg my intrigued readers to stop reading here and to try to solve
the stated problem themselves, for only then will they (slowly!) build up
some sympathy with my difficulties: the problem has been with me for
many months, while I was oscillating between trying to find a solution