Consensus in π-calculus
Ezra e. k. Cooper
May 7, 2006

Statement π-calculus solves consensus for any n processes.

I give a π-calculus system that solves consensus for any number of processes. For each process i, we have a numbered sequence of channels p_i^r. Each process also has a special channel $DECIDE_i$ for its output; it should only signal on this channel once, and all processes that send on such a channel should send the same value, for correctness of consensus. Each process starts out as $P_i^0(i)$, using i, its own process identifier, as the input to consensus. The processes are arranged in a circle; $i+1$ should be understood as $(i+1) \bmod n$, that is, the identifier of the next process around the circle.

$$\begin{aligned} P_i^r(v) &\equiv \sum_{j \neq i} \overline{p_j^r} v.I_i^r(v).P_i^{r+1}(v) + p_i^r(x).I_i^r(x).P_i^{r+1}(x) \\ I_i^r(v) &\equiv \overline{p_0^r} v \mid \ldots \mid \overline{p_n^r} v \\ P_i^{n-1}(v) &\equiv \overline{DECIDE_i}(v) \end{aligned}$$

The complete system is $S \equiv P_0^0(0) \mid P_1^0(1) \mid \ldots \mid P_n^0(n)$.

Correctness Each time a process i sends on p_j^r, the receiving process throws away its input value and adopts the one it receives. This reduces the number of distinct values amongst all processes by one. After such an event, the "winning" process signals all others asynchronously using the I_i^r process; any value received from one of these actions for round r is one of the values remaining after that round. So after $n-1$ such events there can be only one value held by any of the processes.

Wait-freedom We need to show that a process that has not decided always has an enabled transition, and that there is no infinite execution. It is obvious that a non-failing process decides in a finite number of rounds, if it can complete each round without waiting. It is obvious, too, that the I_i^r phase is wait-free, since it consists only of asynchronous message-sending.

As long as at least two processes have not failed, say i and j, and they have both reached the same round, a transition on p_i^r (or p_j^r) is enabled and they can proceed to the next round. Suppose two active processes (i and j) are not on the same round. Then the process (say, i) with the higher round number r' must have passed through the lower one, r, in which case it executed I_i^r and there is a transition enabled on p_j^r, thus it can proceed to round $r+1$.

For a process in round r, only transitions involving a channel p_x^r are blocking its progress, of which there are finitely many. Each channel receives exactly one message in its lifetime. Thus there can be no execution in which a process takes infinitely many steps but does not reach the next round.