Bakery Algorithm

This algorithm solves the critical section problem for n processes in software. The basic idea is that of a bakery; customers take numbers, and whoever has the lowest number gets service next. Here, of course, "service" means entry to the critical section.

```
 1  var choosing: shared array[0..n-1] of boolean;
 2       number: shared array[0..n-1] of integer;
               ...
 3  repeat
 4       choosing[i] := true;
 5       number[i] := max(number[0],number[1],...,number[n-1]) + 1;
 6       choosing[i] := false;
 7       for j := 0 to n-1 do begin
 8            while choosing[j] do (* nothing *);
 9            while number[j] <> 0 and
10                          (number[j], j) < (number[i],i) do
11                 (* nothing *);
12      end;
13      (* critical section *)
14      number[i] := 0;
15      (* remainder section *)
16      until false;
```

lines 1-2: Here, *choosing*[*i*] is true if process *i* is choosing a number. The number that process *i* will use to enter the critical section is in *number*[*i*]; it is 0 if process *i* is not trying to enter its critical section.

lines 4-6: These three lines first indicate that the process is choosing a number (line 4), then try to assign a unique number to the process process *i* (line 5); however, that does not always happen. Afterwards, process *i* indicates it is done (line 6).

lines 7-12: Now we select which process goes into the critical section. Process *i* waits until it has the lowest number of all the processes waiting to enter the critical section. If two processes have the same number, the one with the smaller name—the value of the subscript—goes in; the notation "$(a,b) < (c,d)$" means true if $a < c$ or if both $a = c$ and $b < d$ (lines 9-10). Note that if a process is not trying to enter the critical section, its number is 0. Also, if a process is choosing a number when process *i* tries to look at it, process *i* waits until it has done so before looking (line 8).

line 14: Now process *i* is no longer interested in entering its critical section, so it sets *number*[*i*] to 0.