Software Implementation – Lamport's Bakery Algorithm

CPUi

Shared variables: choosing, number: array[1..N] of integer;
Initialized to false and 0.

```
While(true) {

L1:     choosing[i] = 1;
        number[i] = 1 + max(number[1], …, number[N]);
        choosing[i] = 0;
        for (int j = 1; j <= N; j++) {
L2:     while (choosing[j] = 1) {}
L3:     while(number[j] <>0 and (number[j],j) < (number[i], i)) {}
        }
                Critical Section

        number[i] = 0;
}
```

Note: **(a, b) > (c, d) iff (a > c) OR [(a == c) AND (b > d)]**

Next we will go over the proof of two assertions that are necessary for a correct mathematical proof of the Mutual Exclusion. The first Assertion is used in proving the second one. The second assertion will have as direct consequence the proof of Mutual Exclusion.

Without lost of generality we will consider only two processes that are concurrently executing on a distributed system.

Assertion 1: if processes i and k are in the bakery, and i entered the bakery before k entered the doorway,

then **number[i] < number[k]** (1)

Proof: number[i] has been assigned before number[k] is assigned.

number[k] > = 1 + number[i] > number[i]

Assertion 2: if process i executes its Critical Section, and process K is in the bakery (k !=i)

then **(number[i], i) < (number[k], k)** (2)

Proof: Consider the times of events that occur from the point of view of CPUi (during the execution of process i)

T(L2): the time of last execution of L2
T(L3): the time of last execution of L3

T(L2) < T(L3) (3) /sequential execution/

Consider the times of events that occur from the point of view of CPUk (during the execution of process k)

T(e): time when k entered the doorway (choosing[k] = 1)
T(w): time when k finished writing its number[k]
T(c): time k left doorway (choosing[k] = 0)

T(e) < T(w) < T(c) (4) /sequential execution/

At T(L2) (last execution of L2) **choosing[k] = 0**

There are two situations: **T(L2) < T(e)** (before k enters the doorway) (5) Case 1.
OR
T(c) < T(L2) (after k leaves the doorway) (6) Case 2.

Case 1
(5) + (assertion 1) implies **number[i] < number[k]**

Case2
(6) + (3) + (4) implies T(w) < T(c) < T(L2) < T(L3) → **T(w) < T(L3)** (7)

(7) means that the last execution of L3 is done after the value of number[k] is stored (written, updated).
This means that the truth condition for second while will not changed. (until k will re-execute again the code)

Q: How many times L3 is executed by process i ? Answer: Only once.
Q: what is the truth value of the L3 loop when i executes L3? A: **False**. (our hypothesis says that I is in the critical section)

(7) implies that **number [k] > number[i]**

Conclusion:
number[i] < number[k] for any possible situation (on the given hypothesis)
Assertion2 is true.