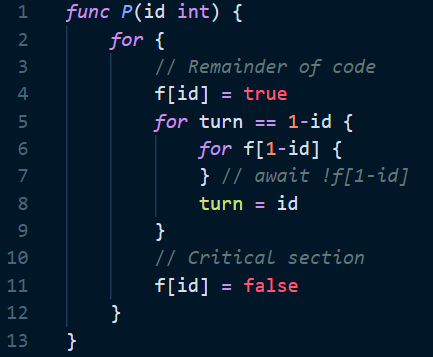
# Hyman’s Protocol mutual exclusion problem.

Demonstrated by Ýmir Þórleifsson & Bjartur Þórhallsson.

This code snippet includes the Hyman’s protocol to the mutual and we have found a counterexample that shows the error in the algorithm.

Claim: Processes P0 and P1 can execute line 10 and 11 at the same time (both can access the critical section at the same time).

If P0 is in the critical section (line 10) then f[0] == true and turn == 0.

Then P1 executes lines 4 (f[1] == true) and 5. Line 5 is false since turn == 0, thus 0 ≠ 1 and P1 never enters the loop and skips straight to the critical section while P0 is still there.

The following table shows the timeline of the two processes executing the algorithm:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Processes** | | **Variables** | | | **Comments** |
| **P0** | **P1** | **f[0]** | **f[1]** | **turn** |  |
| 1 |  | F | F | 0 | P0 starts execution. |
| 2 |  | F | F | 0 |  |
| 3 |  | F | F | 0 |  |
| 4 |  | T | F | 0 |  |
| 5 |  | T | F | 0 |  |
| 6 |  | T | F | 0 |  |
| 8 |  | T | F | 0 |  |
| 10 |  | T | F | 0 | P0 accesses critical section. |
|  | 1 | T | F | 0 | P1 starts execution. |
|  | 2 | T | F | 0 |  |
|  | 3 | T | F | 0 |  |
|  | 4 | T | T | 0 |  |
|  | 5 | T | T | 0 | P1 does not wait for P0 to finish execution. |
|  | 10 | T | T | 0 | P1 accesses critical section, mutual exclusion has been violated. |

We can confidently say that this algorithm does not solve the mutual exclusion problem, every time P0 is in the critical section, P1 is not blocked by anything and is able to access it at the same time.