Modelling Concurrent Systems

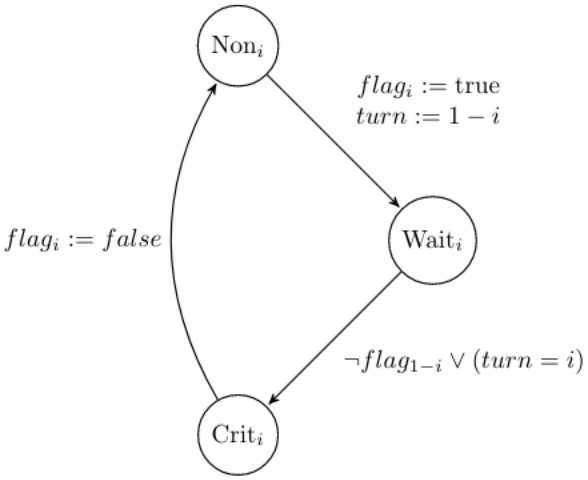
1 Peterson's Algorithm for Mutual Exclusion

Two processes P_0 and P_1, with shared variables: boolean $flag_i, i \in \{0,1\}$ and $turn \in \{0,1\}$

Listing 1 Process P_i

```
1 Non critical:
2 flag_i=true
3 turn=1-i
4 while flag_{1-i} ∧ (turn = 1 − i)
5     do
6         wait
7 end
8 Critical ection:
9 flag_i=false
```

Program Graph



2 Program Graph

Definition: Program Graph

A program graph over a finite set of Boolean variables has

1. A finite set of states S called locations
2. A deterministic transition relation $\rightarrow \subseteq S \times S \times (Act \cup Cond)$ where
   (a) Act is a set of atomic actions that change the values of some variables - if a transition labelled by an action is taken, the respective variables are updated accordingly
   (b) Cond is a set of formulae over the variables - such a transition can be taken only if the respective condition (formula) is true under the current valuation of the variables

3 Interleaved program graphs as a transition system

Given two program graphs $(S_1, \rightarrow_1)$ and $(S_2, \rightarrow_2)$ over a joint set of Boolean variables Vars, we create a transition system with states $S_1 \times S_2 \times 2^{vars}$ that simulates a concurrent execution of the two programs.

The transition relation of the system is defined as follows. Either

$$(s_1, s_2, v) \to (S_1', s_2, v_1)$$

Where $s_1 \to_1 s_1'$ and $\to_1$ changes/checks the values v into v_1 accordingly or

$$(s_1, s_2, v) \to (s_1, s_2', v_2)$$

where $s_2 \to_2 s_2'$ and $\to_2$ changes/checks the values v into v_2 accordingly. In words - one of the programs makes a step while the other stands still.