Kripke Structures

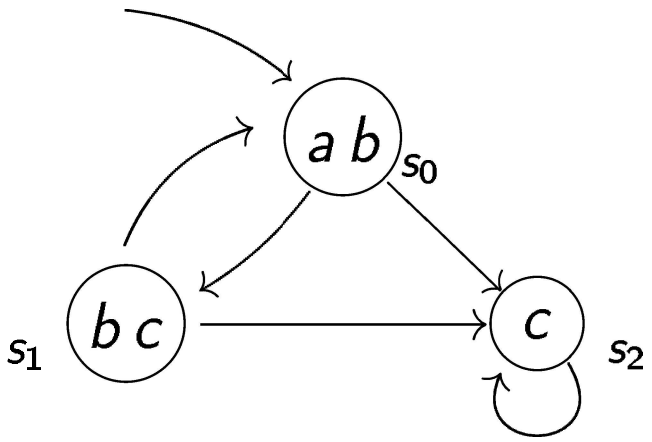
A **Kripke structure** M over a set *Prop* of *atomic propositions* is given by:

- a *finite* set S of states
- a subset $S_0 \subseteq S$ of initial states
- a transition relation $R \subseteq S \times S$ between states
- a valuation V giving, for each state s, the atomic propositions which are true in that state, $V(s) \subseteq \mathit{Prop}$

Think of S as modelling the states of a system, of R as modelling the computation steps, and of V as describing basic properties of states.

Paths through the Kripke structure then correspond to possible system executions !

Kripke Structures - Example



- set of states: $S = \{ s_0, s_1, s_2 \}$
- set of initial states: $\{ s_0 \}$
- transition relation:
  $\{ (s_0, s_1), (s_0, s_2), (s_1, s_0), (s_1, s_2), (s_2, s_2) \}$
  $s_0 \longrightarrow s_2 \quad s_2 \not\longrightarrow s_0$
- valuation V gives labelling of states with atomic propositions:
  $V(s_0) = \{a, b\}$
  $V(s_1) = \{b, c\}$
  $V(s_2) = \{c\}$

Example: Mutual Exclusion Protocol

bool $turn$;
$P \;=\; m$: **cobegin** $P_0 \parallel P_1$ **coend** m'
$P_0 \;=\; n_0$: **while** $True$ **do**
    t_0 : **wait** $(turn = 0)$;
    c_0 : `use resource;` $turn := 1$;
endwhile ; n_0'
$P_1 \;=\; n_1$: **while** $True$ **do**
    t_1 : **wait** $(turn = 1)$;
    c_1 : `use resource;` $turn := 0$;
endwhile ; n_1'

Note: **wait** (c) repeatedly tests c until it becomes true.

Extract a Kripke structure which models $P_0 \parallel P_1$:

- states are given by pairs of states of P_0 and P_1, together with the value of the shared variable $turn$
- transitions correspond to execution steps in either P_0 or P_1
- $Prop$ and V will depend on the properties we want to verify ...

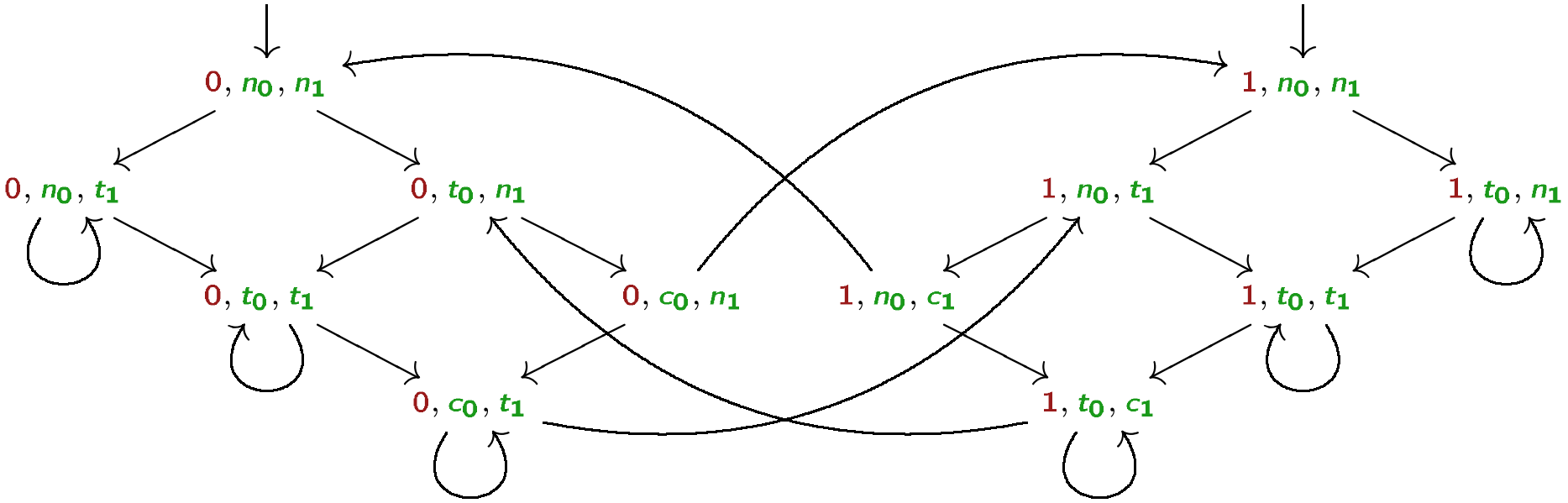
Mutual Exclusion: the Model

bool $turn$;

$P \;=\; m$: **cobegin** $P_0 \parallel P_1$ **coend** m'

$P_0 \;=\; n_0$: **while** $True$ **do**
t_0 : **wait** $(turn = 0)$;
c_0 : `use resource`; $turn := 1$;
endwhile ; n_0'

$P_1 \;=\; n_1$: **while** $True$ **do**
t_1 : **wait** $(turn = 1)$;
c_1 : `use resource`; $turn := 0$;
endwhile ; n_1'



(More on extracting Kripke structures from concurrent programs in Chapter 2 of Clarke, Grumberg and Peled.)