Model-Checking

- Idea of model-checking: establish that the system is a model of a formula (doing a search).
- CTL Model Checking
- SMV input language and its semantics
- SMV examples
- Model checking with fairness
- Binary Decision Diagrams.
- Symbolic model-checking and fixpoints.

CTL Model checking

- Assumptions:
  1. finite number of processes, each having a finite number of finite-valued variables.
  2. finite length of CTL formula
- Problem:Determine whether formula f_0 is true in a finite structure M.
- Algorithm overview:
  1. f_0 = $\mathtt{TRANSLATE}(f_0)$ (in terms of AF, EU, EX, $\wedge$, $\vee$, $\perp$)
  2. Label the states of M with the subformulas of f_0 that are satisfied there and work outwards towards f_0.
     Ex: $\mathrm{AF}(a \wedge \mathrm{E}(b\,\mathrm{U}\,c))$
  3. If starting state s_0 is labeled with f_0, then f_0 is holds on M, i.e.

$$(s_0 \in \{s \mid M, s \models f_0\}) \Rightarrow (M \models f_0)$$

Labeling Algorithm

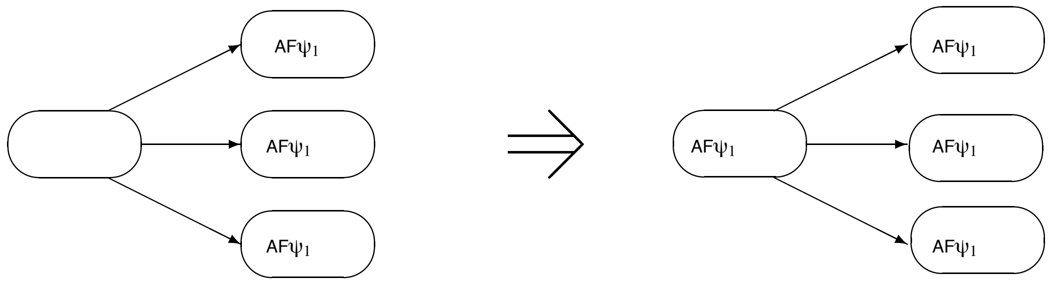
Suppose ψ is a subformula of f and states satisfying all the immediate subformulas of ψ have already been labeled. We want to determine which states to label with ψ. If ψ is:

- $\perp$: then no states are labeled with $\perp$.
- p (prop. formula): label s with p if $p \in I(s)$.
- $\psi_1 \wedge \psi_2$: label s with $\psi_1 \wedge \psi_2$ if s is already labeled both with ψ_1 and with ψ_2.
- $\neg\psi_1$: label s with $\neg\psi_1$ if s is not already labeled with ψ_1.
- EX ψ_1: label any state with EX ψ_1 if one of its successors is labeled with ψ_1.

Labeling Algorithm (Cont'd)

- AF ψ_1:
  - If any state s is labeled with ψ_1, label it with AF ψ_1.
  - Repeat: label any state with AF ψ_1 if all successor states are labeled with AF ψ_1, until there is no change.

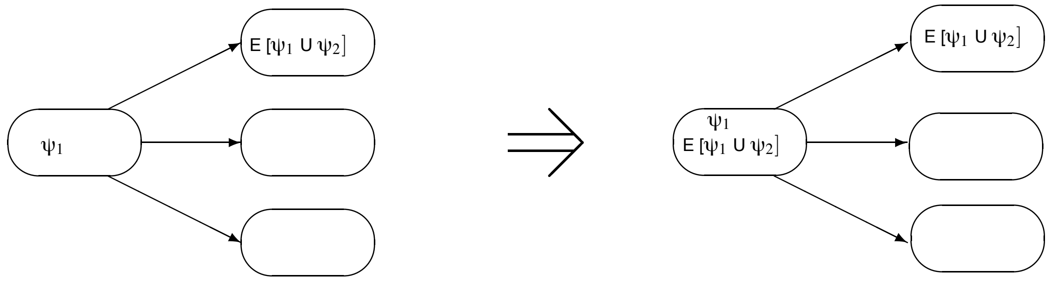
Ex:



Labeling Algorithm (Cont'd)

- E [ψ_1 U ψ_2]:
  - If any state s is labeled with ψ_2, label it with E[ψ_1 U ψ_2].
  - Repeat: label any state with E[ψ_1 U ψ_2] if it is labeled with ψ_1 and at least one of its successors is labeled with E[ψ_1 U ψ_2], until there is no change.

Ex:



Output states labeled with f.

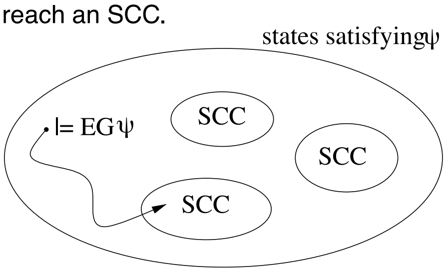
Complexity: $O(|f| \times S \times (S + |R|))$ (linear in the size of the formula and quadratic in the size of the model).

Handling EGψ_1 directly

- EG ψ_1:
  - Label *all* the states with EG ψ_1.
  - If any state s is *not* labeled with ψ_1, *delete* the label EG ψ_1.
  - Repeat: *delete* the label EG ψ_1 from any state if *none* of its successors is labeled with EG ψ_1; until there is no change.

Even Better Handling of EG

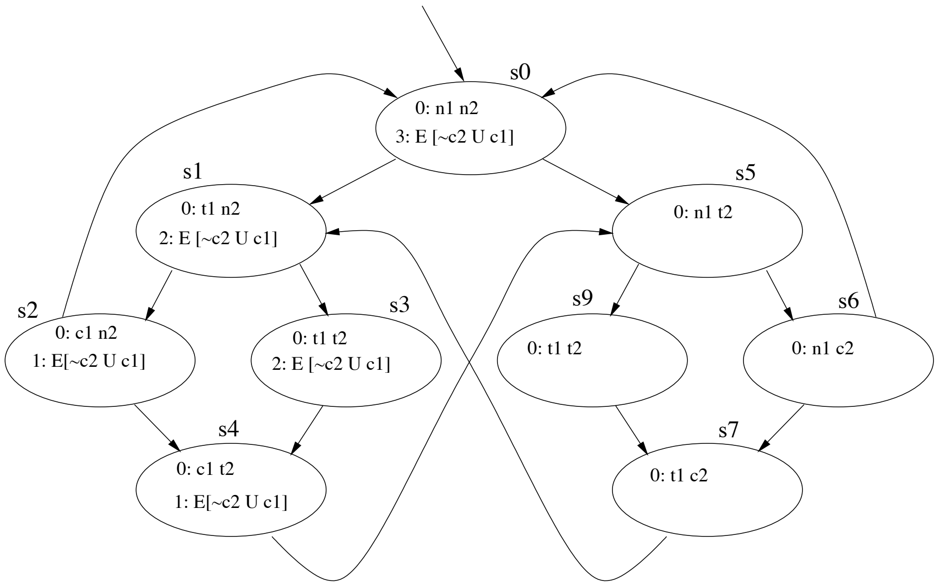
● restrict the graph to states satisfying ψ_1, i.e., delete all other states and their transitions;
● find the maximal *strongly connected components* (SCCs); these are maximal regions of the state space in which every state is linked with every other one in that region.
● use breadth-first searching on the restricted graph to find any state that can reach an SCC.



Complexity: $O(|f| \times (S + |R|))$ (linear in size of model and size of formula).

Example

Verifying E[$\neg c_2$ U c_1] on the mutual exclusion example.



CTL Model-Checking

- Michael Browne, CMU, 1989.
- Usually for verifying concurrent *synchronous* systems (hardware, SCR specs...)
- Specify correctness criteria: safety, liveness...
- Instead of keeping track of labels for each state, keep track of a set of states in which a certain formula holds.

Example

Verifying E[$\neg c_2$ U c_1] on the mutual exclusion example.