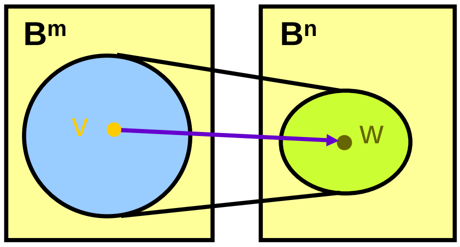
Symbolic Pre-Image Computation

- **Definition**. Let F: $B^m \times B^n$ be a projection and C be a set of minterms in B^m. Then the pre-image of C is the set $PreImg(C, F) = \{ v \in B^m \mid (v, w) \in F \text{ and } w \in C\}$ in B^n.

- Characteristic Function
  - for reachable previous-state computation

$$N_i(\vec{s}) = PreImg(R_i(\vec{s}\,'), T_\exists(\vec{s}, \vec{s}\,'))$$
$$= \exists \vec{s}\,'.(R_i(\vec{s}\,') \wedge T_\exists(\vec{s}, \vec{s}\,'))$$
$$= \exists \vec{s}\,'.(R_i(\vec{s}\,') \wedge (\exists \vec{x}. \prod_i (s_i' \equiv \delta_i(\vec{x}, \vec{s}))))$$



Reachability Analysis

```
ForwardReachability( Transition Relation T, Initial State I )
{
    i := 0
    R^i := I
    repeat
        R_new = Image( R^i, T );
        i := i + 1
        R^i := R^(i-1) ∨ R_new
    until R^i = R^(i-1)
    return R^i
}
```

- The procedures can be realized using BDD package.

- Backward reachability analysis can be done in a similar manner with pre-image computation and starting from final states to see if they can be reached from initial states.

Sequential Equivalence Checking

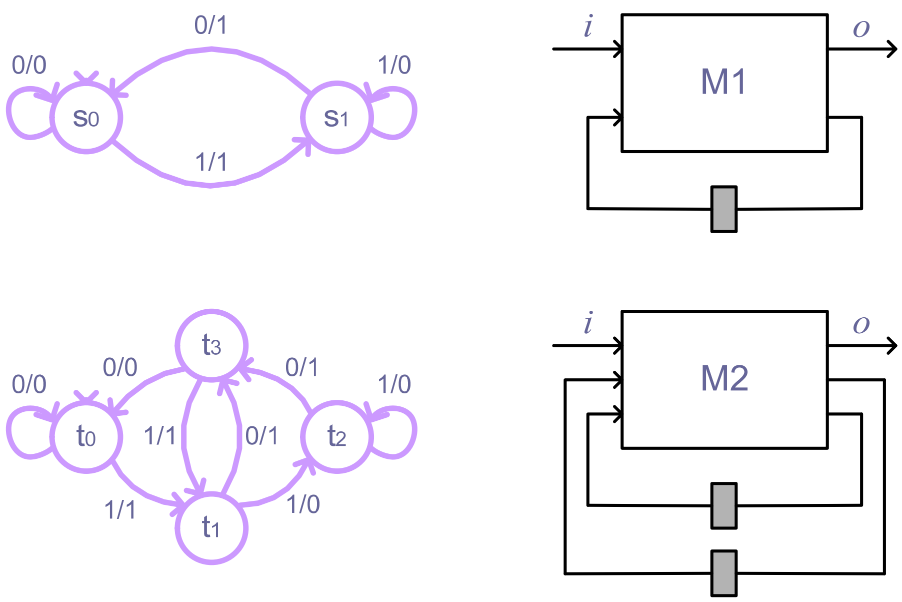
- Let R(s) be the characteristic function of the reachable state set of the product FSM $M_{1\times 2}$ obtained from forward reachability analysis. Then FSMs M_1 and M_2 are equivalent if and only if

$$R(s) \rightarrow (\lambda_{1\times 2}(x,s)\equiv 0)$$

  is valid for all valuations on input variables x and state variables s.
  - This can be checked in constant time for BDD

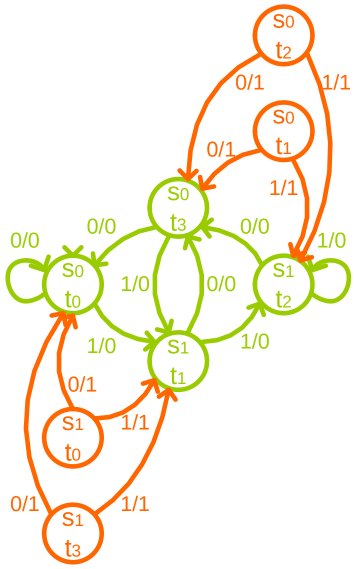
Sequential Equivalence Checking

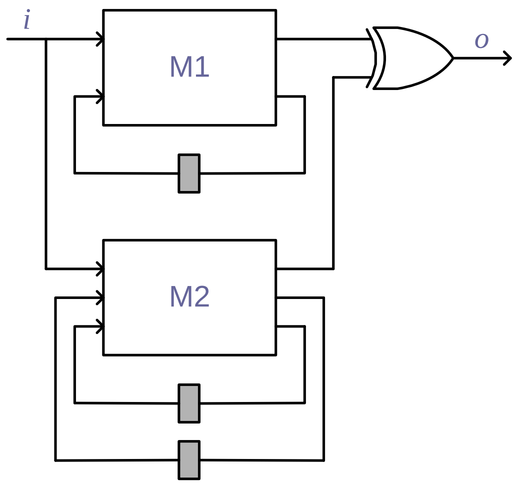
- Example
  - Are M1 and M2 equivalent ?



Sequential Equivalence Checking

- Example (cont'd)
  - Product FSM of M1 and M2

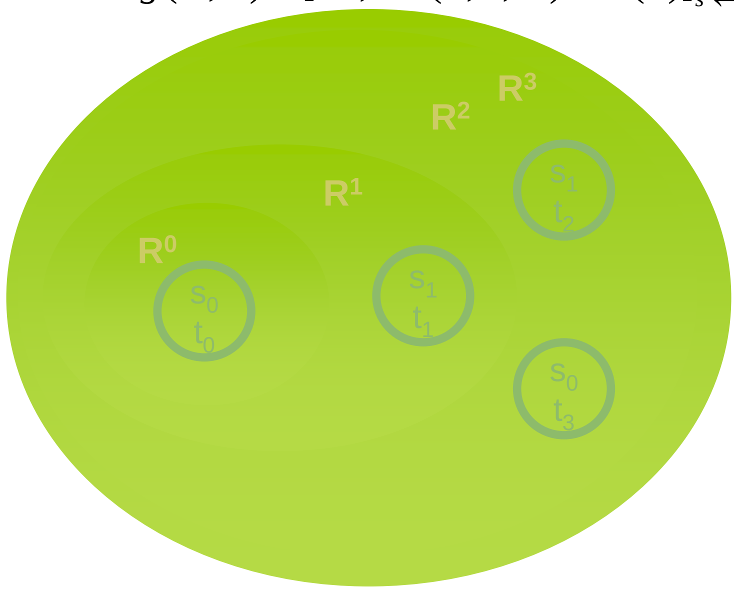


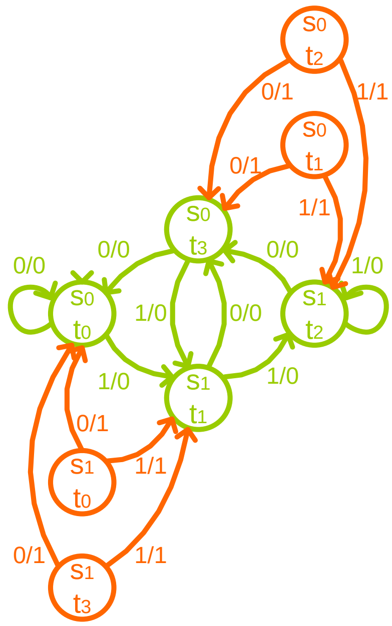


Sequential Equivalence Checking

- Example (cont'd)
  - Forward reachability analysis

$$Img(C,T) = [\exists \vec{x}, \vec{s}.T(\vec{x},\vec{s},\vec{s}\,') \wedge C(\vec{s})]_{\vec{s}\,' \leftarrow \vec{s}}$$

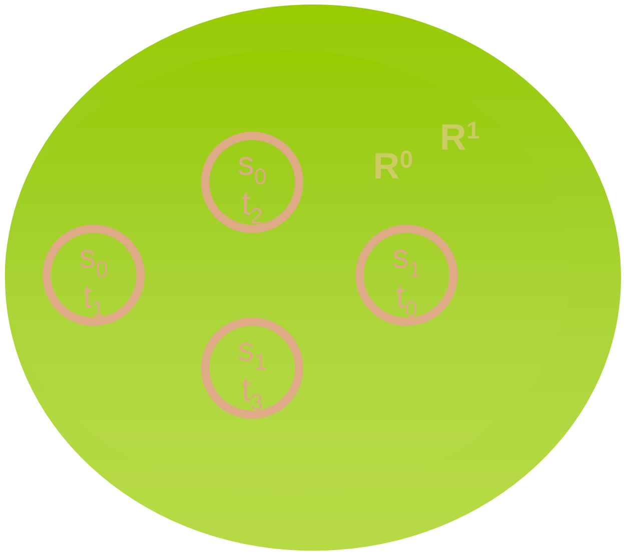


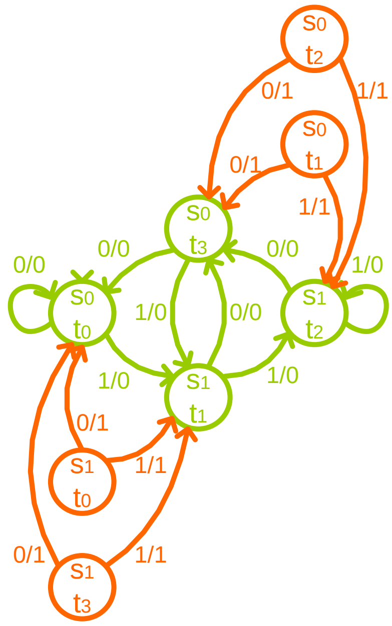


Sequential Equivalence Checking

- Example (cont'd)
  - Backward reachability analysis

    $PreImg(C,T) = \exists \vec{x}, \vec{s}\,'.T(\vec{x}, \vec{s}, \vec{s}\,') \wedge C(\vec{s}\,')$





Remarks on Sequential EC

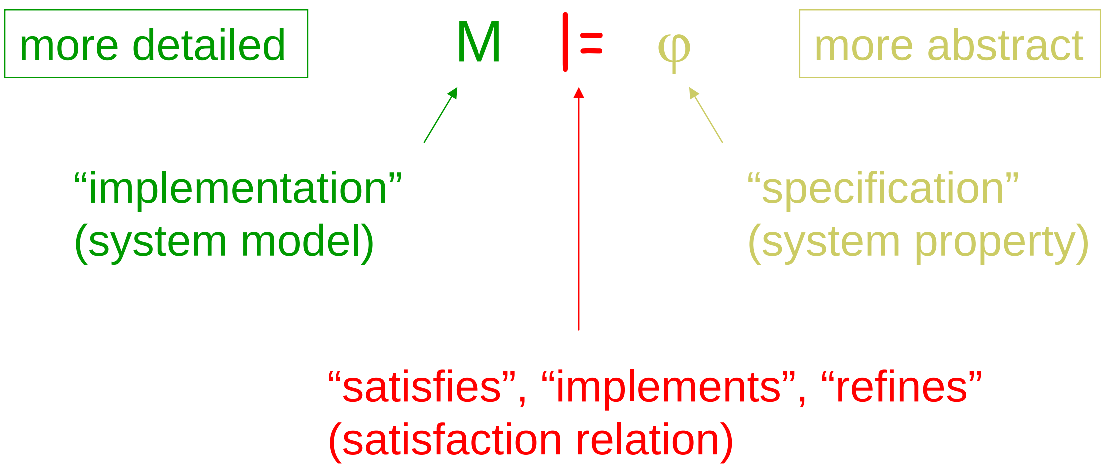
- Industrial equivalence checkers almost exclusively use an combinational EC paradigm even for sequential EC
  - Sequential EC is too complex and can only be applied to design with a few hundred state bits
  - Structure similarity should be identified to simplify sequential EC
- Besides sequential equivalence checking, reachability analysis is useful in sequential circuit optimization
  - In sequential optimization, unreachable states can be used as sequential don't cares to optimize a sequential circuit

Outline

- Introduction
- Boolean reasoning engines
- Equivalence checking
- Property checking
  - Safety property checking

Model Checking

- A specific model-checking problem is defined by



Model Checking

- M |= φ
  - Check if system model M satisfies a system property φ
  - System model M is described with a state transition system
    - finite state or infinite state
  - Temporal property φ can be described with three orthogonal choices:
    1. operational vs. declarative: automata vs. logic
    2. may vs. must: branching vs. linear time
    3. prohibiting bad vs. desiring good behavior: safety vs. liveness

  Different choices lead to different model checking problems.

Property Checking

- Safety property:
  Something "bad" will never happen
  - Safety property violation always has a finite witness
    - if something bad happens on an infinite run, then it happens already on some finite prefix
  - Example
    - Two processes cannot be in their critical sections simultaneously

- Liveness property:
  Something "good" will eventually happen
  - Liveness property violation never has a finite witness
    - no matter what happens along a finite run, something good could still happen later
  - Example
    - Whenever process P1 wants to enter the critical section, provided process P2 never stays in the critical section forever, P1 gets to enter eventually

For finite state systems, liveness can be converted to safety!

Safety Property Checking

- Safety property checking can be formulated as a reachability problem
  - Are bad states reachable from good states?
- Sequential equivalence checking can be considered as one kind of safety property checking
  - M : product machine
  - φ : all states reachable from initial states has output 0

Model Checking

- Data structure evolution
  - State graph (late 70s-80s)
    - Problem size ~10^4 states
  - BDD (late 80s-90s)
    - Problem size ~10^{20} states
    - Critical resource: memory
  - SAT (late 90s-)
    - GRASP, SATO, chaff, berkmin
    - Problem size ~10^{100} (?) states
    - Critical resource: CPU time

Remarks on Model Checking

- Model checking is a very rich subject developed since early 1980's

- It is a variation of mathematical logic and is concerned with automatic temporal reasoning

- Reference

  M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.