Lecture 2: CTL Model Checking

- What is model checking?
- State transition systems
- Computation Tree Logics
- The logic CTL
- Typical CTL formulas
- Structure of the SMV model checker
- SMV examples

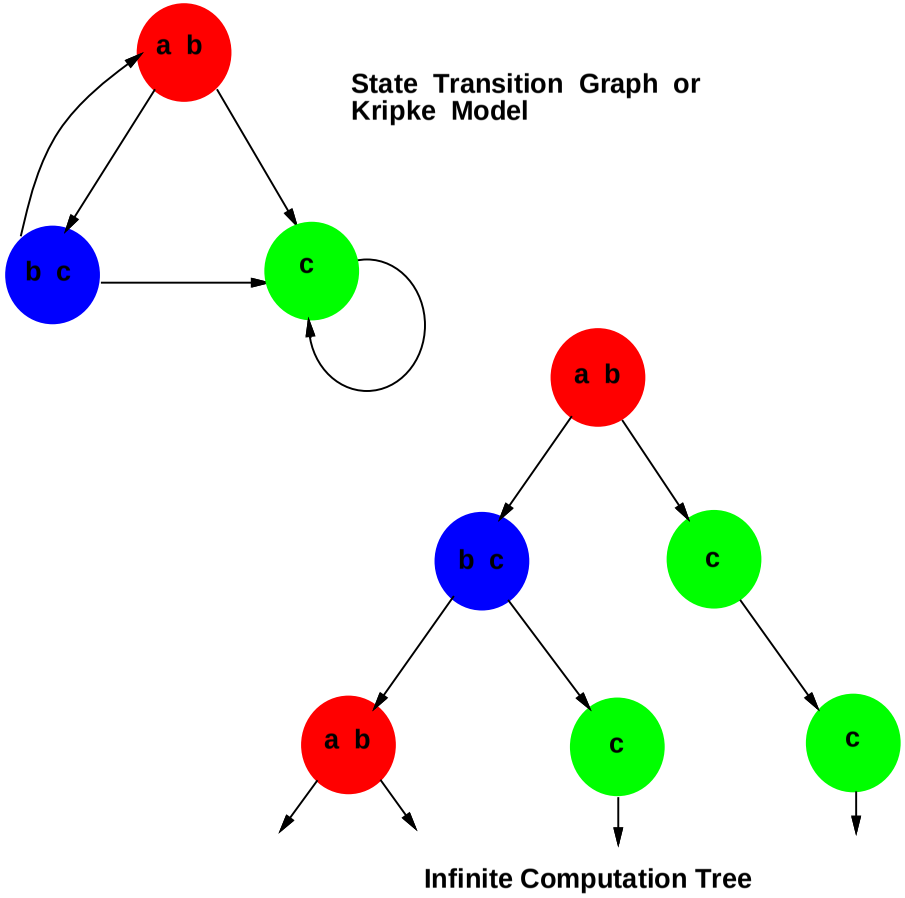
Temporal Logic Model Checking

Specification Language: A propositional temporal logic.

Verification Procedure: Exhaustive search of the state space of the concurrent system to determine truth of specification.

- E. M. Clarke and E. A. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In *Logic of programs: workshop, Yorktown Heights, NY, May 1981*, volume 131 of *Lecture Notes in Computer Science*. Springer-Verlag, 1981.
- J.P. Quielle and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proceedings of the Fifth International Symposium in Programming*, volume 137 of *Lecture Notes in Computer Science*. Springer-Verlag, 1981.

Temporal Logic



(Unwind State Graph to obtain Infinite Tree)

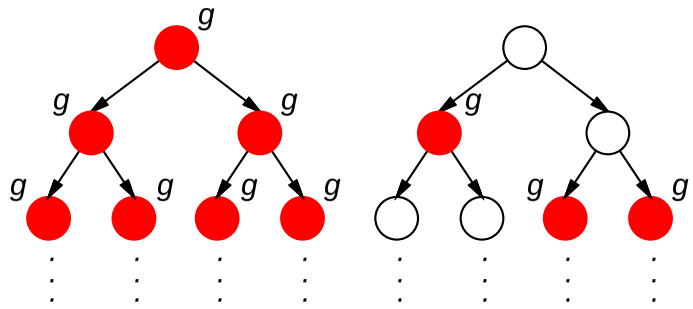
Computation Tree Logics

Formulas are constructed from path quantifiers and temporal operators:

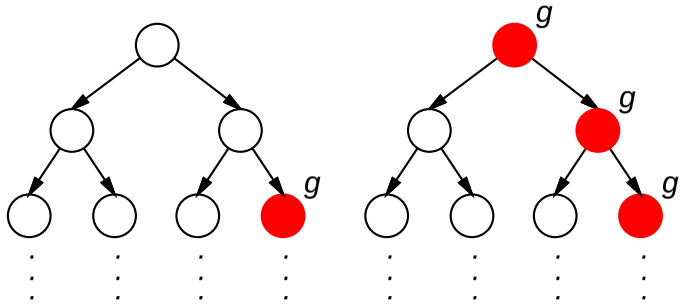
1. Path quantifier:
   - **A**—"for every path"
   - **E**—"there exists a path"
2. Temporal Operator:
   - **X**p—p holds next time.
   - **F**p—p holds sometime in the future
   - **G**p—p holds globally in the future
   - p**U**q—p holds until q holds

The Logic CTL

This lecture will deal primarily with CTL. The four most widely used CTL operators are illustrated below. (s_0 is the root of each computation tree.)



$M, s_0 \models \mathbf{AG}\, g$ $M, s_0 \models \mathbf{AF}\, g$



$M, s_0 \models \mathbf{EF}\, g$ $M, s_0 \models \mathbf{EG}\, g$

Typical CTL Formulas

- **EF**$(Started \wedge \neg Ready)$: it is possible to get to a state where *Started* holds but *Ready* does not hold.
- **AG**$(Req \Rightarrow$ **AF** $Ack)$: if a *Request* occurs, then it will be eventually *Acknowledged*.
- **AG**(**AF** $DeviceEnabled)$: *DeviceEnabled* holds infinitely often on every computation path.
- **AG**(**EF** $Restart)$: from any state it is possible to get to the *Restart* state.

Model Checking Problem

Let M be the state–transition graph obtained from the concurrent system.

Let f be the specification expressed in temporal logic.

Find all states s of M such that

$$M, s \models f.$$

and check if initial states are among these.

Efficient model checking algorithms exist for CTL.

- E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Programming Languages and Systems*, 8(2):pages 244–263, 1986.

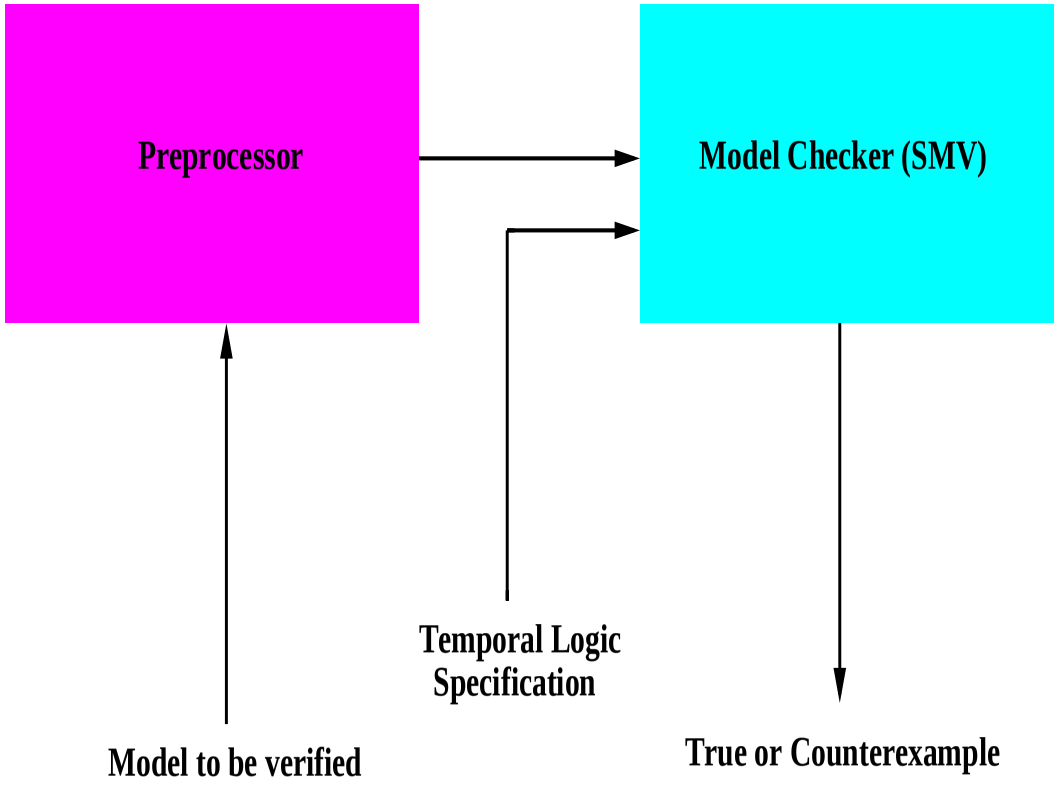
Symbolic Model Checking

Method used by most "industrial strength" model checkers:

- uses boolean encoding for state machine and sets of states.
- can handle much larger designs – hundreds of state variables.
- BDDs traditionally used to represent boolean functions.

Model Checker Structure

Symbolic Model Verifier (SMV)



A Simple SMV Example

```
MODULE main
VAR
  request : boolean;
  state : {ready,busy};
ASSIGN
  init(state) := ready;
  next(state) := case
                   state = ready & request : busy;
                   1 : {ready,busy};
                 esac;
SPEC
  AG(request -> AF state = busy)
```

A Three Bit Counter

```
MODULE main
VAR
  bit0 : counter_cell(1);
  bit1 : counter_cell(bit0.carry_out);
  bit2 : counter_cell(bit1.carry_out);
SPEC
  AG AF bit2.carry_out

SPEC AG(!bit2.carry_out)

MODULE counter_cell(carry_in)
VAR
  value : boolean;
ASSIGN
  init(value) := 0;
  next(value) := (value + carry_in) mod 2;
DEFINE
  carry_out := value & carry_in;
```

Inverter Ring

```
MODULE main
VAR
  gate1 : process inverter(gate3.output);
  gate2 : process inverter(gate1.output);
  gate3 : process inverter(gate2.output);
SPEC
  (AG AF gate1.output) & (AG AF !gate1.output)

MODULE inverter(input)
VAR
  output : boolean;
ASSIGN
  init(output) := 0;
  next(output) := !input;
FAIRNESS
  running
```