Introduction to Software Verification

Orna Grumberg

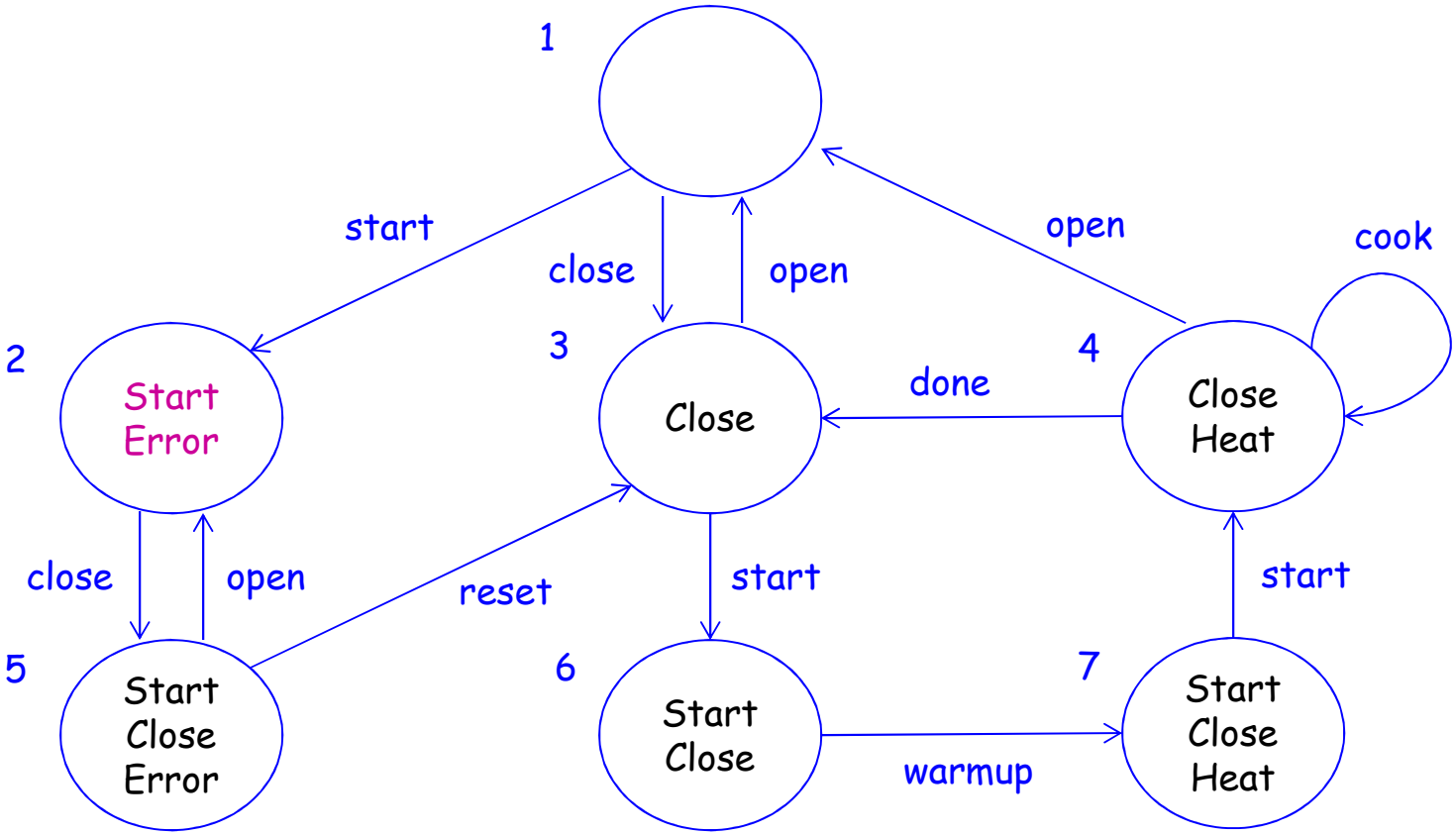
Lectures Material
winter 2017-18

Lecture 7

Model Checking Complexity

- Each subformula requires $O(|M|)$
- Number of subformulas: $O(|f|)$
- Total: $O(|M| \times |f|)$

Microwave Example

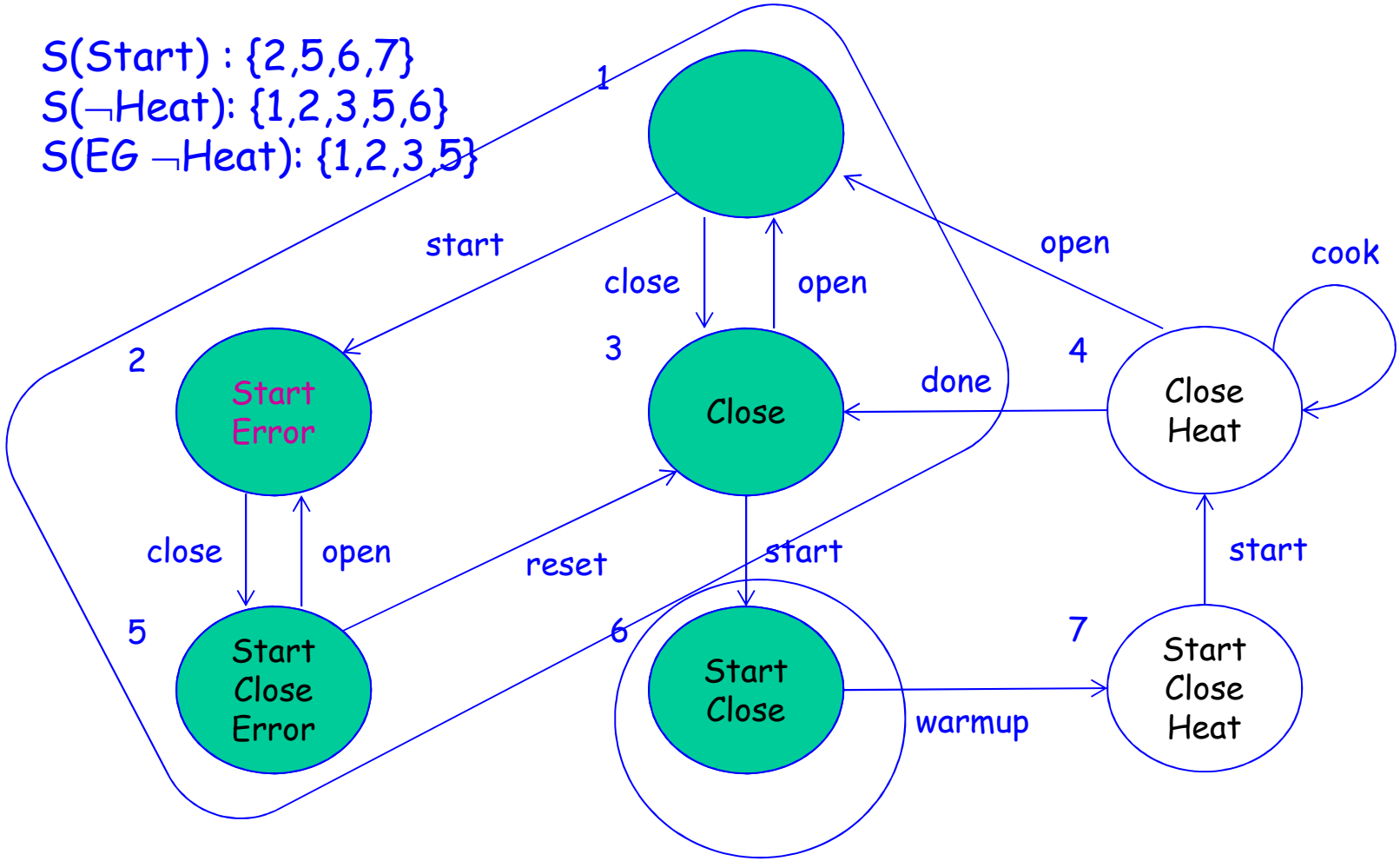


Property

- $AG\ (Start \rightarrow AF\ Heat)$
- $\neg EF\ (Start \wedge EG\ \neg Heat)$
- $\neg E\ (true\ U\ (Start \wedge EG\ \neg Heat))$

Instead of writing the formulas in label(s) for each s, use **S(f)** to denote the set of states s.t. **f $\in$ label(s)**

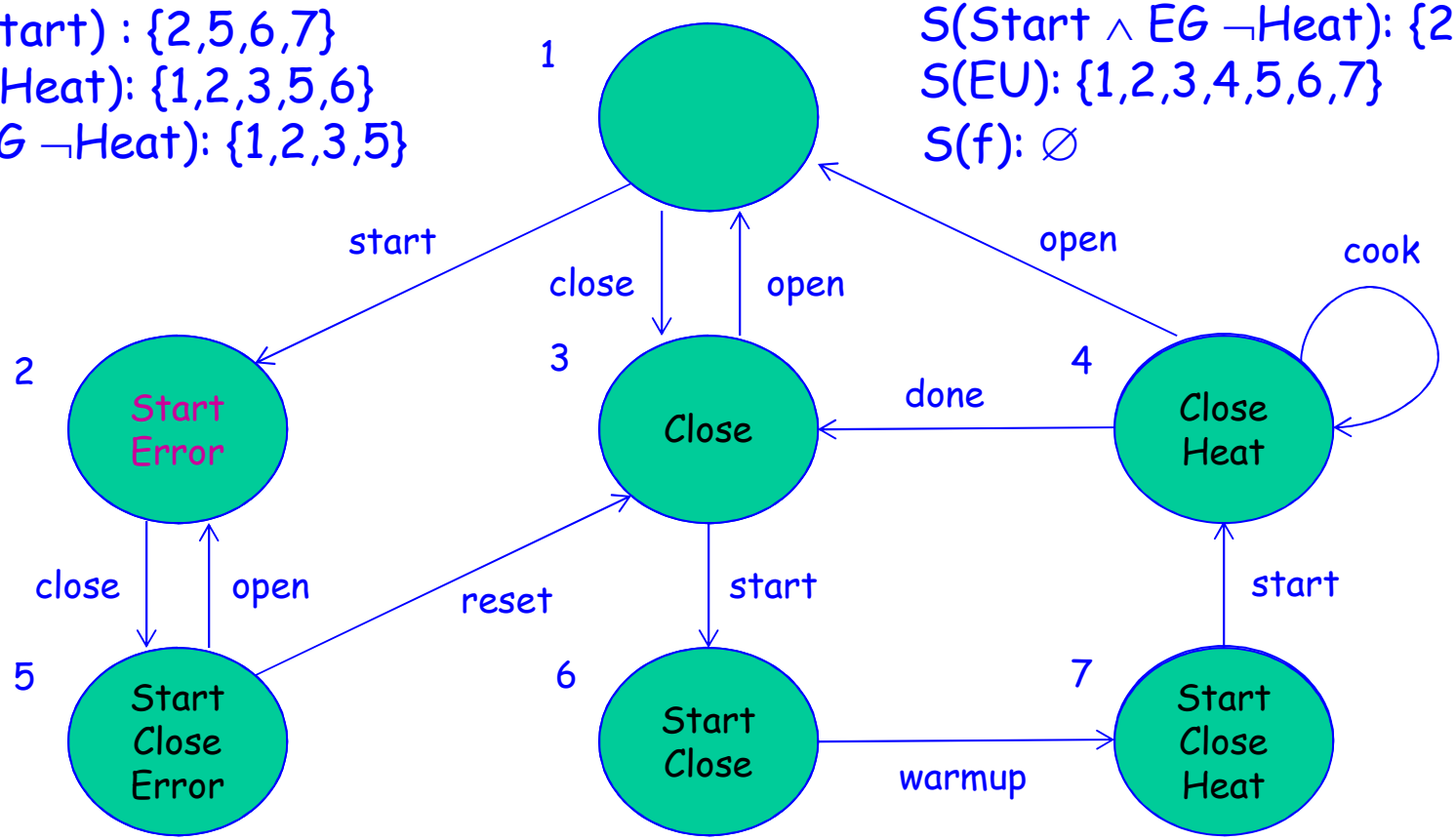
¬E (true U (Start ∧ EG ¬Heat))



¬E (true U (Start ∧ EG ¬Heat))

S(Start) : {2,5,6,7}
S(¬Heat): {1,2,3,5,6}
S(EG ¬Heat): {1,2,3,5}

S(Start ∧ EG ¬Heat): {2, 5}
S(EU): {1,2,3,4,5,6,7}
S(f): ∅



Explicit Model Checking for Fair CTL

Motivation

Fair CTL (CTL^F)

- Same syntax as CTL
- Different semantics

CTL^F formulas are interpreted over fair Kripke structures

Fair Kripke Structures

Fair Kripke structure $M = (S, S_0, R, L, F)$

- S, S_0, R, L – as before
- $F \subseteq 2^S$ is a set of fairness constraints

- $F = \{ P_1, \ldots P_k \}$ where
- $P_i \subseteq S$

  or
- P_i is a CTL formula

Fairness

Fair paths:

- $\pi = s_0, s_1, s_2, \ldots$
- $\inf(\pi) = \{ s \mid s = s_i \text{ for infinitely many } i\}$

π is fair if for every $P \in F$, $\inf(\pi) \cap P \neq \emptyset$

Example

- F = { {1,4}, {1,6}, {3} }

- A path π is fair iff inf(π) includes one of the sets:
- {1, 3}
- {4, 6, 3}
- Or their extensions

Semantics of Fair CTL

- $M,s \models_F EX\ \psi \Leftrightarrow$ there exists a fair path $\pi = s_0,s_1,\ldots$ from s such that $M, s_1 \models_F \psi$

- $M,s \models_F AX\ \psi \Leftrightarrow$ for every fair path $\pi = s_0,s_1,\ldots$ from s, $M, s_1 \models_F \psi$

- Similarly for EG, AG, EU, AU,…

Examples

Fairness constraints for hardware design, expressed as formulas:

- One input that should be 1 infinitely often
- K inputs, each should be 1 infinitely often
- K inputs that should be 1 together infinitely often

Model checking Fair CTL

- Needs to consider only fair paths
- $g \in \text{label}(s) \Leftrightarrow M,s \models_F g$

Reminder: Model Checking $g = EG\ f_1$ without fairness

Observation:

- $s \models EG\ f_1$
  iff
- s is the start of a path where all states satisfy f_1
  iff
- s has a finite path to a nontrivial, Maximal Strongly Connected Component (MSCC), where all states satisfy f_1

Model Checking $g = EG\ f_1$ with fairness

Observation:

- $s \models_F EG\ f_1$
  iff
- s is the start of a fair path where all states satisfy f_1
  iff
- s has a finite path to a nontrivial Maximal Fair Strongly Connected Component (MFSCC), where all states satisfy f_1

$M,s \models_F EG\ f_1$

Strongly connected component C is fair iff for every $P \in F$, $C \cap P \neq \emptyset$

Reduced structure:

Remove from M all states s.t. $f_1 \notin$ label(s).

Resulting model: $M' = (S', R', L', F')$

- $S' = \{ s \mid M, s \models_F f_1 \}$
- R', L' defined as before
- $F' = \{ P_i \cap S' \mid P_i \in F \}$

$M,s \models_F EG\ f_1$

Theorem: $M,s \models_F EG\ f_1$ iff

1. $s \in S'$ and
2. There is a path in M' from s to some state t in a nontrivial maximal fair strongly connected component of M'

Proof: similar to theorem for EG without fairness

$M,s \models_F EG\ f_1$

procedure CheckFairEG (f_1)

$S' := \{s \mid f_1 \in label(s)\}$

MFSCC := { C | C is a nontrivial fair MSCC of M' }

$T := \cup_{C \in MFSCC} \{ s \mid s \in C \}$

For all $s \in T$ do $label(s) := label(s) \cup \{ EG\ f_1 \}$

while $T \neq \emptyset$ do

choose $s \in T$; $T := T \setminus \{s\}$;

for all $t \in S'$ s.t. $R'(t,s)$ do

if $EG\ f_1 \notin label(t)$ then

$label(t) := label(t) \cup \{ EG\ f_1 \}$;

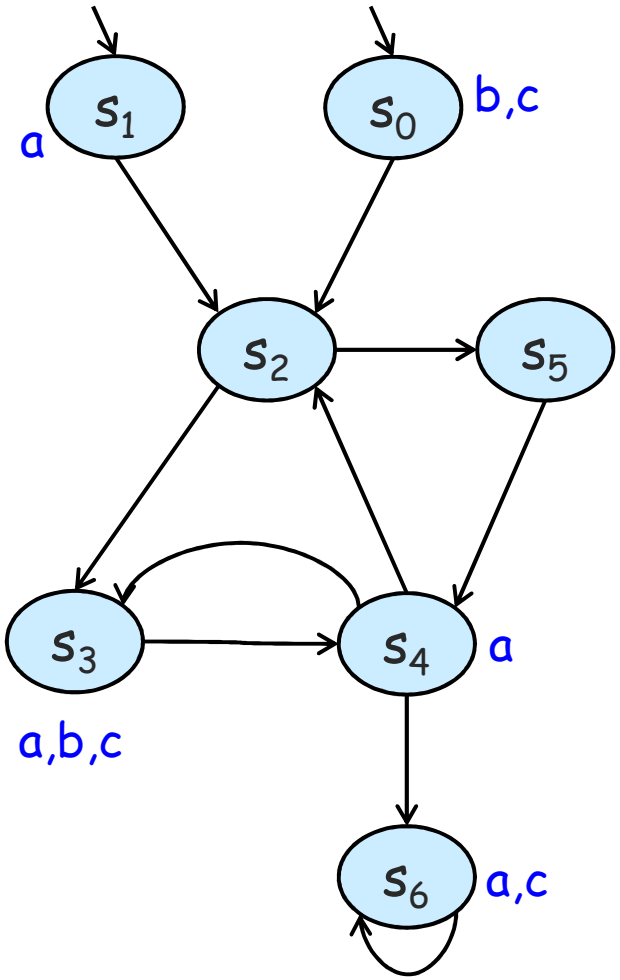
$T := T \cup \{t\}$

end for all

end while

Complexity increases to $O((|S| + |R|) \cdot |F|)$

$M,s \models_F EGa$ with $F=\{ \{1,2\}, \{1,5\} \}$



$F' = \{ \{1\} \}$

$MSCC = \{ \{1\}, \{3,4\}, \{6\} \}$

$FMSCC = \phi$

Set of states satisfying $E_F Ga$ is empty

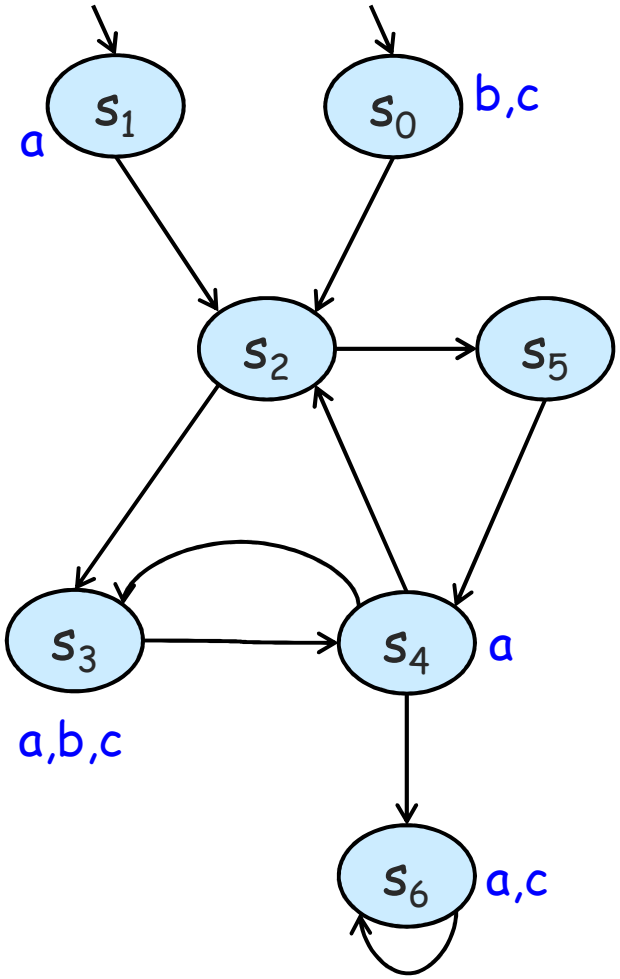
Model Checking other Fair CTL Formulas

- Add atomic proposition fair to all states that satisfy $M,s \models_F EG\ true$

- $M,s \models_F EX\ f_1$ iff $M,s \models EX\ (f_1 \wedge fair)$
- $M,s \models_F E\ [f_1\ U\ f_2]$ iff $M,s \models E\ [f_1\ U\ (f_2 \wedge fair)]$

Overall complexity: $O(|f| \cdot (|S| + |R|) \cdot |F|)$

Computing S_{fair} with F={ {1,2}, {1,5} }

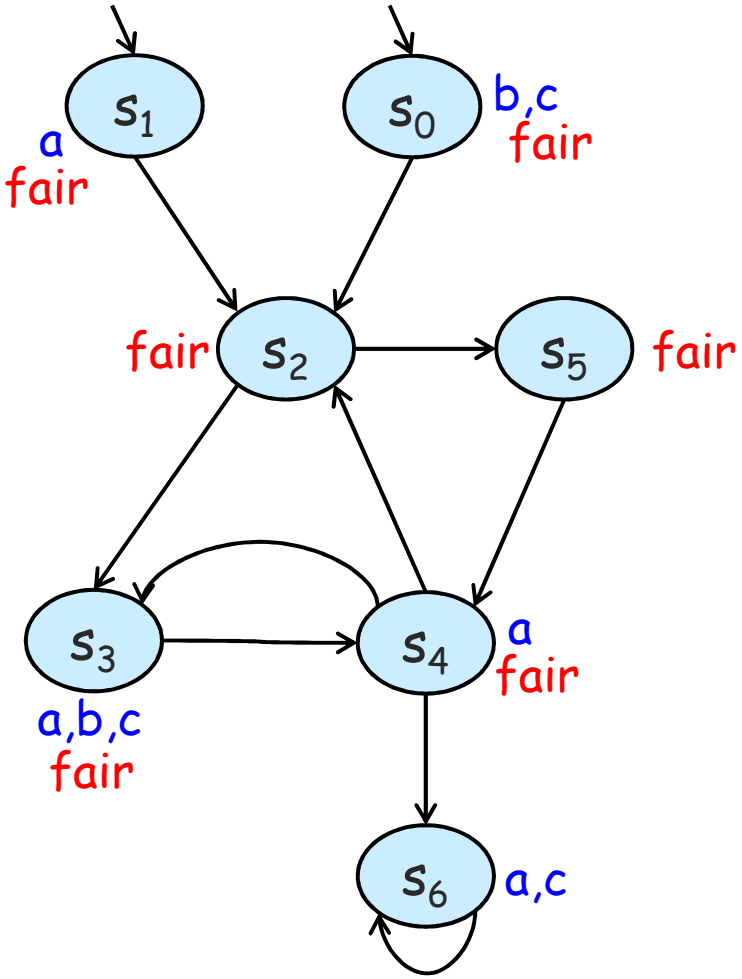


MSCC = { {0}, {1}, {2,3,4,5}, {6} }

FMSCC = { {2,3,4,5} }

$S_{fair} = \{2,3,4,5\} \cup \{0,1\}$

$M,s \models_F E(aUc)$ with F={ {1,2}, {1,5} }



$S_{fair} = \{0,1,2,3,4,5\}$

$E_F(aUc) \equiv E(aU(c \wedge fair)) = \varphi$

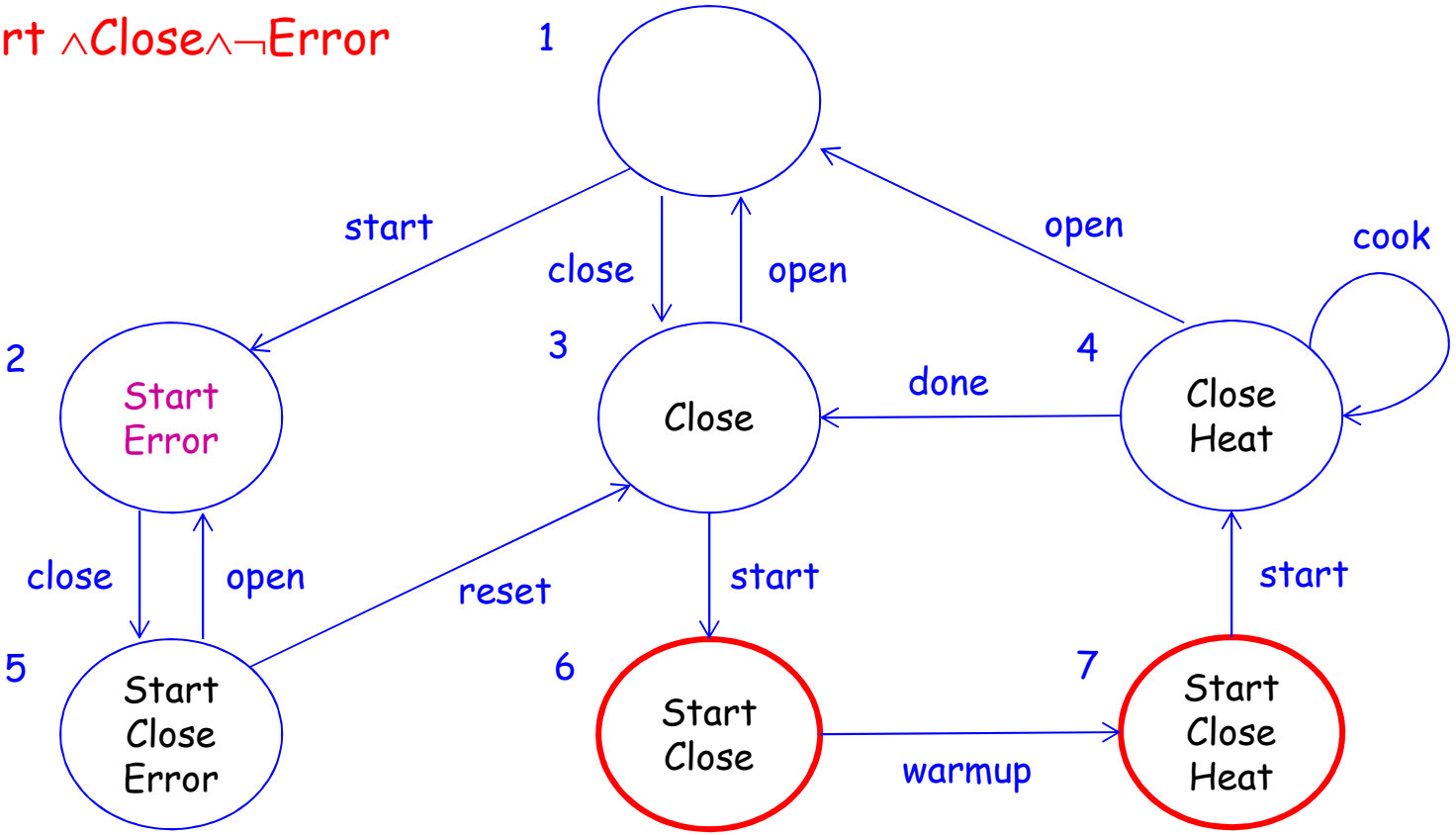
$S_a = \{1,3,4,6\}$ $S_c = \{0,3,6\}$

$S_{fair \wedge c} = \{0,3\}$

$S_\varphi = \{0,3,4\}$

Microwave Example

Fairness:
Start ∧Close∧¬Error

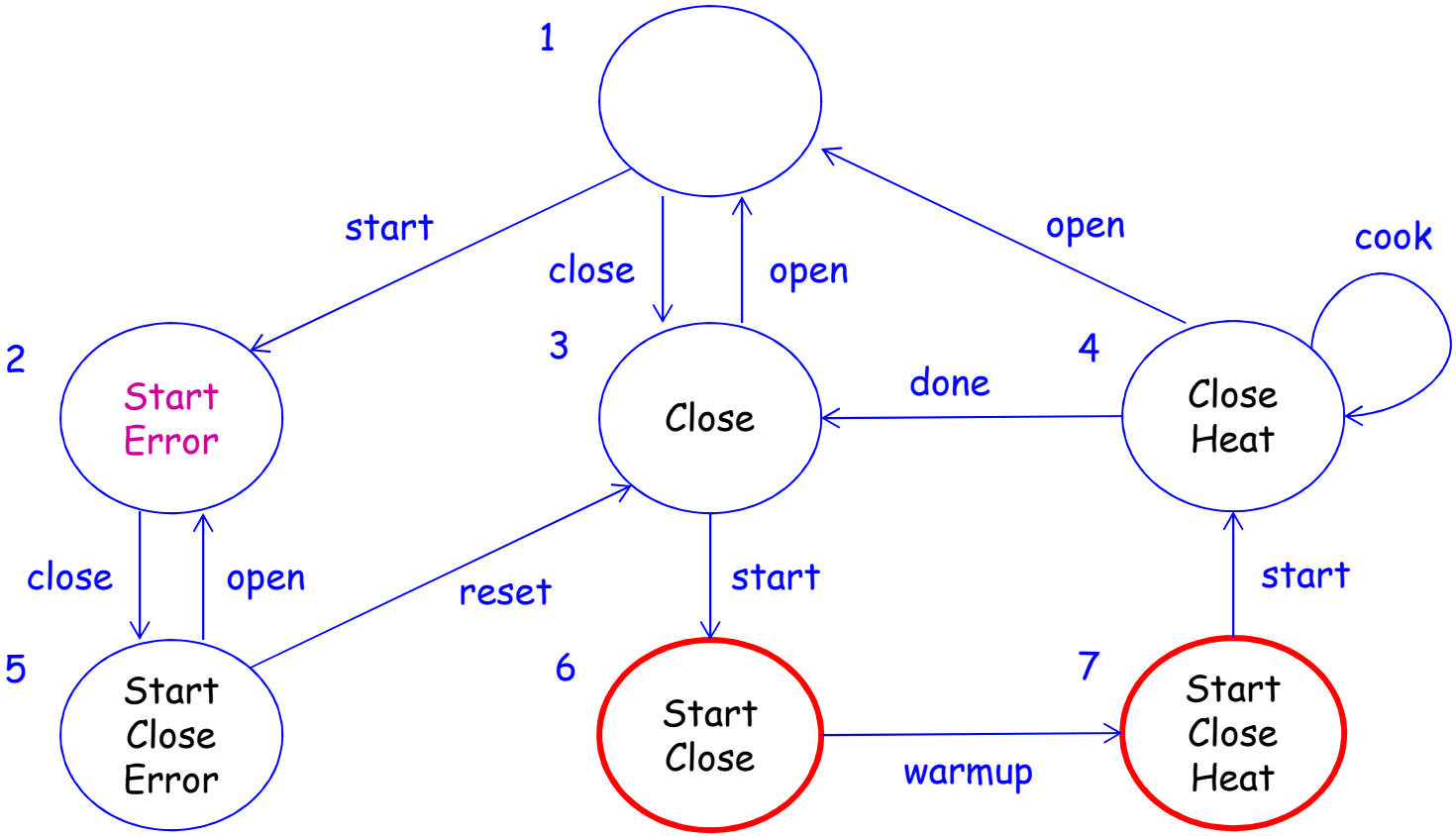


Property

- $AG\ (start \rightarrow AF\ Heat)$
- $\neg EF\ (start \wedge EG\ \neg Heat)$
- $\neg E\ (true\ U\ (start \wedge EG\ \neg Heat))$

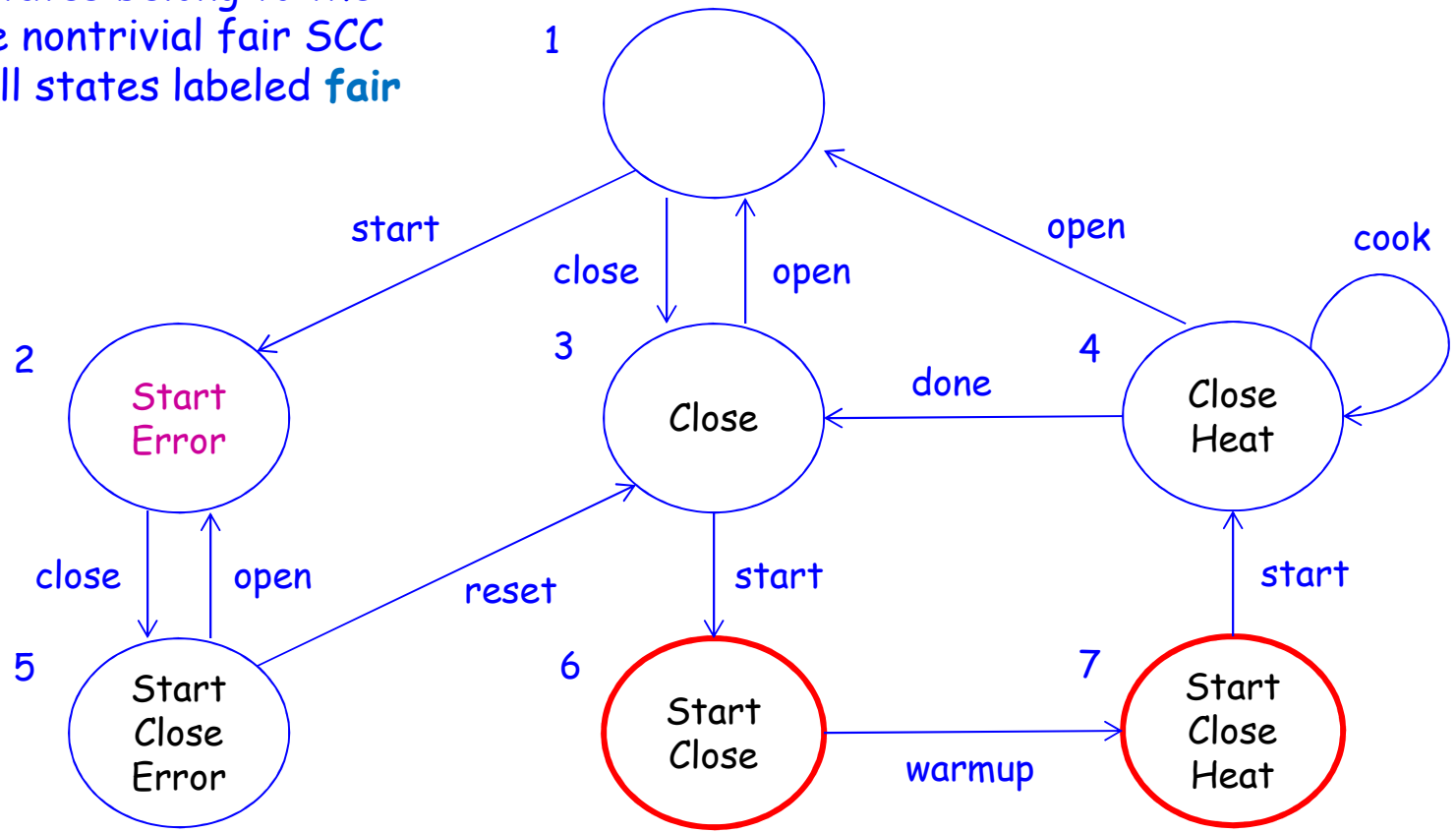
Now we check it with respect to a fair Kripke structure

$\neg E\,(true\ U\ (start \wedge EG\ \neg Heat))$

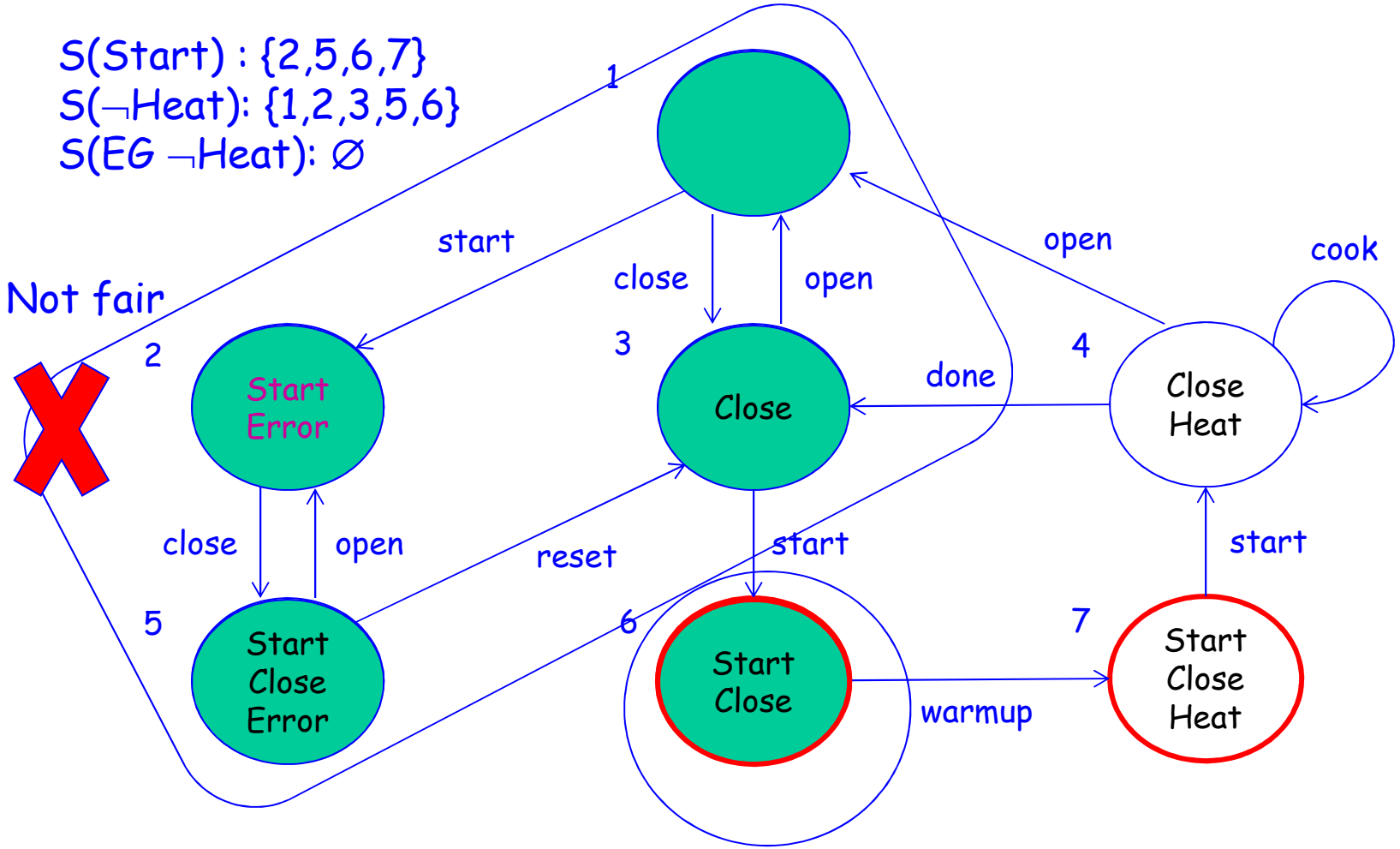


¬E (true U (start ∧ EG ¬Heat))

All states belong to the same nontrivial fair SCC
➔ All states labeled **fair**



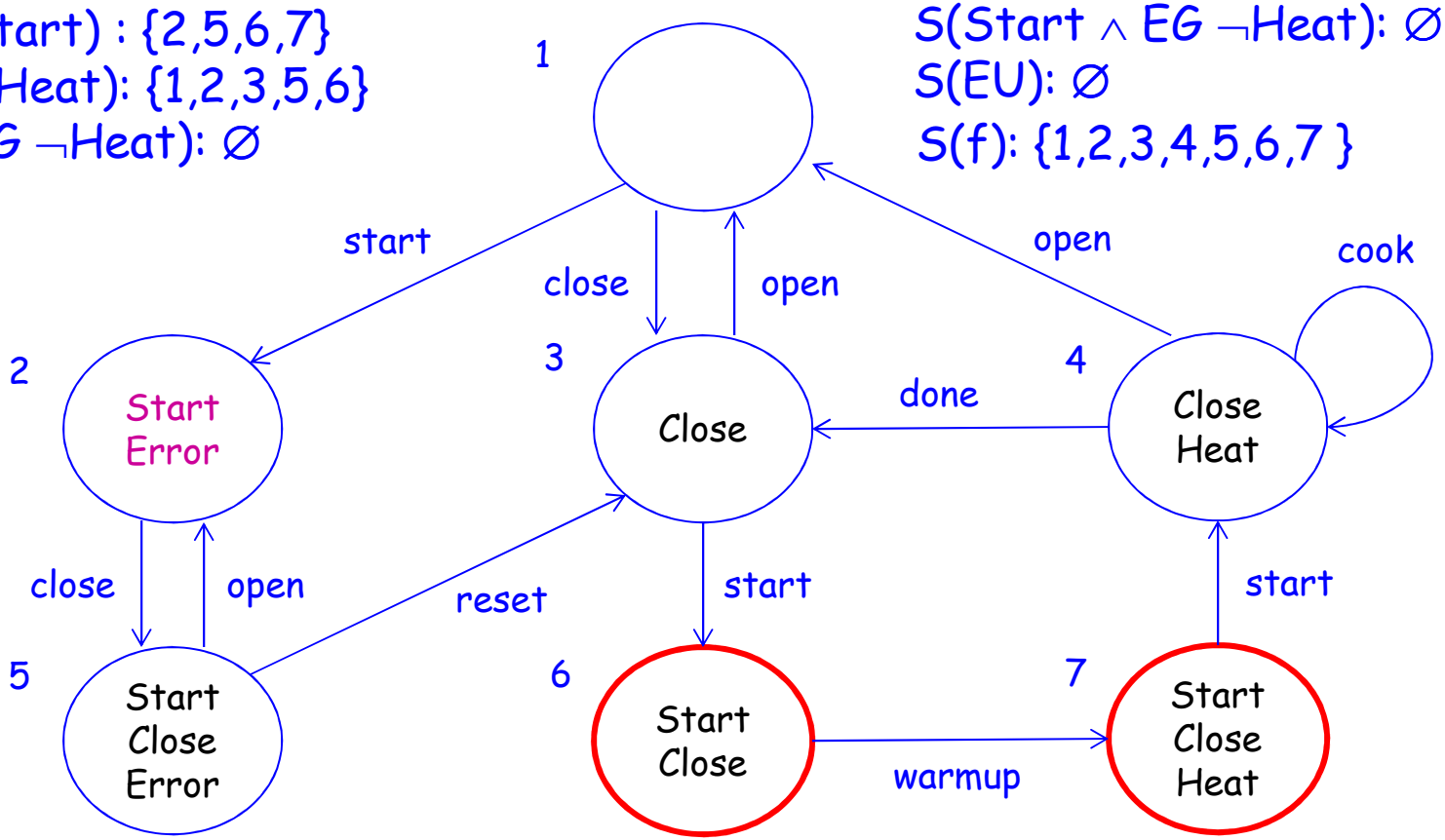
¬E (true U (start ∧ EG ¬Heat))



¬E (true U (start ∧ EG ¬Heat))

S(Start) : {2,5,6,7}
S(¬Heat): {1,2,3,5,6}
S(EG ¬Heat): ∅

S(Start ∧ EG ¬Heat): ∅
S(EU): ∅
S(f): {1,2,3,4,5,6,7 }



Model Checking

- Emerging as an industrial standard tool for verification of hardware designs: Intel, IBM, Cadence, Mellanox, …
- Recently applied successfully also for software verification: SLAM (Microsoft), Java PathFinder and SPIN (NASA), BLAST (EPFL), CBMC (Oxford),…

Clarke, Emerson, and Sifakis won the 2007 Turing award for their contribution to Model Checking

Main Limitation of Model Checking:

The state explosion problem:

Model checking is efficient in time but suffers from high space requirements:

The number of states in the system model grows exponentially with

- the number of variables
- the number of components in the system

Solutions to the state-explosion problem

Symbolic model checking:

The model is represented symbolically

- BDD-based model checking
- SAT-based Bounded Model Checking (BMC)
- SAT-based Unbounded Model Checking

Other solutions to the state-explosion problem

Small models replace the full, concrete model:

- Abstraction
- Compositional verification
- Partial order reduction
- Symmetry

Symbolic (BDD-based) Model Checking for CTL

BDD-based Symbolic Model Checking

A solution to the state explosion problem:
BDD-based model checking

- **Binary Decision Diagrams (BDDs)** are used to represent the **model** and **sets of states**.

- It can handle systems with **hundreds** of Boolean variables.

Binary Decision Diagrams (BDDs)

- Data structure for representing Boolean functions

- Boolean function:

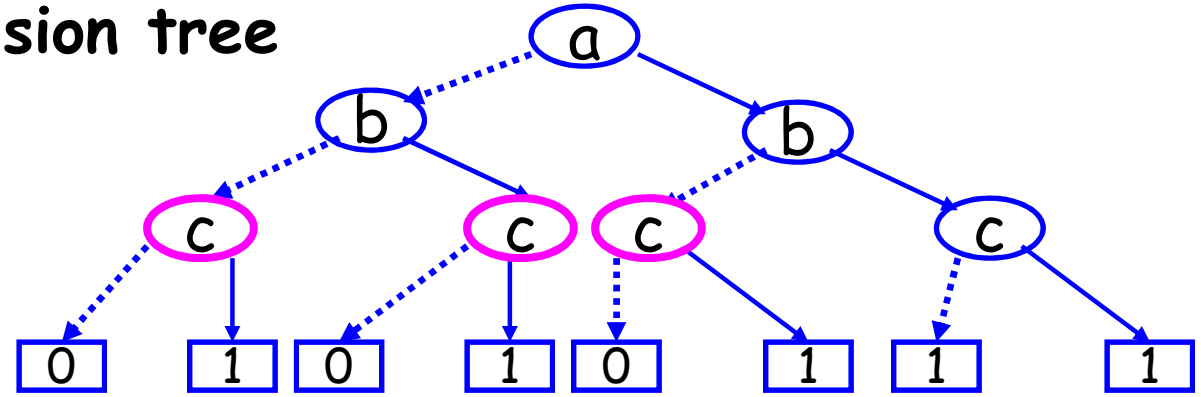
$$f: \{0,1\}^k \rightarrow \{0,1\}$$

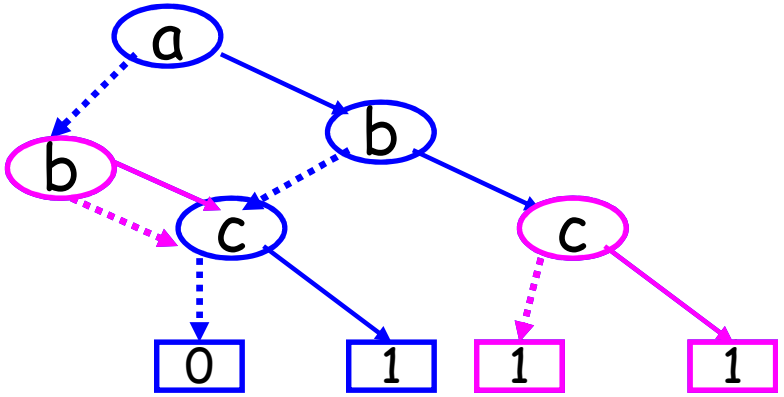
$$f(x_1, \ldots, x_k) = x_{k+1}$$

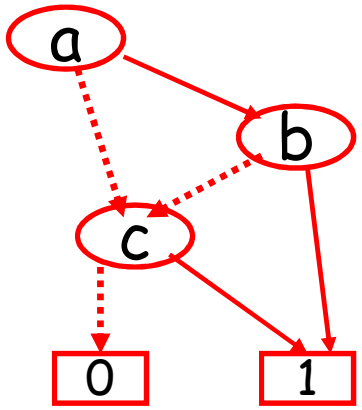
where $x_1, \ldots, x_k, x_{k+1} \in \{0,1\}$

BDD for f(a,b,c) = (a ∧ b) ∨ c

Decision tree







BDD

Binary Decision Diagrams (BDDs)

Advantages of BDDs:

- Often (but not always) **concise** in size
- **Canonical** representation
- Most **Boolean operations** can be performed on BDDs in **polynomial time** in the BDD size

BDDs in Model Checking

- Every set $A \subseteq U$ can be represented by its **characteristic function**

$$f_A(u) = \begin{cases} 1 & \text{if } u \in A \\ 0 & \text{if } u \notin A \end{cases}$$

- If the elements of U are encoded by sequences over $\{0,1\}^n$ then f_A is a **Boolean function** and can be represented by a BDD

- A Boolean function represents the set of all elements for which the function is 1

Representing a Model with BDDs

- Assume that **states** in model M are **encoded by $\{0,1\}^n$** and described by Boolean variables $v_1 \ldots v_n$
- S_f can be represented by a Boolean function (BDD) over $v_1 \ldots v_n$
- **R** (a set of pairs of states **(s,s')**) can be represented by a BDD over $v_1 \ldots v_n \; v_1' \ldots v_n'$

Example: Representing a Model with BDDs

$S = \{ s_1, s_2, s_3 \}$

$R = \{ (s_1,s_2), (s_2,s_2), (s_3,s_1) \}$

State encoding:

s_1: v_1v_2=00 s_2: v_1v_2=01 s_3: v_1v_2=11

For $A = \{s_1, s_2\}$ the Boolean formula representing A:

$f_A(v_1,v_2) = (\neg v_1 \wedge \neg v_2) \vee (\neg v_1 \wedge v_2) = \neg v_1$

$R = \{ (s_1,s_2), (s_2,s_2), (s_3,s_1) \}$

s_1: v_1v_2=00 s_2: v_1v_2=01 s_3: v_1v_2=11

$f_R(v_1, v_2, v'_1, v'_2) =$

$(\neg v_1 \wedge \neg v_2 \wedge \neg v'_1 \wedge v'_2) \vee$

$(\neg v_1 \wedge v_2 \wedge \neg v'_1 \wedge v'_2) \vee$

$(v_1 \wedge v_2 \wedge \neg v'_1 \wedge \neg v'_2)$

f_A and f_R can be represented by **BDDs**.