NPTEL Phase-II
Video course on

Design Verification and Test of Digital VLSI Designs

Dr. Santosh Biswas
Dr. Jatindra Kumar Deka
IIT Guwahati

Module V: Verification Techniques

Lecture I: Introduction to Model Checking

Verification Technique: Model checking

- **Process of Model Checking:**
  - Modeling
  - Specification
  - Verification Method

Model checking

- **Example: Mutual Exclusion**
  - When concurrent processes share a resource (e.g. file or database record), it may be necessary to ensure that they do not have access to it at the same time.
    - Identification of critical section

Model checking

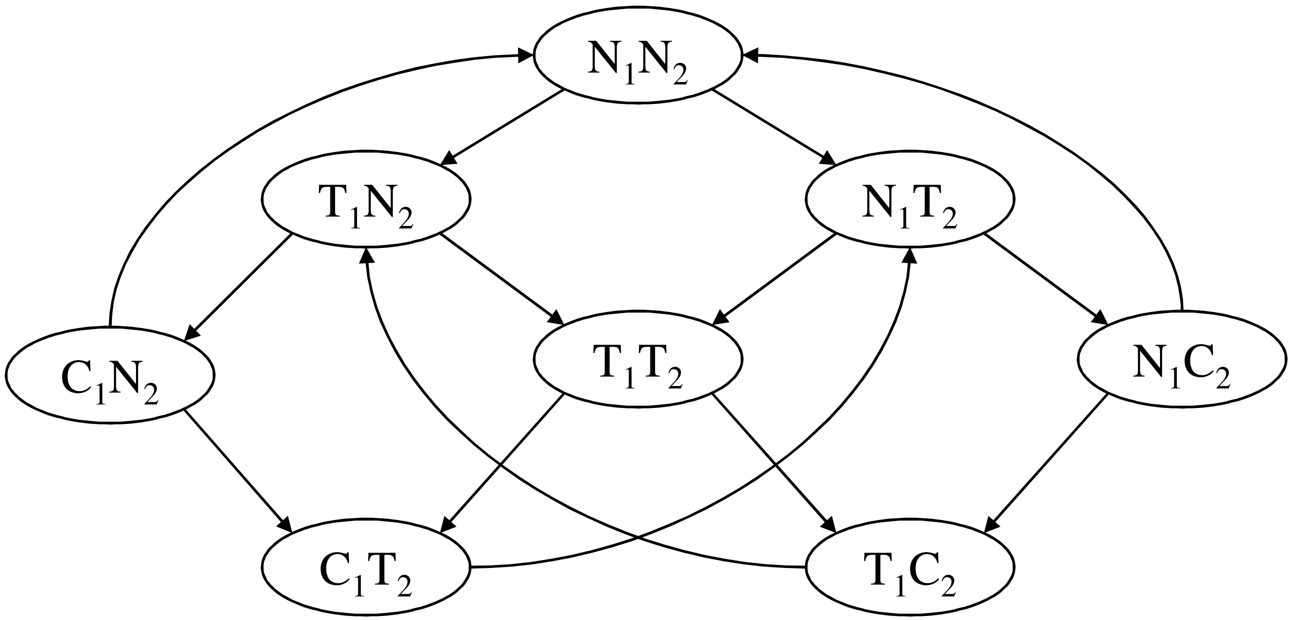
- **Example: Mutual Exclusion**
  - When concurrent processes share a resource (e.g. file or database record), it may be necessary to ensure that they do not have access to it at the same time.
    - Identification of critical section
  - How to model the system
  - What are the specifications

Mutual Exclusion Example

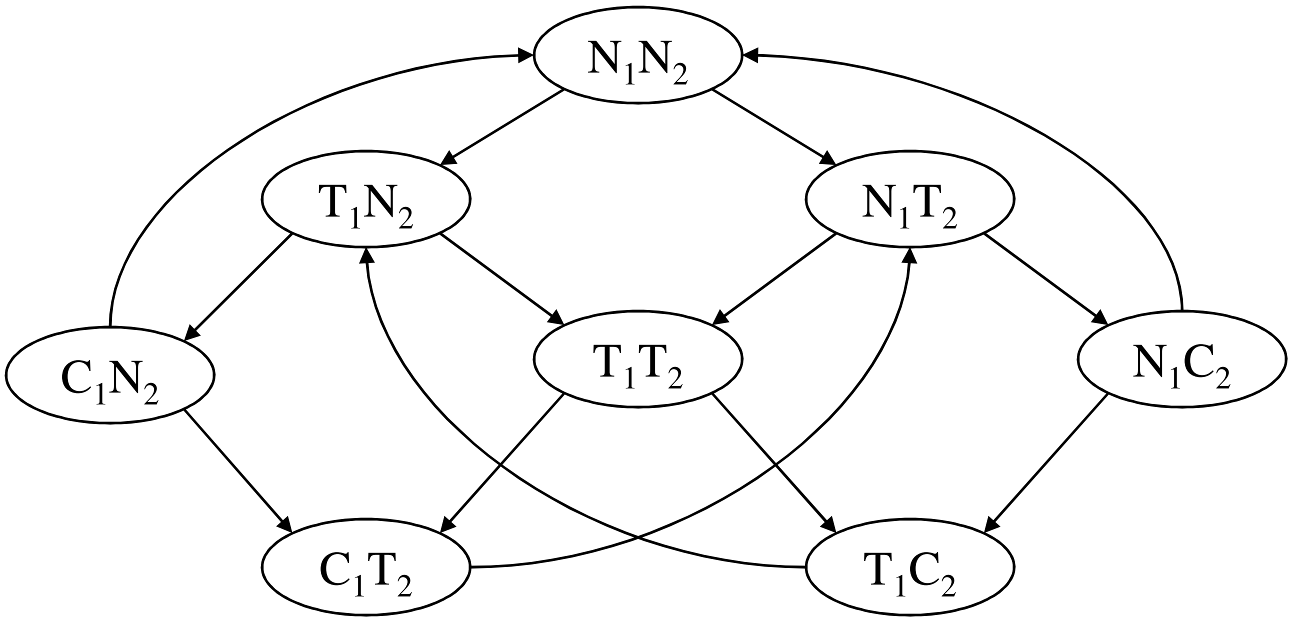
- Two process mutual exclusion for shared resources
- Each process has three states
  - Non-critical (N)
  - Trying (T)
  - Critical (C)
- Initially both processes are in the Non-critical state --- $N_1 N_2$

$$
\begin{array}{lll}
N_1 \rightarrow T_1 & & N_2 \rightarrow T_2 \\
T_1 \rightarrow C_1 & \| & T_2 \rightarrow C_2 \\
C_1 \rightarrow N_1 & & C_2 \rightarrow N_2
\end{array}
$$

Mutual Exclusion Example

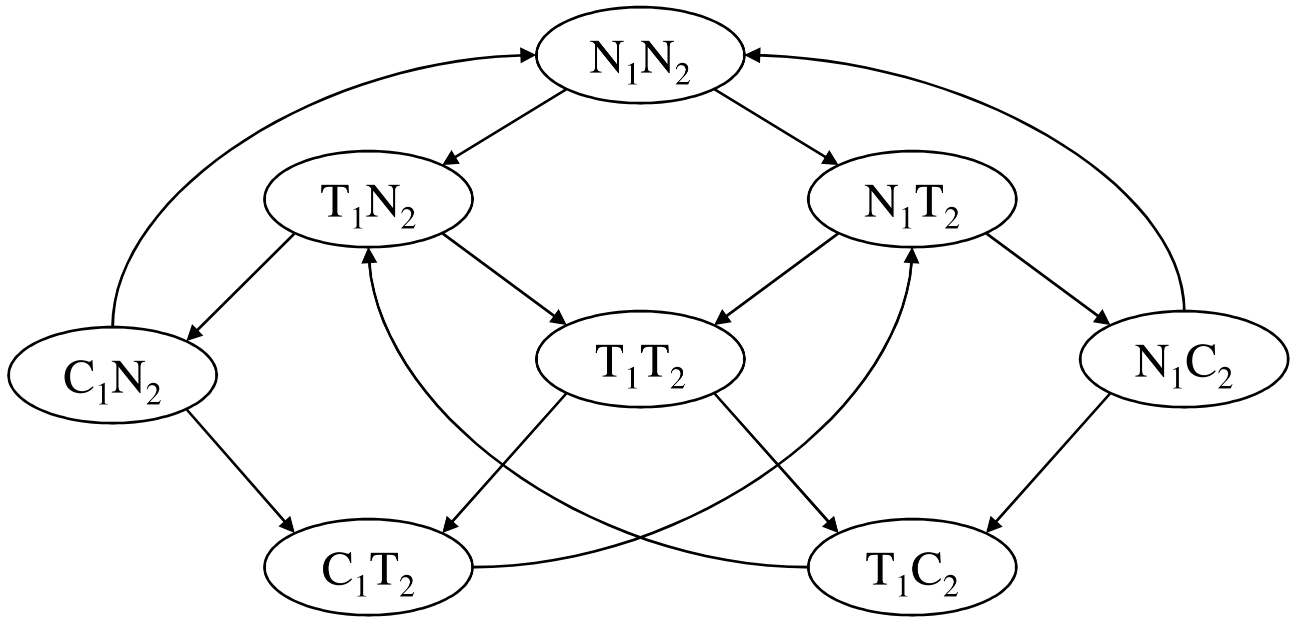


Mutual Exclusion Example



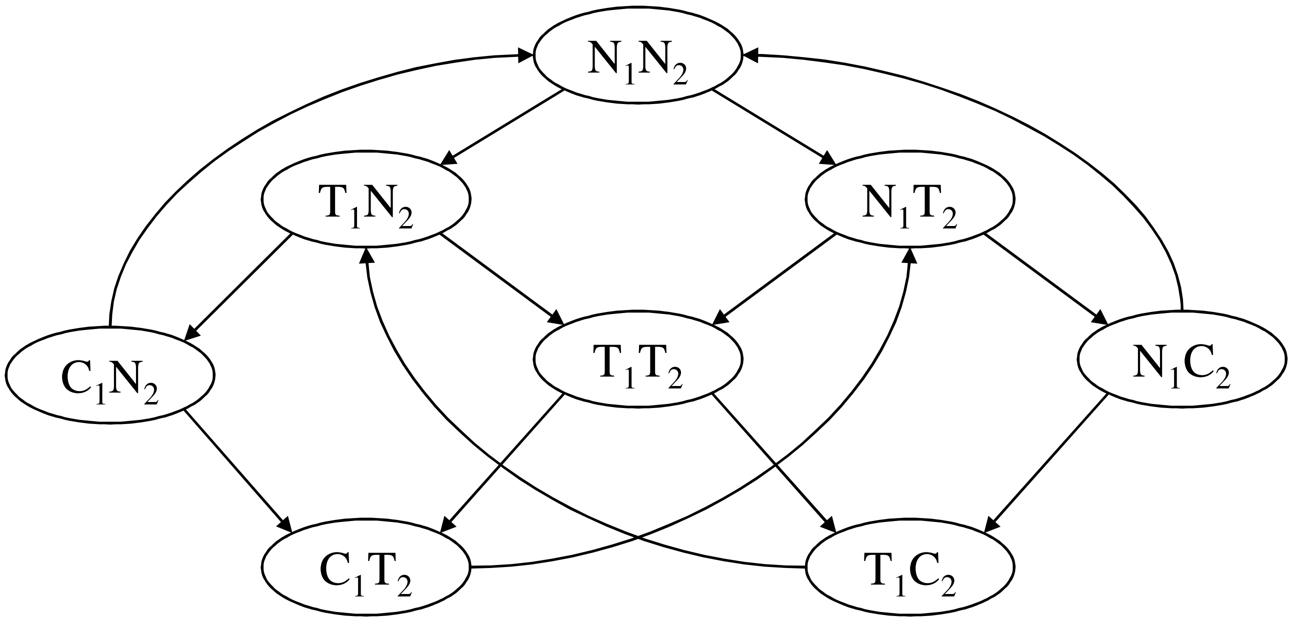
Reachable states

Mutual Exclusion Example



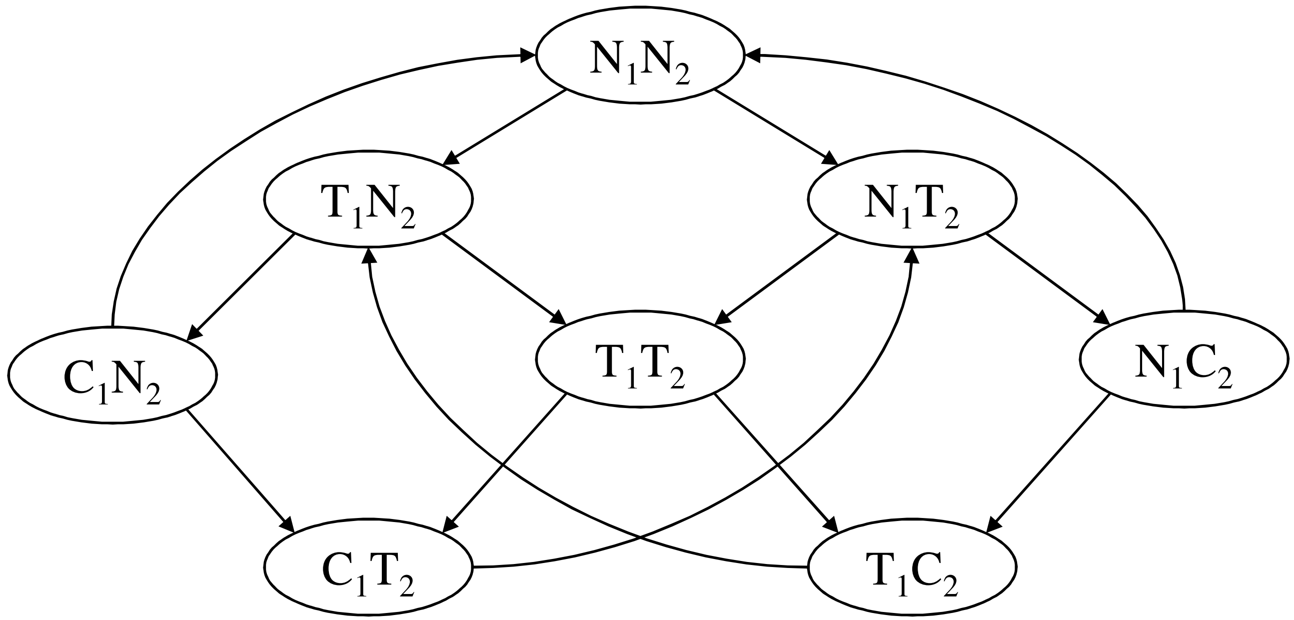
Total Number of states

Mutual Exclusion Example



Kripke structure

Mutual Exclusion Example



$AG\ EF\ (N_1 \wedge N_2)$

No matter where you are there is always a way to get to the initial state

Some Properties

Safety: only one process to be in its critical section at any time.

Liveness: Whenever any process wants to enter its critical section, it will eventually be permitted to do so.

Some Properties

Non-blocking: A process can always request to enter its critical section.

No strict sequencing: Processes need not enter their critical section in strict sequence

Some CTL Properties

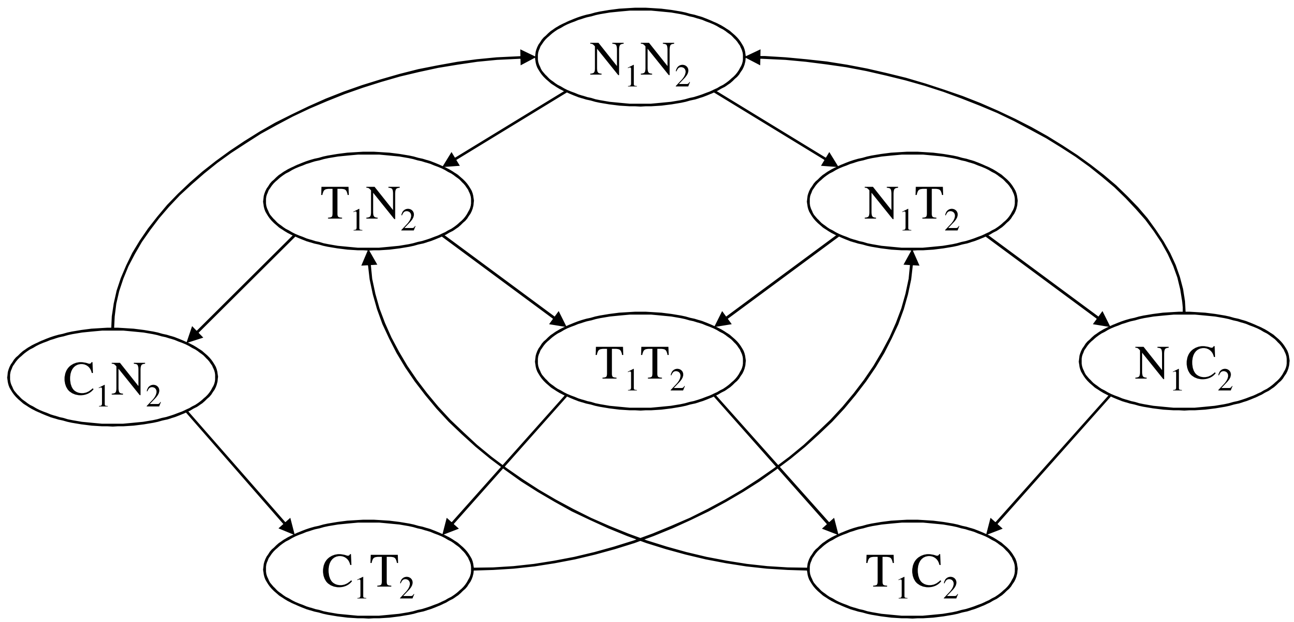
Safety: only one process to be in its critical section at any time.

$$AG\, \neg(c_1 \wedge c_2)$$

Liveness: Whenever any process wants to enter its critical section, it will eventually be permitted to do so.

$$AG(t_1 \rightarrow AFc_1)$$

Mutual Exclusion Example



$AG\, \neg(c_1 \wedge c_2)$

$AG(t_1 \rightarrow AFc_1)$

Some CTL Properties

Non-blocking: A process can always request to enter its critical section.

No strict sequencing: Processes need not enter their critical section in strict sequence

Some CTL Properties

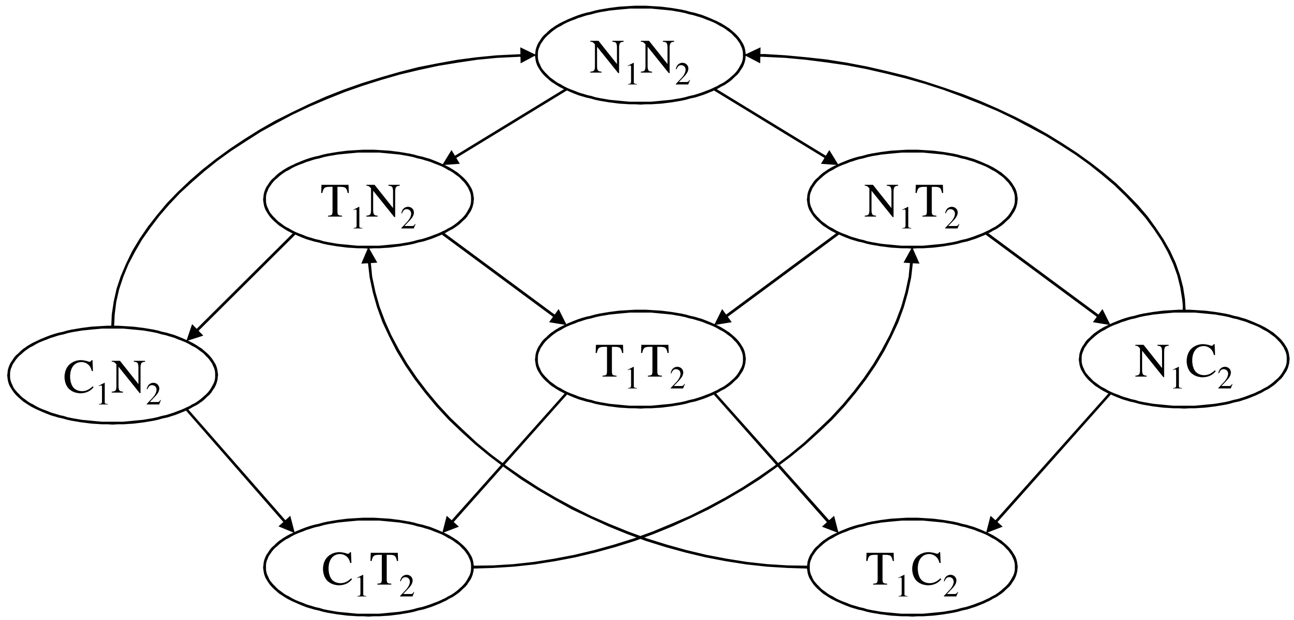
Non-blocking: A process can always request to enter its critical section.

$$AG(n_1 \rightarrow EXt_1)$$

No strict sequencing: Processes need not enter their critical section in strict sequence

$$EF(c_1 \wedge E[\, c_1 \, U \, (\neg\, c_1 \wedge E[\neg\, c_2 \, U \, c_1])])$$

Mutual Exclusion Example



$AG(n_1 \rightarrow EXt_1)$

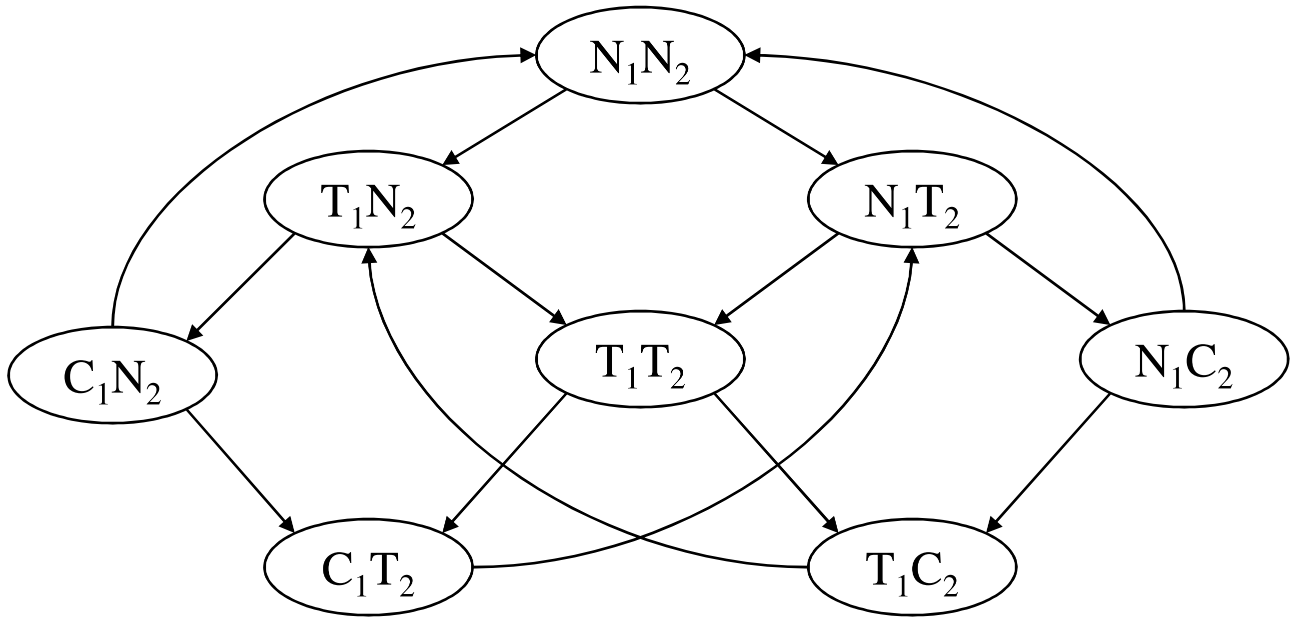
$EF(c_1 \wedge E[\, c_1\, U\ (\neg\, c_1 \wedge\ E[\neg\, c_2\, U\, c_1])])$

- Observation
  - $AG \neg(c_1 \wedge c_2)$
  - $AG(t_1 \rightarrow AFc_1)$
  - $AG(n_1 \rightarrow EXt_1)$
  - $EF(c_1 \wedge E[\, c_1 \, U \, (\neg c_1 \wedge E[\neg c_2 \, U \, c_1])])$

- Observation
  - $AG \neg(c_1 \wedge c_2)$
  - $AG(t_1 \rightarrow AFc_1)$
  - $AG(n_1 \rightarrow EXt_1)$
  - $EF(c_1 \wedge E[c_1 \; U \; (\neg c_1 \wedge E[\neg c_2 \; U \; c_1])])$

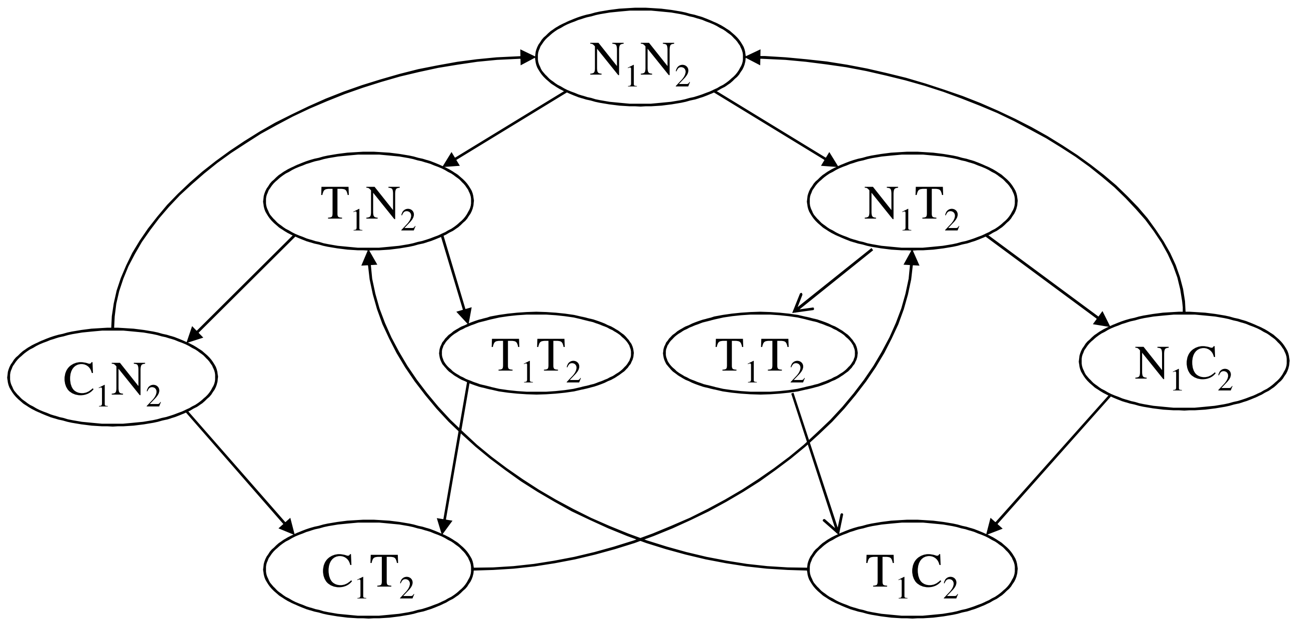
One property is not true: $AG(t_1 \rightarrow AFc_1)$

Mutual Exclusion Example

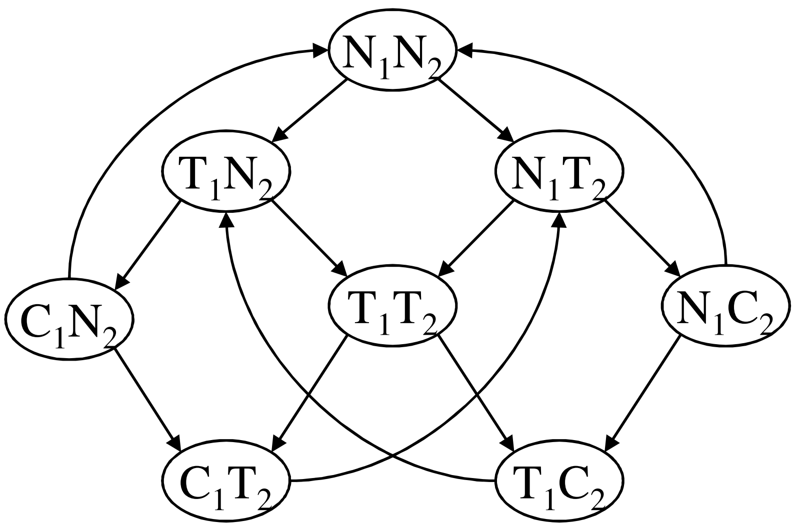


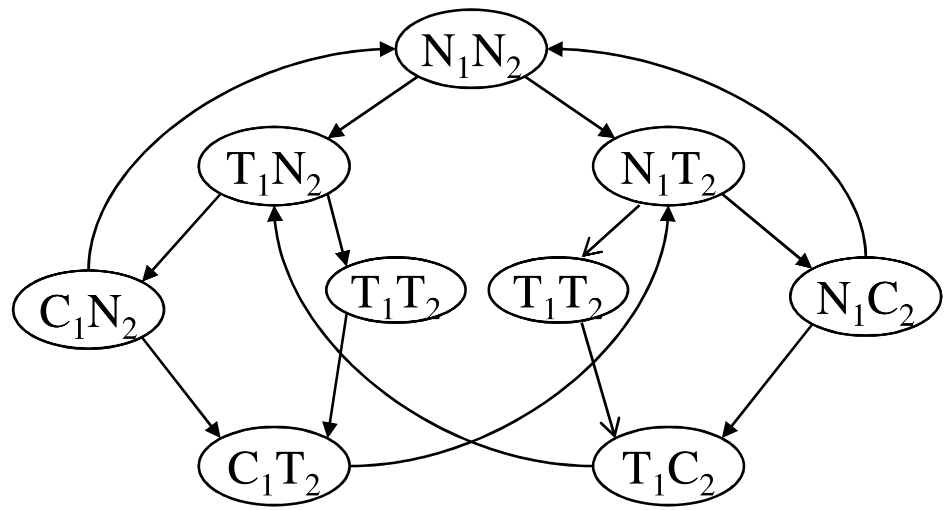
$AG(t_1 \rightarrow AFc_1)$

Mutual Exclusion Example



$AG(t_1 \rightarrow AFc_1)$


N1N2
T1N2
N1T2
C1N2
T1T2
N1C2
C1T2
T1C2


N1N2
T1N2
N1T2
C1N2
T1T2
T1T2
N1C2
C1T2
T1C2

Model checking algorithm

Model Checking Algorithm

Given the model '*M*', the CTL formula Φ and a state s_0 of S as input

Model checking algorithm generates answer 'yes' (M, $s_0 \models \Phi$ holds), or 'no' (M , $s_0 \models \Phi$ does not hold).

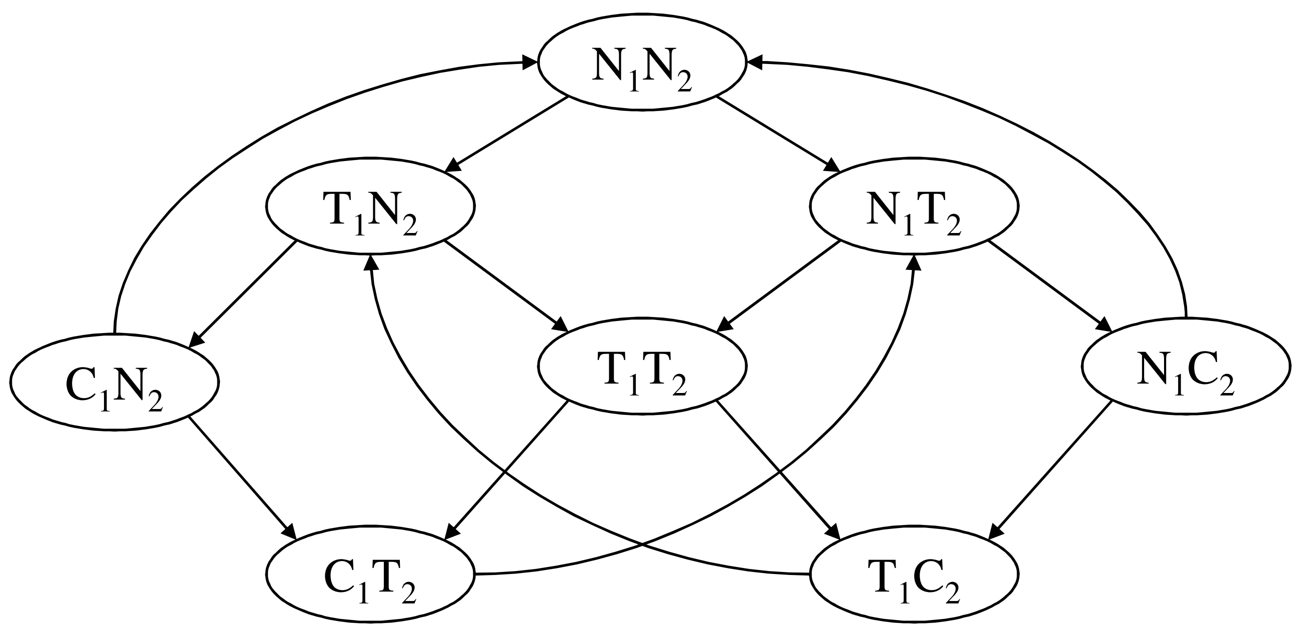
Model Checking Algorithm

Given the model 'M' and a CTL formula Φ as input.

Model checking algorithm provides all the states of model M which satisfy Φ

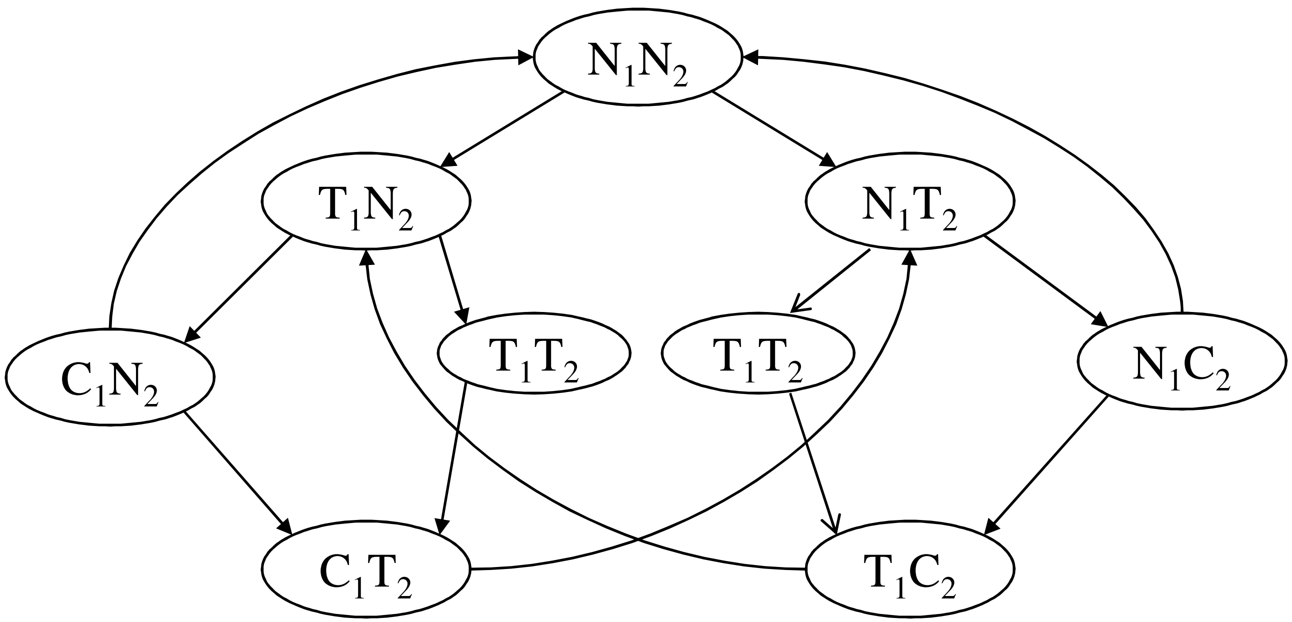
Questions

- Checking for liveness property.
  - $AG(t_1 \rightarrow AFc_1)$



Questions

- Second modeling of mutual exclusion is also a over simplified model.



NPTEL Phase-II
Video course on

Design Verification and Test of Digital VLSI Designs

Dr. Santosh Biswas
Dr. Jatindra Kumar Deka
IIT Guwahati

Module V: Verification Techniques

Lecture II: Model Checking Algorithms

Verification Technique: Model checking

- **Process of Model Checking:**
  - Modeling
  - Specification
  - Verification Method: Model Checking Algorithm

Model Checking Algorithm

Given the model '*M*' and a CTL formula Φ as input.

Model checking algorithm provides all the states of model M which satisfy Φ

Model Checking Algorithm

Given the model 'M' and a CTL formula Φ as input.

Model checking algorithm provides all the states of model M which satisfy Φ

Labeling Algorithm

Labeling Algorithms

CTL model checking algorithm basically works by iteratively determining (i.e., labeling) states which satisfy a given CTL formula.

The basic input/output of labeling algorithm are as follows:

INPUT : A **CTL model *'M'* = *(S,* $\rightarrow$*, L)*** where S is the set of states, $\rightarrow$ is the transition relation and L is the labeling function and a **CTL formula Φ**.

OUTPUT : The set of states of M which satisfy Φ.

Labeling Algorithm

- The adequate set of temporal operators for CTL is AF, EU and EX.
- First, we write the given formula Φ in terms of the connectives AF, EU and EX along with other logical connectives and truth value T.
- Suppose ψ is a subformula of Φ and states satisfying all the immediate subformulas of ψ have already been labeled.

- Formula and subformula:

CTL Model Checking

Function SAT(Φ)

/* determines the set of states satisfying Ø */

Begin

Case

Ø is $\top$: retune S

Ø is $\perp$: return Ø

Ø is atomic: return {s ∈S | Ø ∈ L(S)}

Ø is ¬ $Ø_1$: return S − SAT($Ø_1$)

Ø is $Ø_1$ ∧ $Ø_2$: return SAT ($Ø_1$)∩ SAT ($Ø_2$)

Ø is $Ø_1$ ∨ $Ø_2$: return SAT ($Ø_1$)∪ SAT ($Ø_2$)

Ø is $Ø_1$→ $Ø_2$: return SAT (¬ $Ø_1$ ∨ $Ø_2$)

Ø is AX $Ø_1$: return SAT (¬EX ¬Ø1)

Ø is EX $Ø_1$: return SAT_{EX} ($Ø_1$)

Ø is A($Ø_1$ U $Ø_2$) : return SAT(¬(E [Ø2U(¬ $Ø_1$∧¬ $Ø_2$)]∨EG¬ $Ø_2$))

Ø is E($Ø_1$ U $Ø_2$):return SAT_{EU} ($Ø_1$, $Ø_2$)

Ø is EF $Ø_1$: return SAT (E(T U $Ø_1$))

Ø is EG($Ø_1$):return SAT (E(T U $Ø_1$)

Ø is AF $Ø_1$: return SAT_{AF} ($Ø_1$)

Ø is AG $Ø_1$: return (¬EF ¬ $Ø_1$)

end case

end function

CTL Model Checking

- Atomic proposition
  - p: label state s with p if $p \in L(s)$

- Logical connectives
  - $p \wedge q$: label s with $p \wedge q$ if s is already labeled with p and q

CTL Model Checking

Temporal Operator:

EX p

Label any state with EX p if one of its successor is labeled with p

CTL Model Checking

Function $SAT_{EX}(p)$

/* determines the set of states satisfying EXp */

local var X,Y

begin

  X := SAT(p)

  Y := $\{s_0 \in S \mid s_0 \rightarrow s_1 \text{ for some } s_1 \in X\}$

  return Y

end

CTL Model Checking

Temporal Operator:

AF p

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

CTL Model Checking

Temporal Operator:

AF p

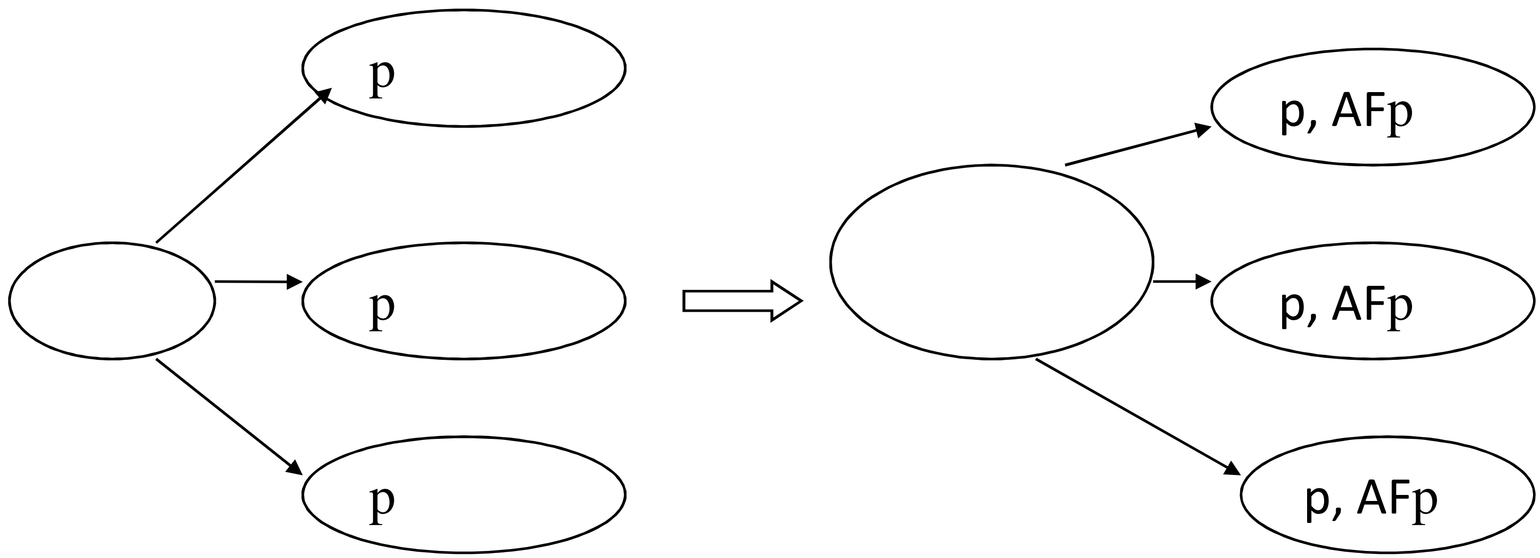
- If any state s is labeled with p, label it with AF p
- Repeat: label any state with AF p if all successor states are labeled with AF p until there is no change.

$$AF\ p \equiv p \vee AX\ AF\ p$$

CTL Model Checking

Function $SAT_{AF}(p)$

/* determines the set of states satisfying AFp */

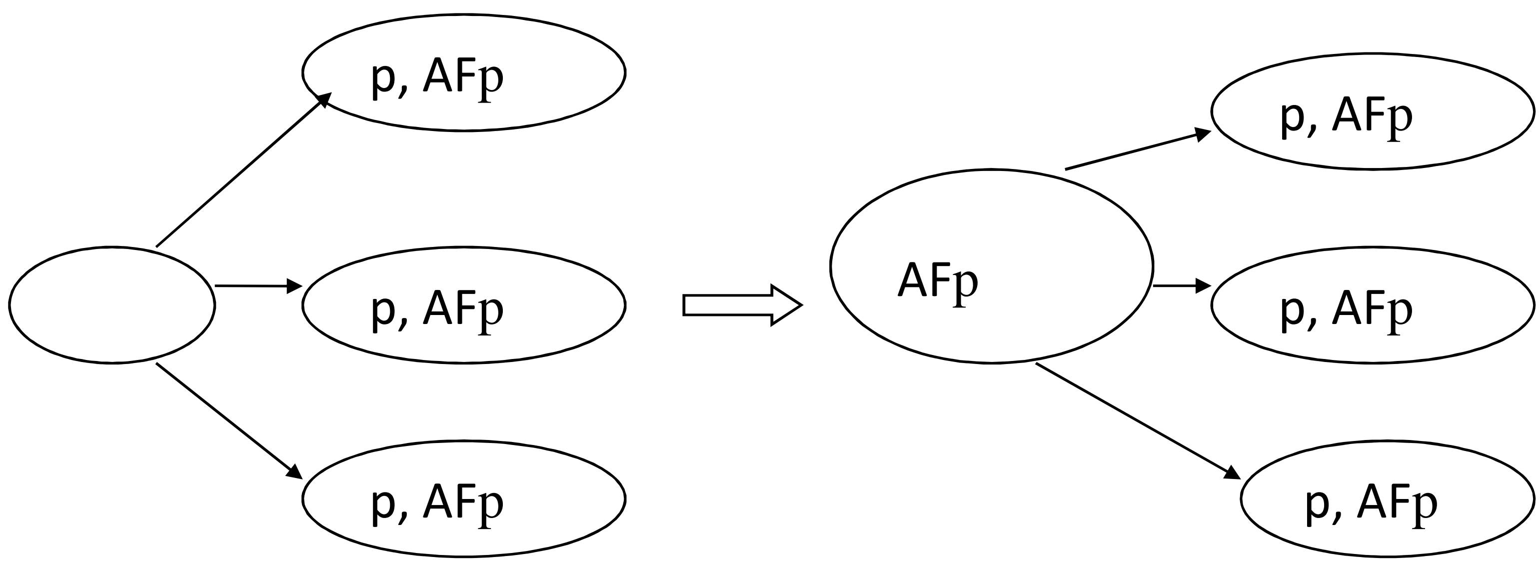


…Until no change

CTL Model Checking

Function $SAT_{AF}(p)$

/* determines the set of states satisfying AFp */



…Until no change

CTL Model Checking

Function $SAT_{AF}(p)$
/* determines the set of states satisfying AFp */
local var X, Y
begin
    X :=S, Y := SAT(p),
    repeat until X = Y
    begin
        X:=Y
        $Y := Y \cup \{s \mid \text{for all } s' \text{ with } s \rightarrow s' \text{ we have } s' \in Y\}$
    end
    return Y
end

CTL Model Checking

Temporal Operator:

E(p U q)

- If any state s is labeled with q, label it with E(p U q)
- Repeat: label any state with E(p U q) if it is labeled with p and at least one of its successor is labeled with E(p U q) until there is no change.

CTL Model Checking

Temporal Operator:

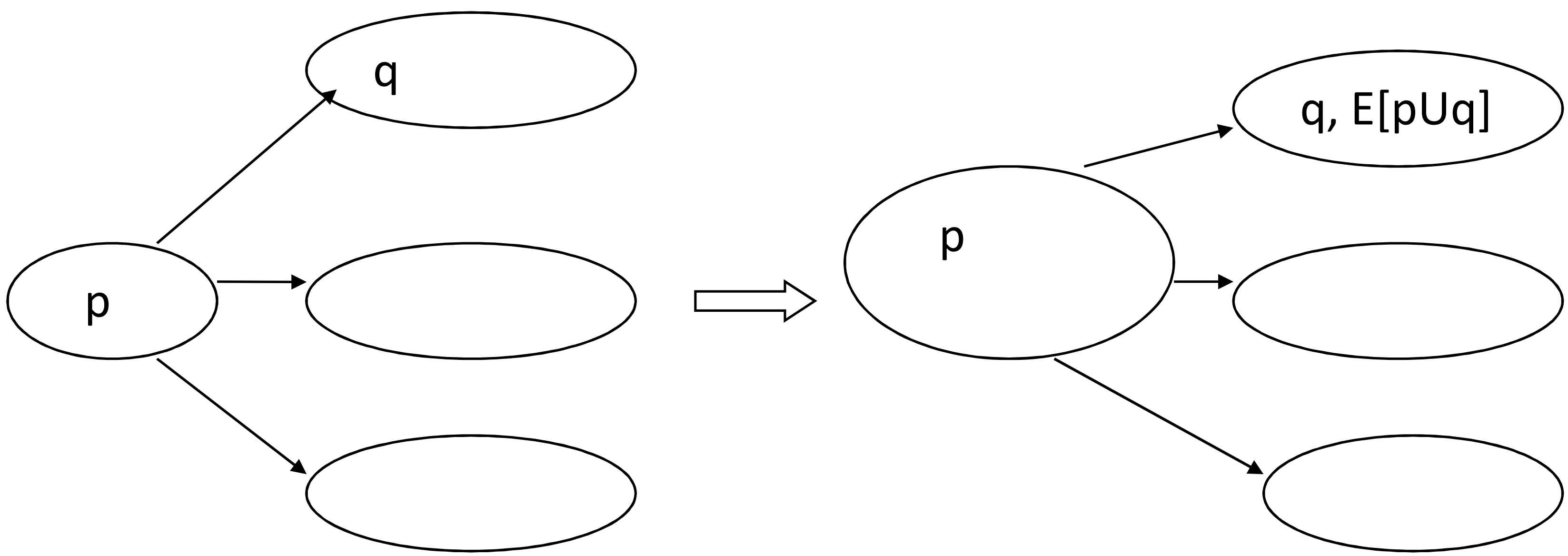
E(p U q)

- If any state s is labeled with q, label it with E(p U q)
- Repeat: label any state with E(p U q) if it is labeled with p and at least one of its successor is labeled with E(p U q) until there is no change.

$$E[p \; U \; q] \equiv q \vee (p \wedge EX \; E[p \; U \; q])$$

CTL Model Checking

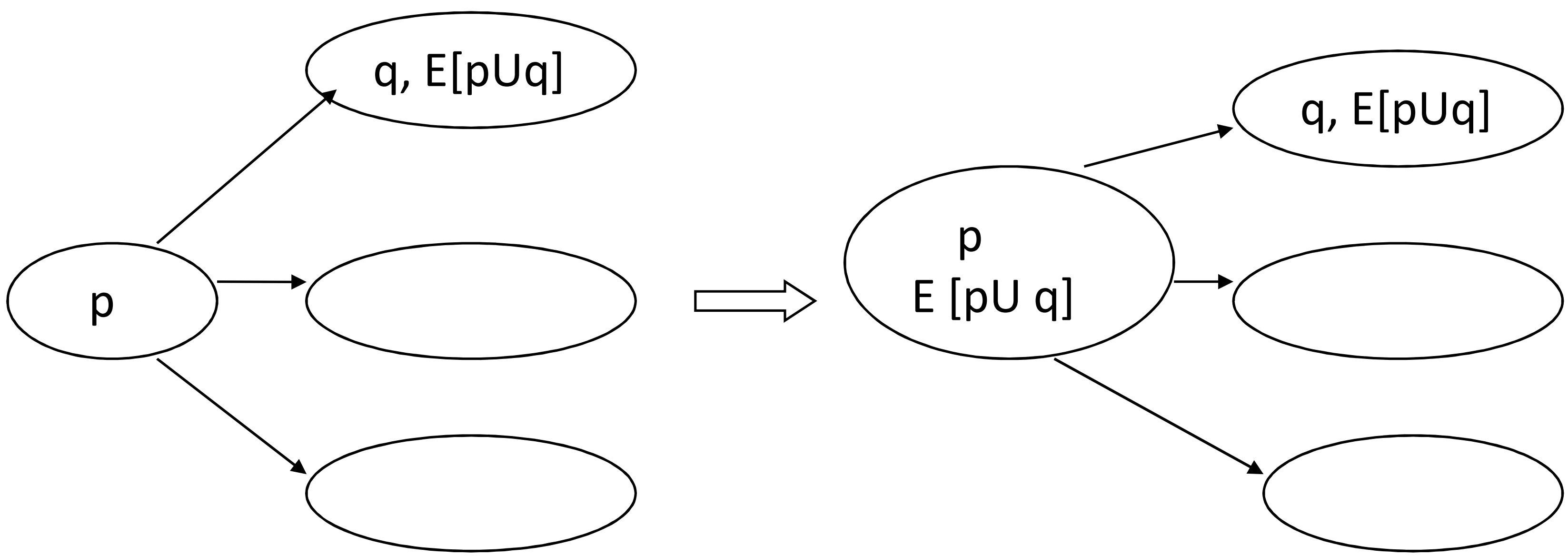
Function $SAT_{EU}(p,q)$
/* determines the set of states satisfying E(p U q) */



…Until no change

CTL Model Checking

Function $SAT_{EU}(p,q)$
/* determines the set of states satisfying E(p U q) */



…Until no change

CTL Model Checking

Function $SAT_{EU}(p,q)$

```
/* determines the set of states satisfying E(p U q) */
local var W,X,Y
begin
    W := SAT(p), X := S, Y := SAT(q)
    repeat until X = Y
    begin
        X := Y
        Y := Y ∪ (W ∩ {s | exists s' such that s → s' and s' ∈ Y}
    end
    return Y
end
```

CTL Model Checking

- After performing the labeling for all the subformulas of Φ (including Φ itself), we output the states which are labeled Φ .

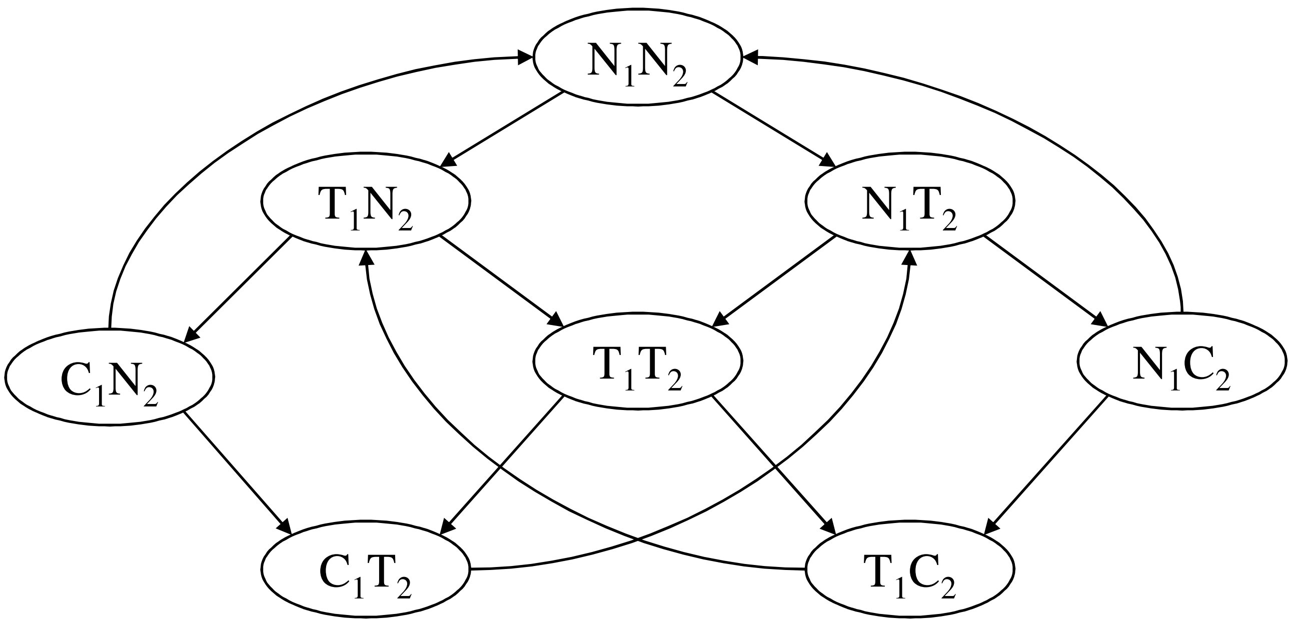
CTL Model Checking

- After performing the labeling for all the subformulas of Φ (including Φ itself), we output the states which are labeled Φ .

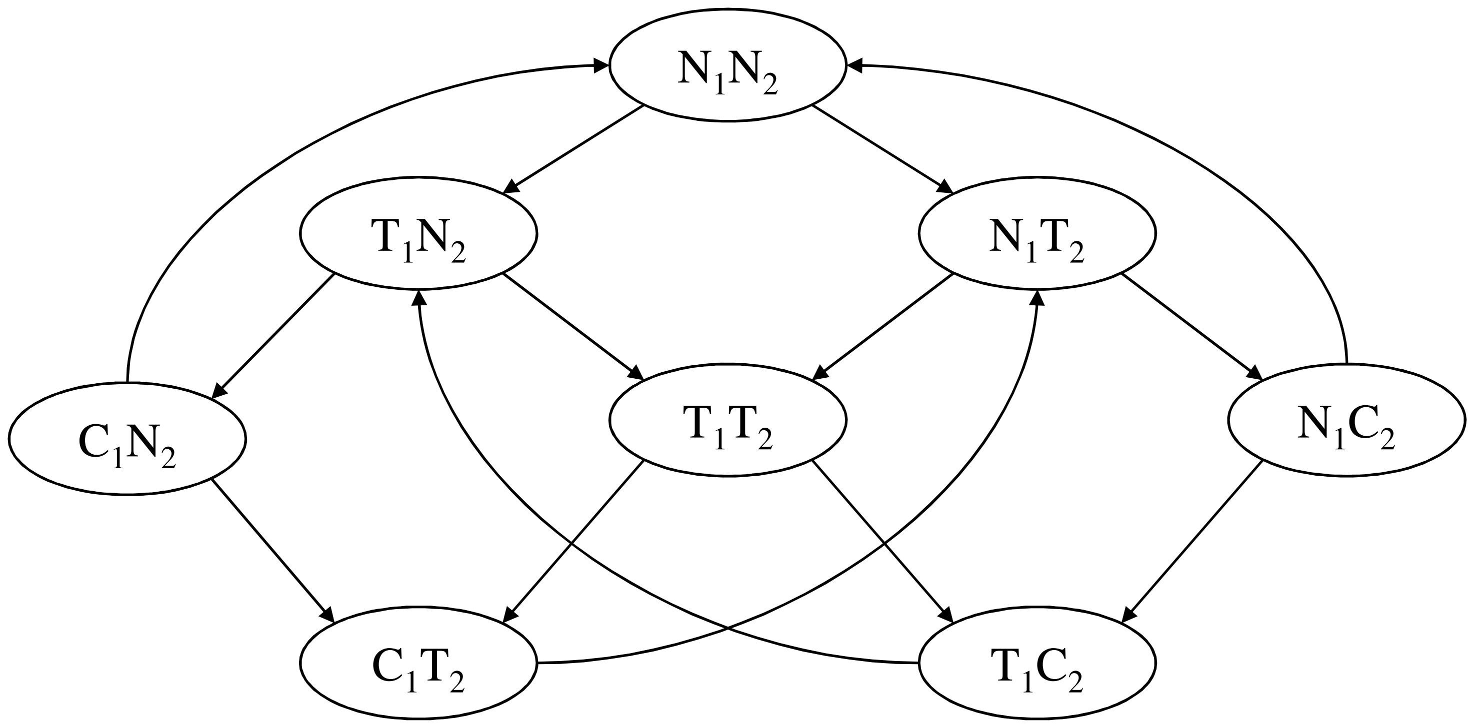
- The complexity of the algorithm is
  - O(|f|.|V|.(V+E))
    - f : number of connectives in the formula
    - V: number of states
    - E: number of transitions

Questions

- Apply the model checking algorithm to label the states with the formula AG $\neg(c_1 \wedge c_2)$ (safety property)



Questions



$AG \neg(c_1 \wedge c_2)$

Questions

- We have the methods for EX, AF and EU

$$AG \neg(c_1 \wedge c_2) \equiv \neg EF (c_1 \wedge c_2)$$
$$\equiv \neg E(T\ U\ (c_1 \wedge c_2))$$

$AGp \equiv \neg EF \neg p$

$EFp \equiv E(true\ U\ p)$

Questions

- We have the methods for EX, AF and EU

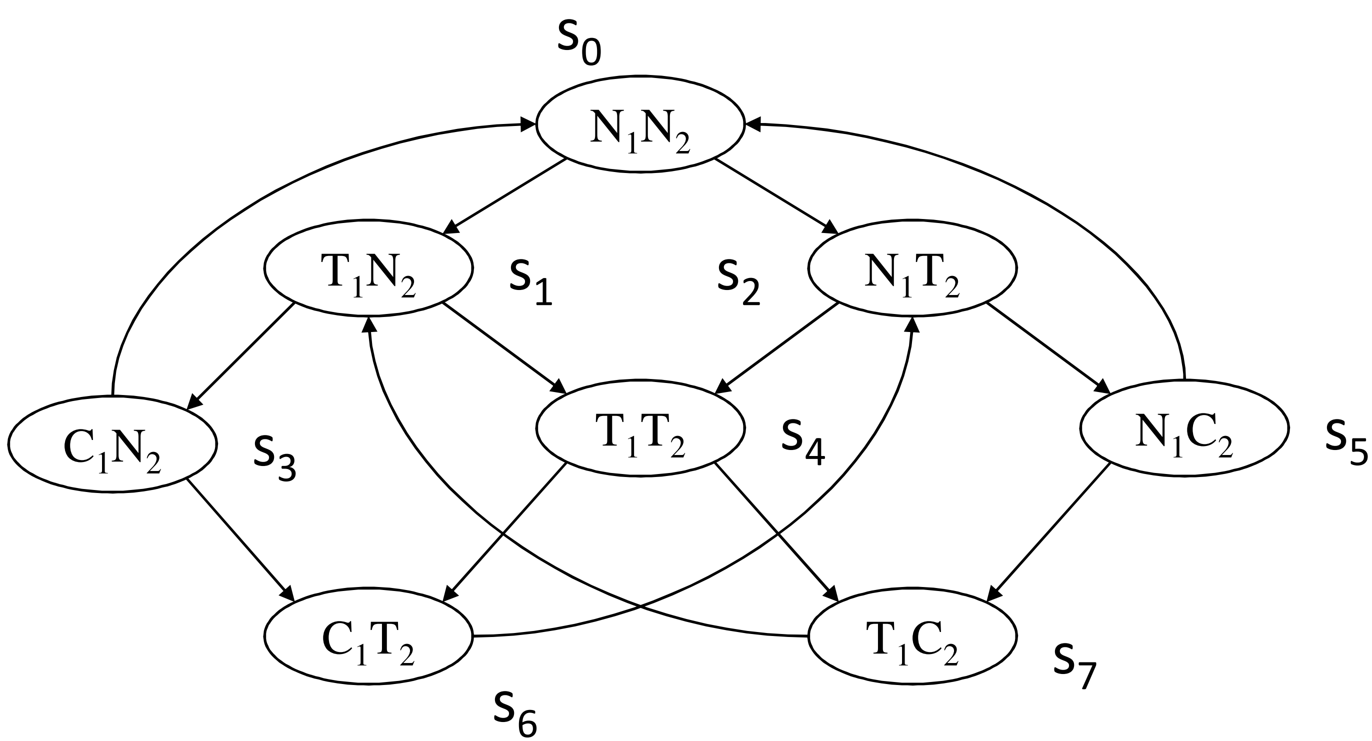
$$AG \neg(c_1 \wedge c_2) \equiv \neg EF\,(c_1 \wedge c_2)$$
$$\equiv \neg E(T\,U\,(c_1 \wedge c_2))$$

Subformulas:

$c_1, c_2, c_1 \wedge c_2, E(T\,U\,(c_1 \wedge c_2))$

$\neg E(T\,U\,(c_1 \wedge c_2))$

Questions



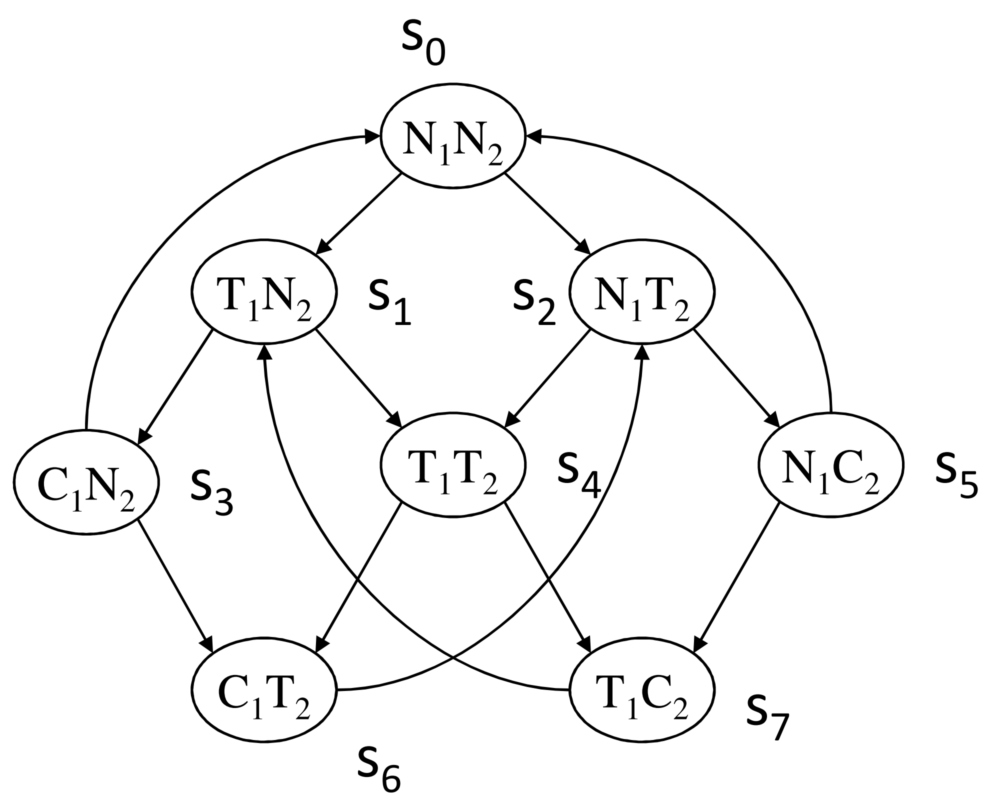
c_1: $\{s_3, s_6\}$ c_2: $\{s_5, s_7\}$

Questions

c_1: $\{s_3, s_6\}$

c_2: $\{s_5, s_7\}$

$c_1 \wedge c_2$: $\{\}$



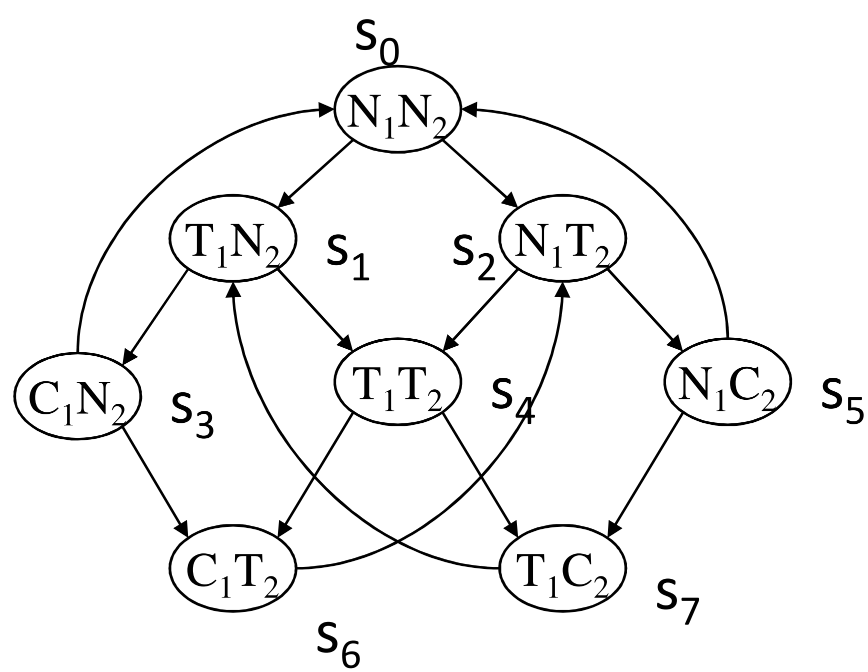
Questions

c_1: $\{s_3, s_6\}$

c_2: $\{s_5, s_7\}$

$c_1 \wedge c_2$: $\{\}$

$E(T \; U \; (c_1 \wedge c_2))$: $\{\}$



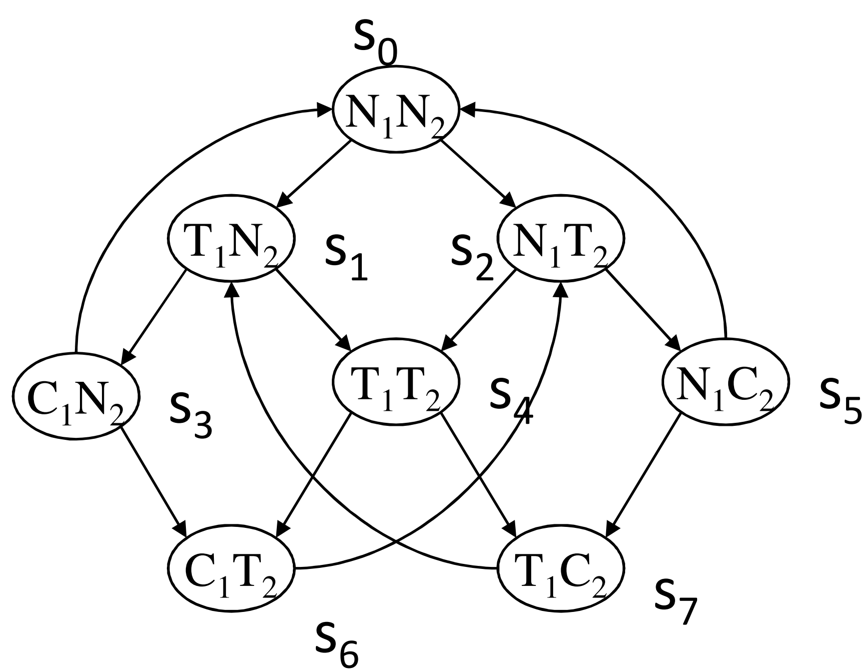
Questions

c_1: $\{s_3, s_6\}$

c_2: $\{s_5, s_7\}$

$c_1 \wedge c_2$: $\{\}$

$E(T\ U\ (c_1 \wedge c_2))$: $\{\}$

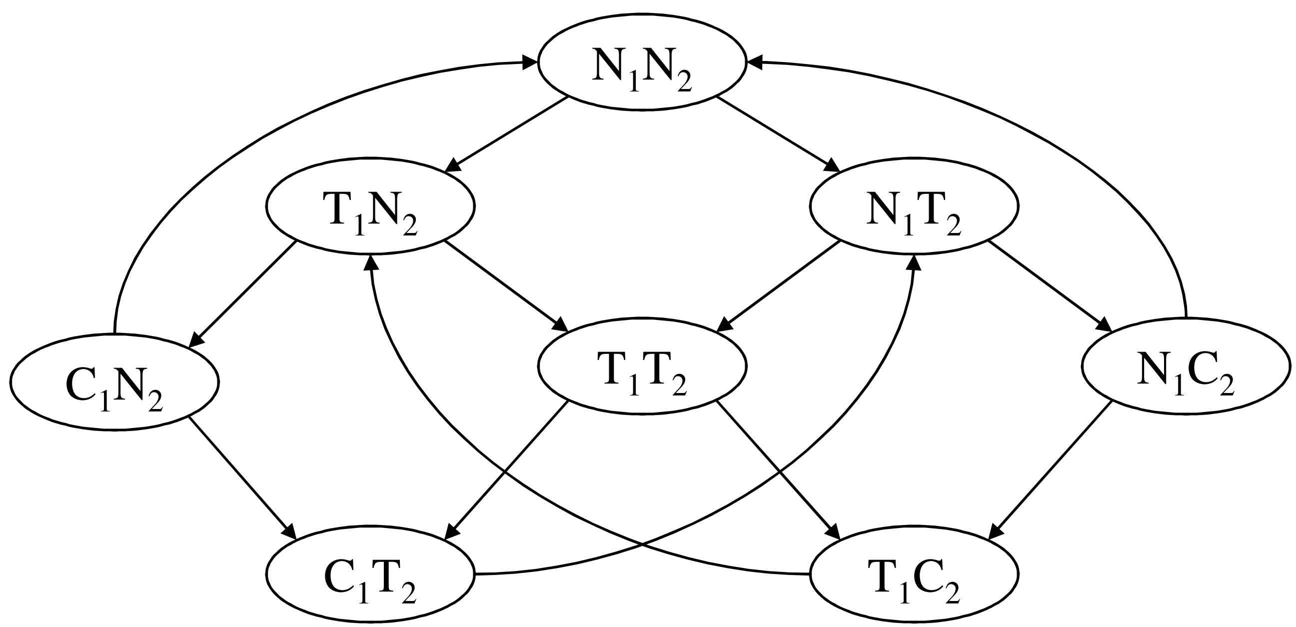


$\neg\ E(T\ U\ (c_1 \wedge c_2))$: $\{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$

$AG\ \neg(c_1 \wedge c_2) \equiv \neg\ E(T\ U\ (c_1 \wedge c_2))$

Questions

- Apply the model checking algorithm to label the states with the formula $AG(t_1 \rightarrow AFc_1)$



Questions

- We have the methods for EX, AF and EU

$$AG(t_1 \rightarrow AFc_1) \equiv \neg EF(\neg (t_1 \rightarrow AFc_1))$$
$$\equiv \neg E(T\ U\ (\neg (t_1 \rightarrow AFc_1)))$$

$$AGp \equiv \neg EF \neg p$$
$$EFp \equiv E(\text{true}\ U\ p)$$

Questions

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\, (\neg\, (t_1 \rightarrow AFc_1))$$

$$= \neg\, E(T\; U\; (\neg\, (t_1 \rightarrow AFc_1)))$$

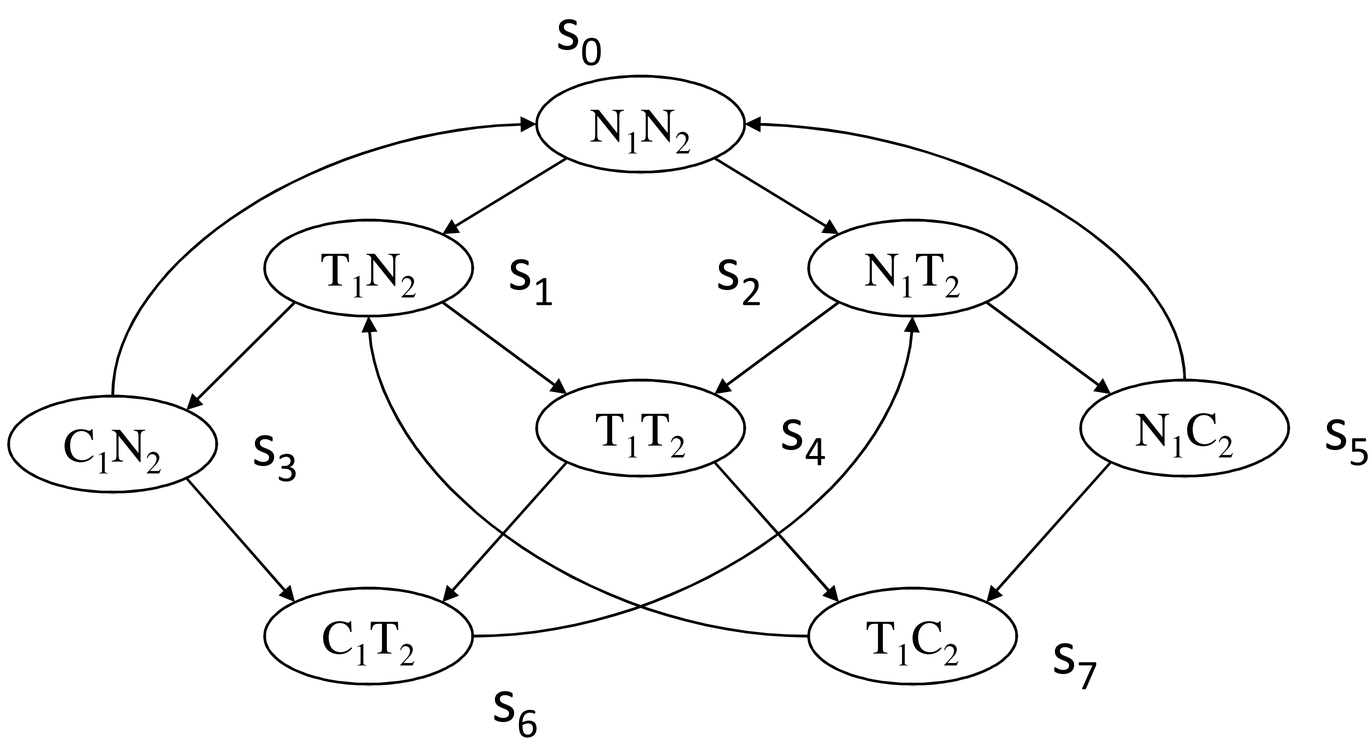
Subformuals:

t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\, (t_1 \rightarrow AFc_1)$,

$E(T\; U\; (\neg\, (t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\; U\; (\neg\, (t_1 \rightarrow AFc_1)))$

Questions



t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

Questions

Temporal Operator:

AF c_1

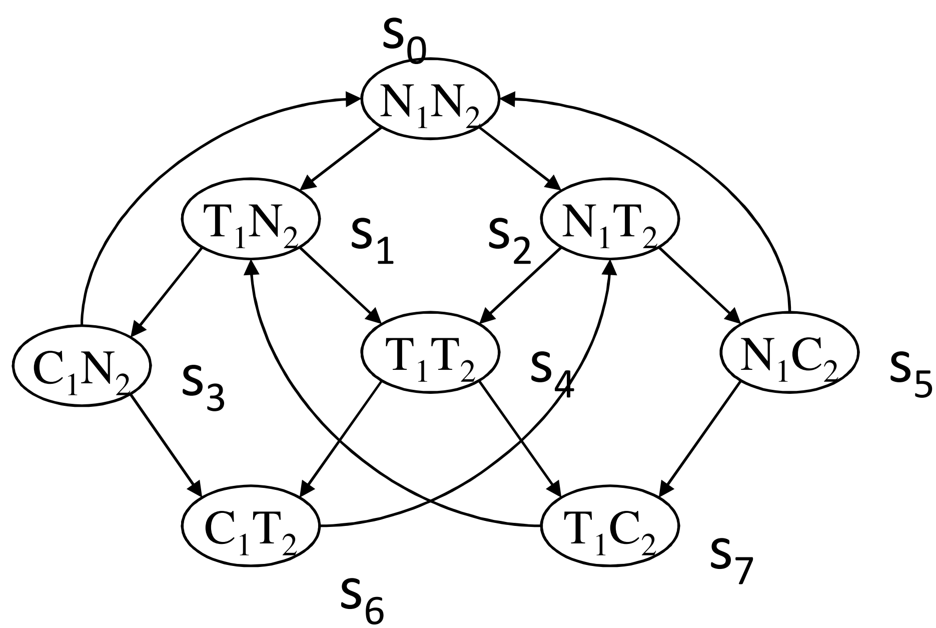
- If any state s is labeled with c_1, label it with AF c_1
- Repeat: label any state with AF c_1 if all successor states are labeled with AF c_1 until there is no change.

Questions

t_1: $\{s_1, s_4, s_7\}$

c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$



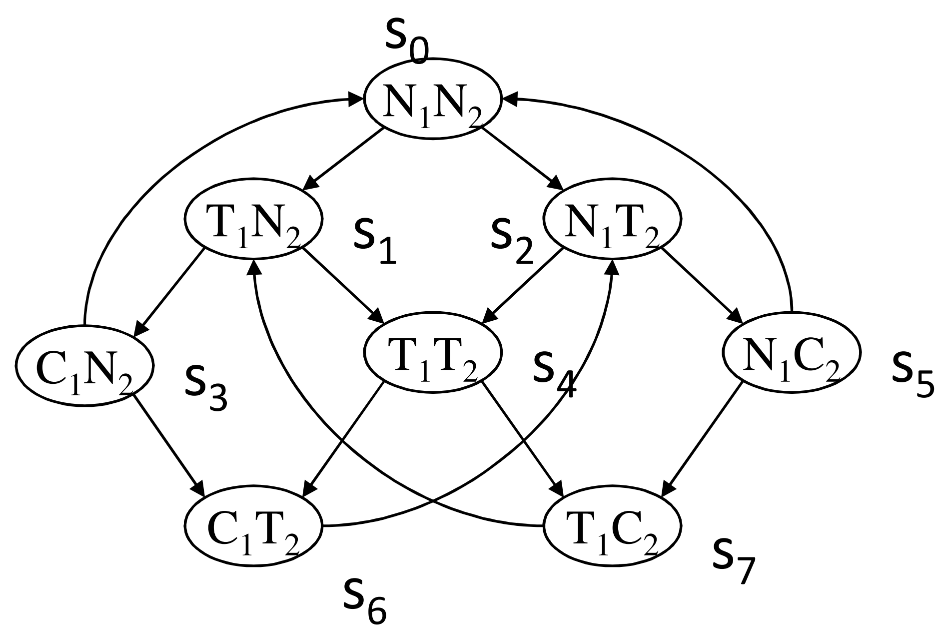
Questions

t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$



Questions

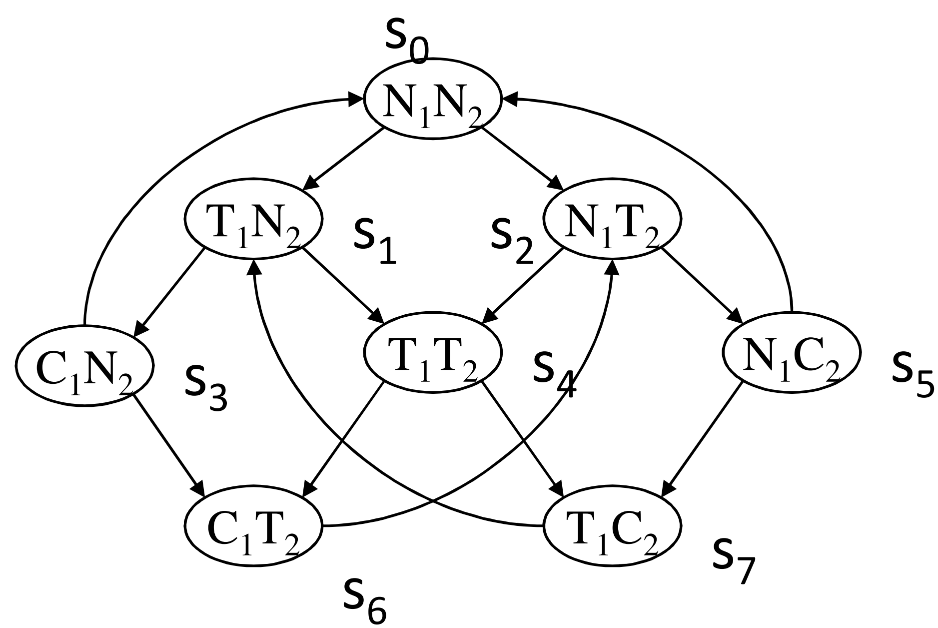
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



Questions

Temporal Operator:

E(p U q)

- If any state s is labeled with q, label it with E(p U q)
- Repeat: label any state with E(p U q) if it is labeled with p and at least one of its successor is labeled with E(p U q) until there is no change.

Questions

t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

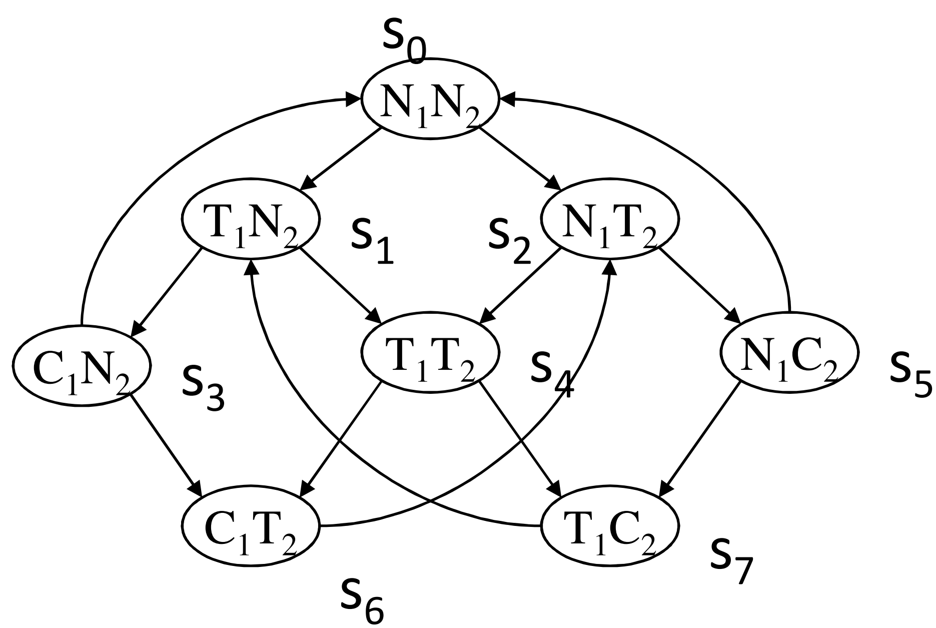
AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$

$E(T \, U \, \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7\}$



Questions

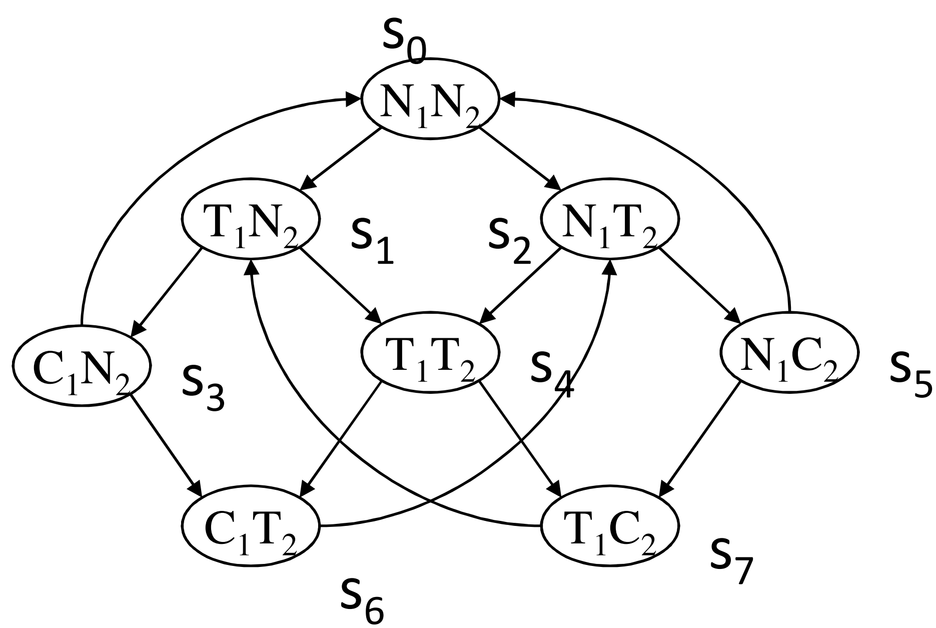
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5\}$

Questions

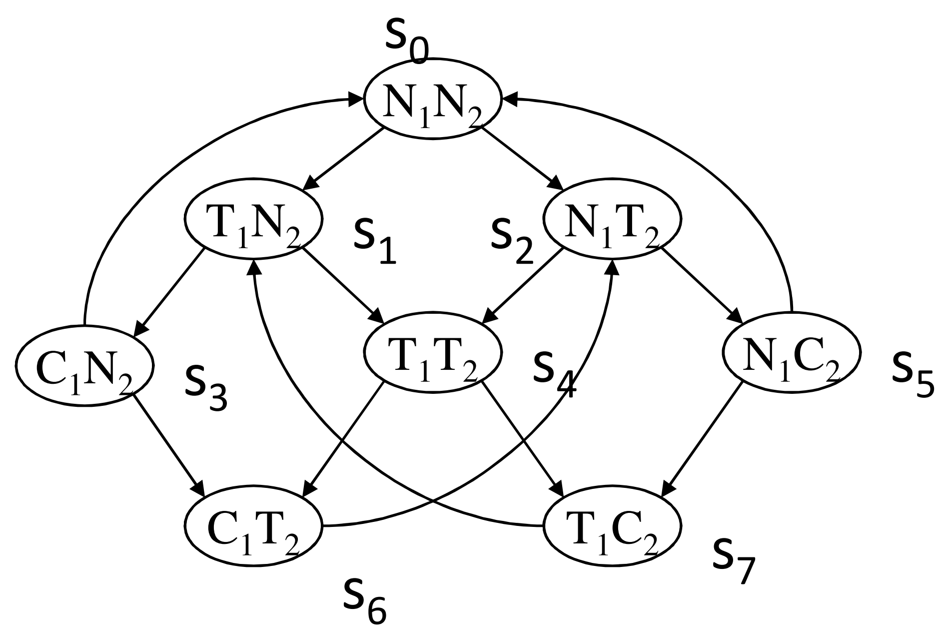
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5, s_3\}$

Questions

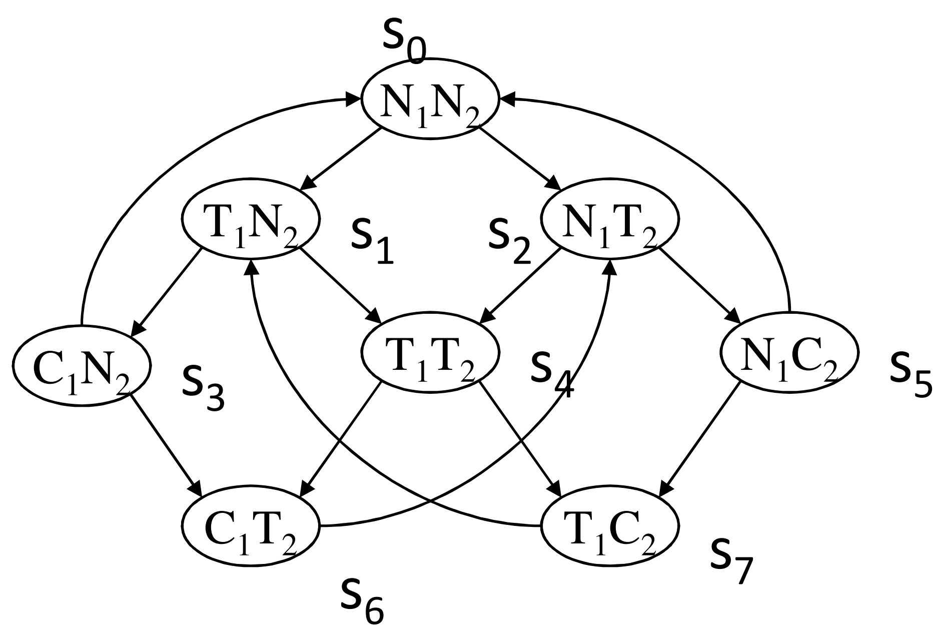
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

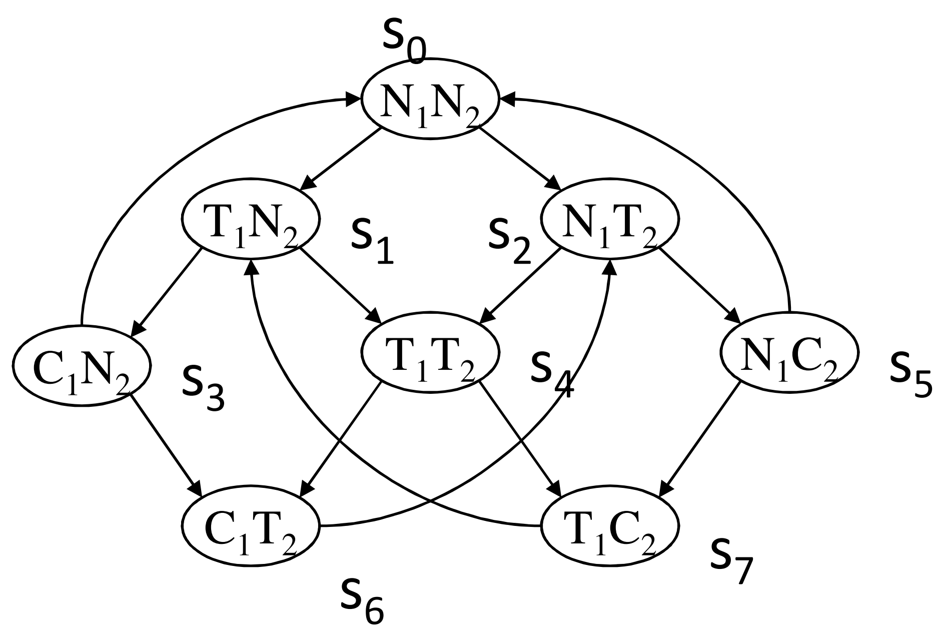
$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5, s_3, s_6\}$

Questions

t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$



AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$

$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5, s_3, s_6\}$

$\neg\, E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{\}$

NPTEL Phase-II
Video course on

Design Verification and Test of Digital VLSI Designs

Dr. Santosh Biswas
Dr. Jatindra Kumar Deka
IIT Guwahati

Module V: Verification Techniques

Lecture III: Model Checking Algorithms

Model Checking Algorithm

Given the model '*M*' and a CTL formula Φ as input.

Model checking algorithm provides all the states of model M which satisfy Φ

Labeling Algorithm

Labeling Algorithms

CTL model checking algorithm basically works by iteratively determining (i.e., labeling) states which satisfy a given CTL formula.

The basic input/output of labeling algorithm are as follows:

INPUT : A **CTL model *'M'* = *(S, →, L)***

CTL formula Φ.

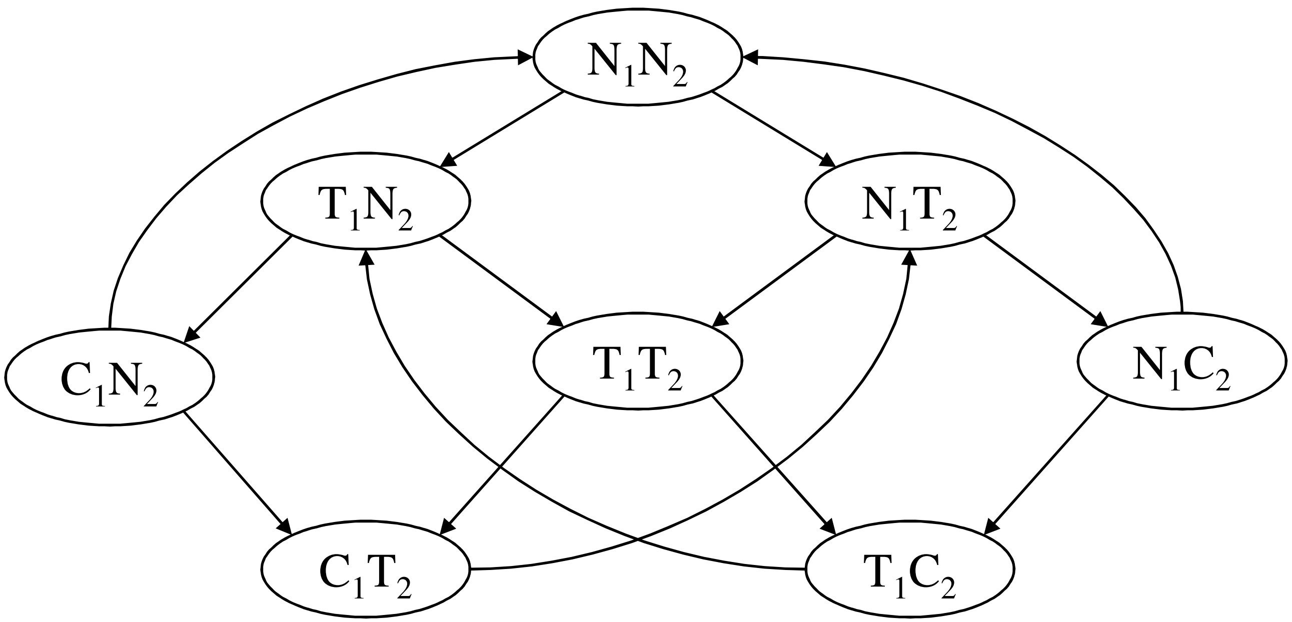
OUTPUT : The set of states of M which satisfy Φ.

CTL Model Checking

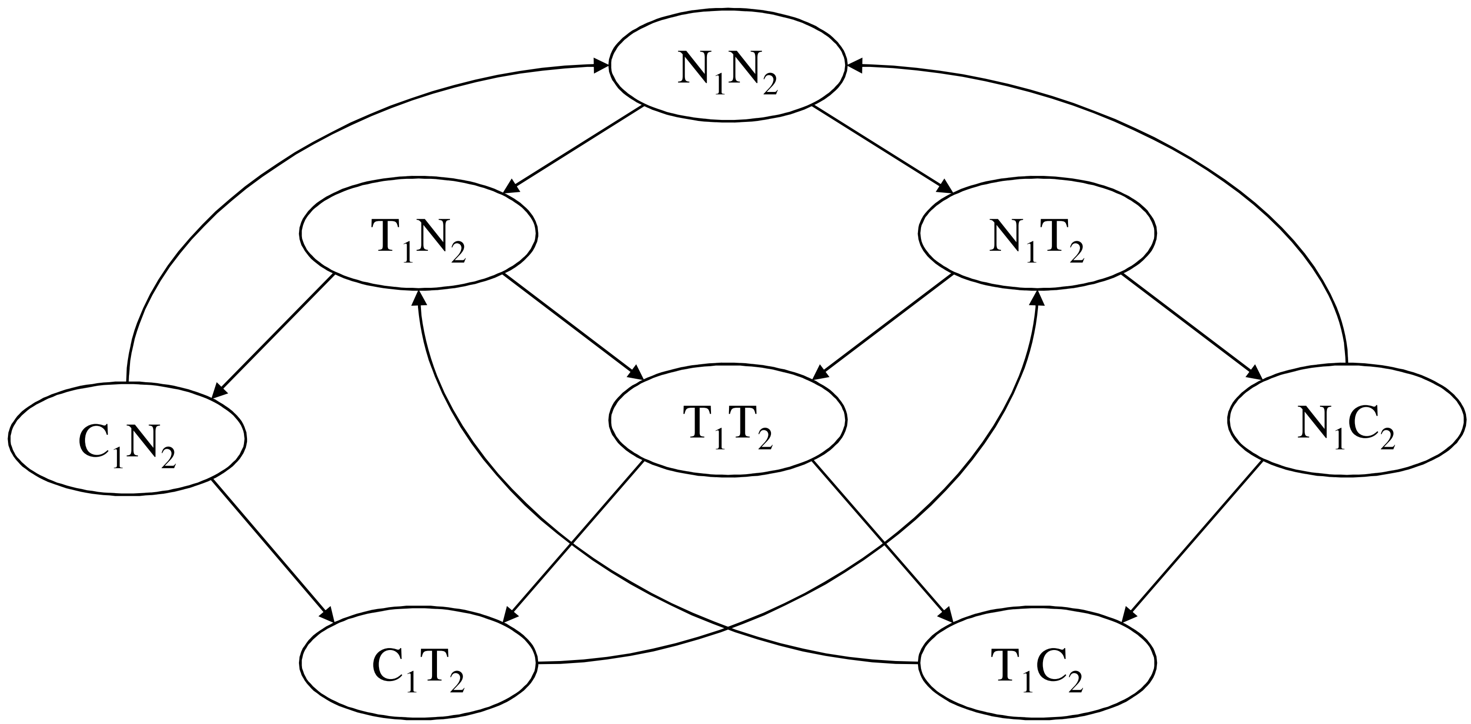
- Algorithms for the operators:
  - EX
  - AF
  - EU

Examples

- Apply the model checking algorithm to label the states with the formula AG $\neg(c_1 \wedge c_2)$ (safety property)



Examples



$AG\ \neg(c_1 \wedge c_2)$

Examples

- We have the methods for EX, AF and EU

$$AG\,\neg(c_1 \wedge c_2) \equiv \neg\, EF\,(c_1 \wedge c_2)$$
$$\equiv \neg\, E(T\ U\ (c_1 \wedge c_2))$$

$$AGp \equiv \neg EF \neg p$$

$$EFp \equiv E(true\ U\ p)$$

Examples

- We have the methods for EX, AF and EU

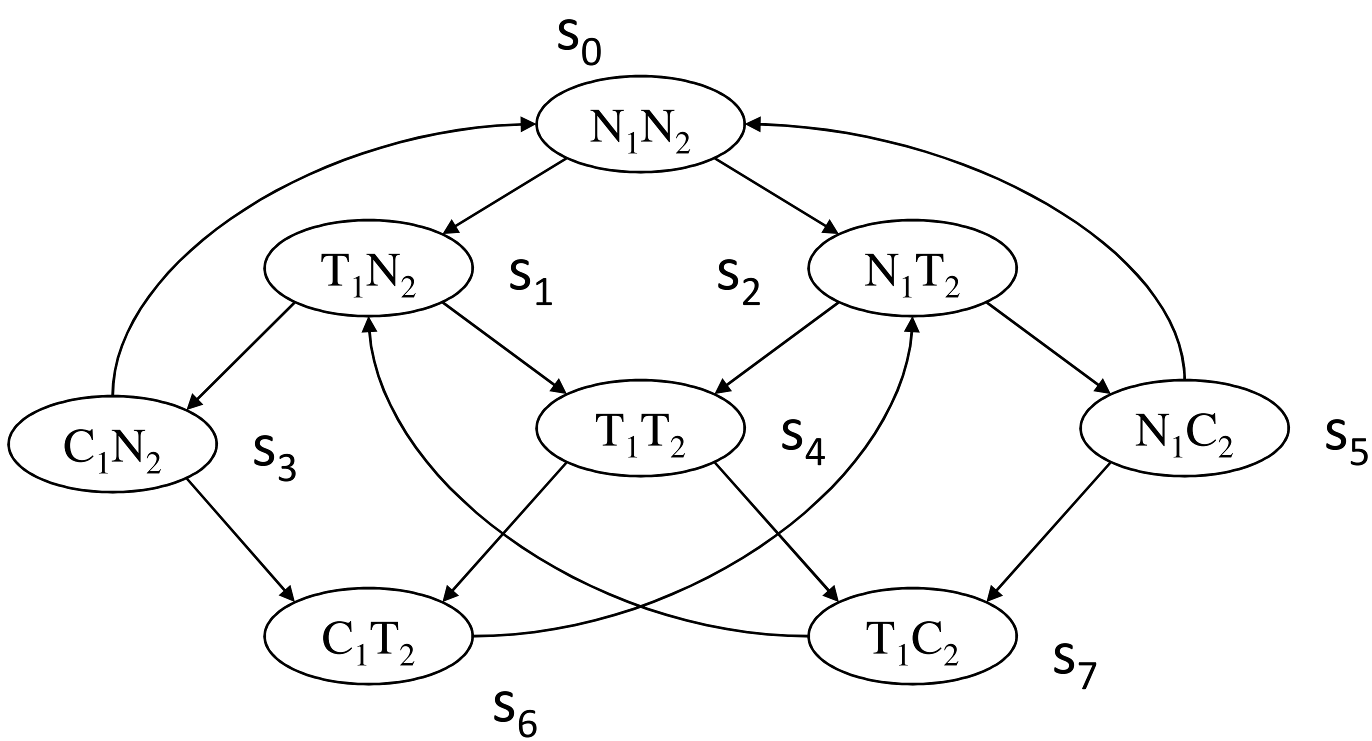
$$AG \neg(c_1 \wedge c_2) \equiv \neg EF\ (c_1 \wedge c_2)$$
$$\equiv \neg E(T\ U\ (c_1 \wedge c_2))$$

Subformulas:

c_1, c_2 , $c_1 \wedge c_2$, $E(T\ U\ (c_1 \wedge c_2))$

$\neg E(T\ U\ (c_1 \wedge c_2))$

Examples



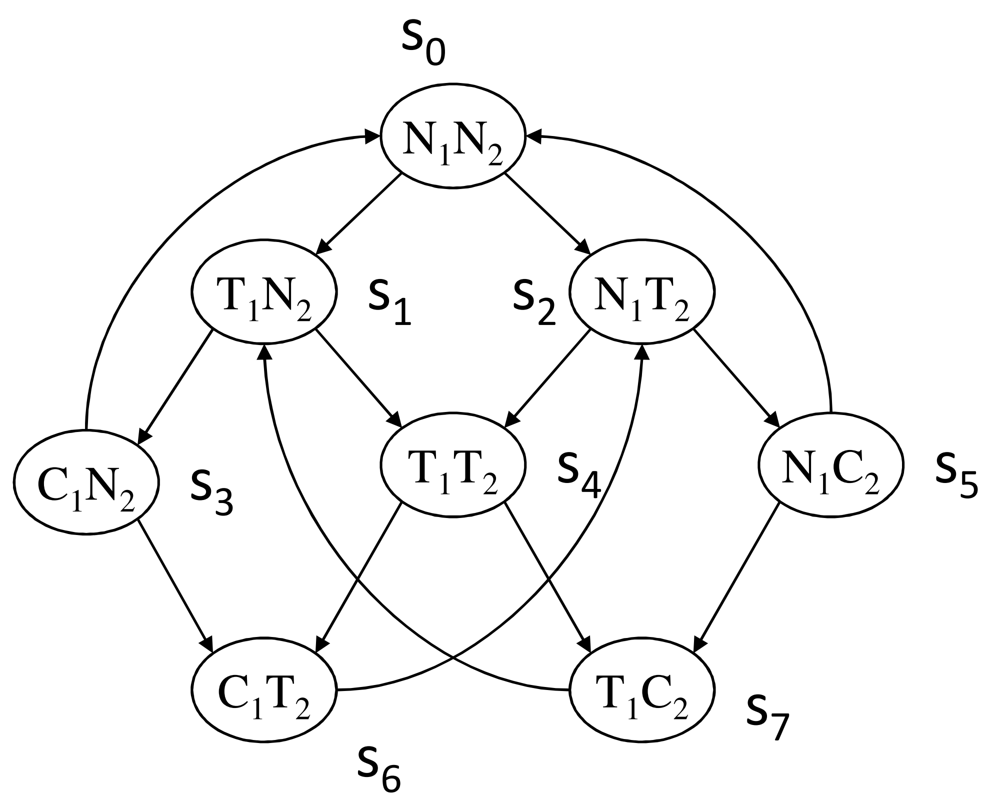
c_1: $\{s_3, s_6\}$ c_2: $\{s_5, s_7\}$

Examples

c_1: $\{s_3, s_6\}$

c_2: $\{s_5, s_7\}$

$c_1 \wedge c_2$: $\{\}$



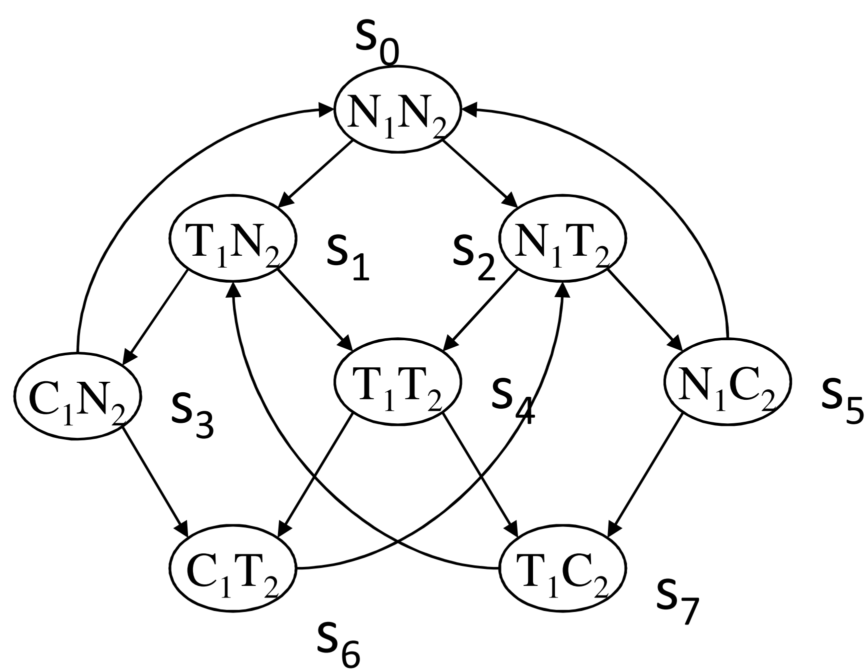
Examples

c_1: $\{s_3, s_6\}$

c_2: $\{s_5, s_7\}$

$c_1 \wedge c_2$: $\{\}$

$E(T\ U\ (c_1 \wedge c_2))$: $\{\}$



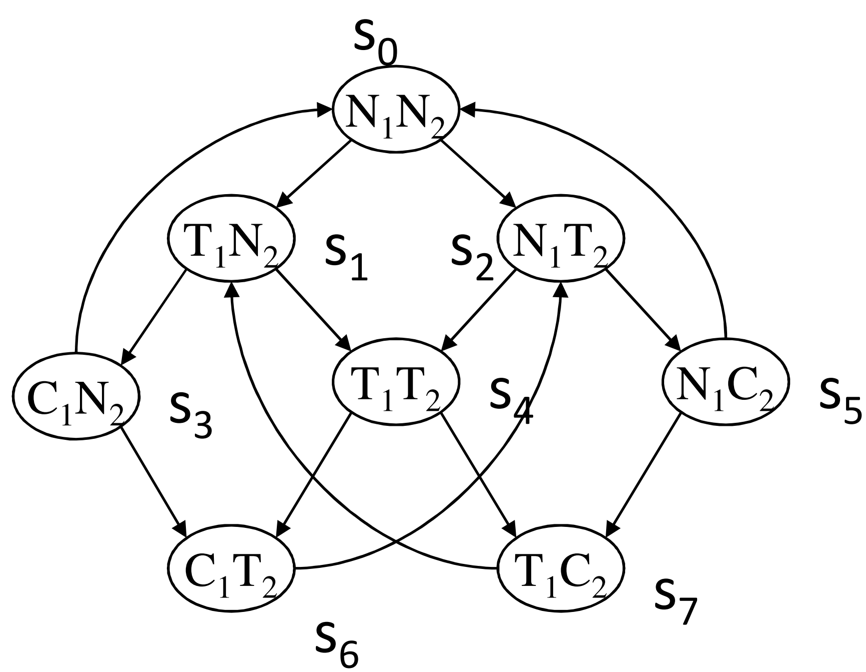
Examples

c_1: $\{s_3, s_6\}$

c_2: $\{s_5, s_7\}$

$c_1 \wedge c_2$: $\{\}$

$E(T\ U\ (c_1 \wedge c_2))$: $\{\}$

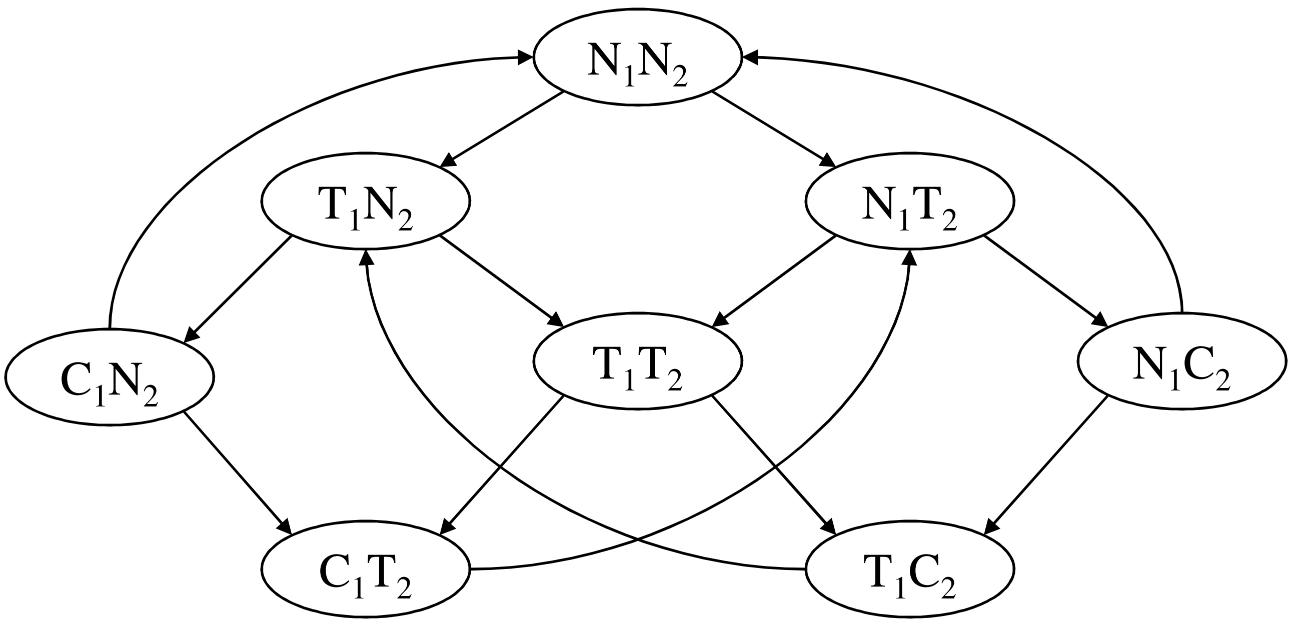


$\neg\ E(T\ U\ (c_1 \wedge c_2))$: $\{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$

$AG\ \neg(c_1 \wedge c_2) \equiv \neg\ E(T\ U\ (c_1 \wedge c_2))$

Examples

- Apply the model checking algorithm to label the states with the formula $AG(t_1 \rightarrow AFc_1)$



Examples

- We have the methods for EX, AF and EU

$$AG(t_1 \rightarrow AFc_1) \equiv \neg EF(\neg(t_1 \rightarrow AFc_1))$$
$$\equiv \neg E(T\ U\ (\neg(t_1 \rightarrow AFc_1)))$$

$$AGp \equiv \neg EF \neg p$$
$$EFp \equiv E(true\ U\ p)$$

Examples

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\, (\neg\, (t_1 \rightarrow AFc_1))$$
$$= \neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$$

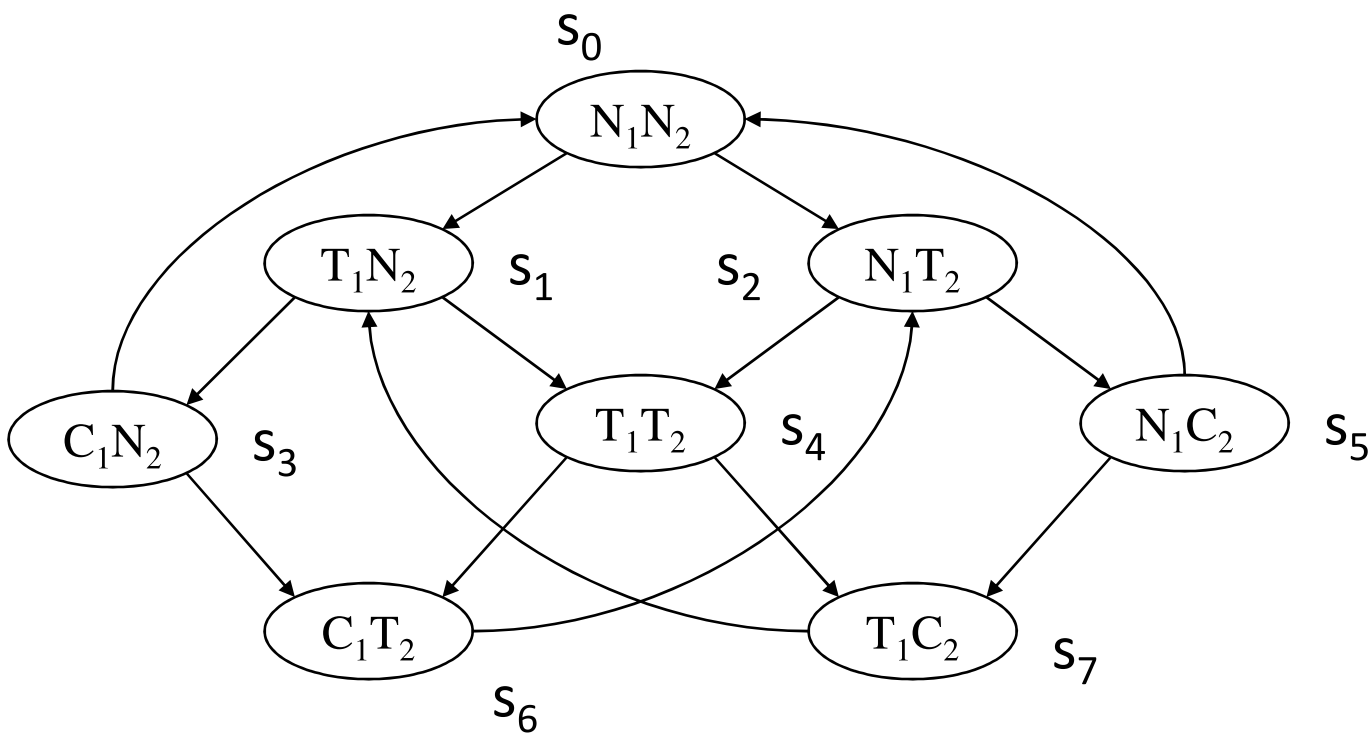
Subformuals:

t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\, (t_1 \rightarrow AFc_1)$,

$E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$

Examples



t_1: {s_1, s_4, s_7}

c_1: {s_3, s_6}

Examples

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\,(\neg\,(t_1 \rightarrow AFc_1))$$
$$= \neg\, E(T\ U\ (\neg\,(t_1 \rightarrow AFc_1)))$$

Subformuals:

t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\,(t_1 \rightarrow AFc_1)$,

$E(T\ U\ (\neg\,(t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\ U\ (\neg\,(t_1 \rightarrow AFc_1)))$

Examples

Temporal Operator:

$AF\ c_1$

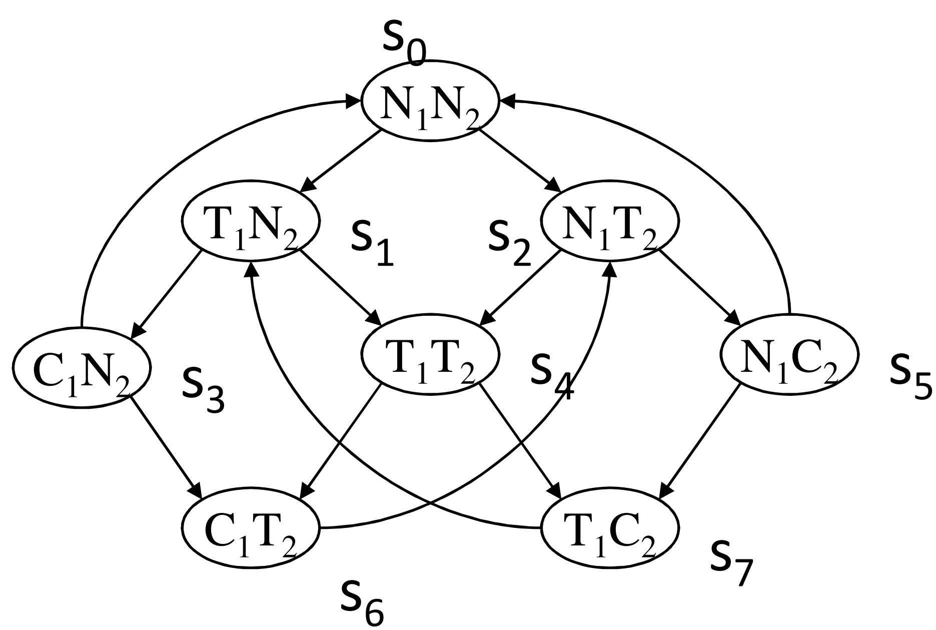
- If any state s is labeled with c_1, label it with $AF\ c_1$
- Repeat: label any state with $AF\ c_1$ if all successor states are labeled with $AF\ c_1$ until there is no change.

Examples

t_1: $\{s_1, s_4, s_7\}$

c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$



Examples

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\, (\neg\, (t_1 \rightarrow AFc_1))$$

$$= \neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$$

Subformuals:

t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\, (t_1 \rightarrow AFc_1)$,

$E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$

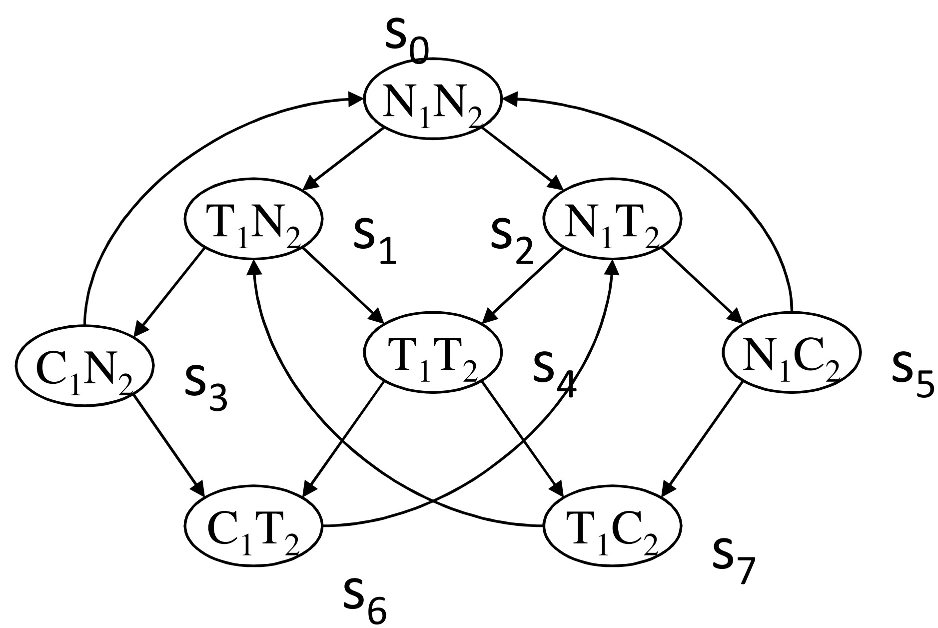
Examples

t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$



Examples

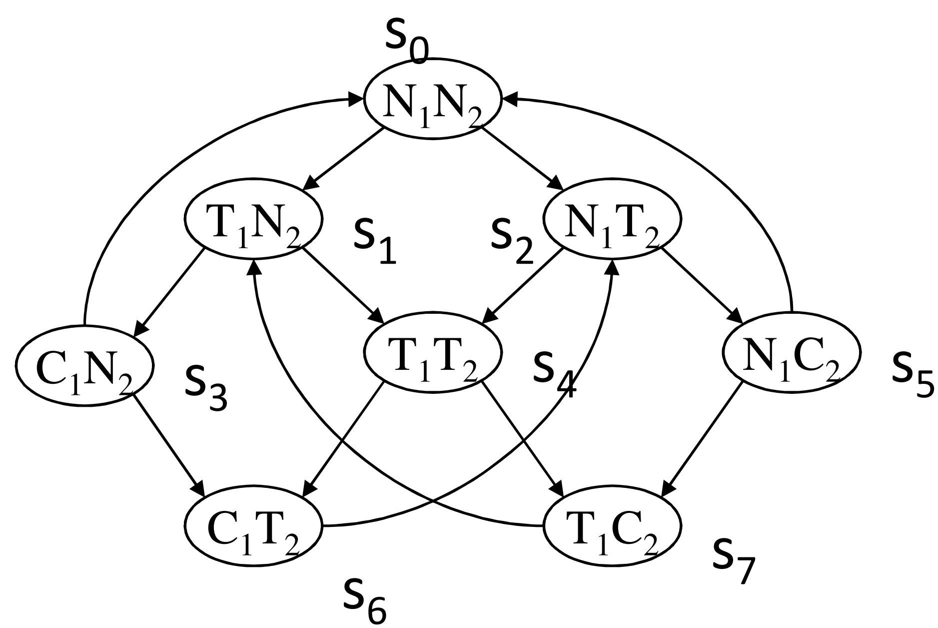
t_1: {s_1, s_4, s_7} c_1: {s_3, s_6}

AFc_1: {s_3, s_6}

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: {s_0, s_2, s_3, s_5, s_6 }

$\neg(t_1 \rightarrow AFc_1)$: {$s_1$, s_4, s_7 }



Examples

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\, (\neg\, (t_1 \rightarrow AFc_1))$$
$$= \neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$$

Subformuals:

t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\, (t_1 \rightarrow AFc_1)$,

$E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$

Examples

Temporal Operator:

E(p U q)

- If any state s is labeled with q, label it with E(p U q)
- Repeat: label any state with E(p U q) if it is labeled with p and at least one of its successor is labeled with E(p U q) until there is no change.

Examples

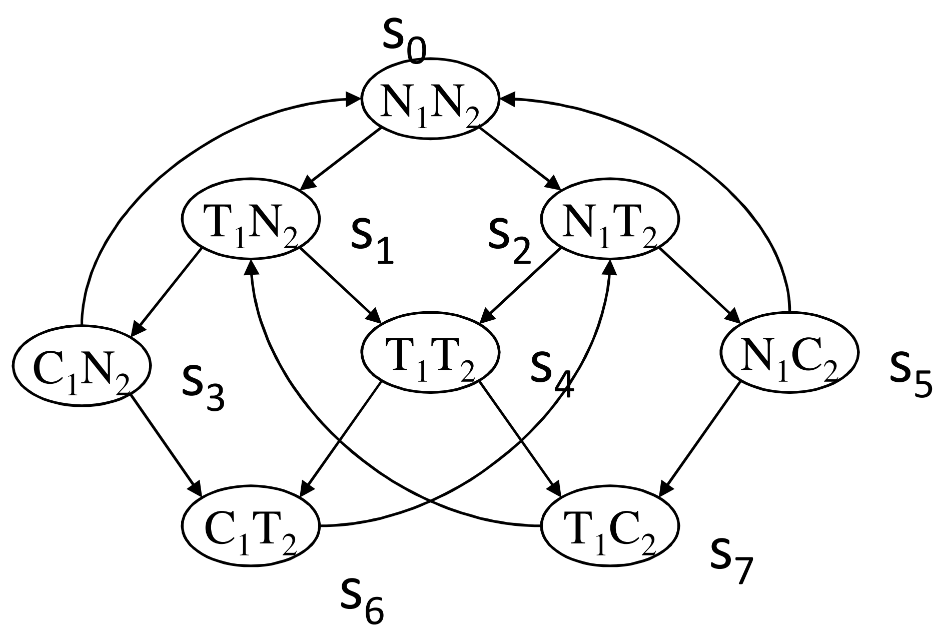
t_1: {s_1, s_4, s_7} c_1: {s_3, s_6}

AFc_1: {s_3, s_6}

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: {s_0, s_2, s_3, s_5, s_6 }

$\neg(t_1 \rightarrow AFc_1)$: {$s_1$, s_4, s_7 }



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: {$s_1$, s_4, s_7}

Examples

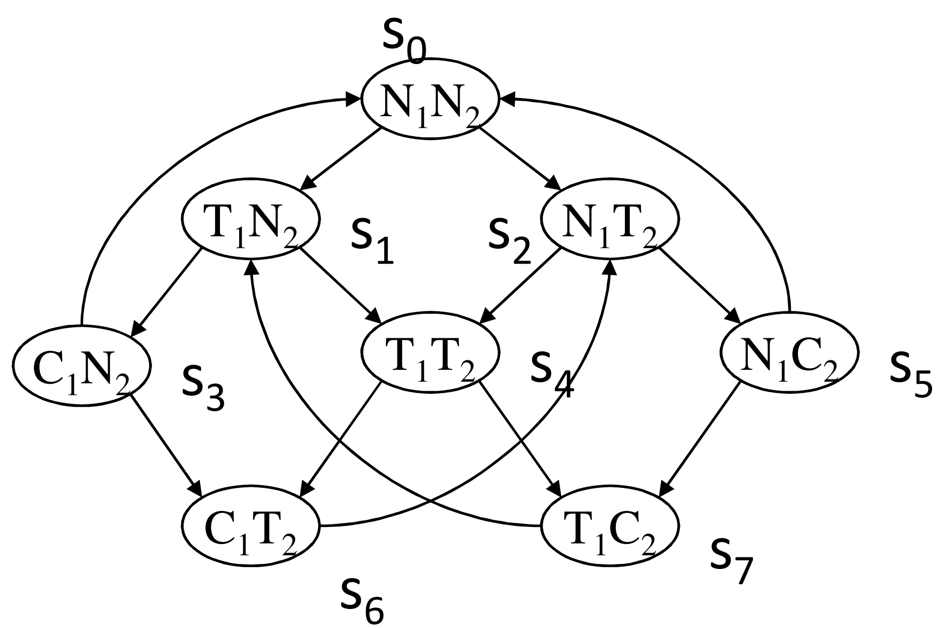
t_1: {s_1, s_4, s_7} c_1: {s_3, s_6}

AFc_1: {s_3, s_6}

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: {s_0, s_2, s_3, s_5, s_6 }

$\neg(t_1 \rightarrow AFc_1)$: {$s_1$, s_4, s_7 }



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: {$s_1$, s_4, s_7 , s_0, s_2, s_5}

Examples

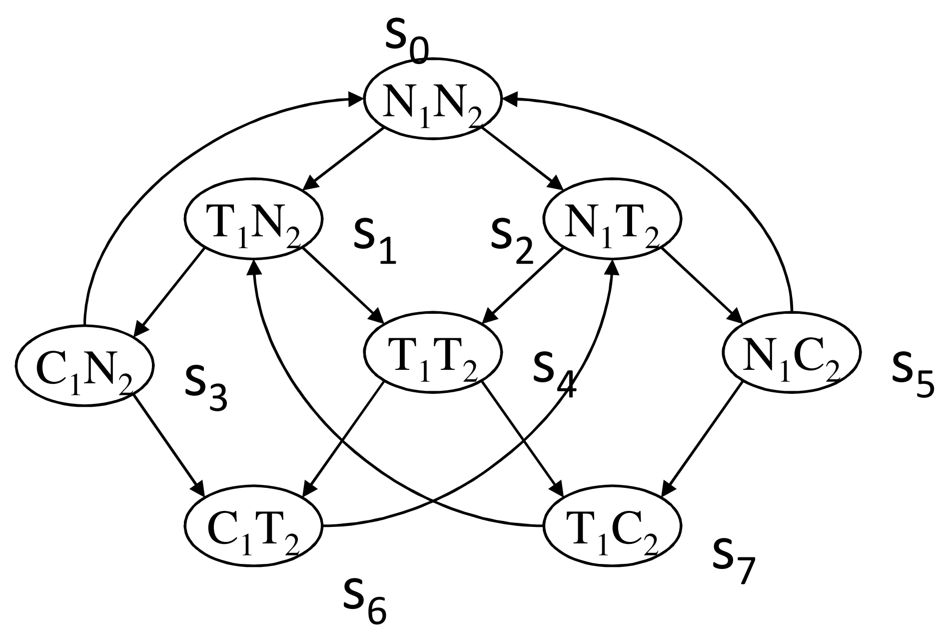
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



$E(T \cup \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5, s_3\}$

Examples

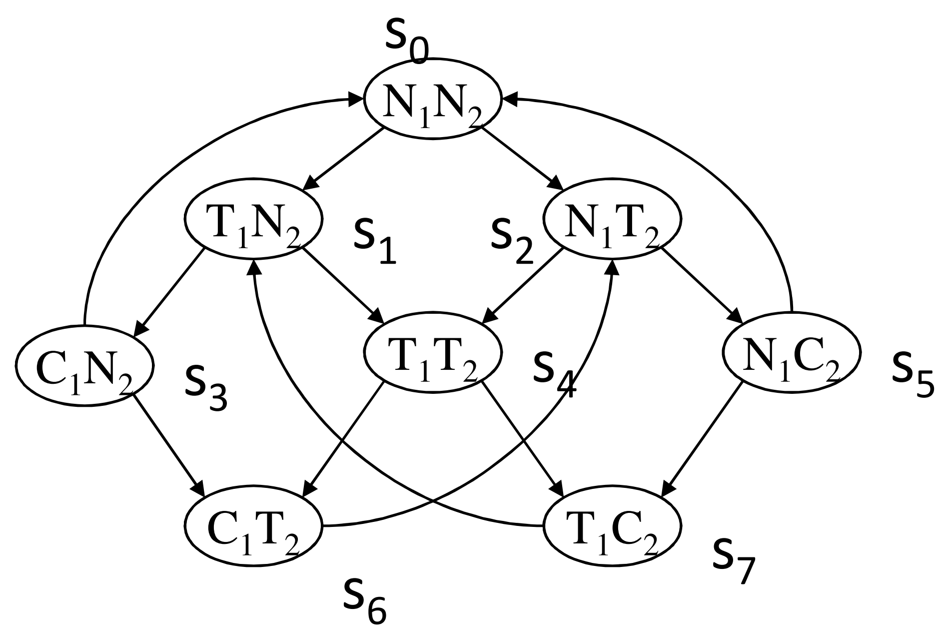
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg t_1 \vee AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5, s_3, s_6\}$

Examples

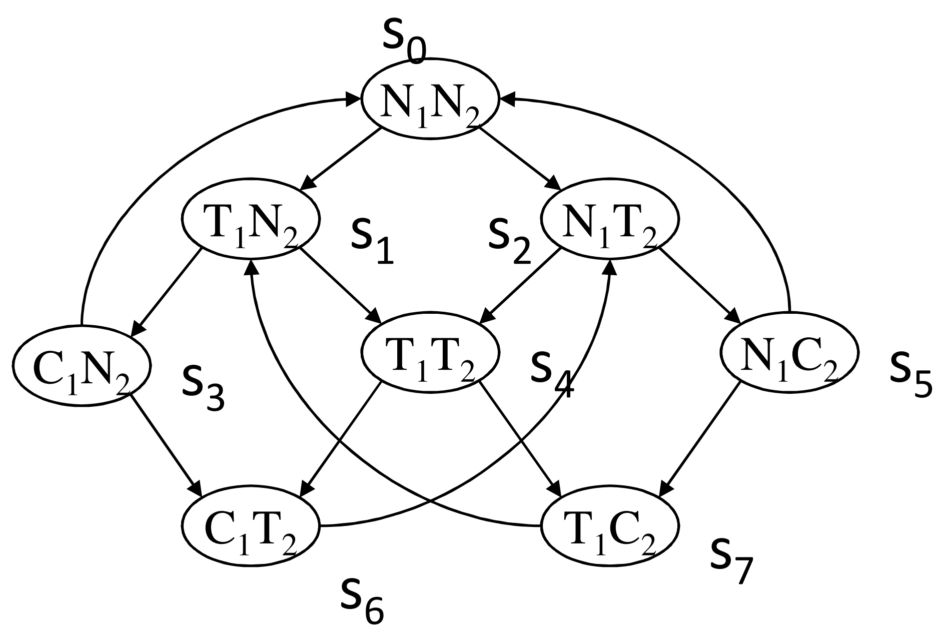
t_1: $\{s_1, s_4, s_7\}$ c_1: $\{s_3, s_6\}$

AFc_1: $\{s_3, s_6\}$

$t_1 \rightarrow AFc_1 = \neg\, t_1 \lor AFc_1$

$(t_1 \rightarrow AFc_1)$: $\{s_0, s_2, s_3, s_5, s_6\}$

$\neg(t_1 \rightarrow AFc_1)$: $\{s_1, s_4, s_7\}$



$E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{s_1, s_4, s_7, s_0, s_2, s_5, s_3, s_6\}$

$\neg\, E(T\ U\ \neg(t_1 \rightarrow AFc_1))$: $\{\}$

Examples

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\, (\neg\, (t_1 \rightarrow AFc_1))$$
$$= \neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$$

Subformuals:

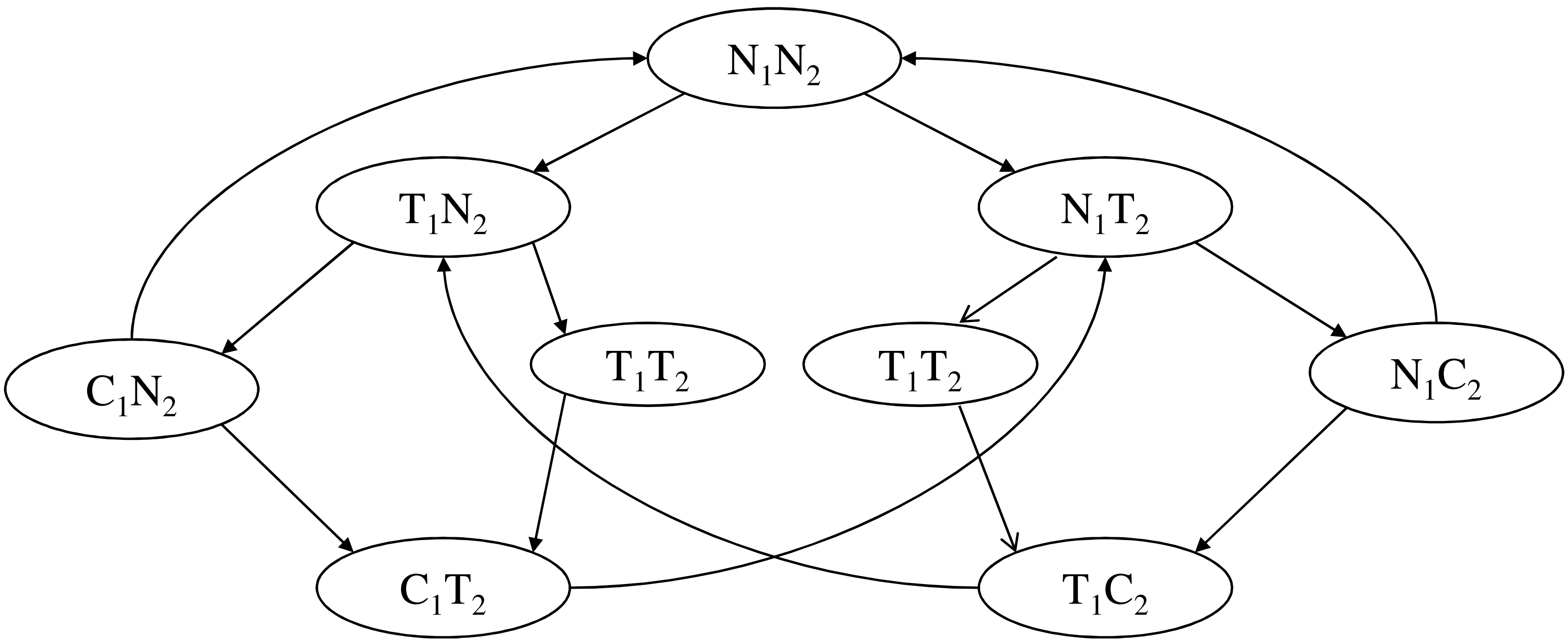
t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\, (t_1 \rightarrow AFc_1)$,

$E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$

Questions

Apply the model checking algorithm to label the states with the formula $AG(t_1 \rightarrow AFc_1)$



Examples

$$AG(t_1 \rightarrow AFc_1) = \neg\, EF\, (\neg\, (t_1 \rightarrow AFc_1))$$

$$= \neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$$

Subformuals:

t_1, c_1 , AFc_1, $(t_1 \rightarrow AFc_1)$, $\neg\, (t_1 \rightarrow AFc_1)$,

$E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$,

$\neg\, E(T\ U\ (\neg\, (t_1 \rightarrow AFc_1)))$

CTL Model Checking

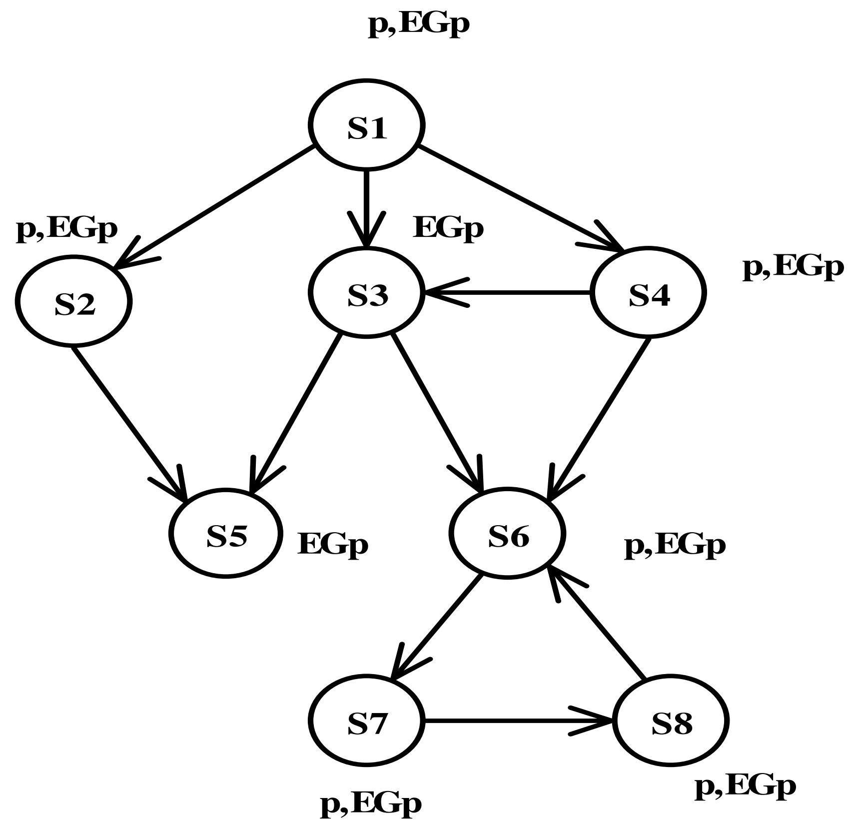
- Algorithms for the operators:
  - EX
  - AF
  - EU
- We may write procedure for other operators also
  - EG or AG

Labeling algorithm for EG

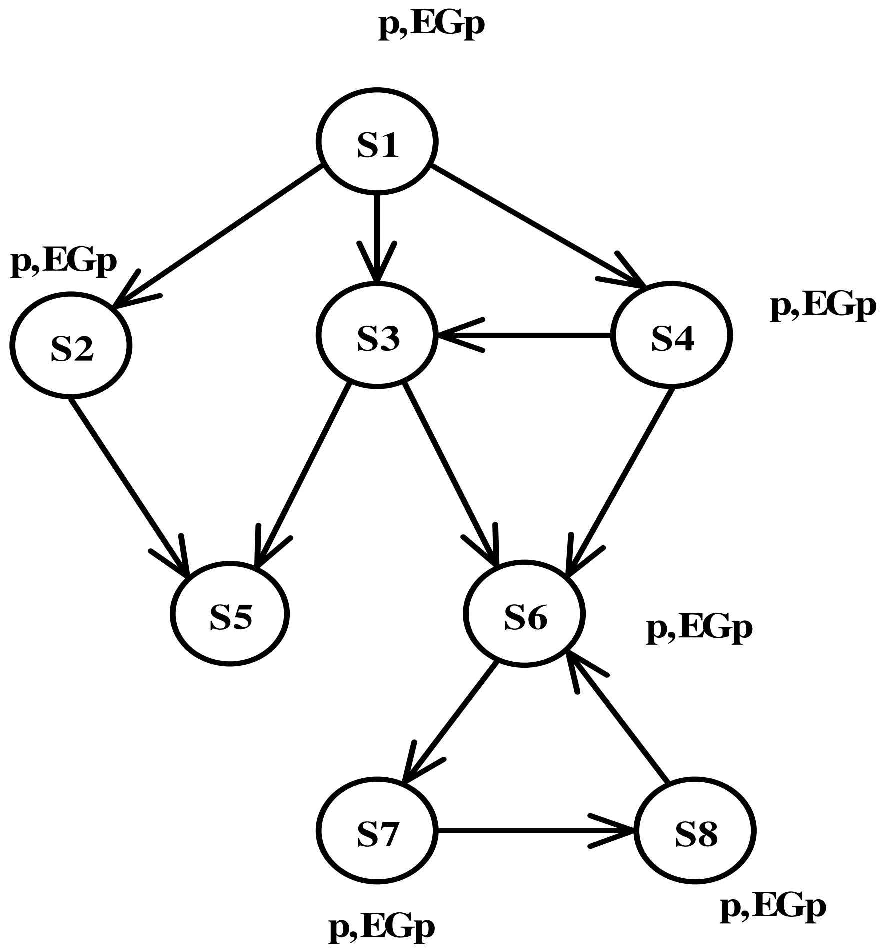
Step1: Label all the states with EGp.

Step2: If any state s is not labeled with p, delete the label EGp.

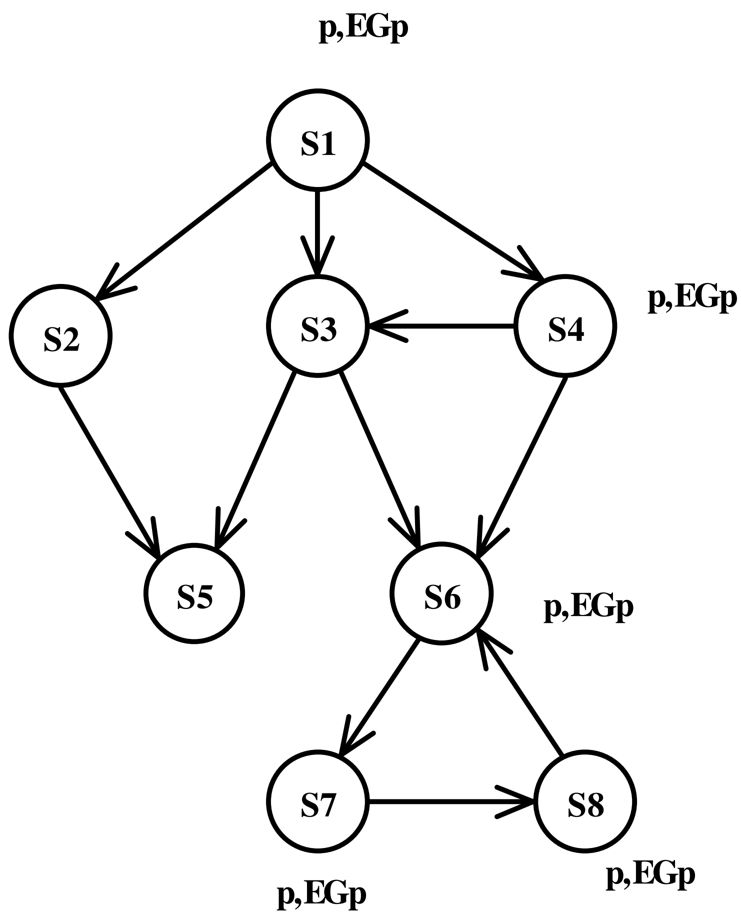
Step3: Repeat: delete the label EGp from any state if none of its successors is labeled with EGp until there is no change.



Label each state by EGp



Delete the label EGp if the sate is not labeled with p



Delete the label EGp if non of its successor is labeled with EGp

- For the operators AFq and E(p U q)
  - We start from nothing
  - Collecting the states that are labeled with q
  - Repeat the process for collection
- For the operator EG
  - We start from complete state space
  - Delete states from this set

Questions

- Write the labeling algorithm for the temporal operator AG.

Labeling algorithm for EG

Step1: Label all the states with EGp.

Step2: If any state s is not labeled with p, delete the label EGp.

Step3: Repeat: delete the label EGp from any state if none of its successors is labeled with EGp until there is no change.

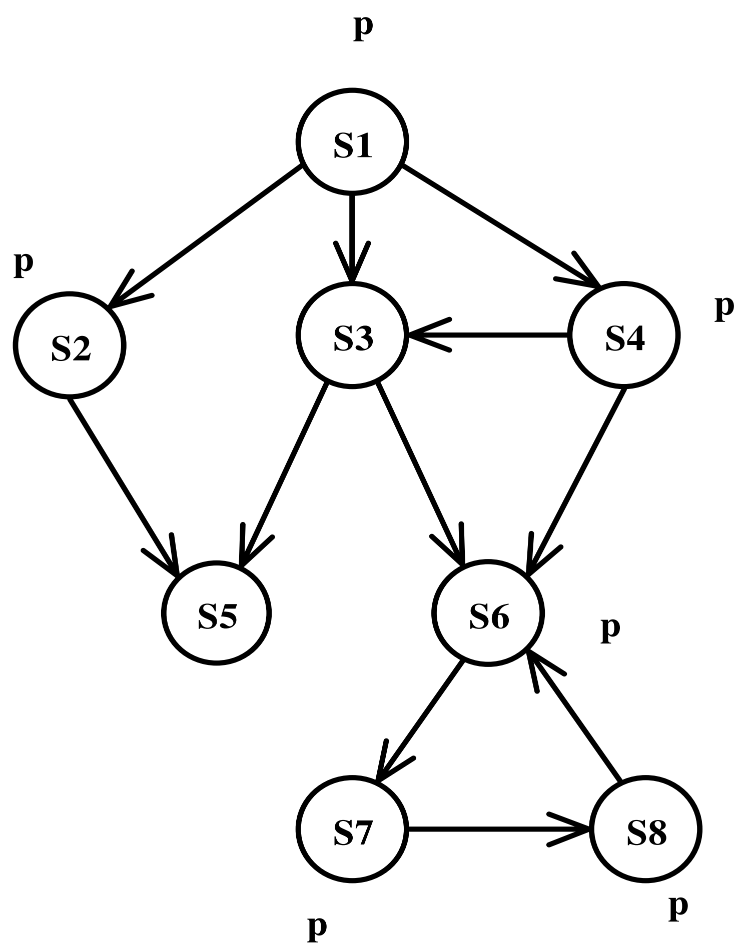
- Complexity Issue

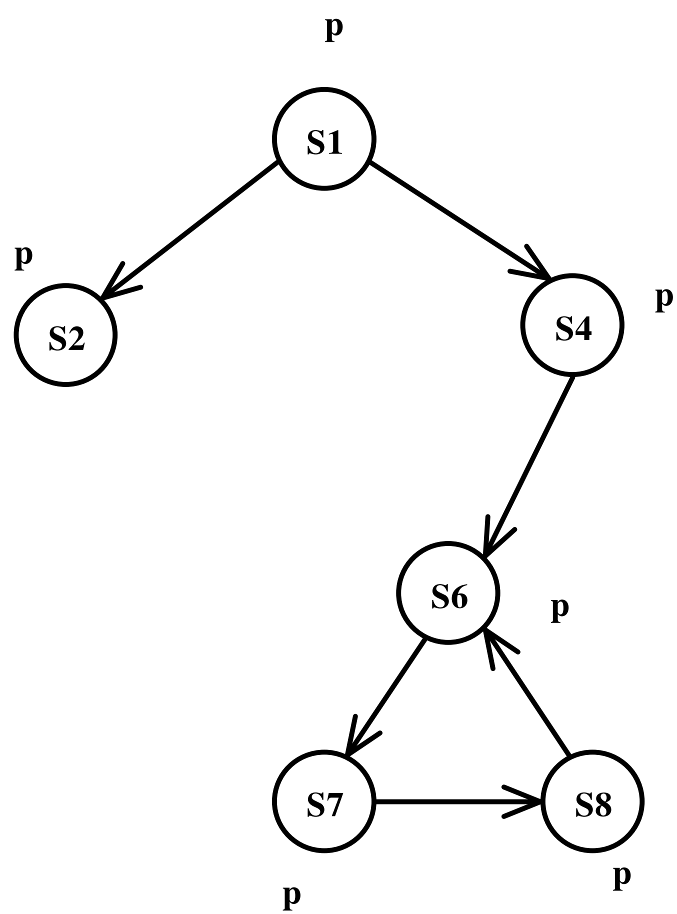
Different way of handling EG

Step1: Restrict the graph to states satisfying p, i.e., delete all other states and their transitions.

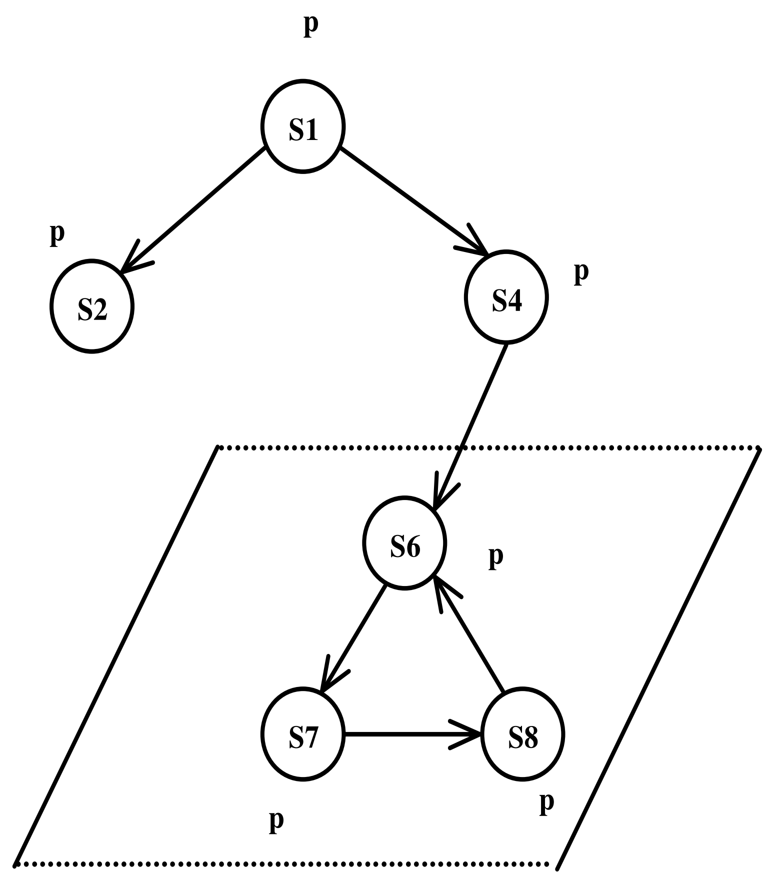
Step2: Find the maximal strongly connected components (SCCs); These are maximal regions of the state space in which every state is linked with every other one in that region.

Step3: Use backwards breadth-first searching on the restricted graph to find any state that can reach an SCC.

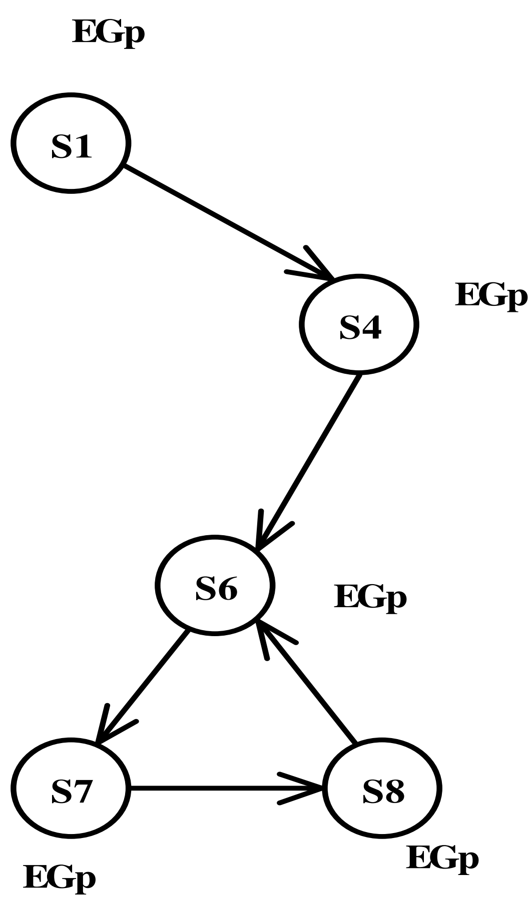

p
S1
p
S2
S3
S4
p
S5
S6
p
S7
S8
p
p



Restrict the graph for the states where p is true



Find the strongly connected component (SCC)


EGp
S1
S4
EGp
S6
EGp
S7
S8
EGp
EGp

NPTEL Phase-II
Video course on

Design Verification and Test of Digital VLSI Designs

Dr. Santosh Biswas
Dr. Jatindra Kumar Deka
IIT Guwahati

Module V: Verification Techniques

Lecture IV: Model Checking with Fairness

Labeling Algorithms

CTL model checking algorithm basically works by iteratively determining (i.e., labeling) states which satisfy a given CTL formula.

The basic input/output of labeling algorithm are as follows:

INPUT : A **CTL model *'M'* = *(S, →, L)***

CTL formula Φ.

OUTPUT : The set of states of M which satisfy Φ.

Example

- Design a controller for a microwave oven.

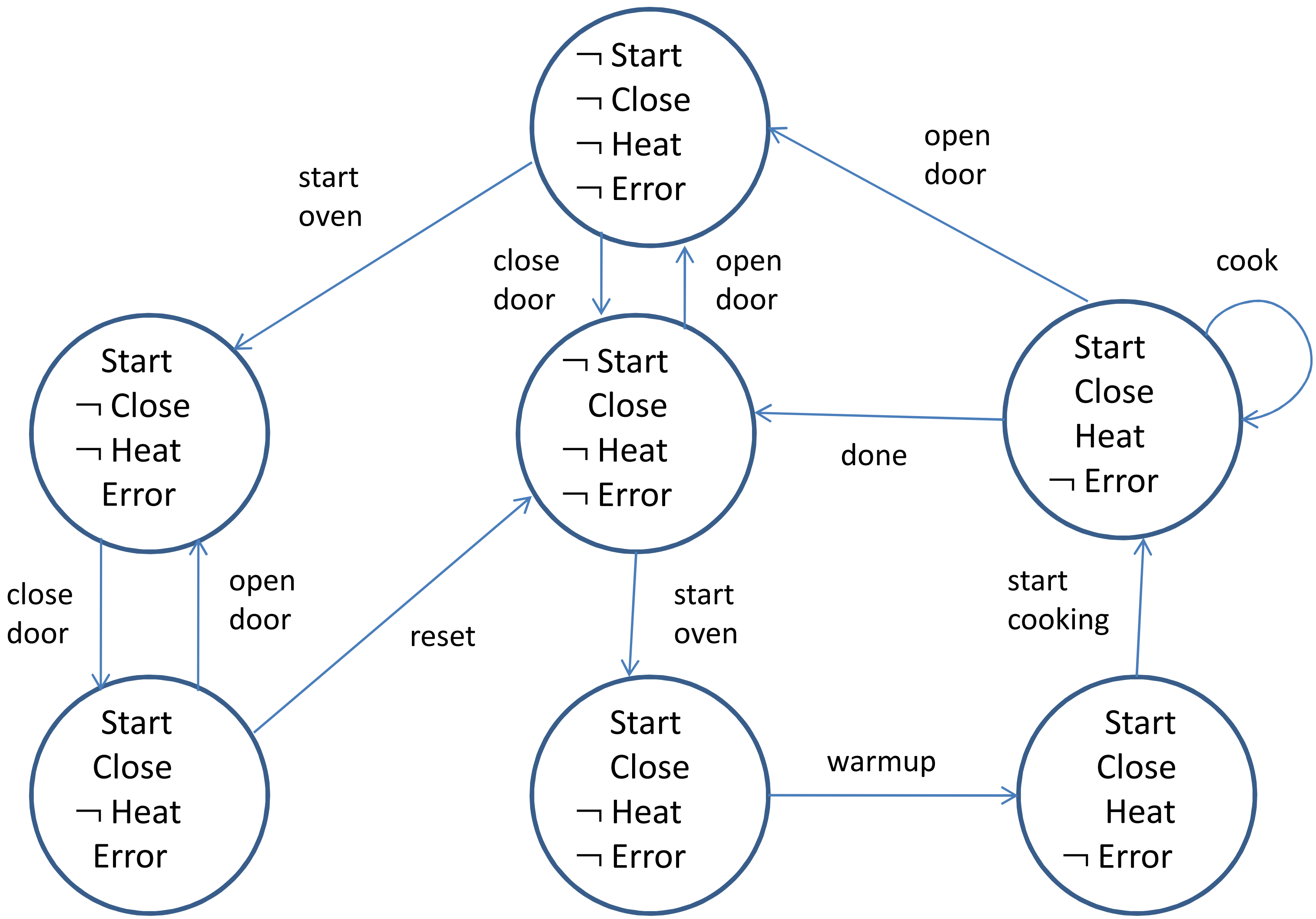
Example

- Design a controller for a microwave oven.
  - Door of the oven: either open or close
  - Start of the oven
  - reset

Example

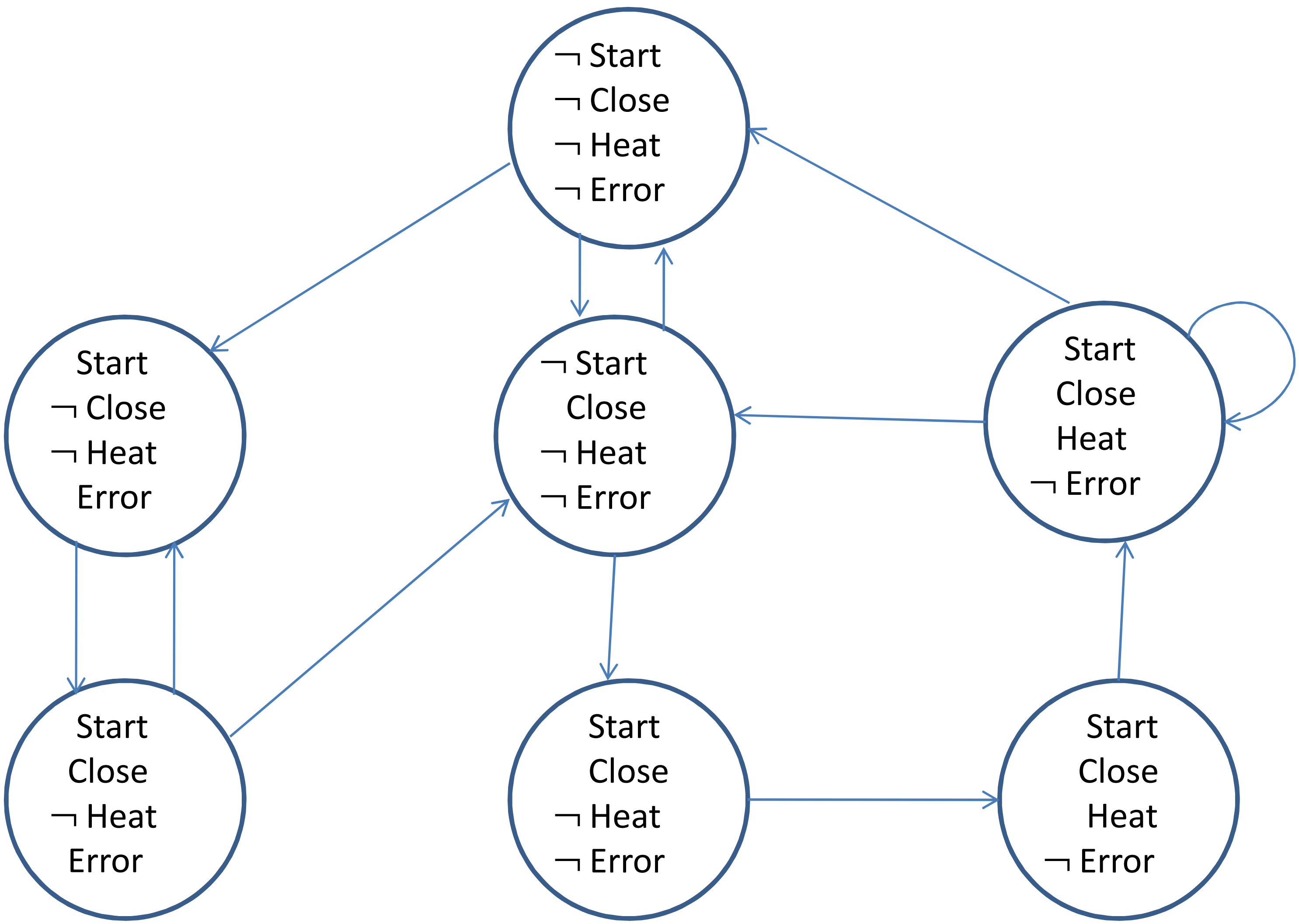
- Simplified model:
  - Start
  - Close
  - Heat
  - Error

Example



¬ Start
¬ Close
¬ Heat
¬ Error
start oven
close door
open door
open door
cook
Start
¬ Close
¬ Heat
Error
¬ Start
Close
¬ Heat
¬ Error
Start
Close
Heat
¬ Error
done
close door
open door
reset
start oven
start cooking
Start
Close
¬ Heat
Error
Start
Close
¬ Heat
¬ Error
warmup
Start
Close
Heat
¬ Error

Example



¬ Start
¬ Close
¬ Heat
¬ Error
Start
¬ Close
¬ Heat
Error
¬ Start
Close
¬ Heat
¬ Error
Start
Close
Heat
¬ Error
Start
Close
¬ Heat
Error
Start
Close
¬ Heat
¬ Error
Start
Close
Heat
¬ Error

Example

- Microwave oven should not heat up with its door open.

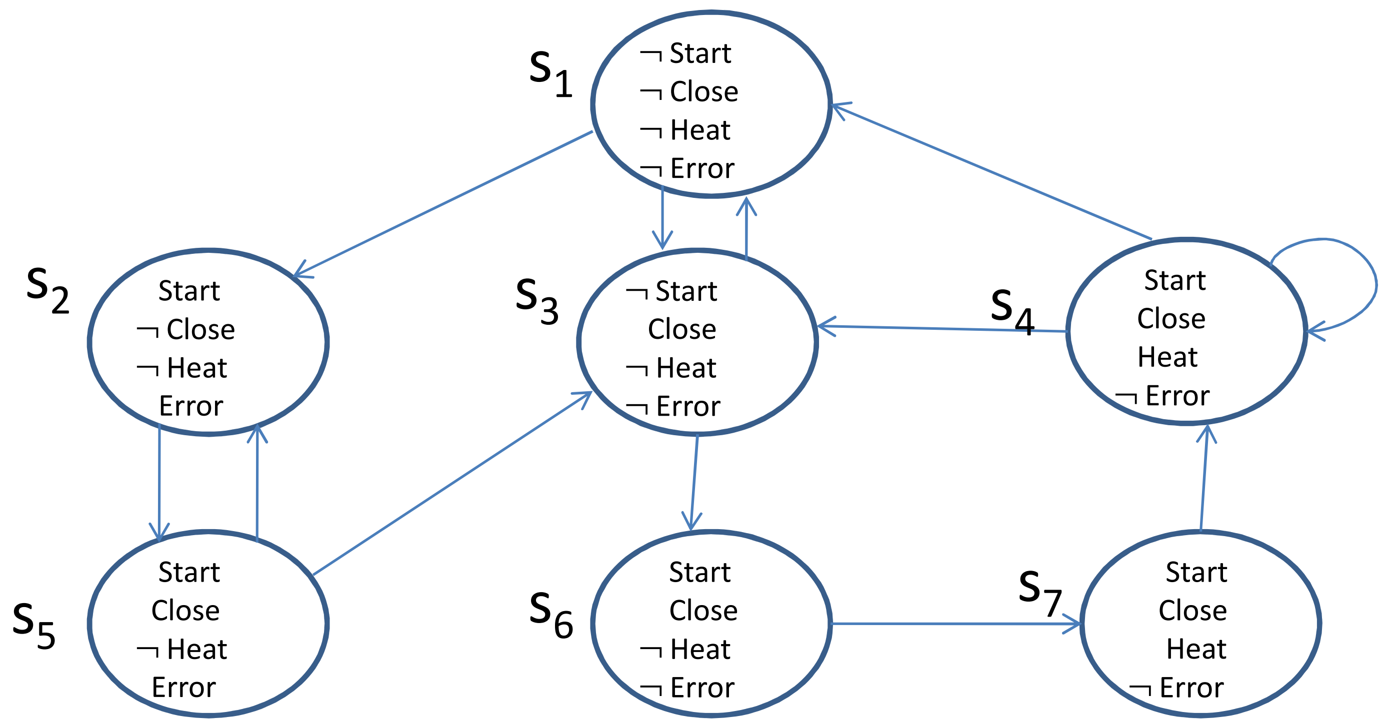
- Once we start the oven, eventually it must turn on the heating coil.

Example

- Microwave oven should not heat up with its door open.
  - $AG(\neg(\neg close \wedge heat))$

- Once we start the oven, eventually it must turn on the heating coil.
  - $AG(start \rightarrow AF\ heat)$

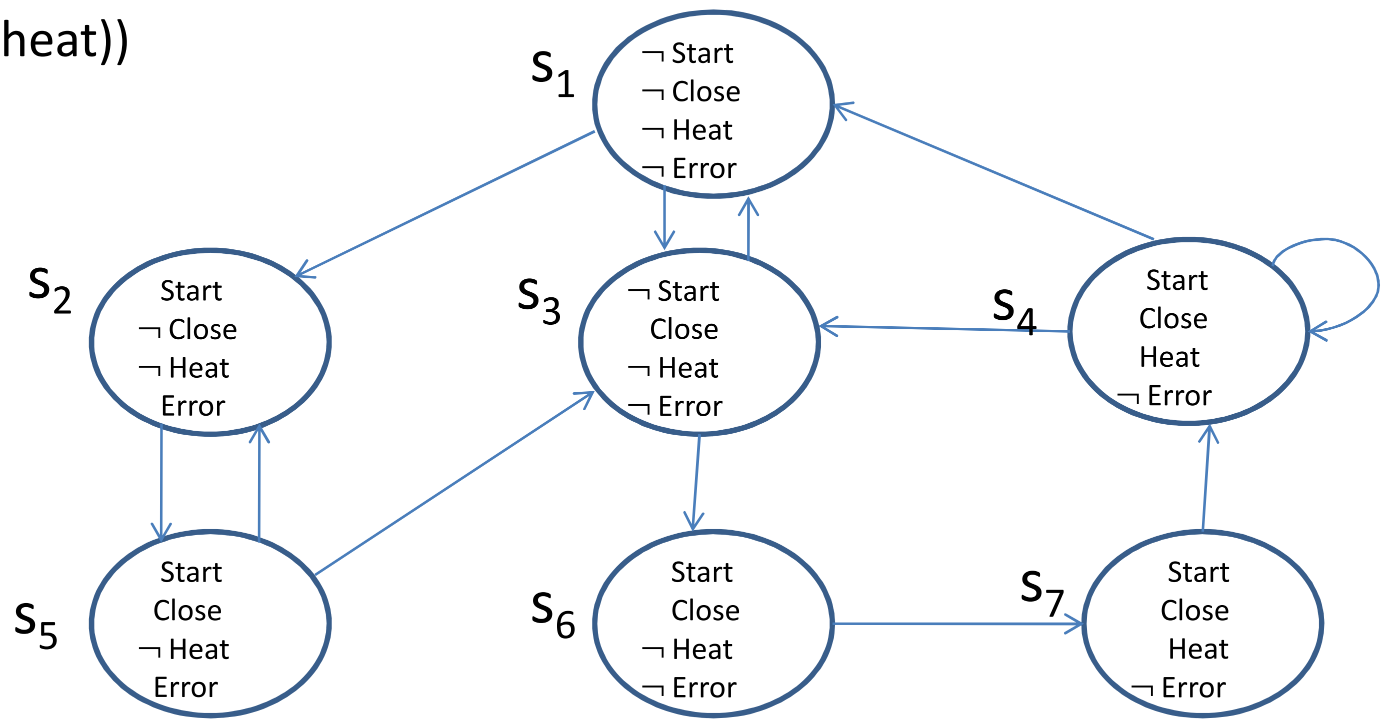
Example



AG (¬ (¬ close ∧ heat))

Example

AG (¬ (¬ close ∧ heat))

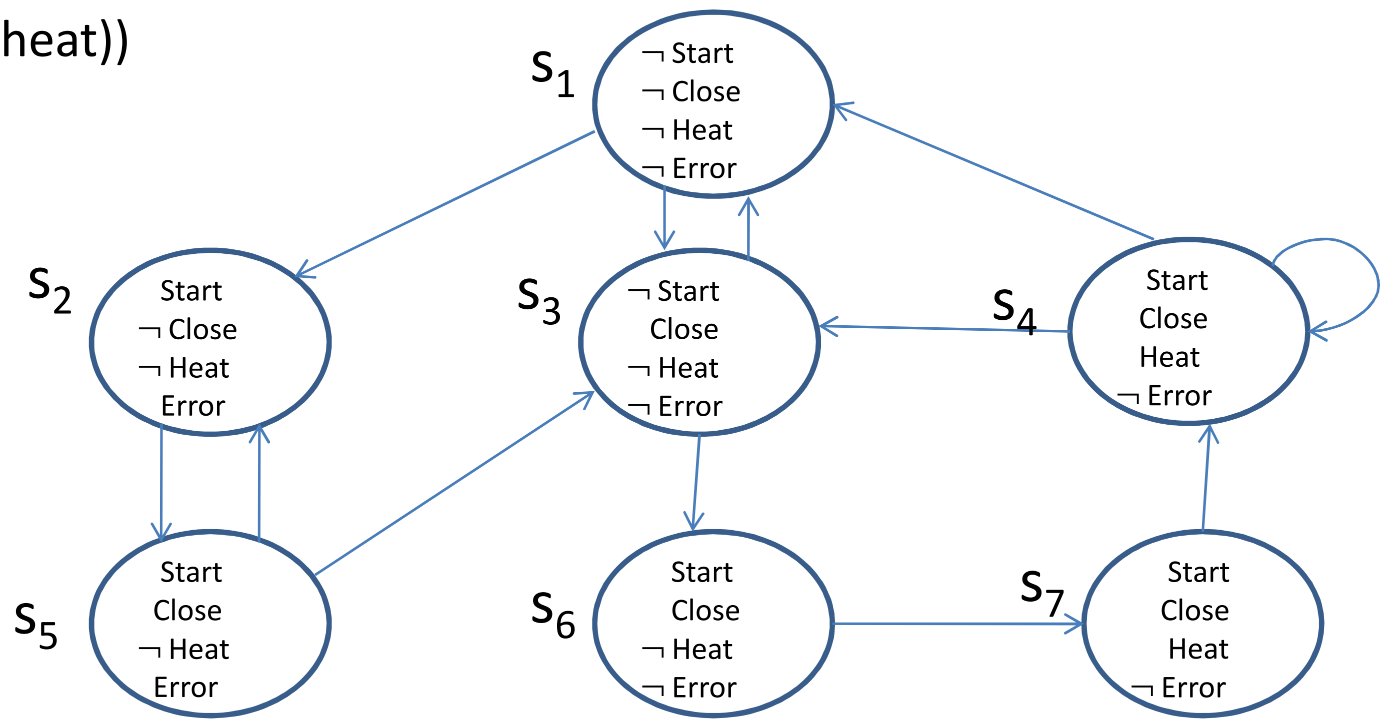


(¬ close ∧ heat)

S1={}

Example

AG (¬ (¬ close ∧ heat))

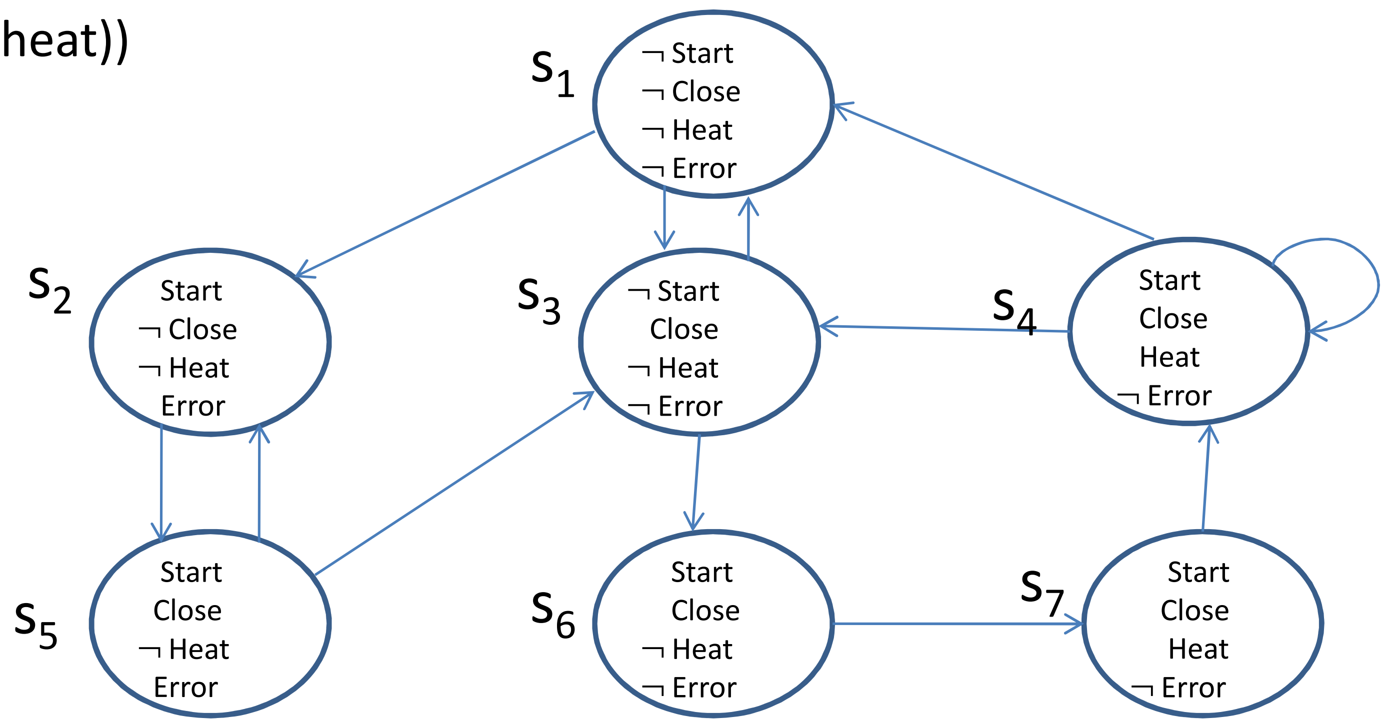


(¬ close ∧ heat) S1={}

¬(¬ close ∧ heat) S2=S

Example

AG (¬ (¬ close ∧ heat))



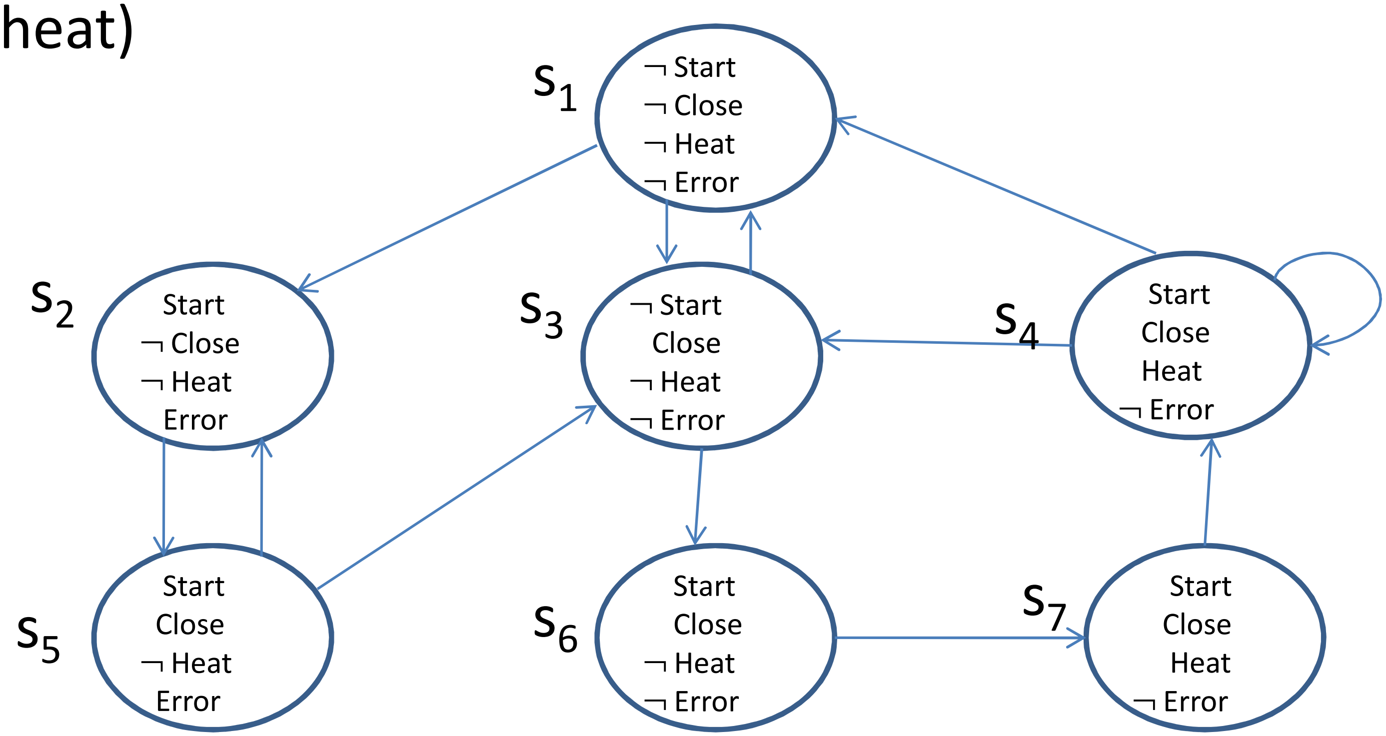
(¬ close ∧ heat) S1={}

¬(¬ close ∧ heat) S2=S

AG (¬ (¬ close ∧ heat)) S2=S

Example

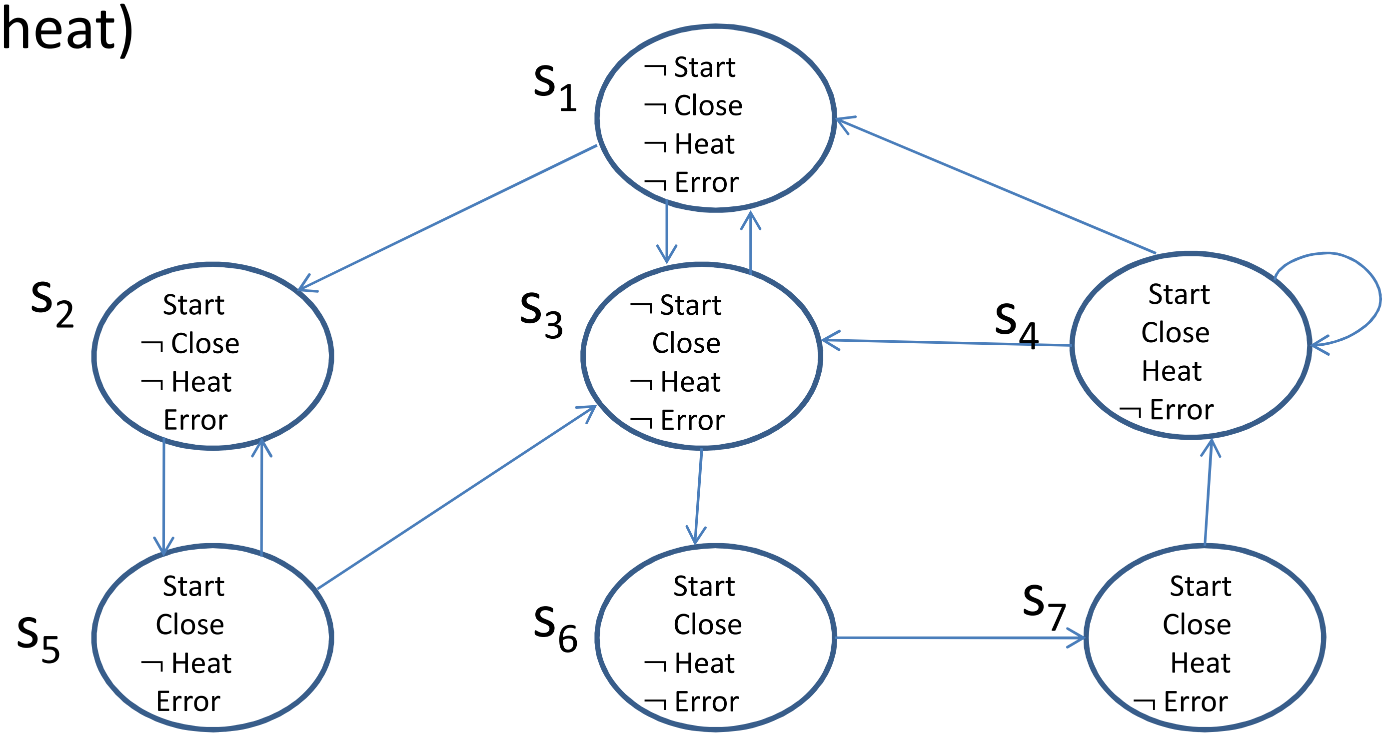
$AG(start \rightarrow AF\ heat)$



(heat) $S1=\{s_4, s_7\}$

Example

AG(start $\rightarrow$ AF heat)



heat $\quad$ S1={s_4, s_7 }

AF heat

Examples

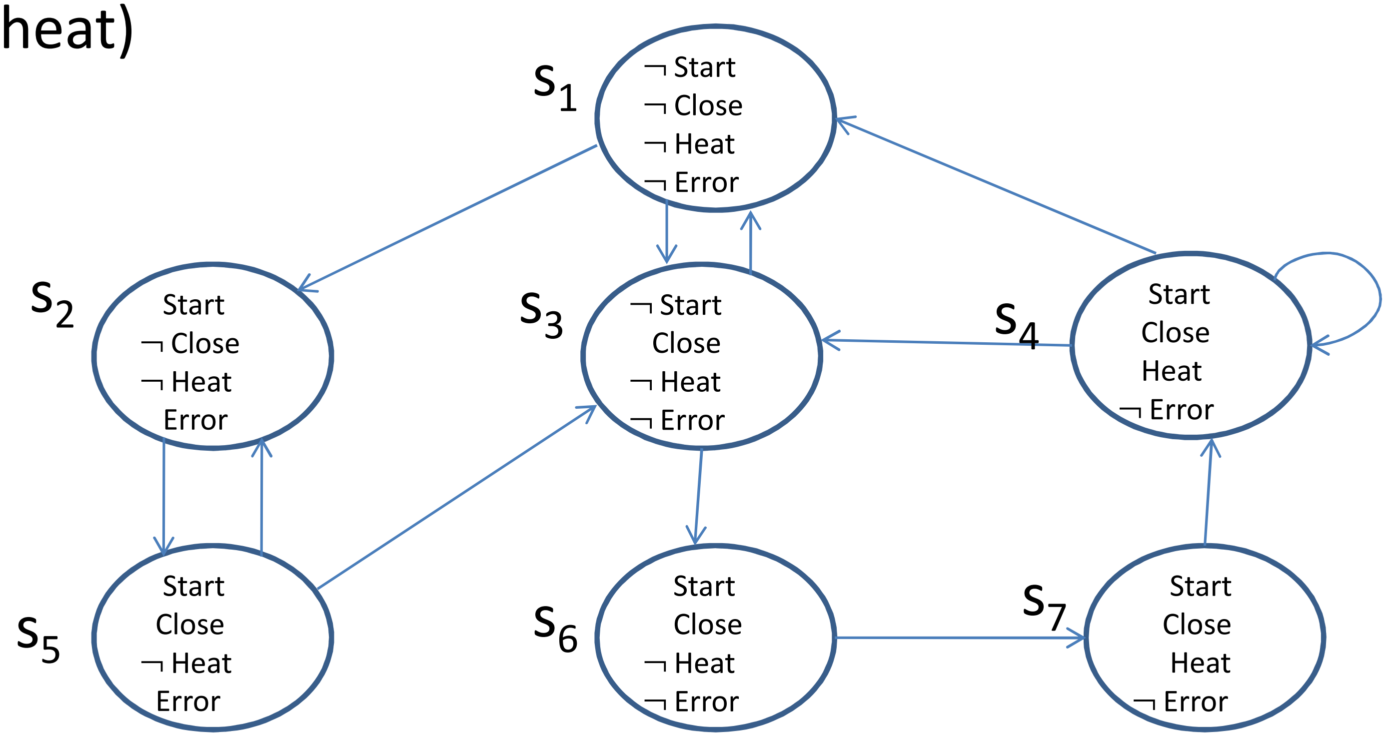
Temporal Operator:

$AF\ c_1$

- If any state s is labeled with c_1, label it with $AF\ c_1$
- Repeat: label any state with $AF\ c_1$ if all successor states are labeled with $AF\ c_1$ until there is no change.

Example

AG(start $\rightarrow$ AF heat)

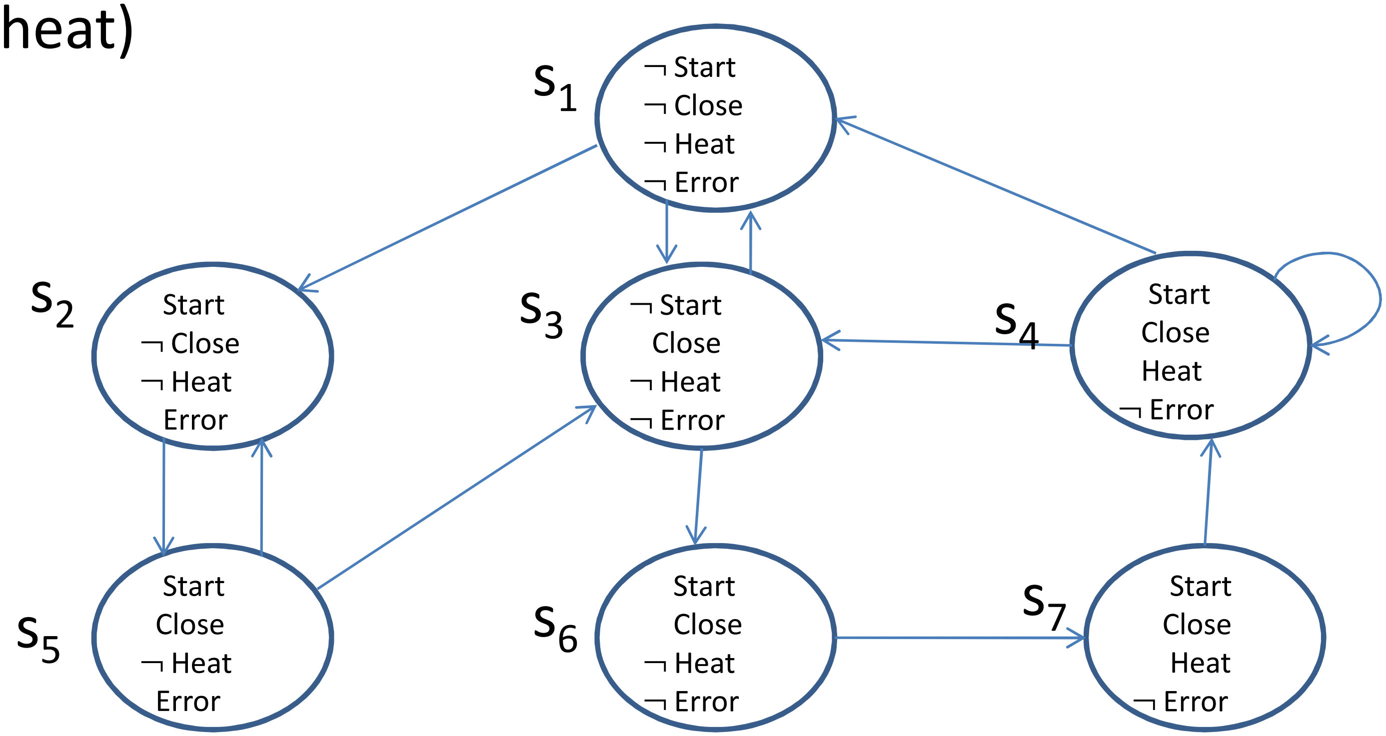


heat $S1=\{s_4, s_7\}$

AF heat $S1=\{s_4, s_7\}$

Example

AG(start $\rightarrow$ AF heat)

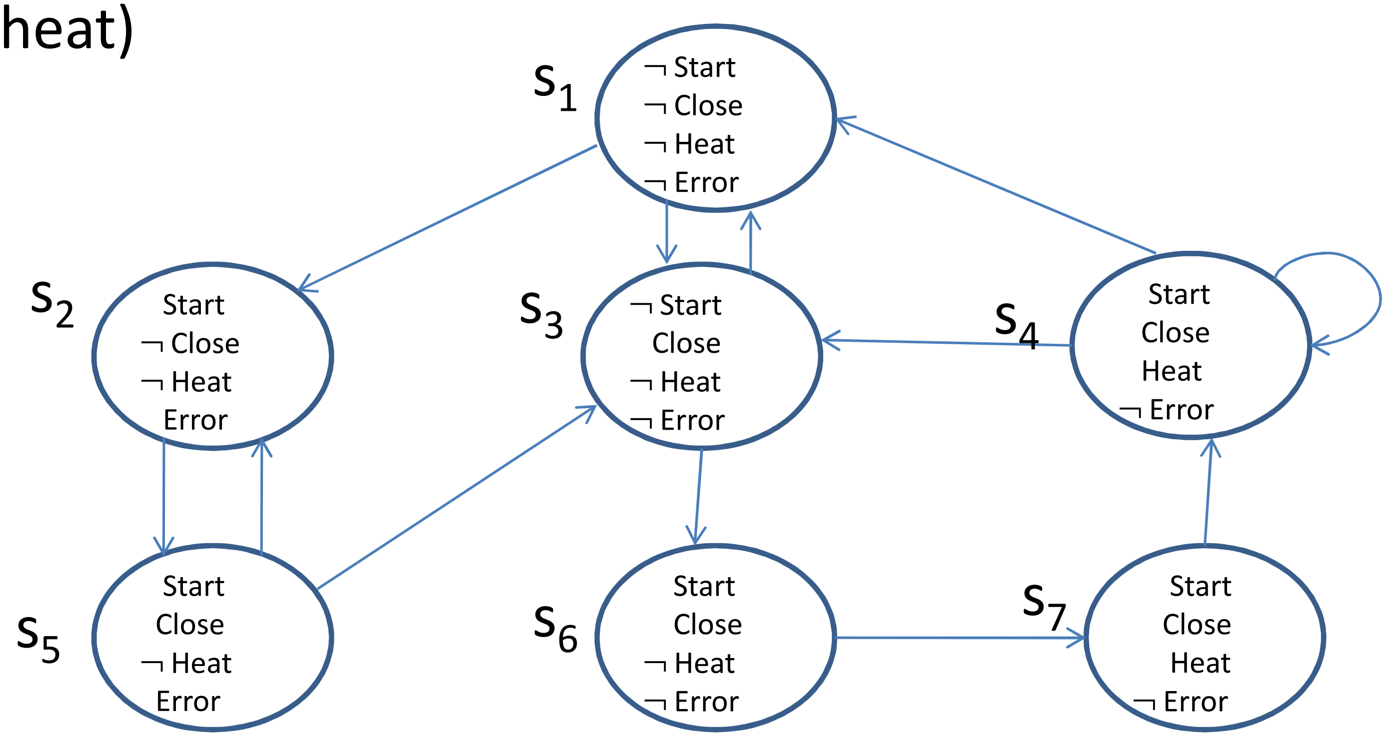


heat $\quad S1=\{s_4, s_7\}$

AF heat $\quad S1=\{s_4, s_7\} \quad S2=\{s_4, s_7, s_6\}$

Example

AG(start $\rightarrow$ AF heat)

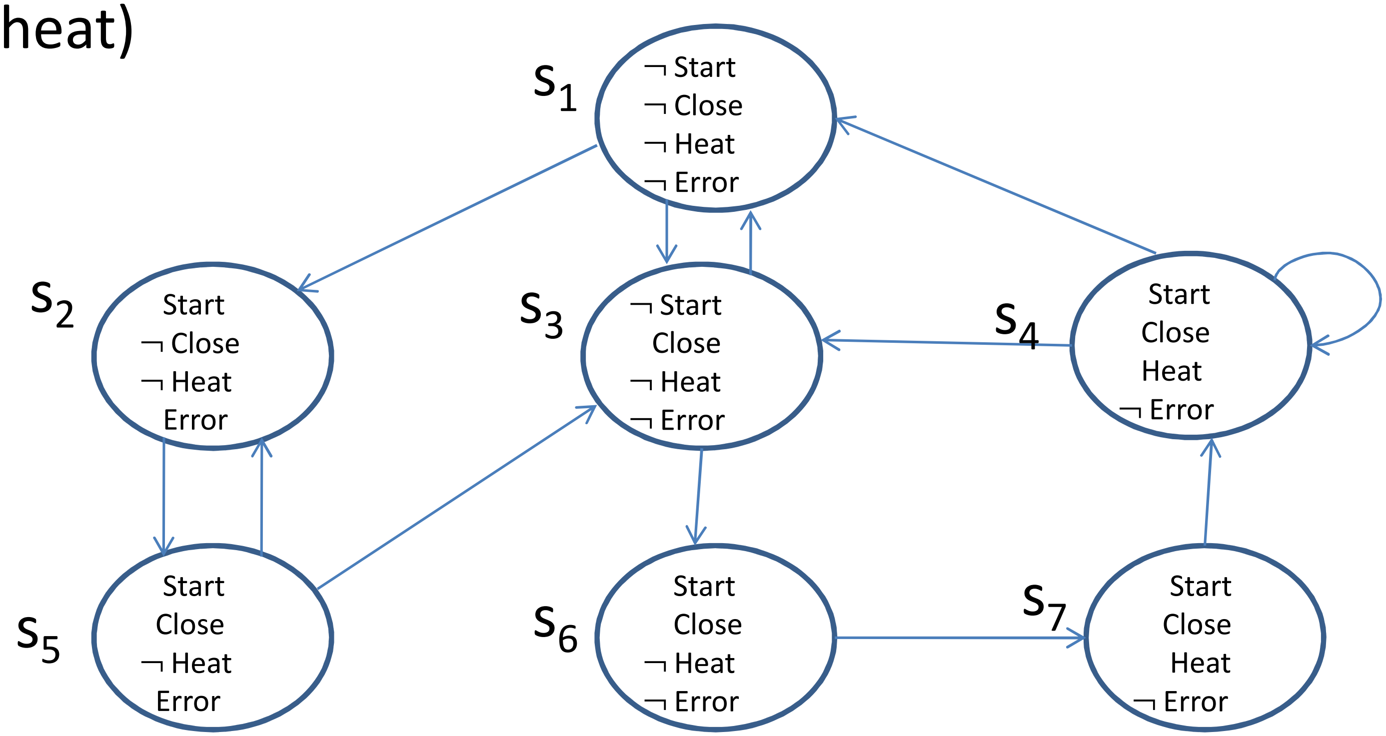


heat $S1=\{s_4, s_7\}$

AF heat $S1=\{s_4, s_7\}$ $S2=\{s_4, s_7, s_6\}$ $S3=\{s_4, s_7, s_6, s_3\}$

Example

AG(start $\rightarrow$ AF heat)



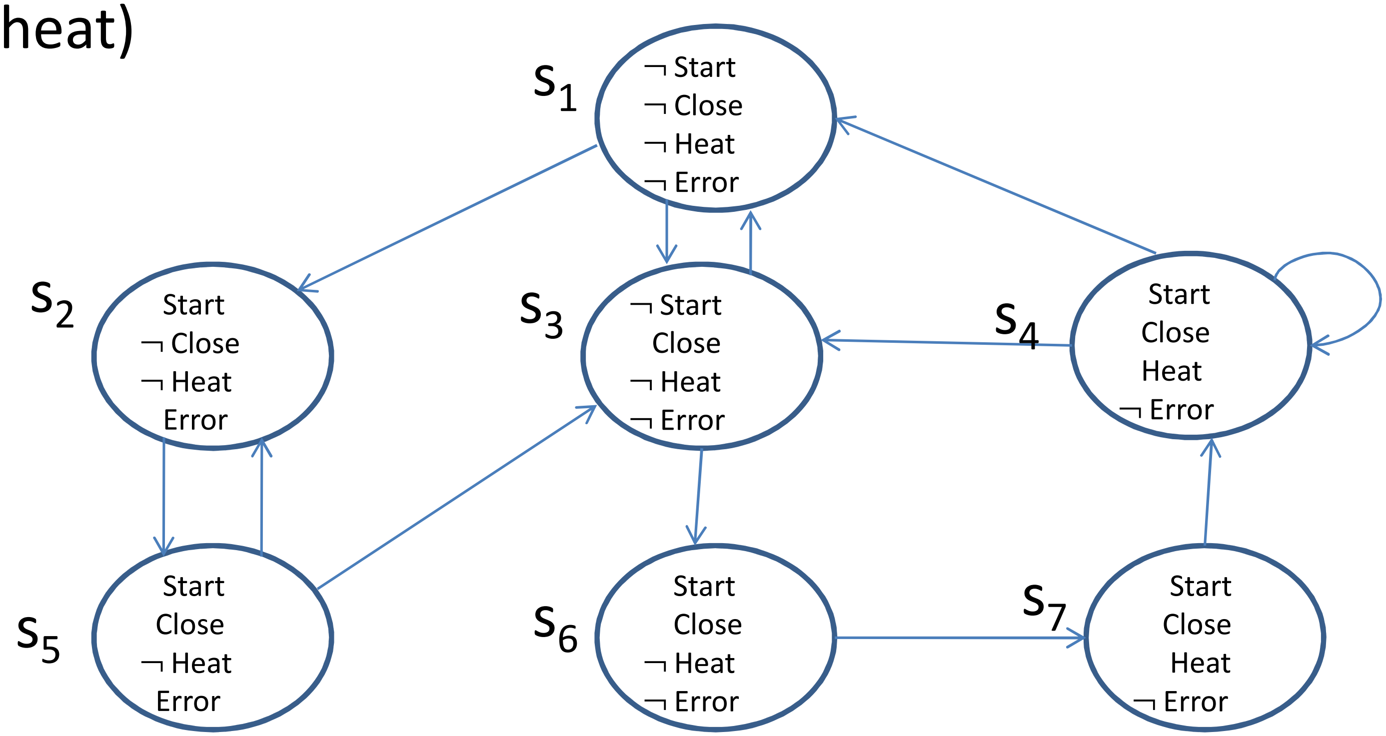
heat $S1=\{s_4, s_7\}$

AF heat $S1=\{s_4, s_7\}$ $S2=\{s_4, s_7, s_6\}$ $S3=\{s_4, s_7, s_6, s_3\}$

$S4=\{s_4, s_7, s_6, s_3\}$

Example

AG(start $\rightarrow$ AF heat)



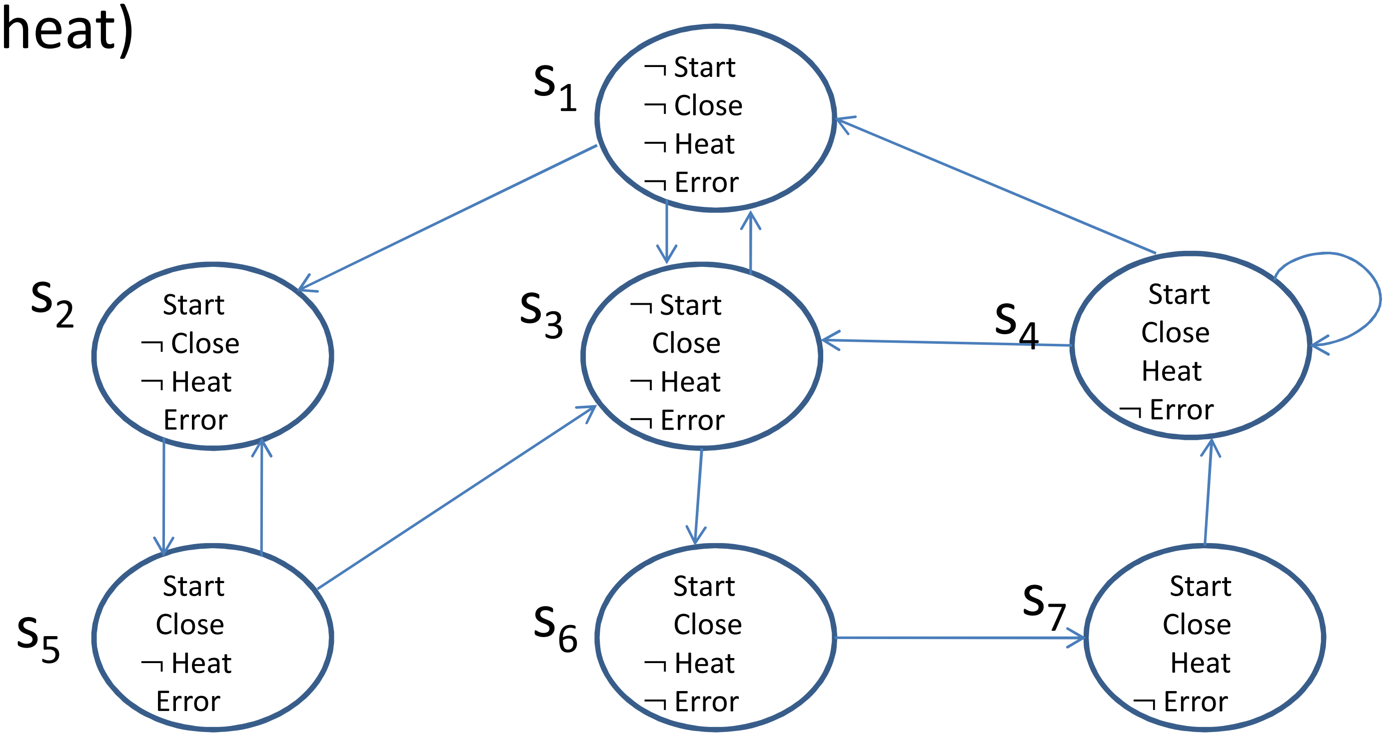
heat $S1=\{s_4, s_7\}$

AF heat $S2=\{s_4, s_7, s_6, s_3\}$

start $S3=\{s_4, s_7, s_6, s_2, s_5\}$

Example

AG(start $\rightarrow$ AF heat)



heat $S1=\{s_4, s_7\}$

AF heat $S2=\{s_4, s_7, s_6, s_3\}$

start $S3=\{s_4, s_7, s_6, s_2, s_5\}$

(start $\rightarrow$ AF heat) $S4=\{s_4, s_7, s_6, s_3, s_1\}$

Labeling algorithm for AGp

AG(start $\rightarrow$ AF heat)

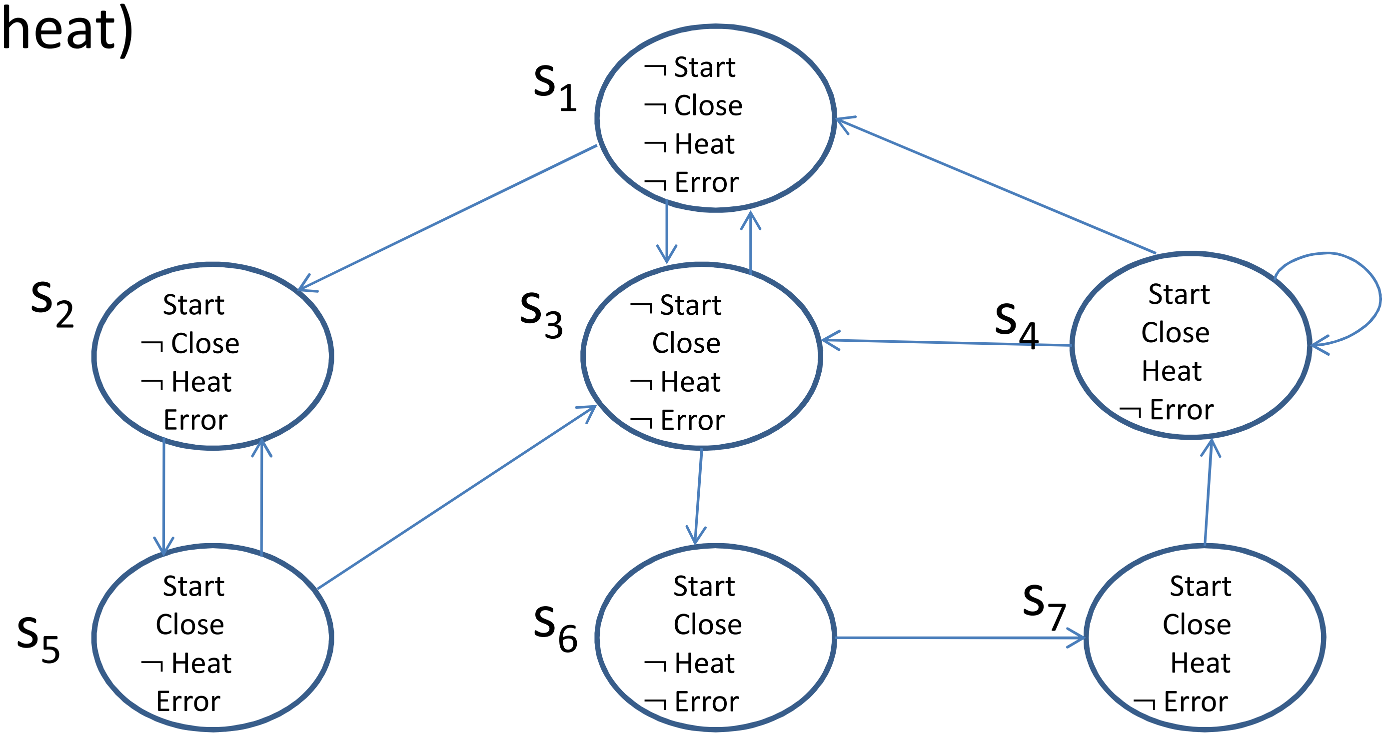
Step1: Label all the states with AGp.

Step2: If any state s is not labeled with p, delete the label AGp.

Step3: Repeat: delete the label AGp from any state if all of its successors are not labeled with AGp until there is no change.

Example

AG(start $\rightarrow$ AF heat)



heat $S1=\{s_4, s_7\}$

AF heat $S2=\{s_4, s_7, s_6, s_3\}$

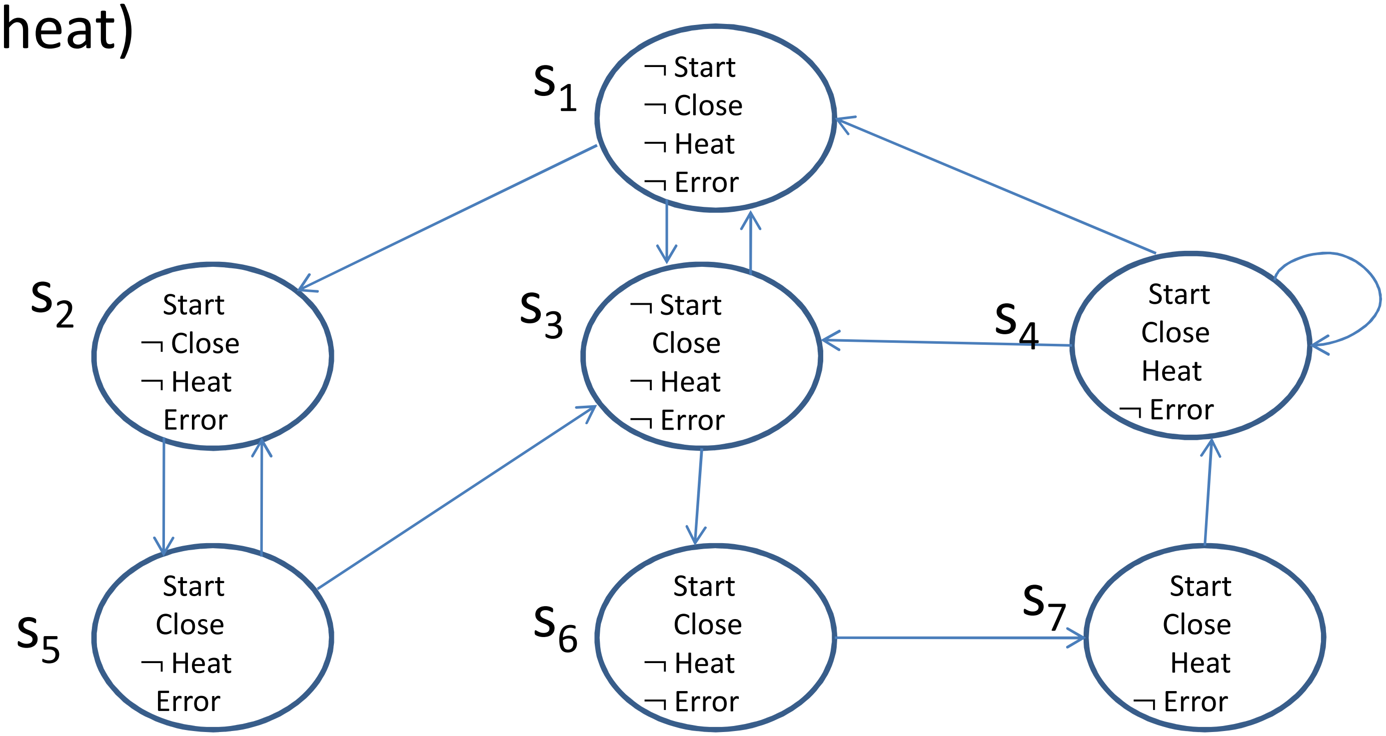
start $S3=\{s_4, s_7, s_6, s_2, s_5\}$

(start $\rightarrow$ AF heat) $S4=\{s_4, s_7, s_6, s_3, s_1\}$

AG(start $\rightarrow$ AF heat) $S5=\{s_4, s_7, s_6, s_3, s_1\}$

Example

AG(start $\rightarrow$ AF heat)



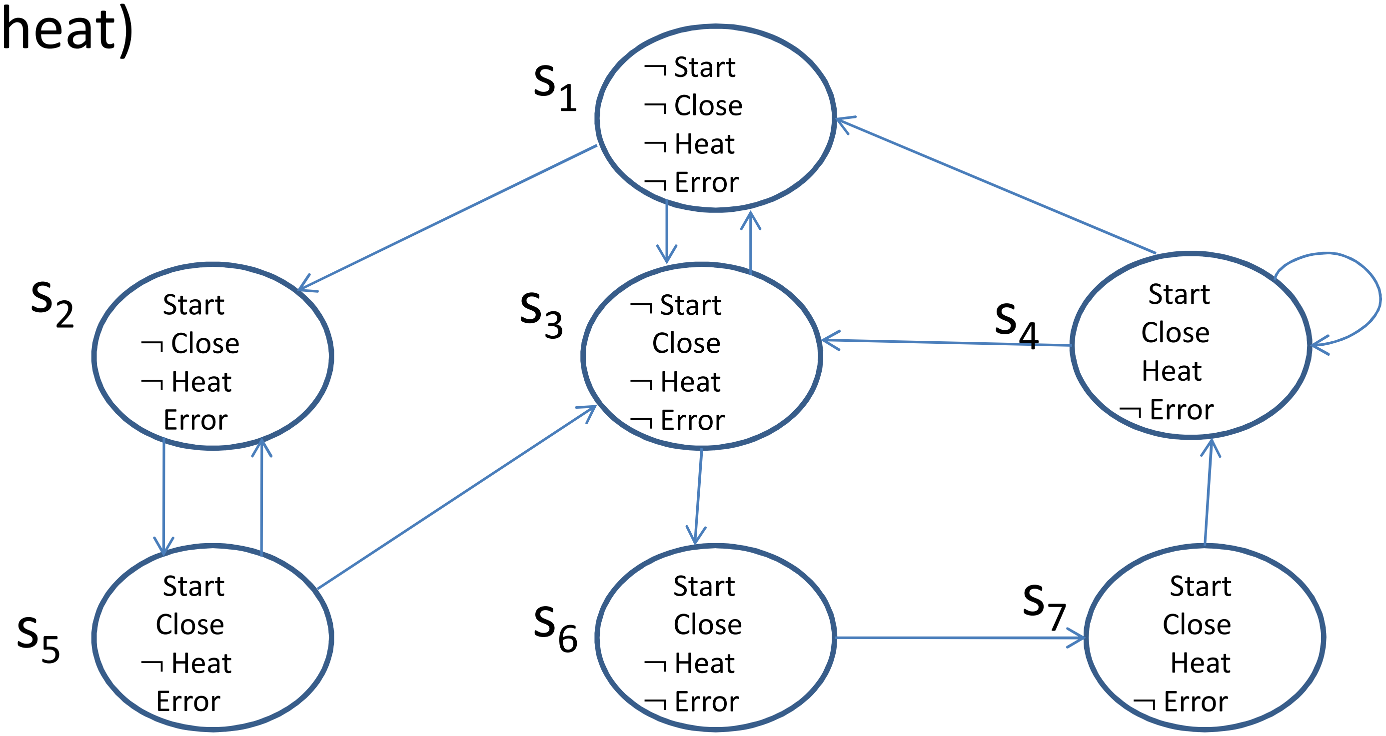
AG(start $\rightarrow$ AF heat)

$S5=\{s_4, s_7, s_6, s_3, s_1\}$

$S6=\{s_4, s_7, s_6, s_3\}$

Example

AG(start $\rightarrow$ AF heat)



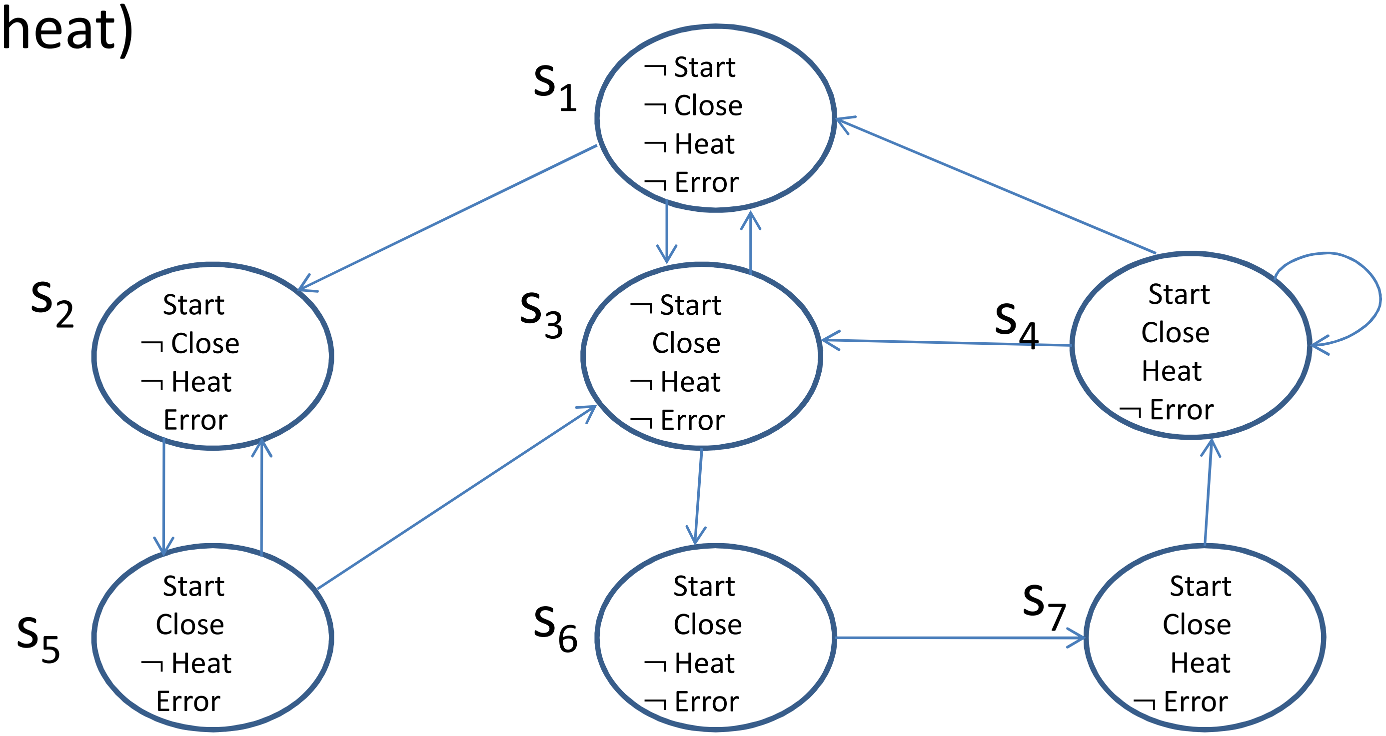
AG(start $\rightarrow$ AF heat)

$S6=\{s_4, s_7, s_6, s_3\}$

$S7=\{s_4, s_7, s_6\}$

Example

AG(start $\rightarrow$ AF heat)



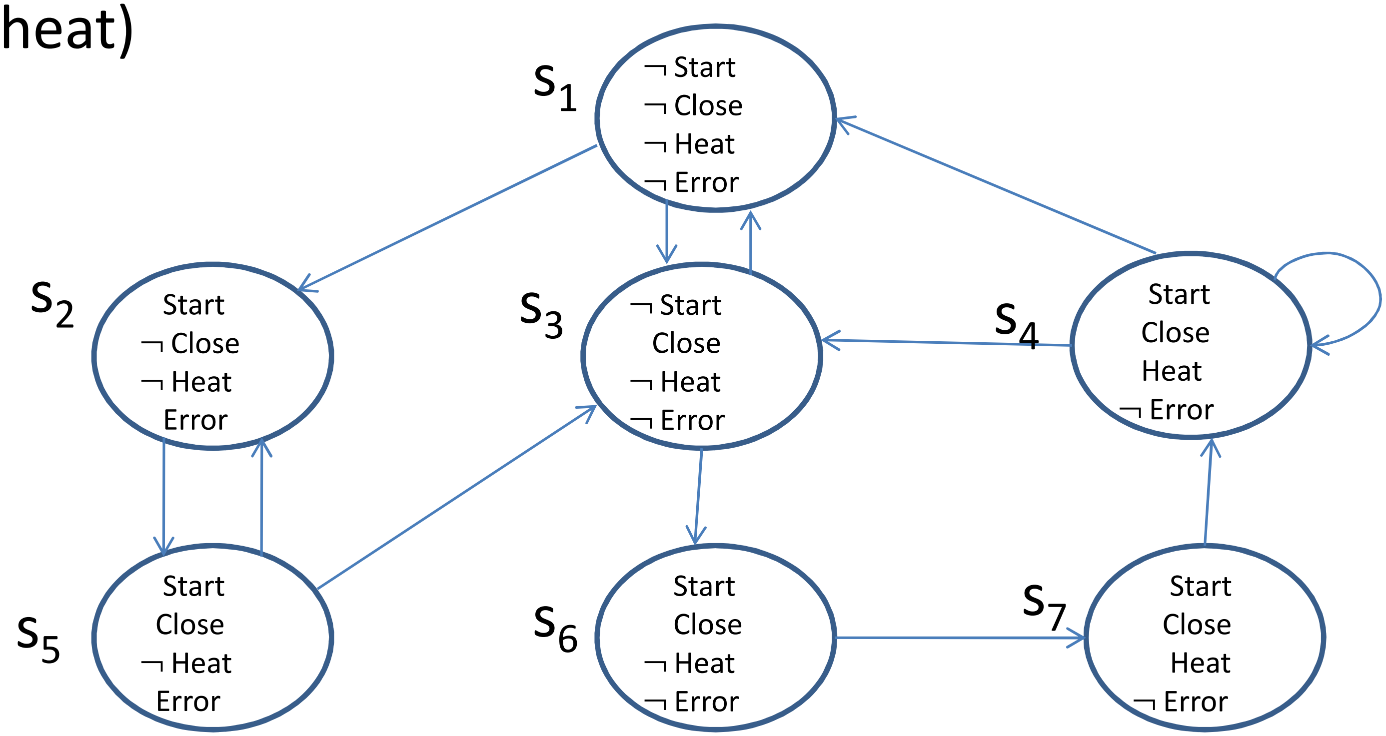
AG(start $\rightarrow$ AF heat)

S7={s_4, s_7 , s_6 }

S8={ s_7 , s_6 }

Example

AG(start $\rightarrow$ AF heat)



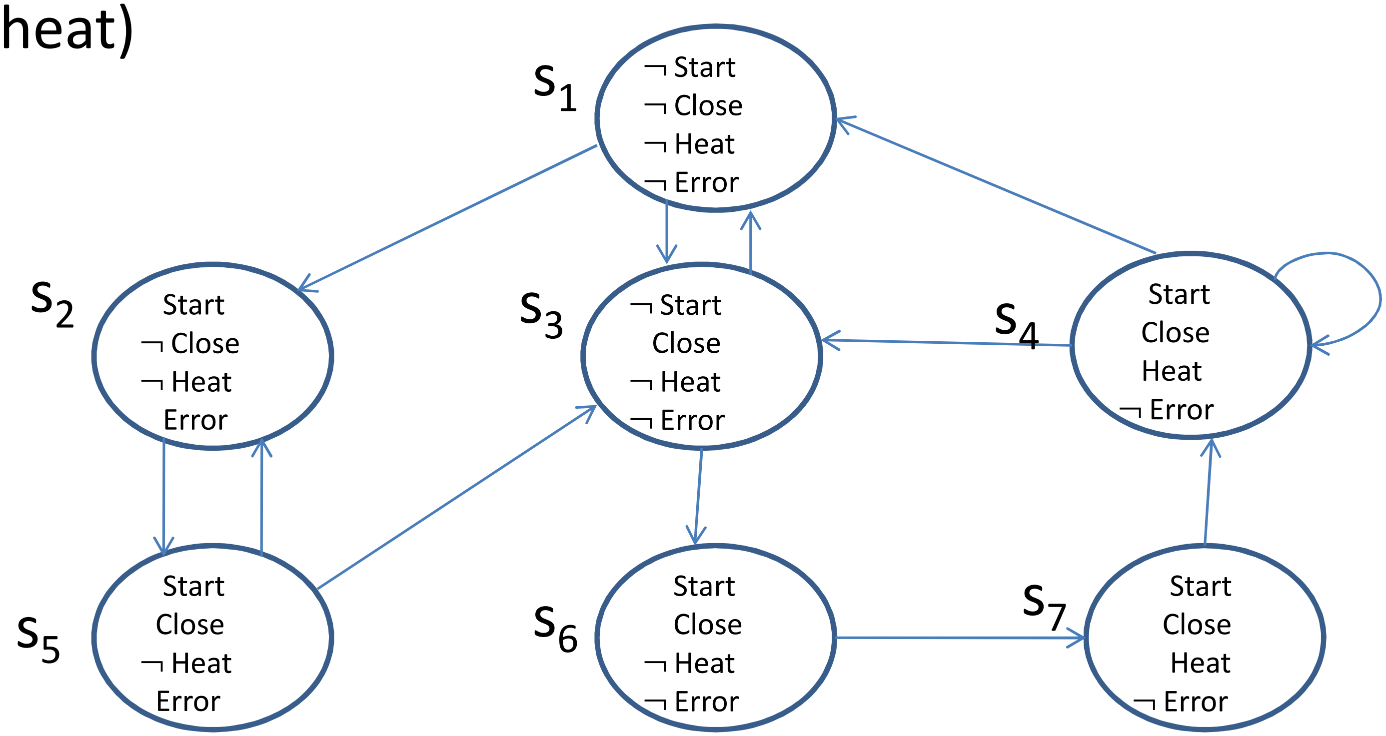
AG(start $\rightarrow$ AF heat)

S8={ s_7 , s_6 }

S9={ s_6 }

Example

$AG(start \rightarrow AF\ heat)$



$AG(start \rightarrow AF\ heat)$

$S9=\{ s_6 \}$

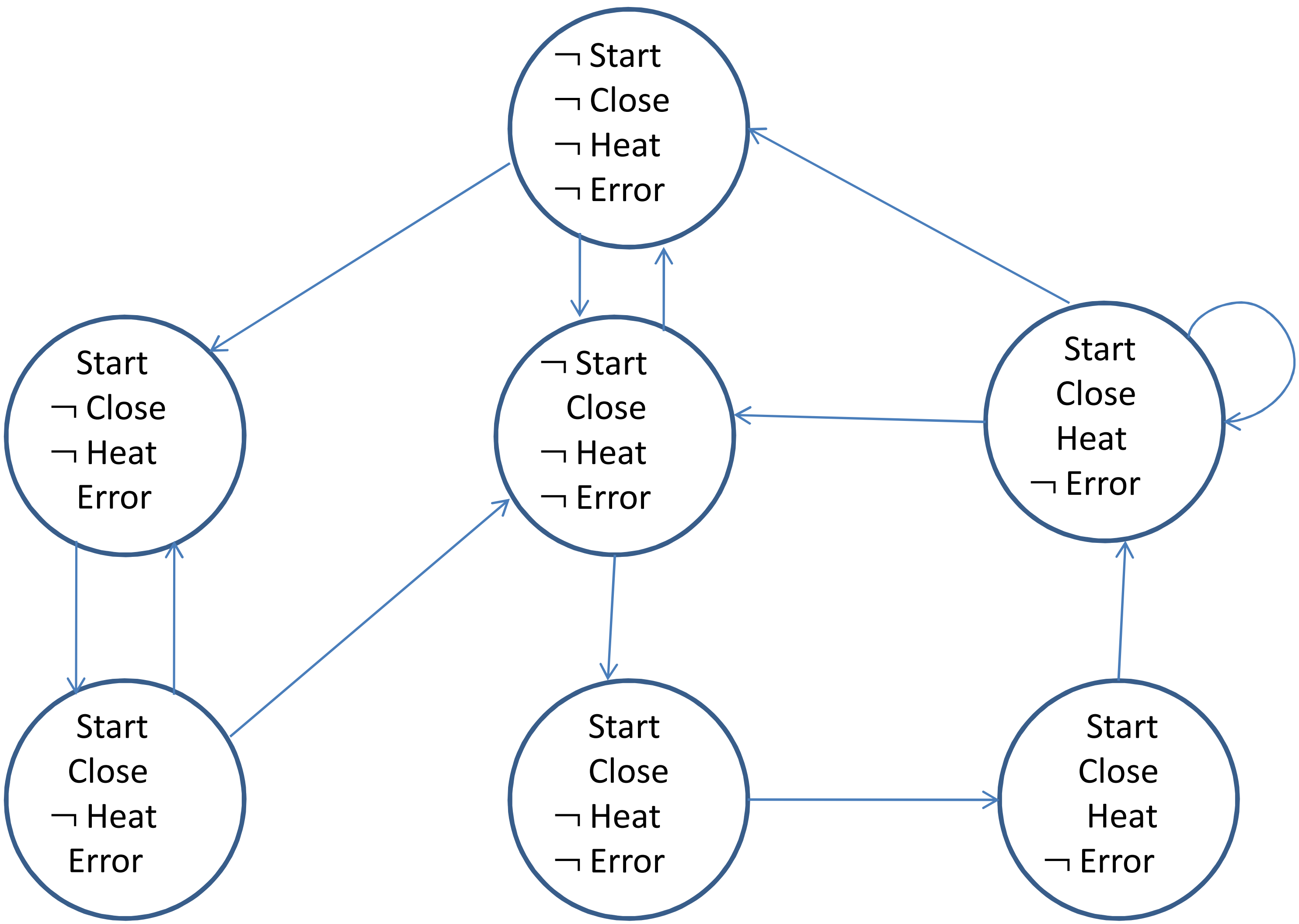
$S10=\{ \}$

- The given specification is not true
  - AG(start $\rightarrow$ AF heat)
- What to do
  - Revisit the design
  - Look for correct sequence of operation

Model Checking with fairness

- The verification of $M, s \models \phi$ might fail because the model M may contain unrealistic behavior.

Example



¬ Start
¬ Close
¬ Heat
¬ Error
Start
¬ Close
¬ Heat
Error
¬ Start
Close
¬ Heat
¬ Error
Start
Close
Heat
¬ Error
Start
Close
¬ Heat
Error
Start
Close
¬ Heat
¬ Error
Start
Close
Heat
¬ Error

Model Checking with fairness

- It may sometimes be better to stick to the original model and to impose a filter on the model check.

Model Checking with fairness

- We verify $M, s \models \psi \rightarrow \phi$, where ψ encodes the refinement of our model expressed as a specification.

Model Checking with fairness

- We verify M, s $\models \psi \rightarrow \phi$, where ψ encodes the refinement of our model expressed as a specification.

- If ψ is true infinitely often, then ϕ is also true infinitely often.

Model Checking with fairness

- Let C = $\{\psi_1, \psi_2, \ldots., \psi_n\}$ be a set of n fairness constraints.
- A computation path $s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow$ … is fair with respect to these fairness constraints if for each i there are infinitely many j such that $s_j \models \psi_i$.

Model Checking with fairness

- Let $C = \{\psi_1, \psi_2, \ldots., \psi_n\}$ be a set of n fairness constraints.
- A computation path $s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots$ is fair with respect to these fairness constraints if for each i there are infinitely many j such that $s_j \models \psi_i$.
- We write A_C and E_C for the path quantifier A and E restricted to fair paths.

Model Checking with fairness

- We write A_C and E_C for the path quantifier A and E restricted to fair paths.

- $M, s_0 \models A_C G\phi$ iff ϕ is true in every state along all fair paths.
- Similarly $A_C F$, $E_C U$, etc.

Model Checking with fairness

- A computation path is fair iff any suffix of it is fair.

Model Checking with fairness

- A computation path is fair iff any suffix of it is fair.

- $E_C[\ \phi\ U\ \psi] \equiv E[\phi\ U\ (\psi \wedge E_C G\ T)]$
- $E_C X\ \phi \equiv\ EX(\ \phi \wedge E_C G\ T)$

Model Checking with fairness

Procedure for EGϕ

- Restrict the graph to state satisfying ϕ.
- Find the strongly connected components (SCC) of the restricted graph.
- Use backward breadth-first searching to find the states on the restricted graph that can reach a SCC.

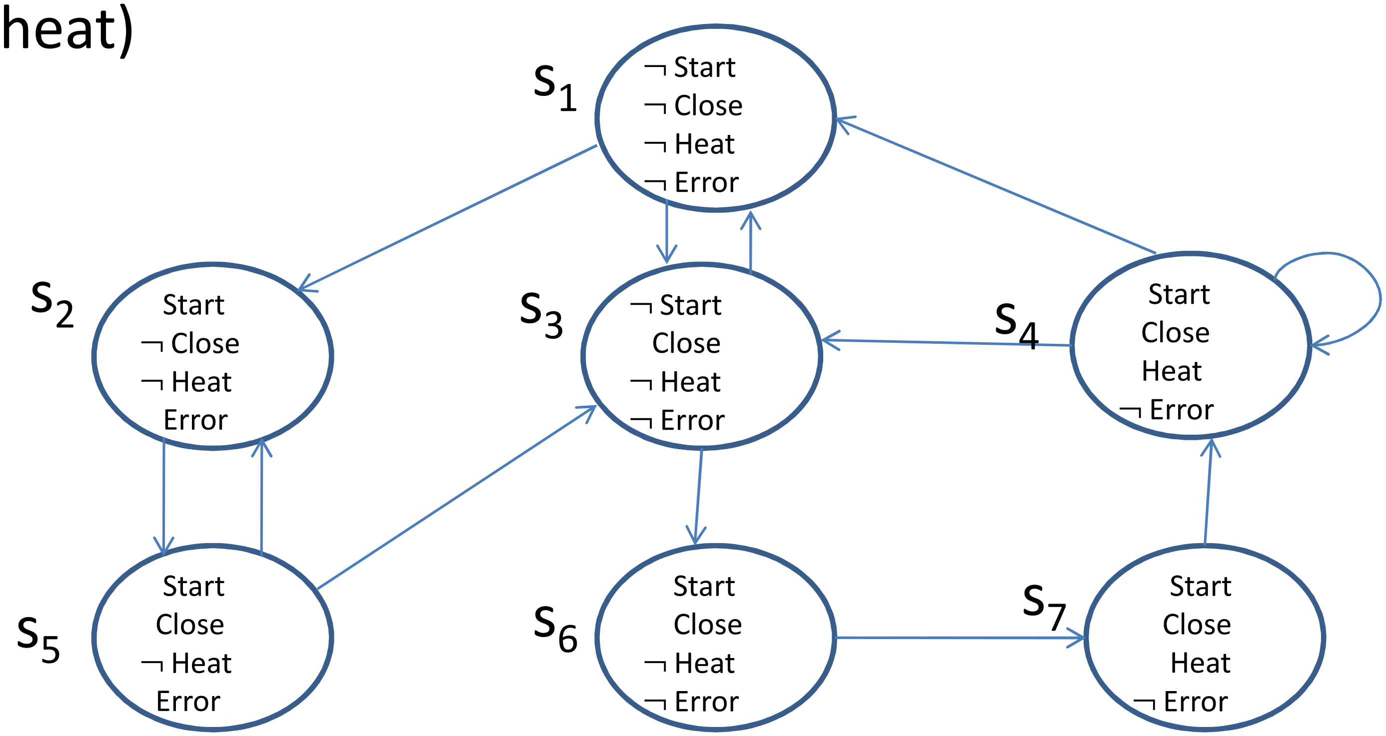
Model Checking with fairness

Procedure for $E_C G\phi$

- Restrict the graph to state satisfying ϕ.
- Find the strongly connected components (SCC) of the restricted graph.
- Remove an SCC if, for some ψ_i, it does not contain a state satisfying ψ_i. The resulting SCCs are fair SCCs.
- Use backward breadth-first searching to find the states on the restricted graph that can reach a fair SCC.

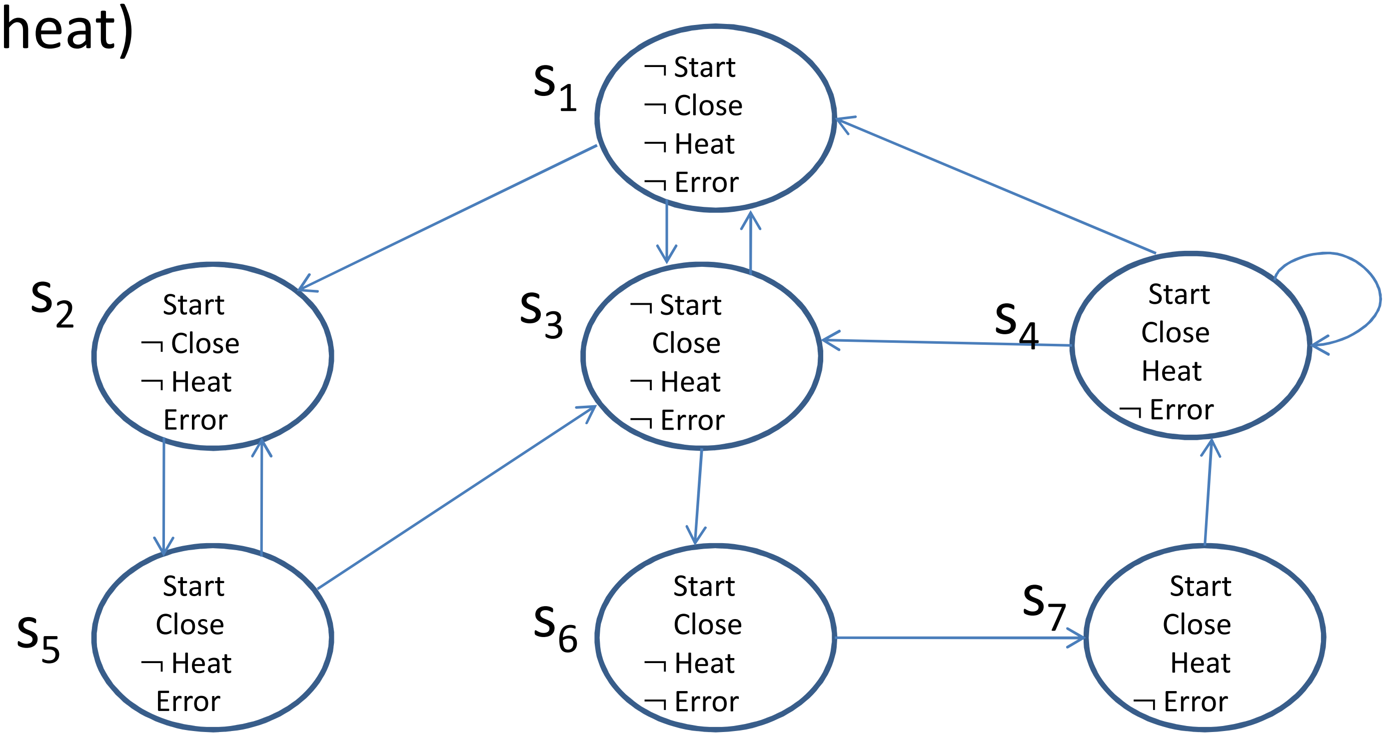
Example

AG(start $\rightarrow$ AF heat)



Example

AG(start $\rightarrow$ AF heat)

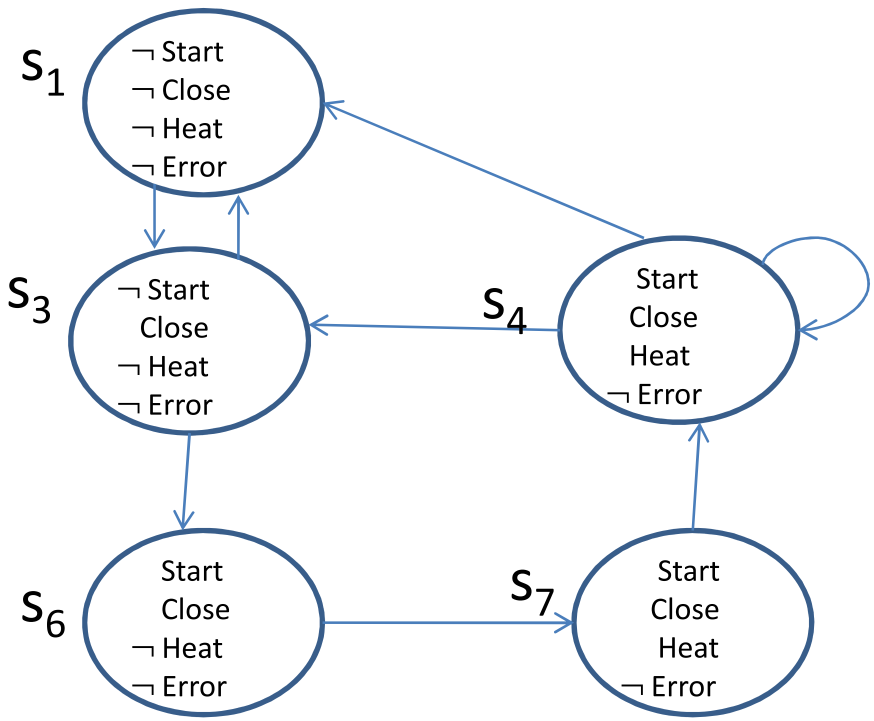


(start $\rightarrow$ AF heat) $\{s_4, s_7, s_6, s_3, s_1\}$

Fairness constraints: {start, close, ¬ error}

Example

AG(start $\rightarrow$ AF heat)



Restrict the graph

(start $\rightarrow$ AF heat) $\{s_4, s_7, s_6, s_3, s_1\}$

Fairness constraints: {start, close, ¬ error}

Question

- Design an elevator controller.

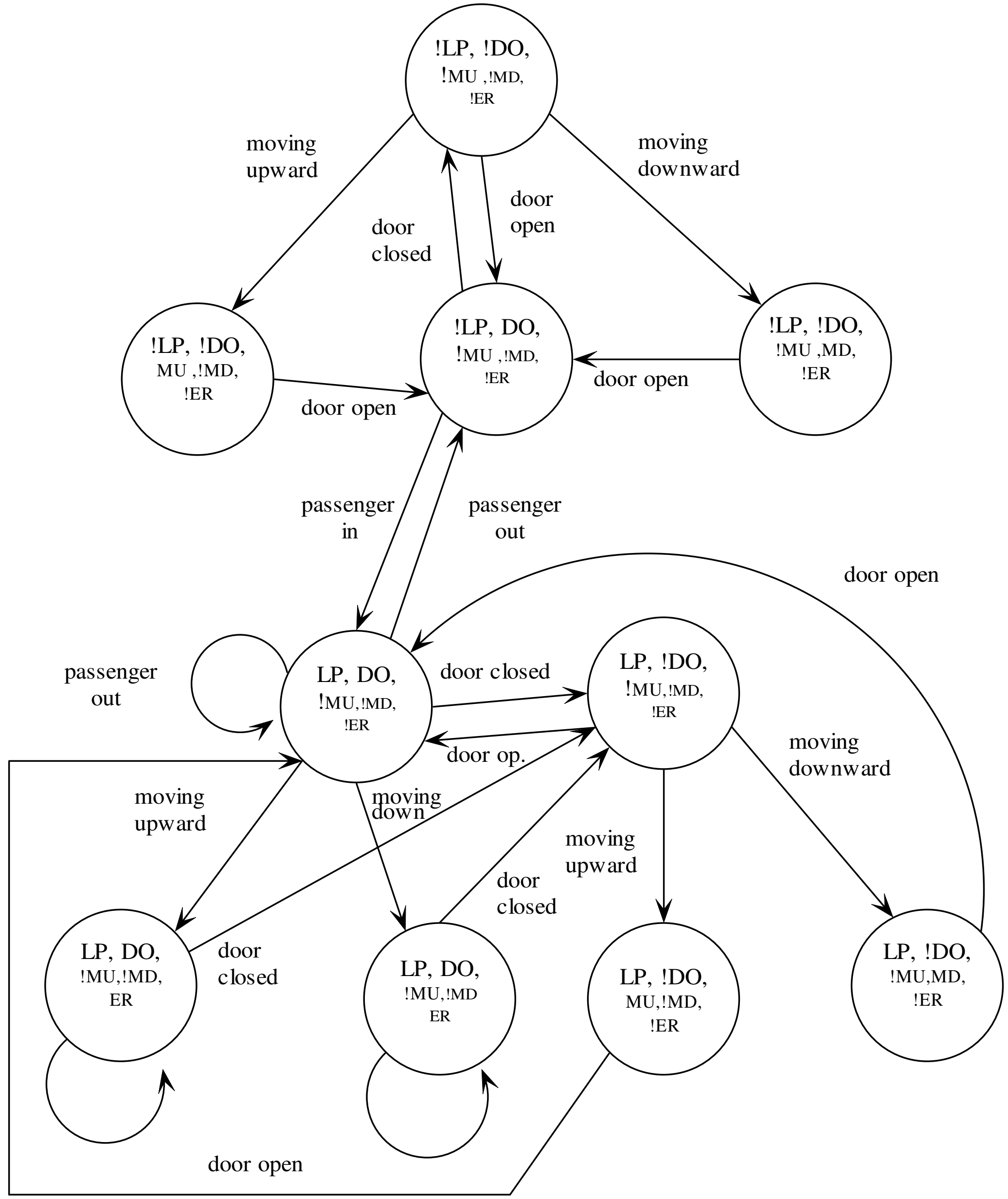
Question

- Design an elevator controller.
  - Abstract model
  - Required control signal

Question

- MU: elevator is moving in the upward direction.
- MD: elevator is moving in the downward direction.
- DO: door is open.
- LP: elevator is loaded with passengers,
- ER: some error occurred.

Question


!LP, !DO, !MU ,!MD, !ER
moving upward
moving downward
door open
door closed
!LP, !DO, MU ,!MD, !ER
!LP, DO, !MU ,!MD, !ER
!LP, !DO, !MU ,MD, !ER
door open
door open
passenger in
passenger out
door open
passenger out
LP, DO, !MU,!MD, !ER
door closed
LP, !DO, !MU,!MD, !ER
door op.
moving downward
moving upward
moving down
moving upward
door closed
door closed
LP, DO, !MU,!MD, ER
LP, DO, !MU,!MD ER
LP, !DO, MU,!MD, !ER
LP, !DO, !MU,MD, !ER
door open

Question

- Specification:
  - The elevator will either move up or move down provided the door is closed.

Question

- Specification:
  - The elevator will either move up or move down provided the door is closed.
  - An upward travelling elevator at the second floor does not change it direction when it has passengers wishing to go to the fifth floor.

Question

- Design the mutual exclusion protocol for n processes.

Question

- Design a controller for Traffic light.
- Mention the property that the traffic light controller should satisfy.

- The “state explosion” problem
  - State space is exponential to the number of state variables.