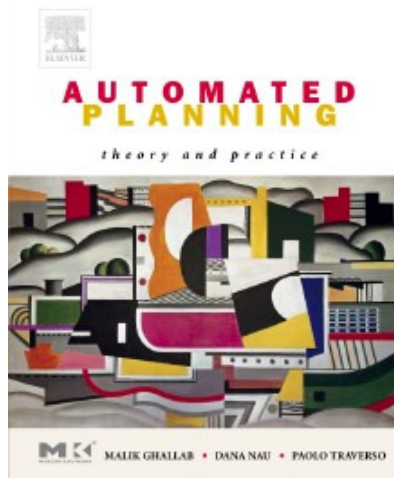


Planning Based on Model Checking

01.06.05 / Planning Systems
by Jan Priegnitz



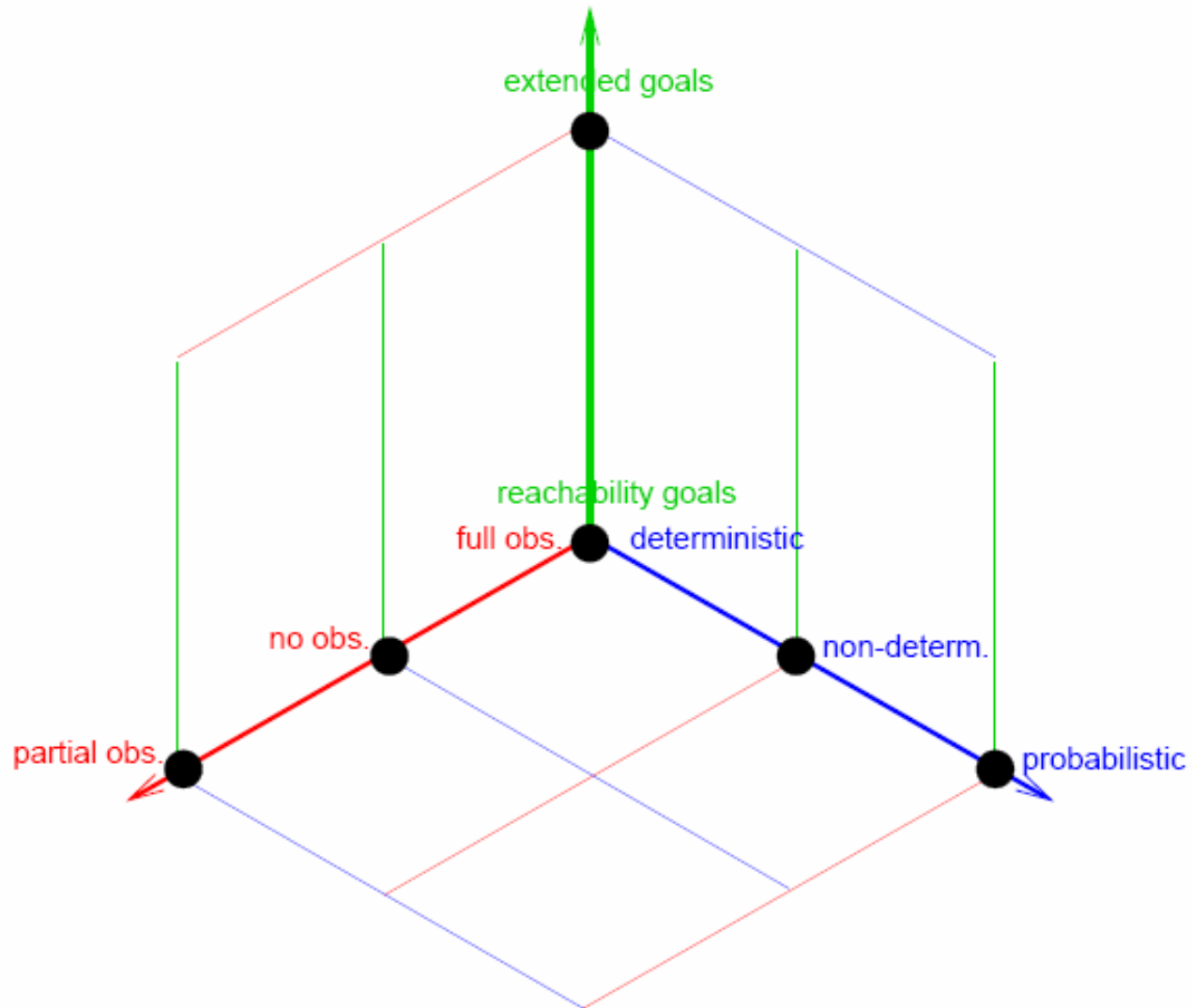
<http://www.laas.fr/planning/nau-lecture-slides.html>



outline

- Dimensions of uncertainty
- Nondeterministic Systems
 - Kinds of solutions and how to find them
- Extended Goals
 - Computation Tree Logic and beyond
 - Symbolic Model Checking (with BDDs)
- Partial Observability

uncertainty



Nondeterministic Systems

are triples $\Sigma = (S, A, \gamma)$ where

- S is a finite set of states
- A is a finite set of actions
- $\gamma: S \times A \rightarrow 2^S$ is the state-transition-function

```
Execute-Policy( $\pi$ )
```

```
  observe the current state  $s$ 
```

```
  while  $s \in S_\pi$  do
```

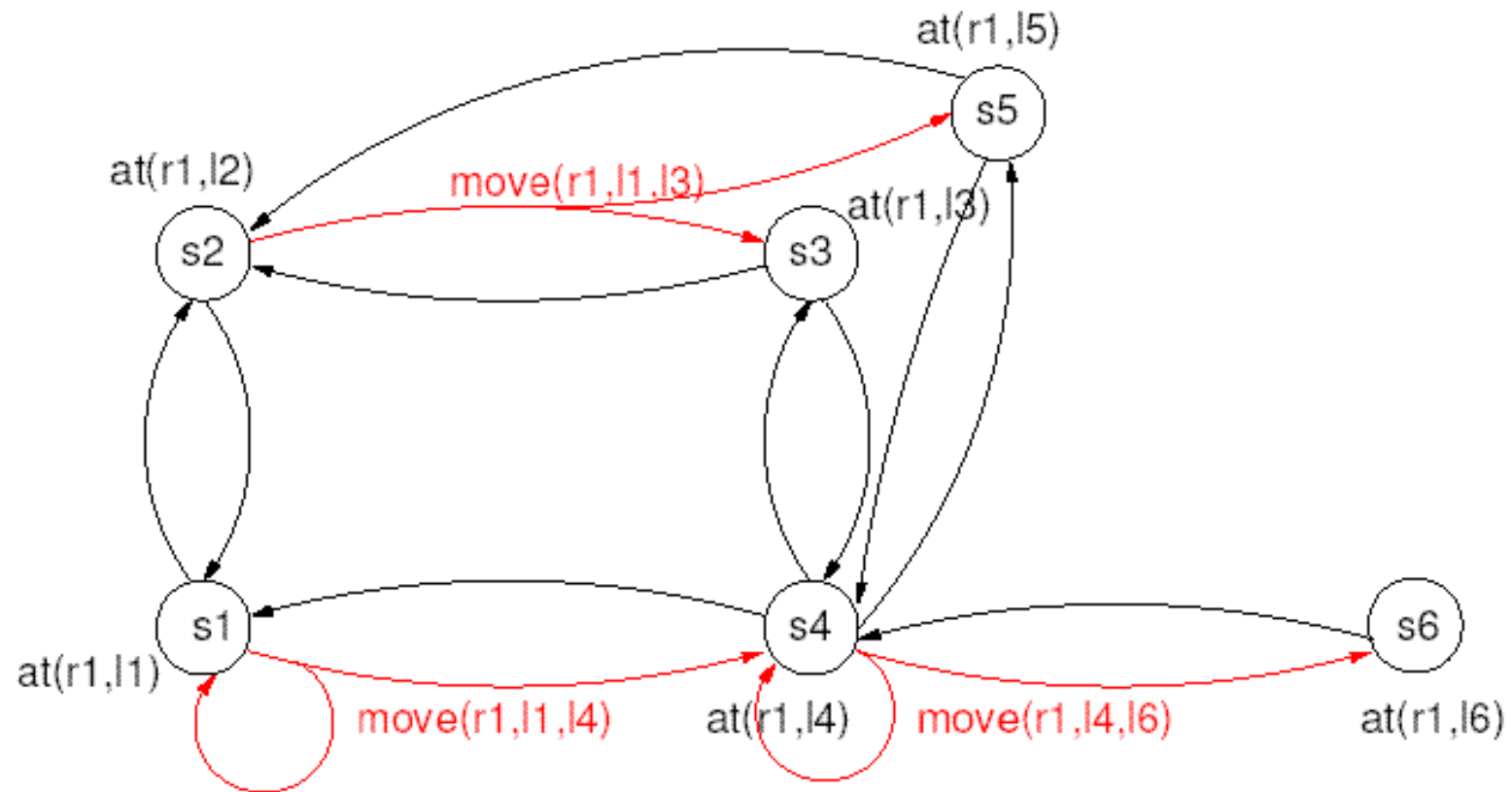
```
    select an action  $a$  such that  $(s, a) \in \pi$ 
```

```
    execute action  $a$ 
```

```
    observe the current state  $s$ 
```

```
end
```

on the blackboard, too ->

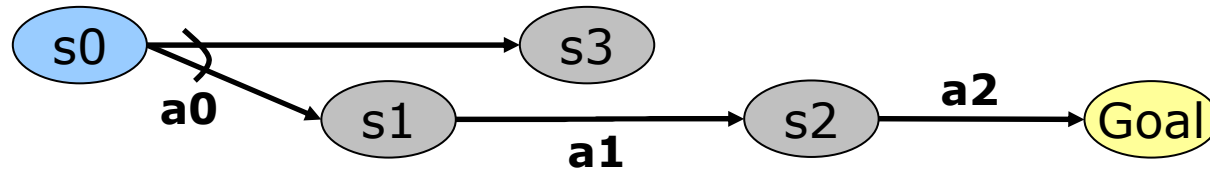


Examples of policies

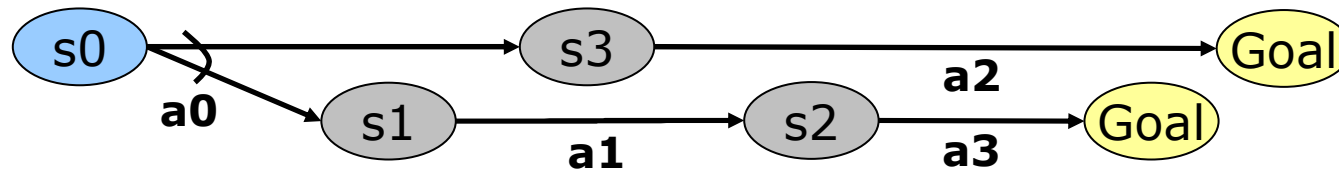
- $\pi_1 = \{(s1, \text{move}(r1,l1,l2)),$
 $(s2, \text{move}(r1,l2,l3)), (s3, \text{move}(r1,l3,l4))\}$
- $\pi_2 = \{(s1, \text{move}(r1,l1,l2)),$
 $(s2, \text{move}(r1,l2,l3)), (s3, \text{move}(r1,l3,l4)),$
 $(s5, \text{move}(r1,l3,l4))\}$
- $\pi_3 = \{(s1, \text{move}(r1,l1,l4))\}$

Types of Solutions

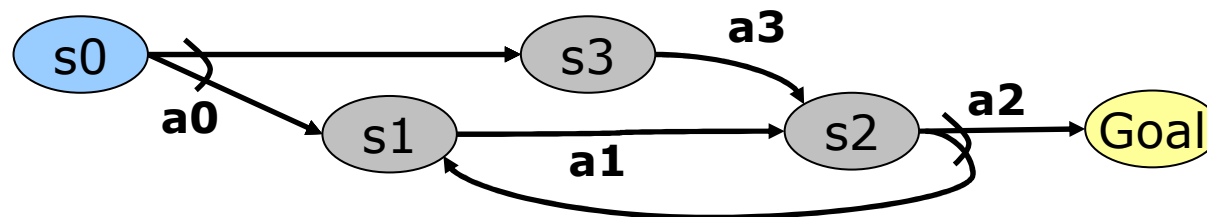
- **Weak solution:** at least one ...



- **Strong solution:** every ...



- **Strong-cyclic solution:** every fair ...



... execution path reaches a goal

Finding Strong and Weak Solutions

Weak-Plan(P)

$\pi \leftarrow \text{failure}; \pi' \leftarrow \emptyset$

While $\pi' \neq \pi$ and $S_0 \not\subseteq (S_g \cup S_{\pi'})$ do

$PreImage \leftarrow \text{WeakPreImg}(S_g \cup S_{\pi'})$

$\pi'' \leftarrow \text{PruneStates}(PreImage, S_g \cup S_{\pi'})$

$\pi \leftarrow \pi'$

$\pi' \leftarrow \pi' \cup \pi''$

if $S_0 \subseteq (S_g \cup S_{\pi'})$ then return($\text{MkDet}(\pi')$)

else return(failure)

end

WeakPreImg(S)

$= \{(s,a) : \gamma(s,a) \cap S \neq \emptyset\}$

PruneStates(π, S)

$= \{(s,a) \in \pi : s \notin S\}$

Finding Strong-Cyclic Solutions

Strong-Cyclic-Plan(S_0, S_g)

$\pi \leftarrow \emptyset; \pi' \leftarrow UnivPol$

while $\pi' \neq \pi$ do

$\pi \leftarrow \pi'$

$\pi' \leftarrow PruneUnconnected(PruneOutGoing(\pi', S_g), S_g)$

if $S_0 \subseteq (S_g \cup S_{\pi'})$

then return($MkDet(RemoveNonProgress(\pi', S_g))$)

else return(failure)

end

Planning for Extended Goals

- *Context*: the internal state of the controller
- *Plan*: $(C, c_0, act, ctxt)$
 - C is a set of execution contexts
 - c_0 is the initial context
 - $act: S \times C \rightarrow A$
 - $ctxt: S \times C \times S \rightarrow C$

Learning by Doing

<i>state</i>	<i>context</i>	<i>action</i>	<i>next state</i>	<i>next context</i>
s1	c0	move(r1,l1,l4)	s1	c1
s1	c0	move(r1,l1,l4)	s4	c4
s4	c0	wait	s4	c0
s1	c1	move(r1,l1,l2)	s2	c2
s2	c2	move(r1,l2,l1)	s1	c2
s1	c2	move(r1,l1,l2)	s2	c2

Let the initial state be s1 and the the goal state s4. Is π a weak, strong, strong cyclic solution?

And, what about the following plan?

<i>state</i>	<i>context</i>	<i>action</i>	<i>next state</i>	<i>next context</i>
s1	c0	move(r1,l1,l4)	s1	c1
s1	c0	move(r1,l1,l4)	s4	c4
s4	c0	wait	s4	c0
s1	c1	move(r1,l1,l2)	s2	c2
s2	c2	move(r1,l2,l1)	s1	c0

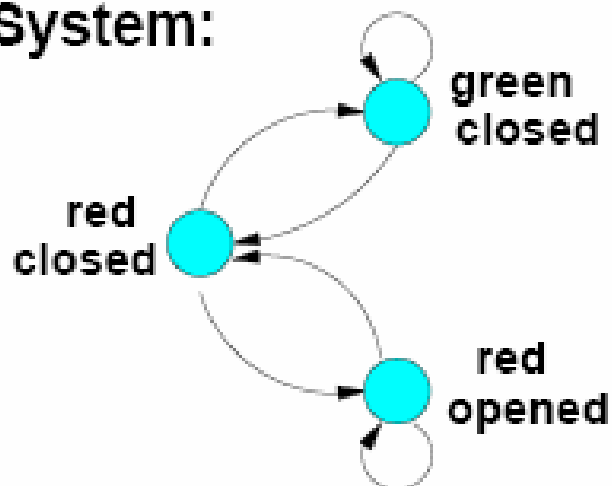
The Model Checker

Properties:

"never green when opened"

"if red, then eventually green"

System:



Model
Checker

yes!

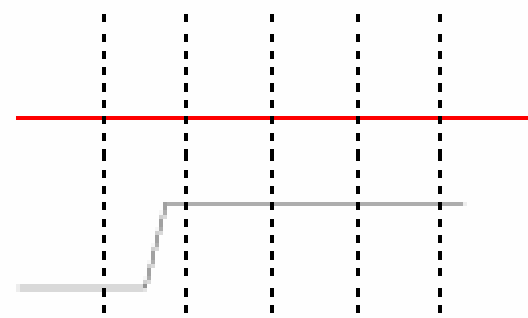
no!

green

red

opened

closed



counterexample

Computation Tree Logic

1. Every atomic proposition $p \in P$ is a CTL formula
2. If p and q are CTL formulas, then so are the following formulas:
 - $\neg p$, $p \vee q$,
 - AXp , EXp ,
 - $A(pUq)$ and $E(pUq)$

Algorithms for CTL

1. $\text{MCHECKEF}(p, K)$
2. $\text{CurrentStates} \leftarrow \emptyset;$
3. $\text{NextStates} \leftarrow \text{STATES}(p, K);$
4. while $\text{NextStates} \neq \text{CurrentStates}$ do
5. if $(S_0 \subseteq \text{NextStates})$
6. then return(True);
7. $\text{CurrentStates} \leftarrow \text{NextStates};$
8. $\text{NextStates} \leftarrow \text{NextStates} \cup \text{PRE-IMG-EF}(\text{NextStates}, K);$
9. return(False);

$\text{PRE IMG EF}(\text{States}, K) = \{s \in S : \exists s'. (s' \in \text{States} \wedge R(s, s'))\}$

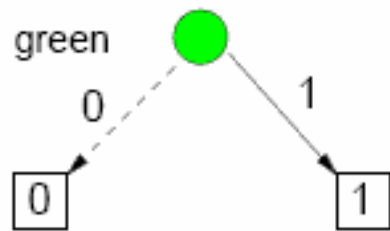
$\text{PRE IMG AF}(\text{States}, K) = \{s \in S : \forall s'. (R(s, s') \rightarrow s' \in \text{States})\}$

Symbolic Model Checking

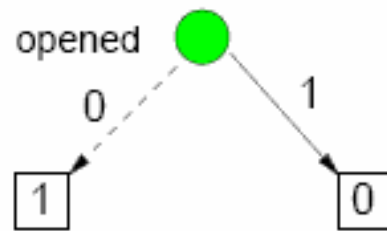
- Extends the Kripke-Structure
- Sets of states are explored
instead of single states
- vector of state variables x

$$\xi(Q) = \bigvee_{s \in Q} \xi(s).$$

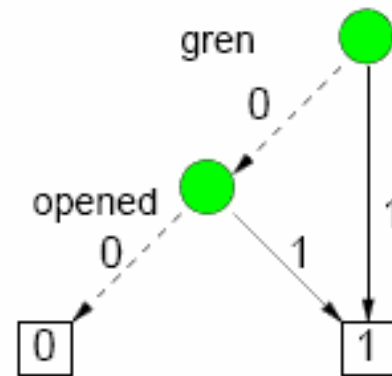
BDD-based Symbolic Model Checking



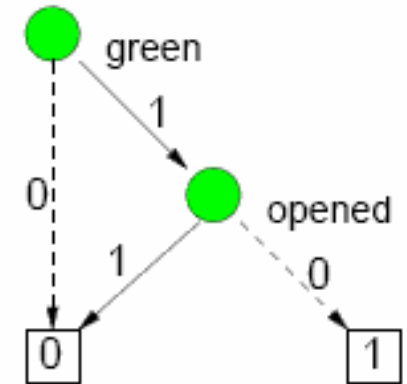
signal is green



door is not opened

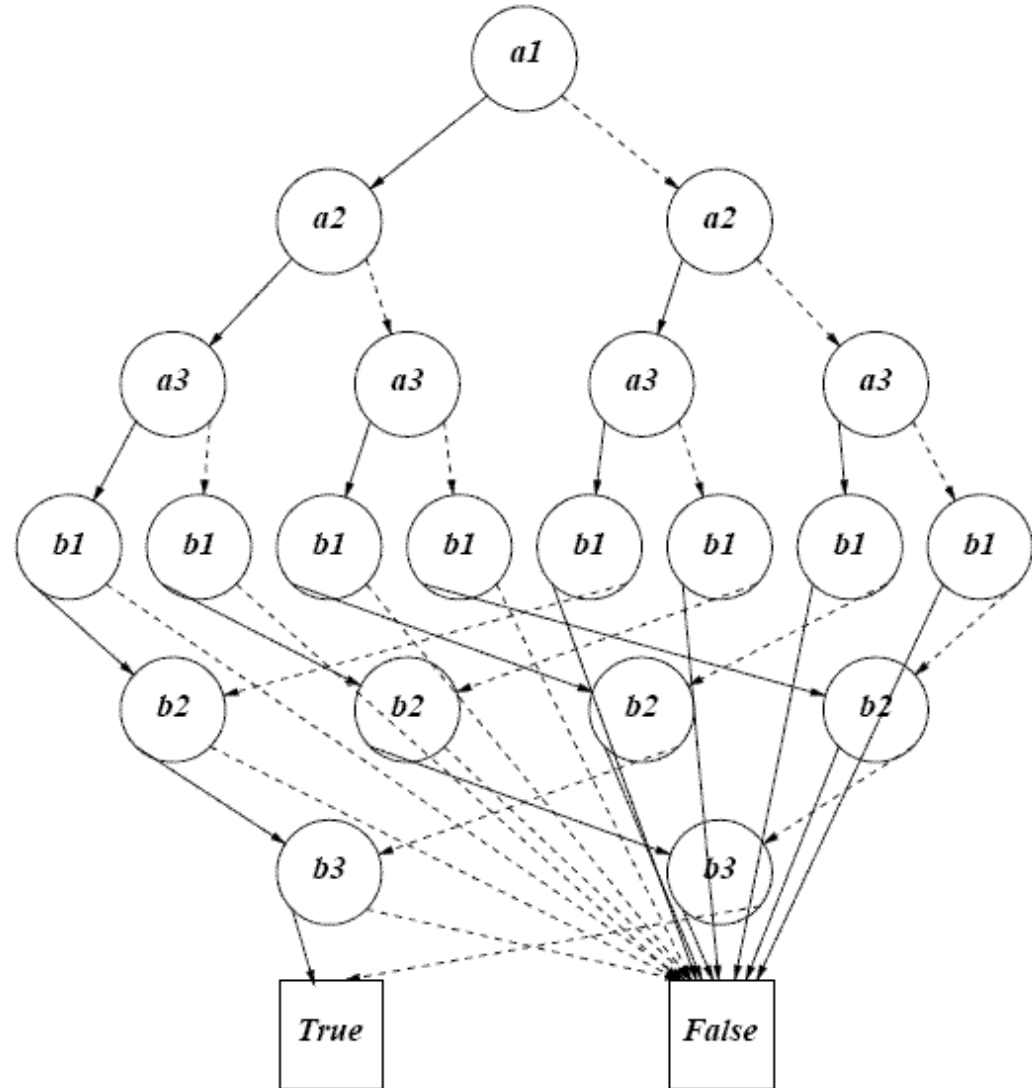
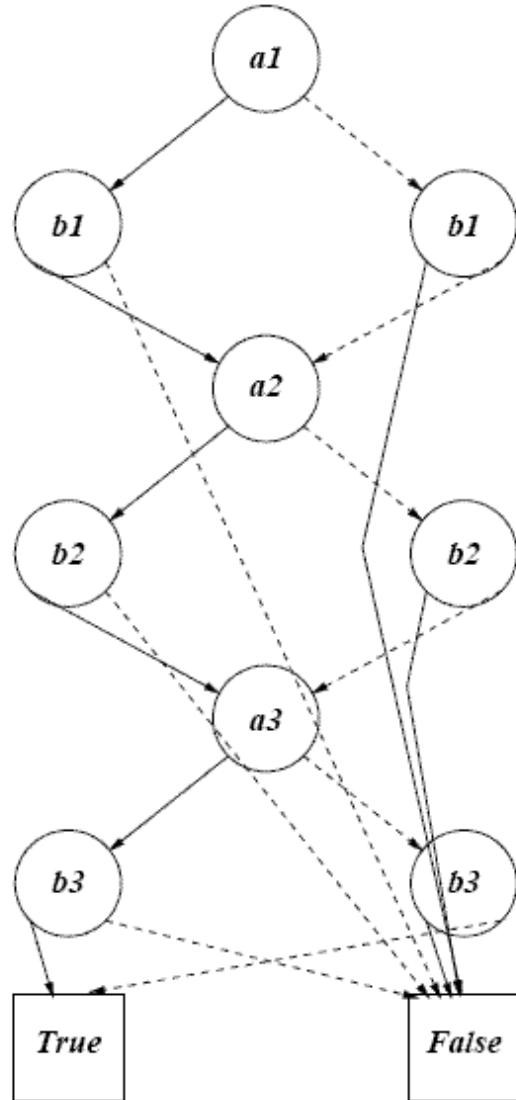


*signal is green or
door is opened*

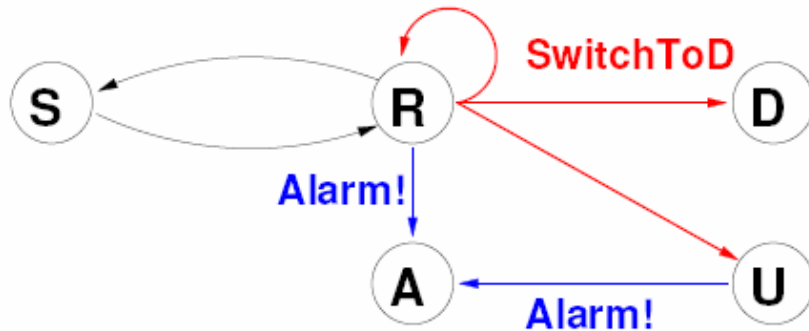


*signal is green and
door is not opened*

$$(a1 \leftrightarrow b1) \wedge (a2 \leftrightarrow b2) \wedge (a3 \leftrightarrow b3).$$

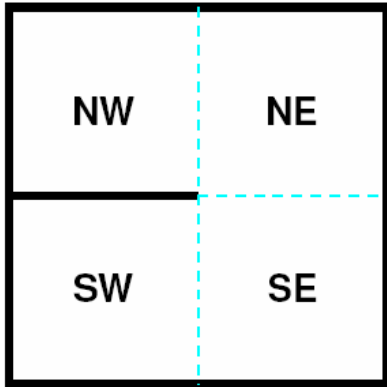


Beyond Temporal Logics



syntax of the **EaGle**-language:

- *reachability (basic) goals*: DoReach p, TryReach p
- *maintenance (basic) goals*: DoMaintain p, TryMaintain p
- *conjunction*: g And g'
- *failure*: g Fail g'
- *control operators*: g Then g', Repeat g



Partial Observability

GoEast ; (if WallN then GoSouth else Nothing) ; GoWest

