

EXPRESSIVE GOAL SPECIFICATION LANGUAGES IN PRESENCE OF NON-DETERMINISTIC ACTIONS

By: Chitta Baral and Jicheng Zhao

Dept. of Computer Science and Engineering
Arizona State University, USA

(presented by [Pedro Cabalar](#))

Outline

- Introduction
- LTL and CTL* overview
- The proposal: π -CTL*
- Related work
- Conclusions

Outline

Introduction

- LTL and CTL* overview

- The proposal: π -CTL*

- Related work

- Conclusions

Introduction: deterministic domains

- Deterministic domain: plan = sequence of actions
what leads to a sequence of states from initial state;

Introduction: deterministic domains

- Deterministic domain: plan = sequence of actions
what leads to a sequence of states from initial state;
- Interest: to specify properties of the agent's plan;

Introduction: deterministic domains

- Deterministic domain: plan = sequence of actions
what leads to a sequence of states from initial state;
- Interest: to specify properties of the agent's plan;
- E.g.: LTL not only specifies properties of the “final state”
but also the intermediate states in the sequence;

Introduction: deterministic domains

- Deterministic domain: plan = sequence of actions
what leads to a sequence of states from initial state;
- Interest: to specify properties of the agent's plan;
- E.g.: LTL not only specifies properties of the “final state”
but also the intermediate states in the sequence;
Example: “move through states that avoid p ”

Introduction: deterministic domains

- Deterministic domain: plan = sequence of actions
what leads to a sequence of states from initial state;
- Interest: to specify properties of the agent's plan;
- E.g.: LTL not only specifies properties of the “final state”
but also the intermediate states in the sequence;
Example: “move through states that avoid p ”
- CTL* also specifies the properties of the states leading from
states in the main path.

Introduction: deterministic domains

- Deterministic domain: plan = sequence of actions
what leads to a sequence of states from initial state;
- Interest: to specify properties of the agent's plan;
- E.g.: LTL not only specifies properties of the “final state”
but also the intermediate states in the sequence;
Example: “move through states that avoid p ”
- CTL* also specifies the properties of the states leading from
states in the main path. Example: “move through states
from which you could reach p (with another action)”

Introduction: **non**-deterministic domains

- Now we cannot associate a sequence of actions to a sequence of states. The action sequence either results in multiple sequences, or is not well defined;

Introduction: **non**-deterministic domains

- Now we cannot associate a sequence of actions to a sequence of states. The action sequence either results in multiple sequences, or is not well defined;

- Instead, we adopt the definition of a policy π to be a **mapping** from each state to an action. Example:

$$\pi_1 = \{(s_1, a_1), (s_2, a_2), (s_3, a_3), (s_4, nop)\}$$

Introduction: **non-deterministic domains**

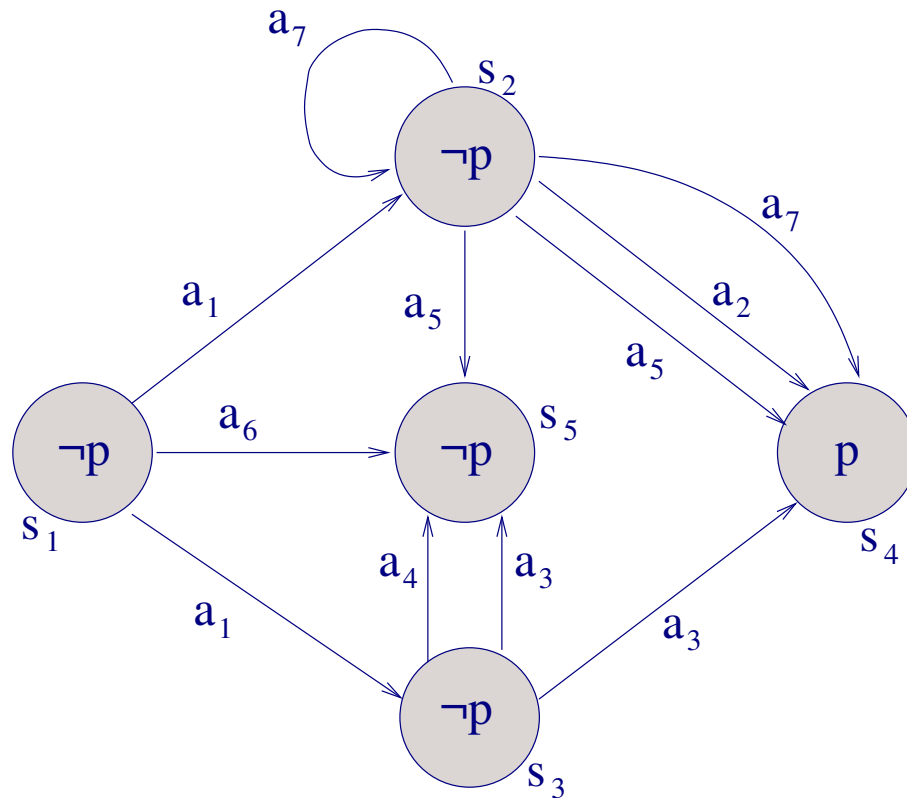
- Now we cannot associate a sequence of actions to a sequence of states. The action sequence either results in multiple sequences, or is not well defined;

- Instead, we adopt the definition of a **policy** π to be a **mapping** from each state to an action. Example:

$$\pi_1 = \{(s_1, a_1), (s_2, a_2), (s_3, a_3), (s_4, nop)\}$$

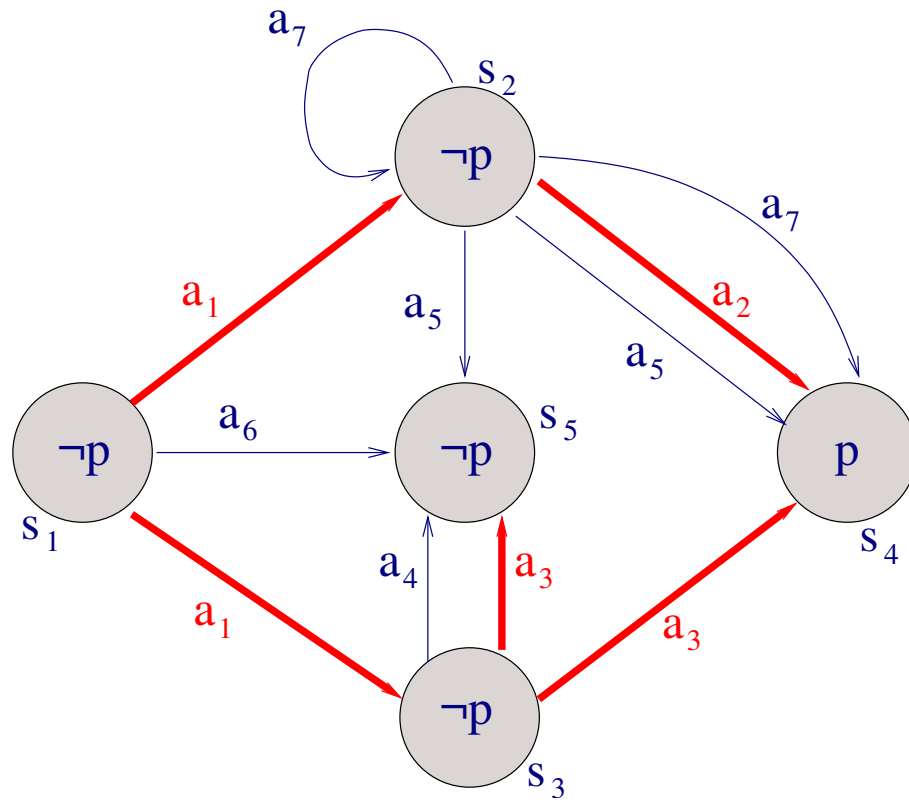
- Under this setting, specifying goals has new challenges: [Dal Lago, Pistore & Traverso 02] talk about “**extended goals.**”

An example



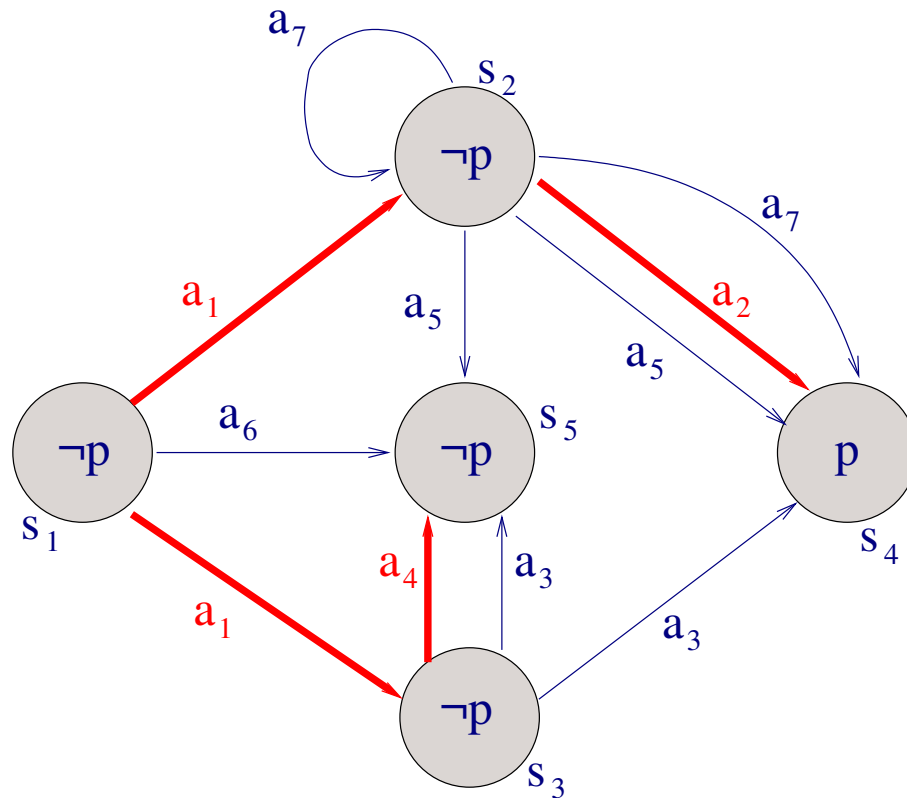
“Try your best to reach p ”

An example



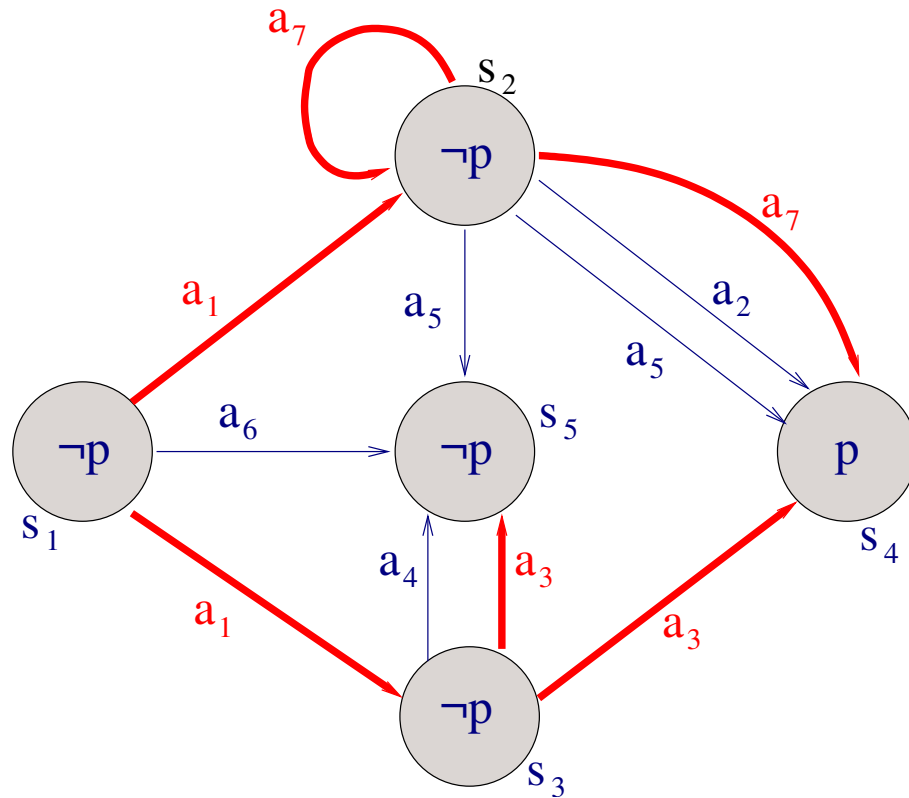
Policy π_1

An example



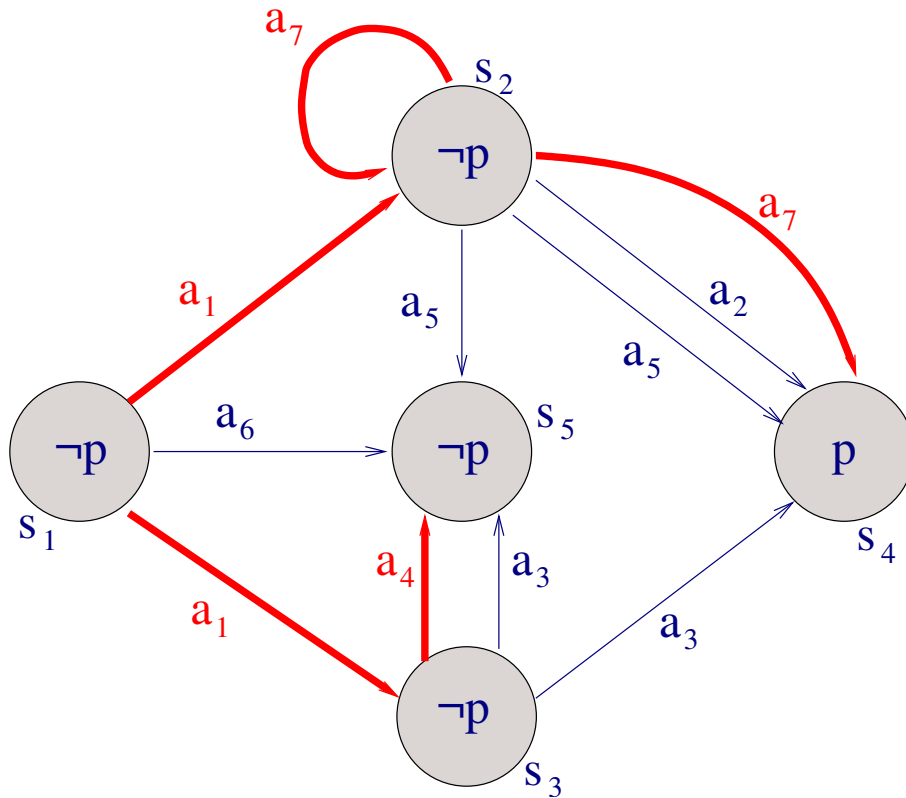
Policy π_2
clearly *worse than* π_1 !

An example



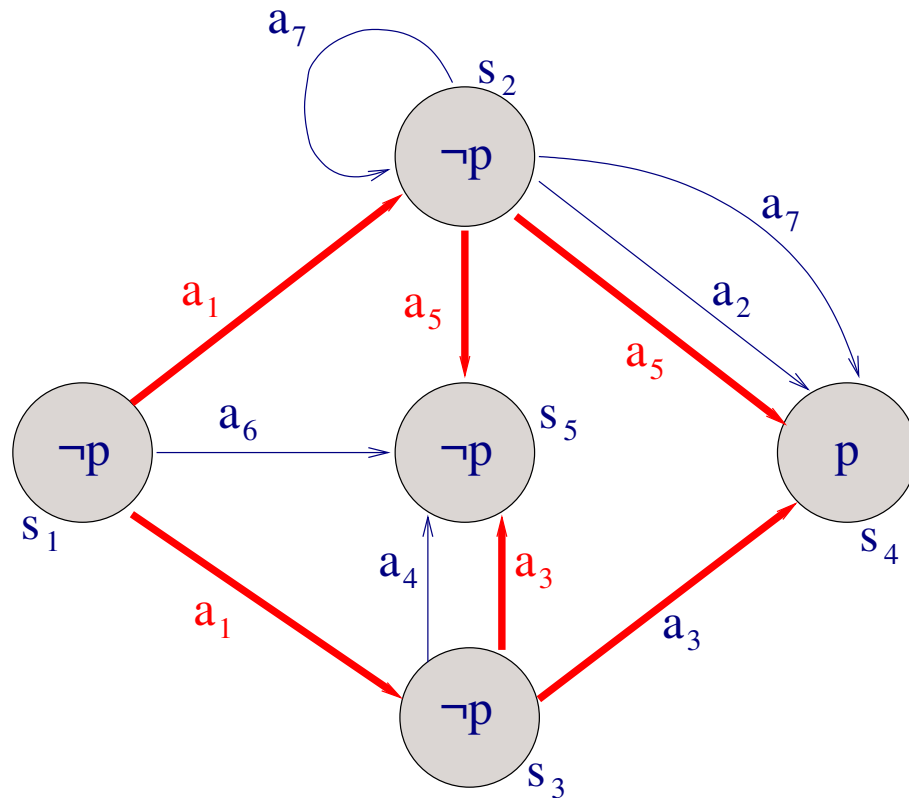
Policy π_3
worse than $\pi_1!$ but $\pi_2?$

An example



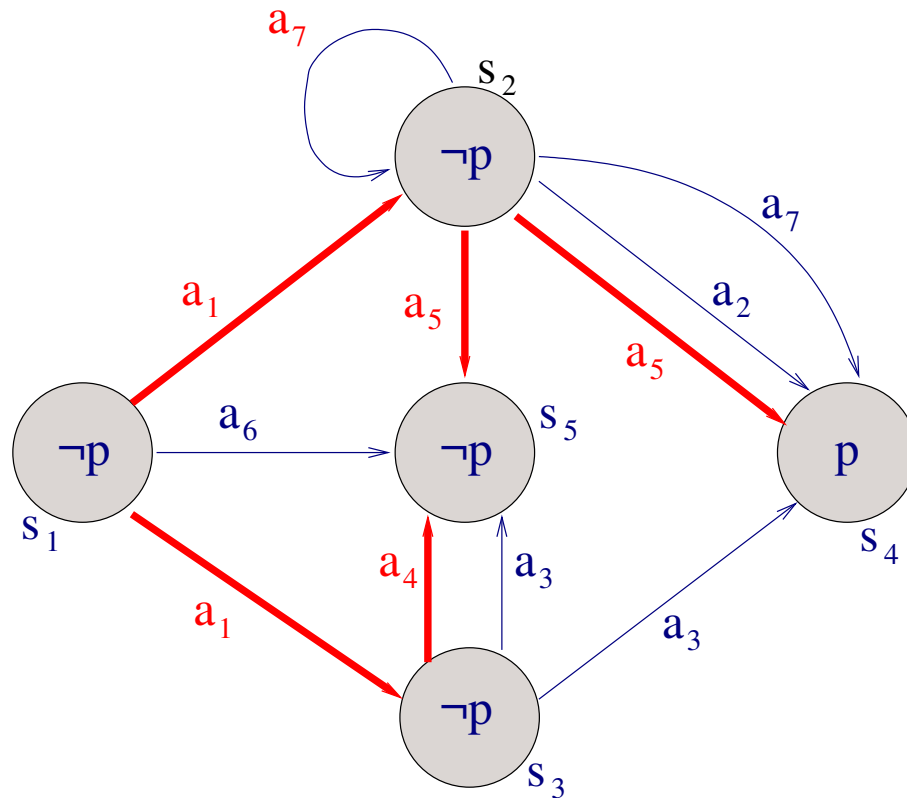
Policy π_4
worse than π_2 and π_3

An example



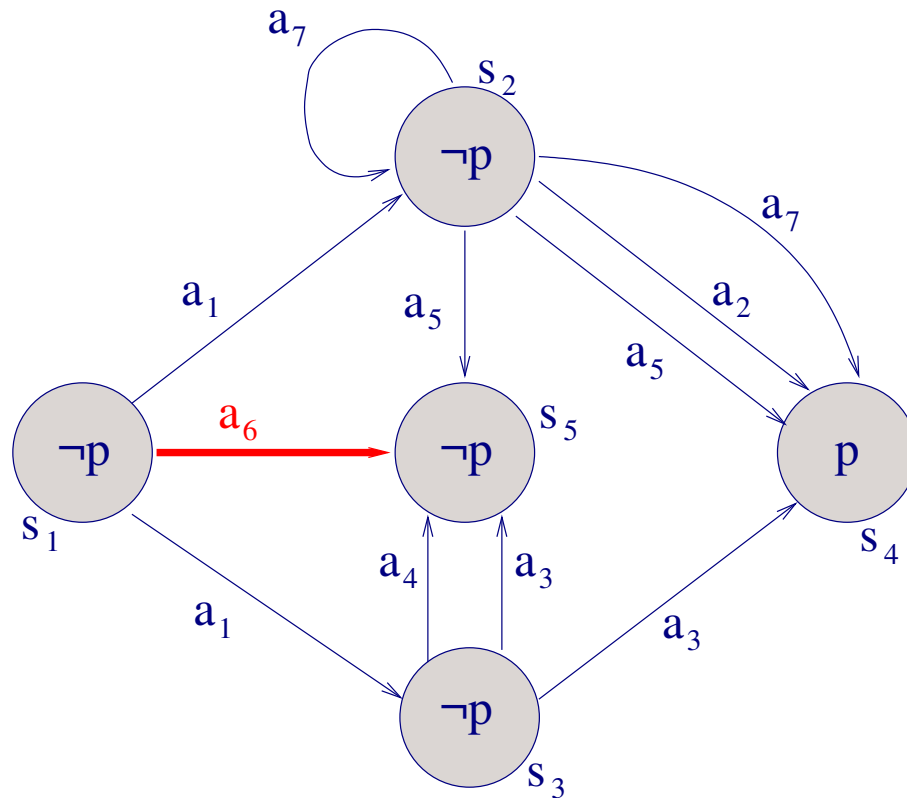
Policy π_5
worse than π_3

An example



Policy π_6
worse than π_4, π_5

An example



Policy π_7
Really bad!

Outline

- ✓ Introduction
- LTL and CTL* overview
- The proposal: π -CTL*
- Related work
- Conclusions

Outline

✓ Introduction

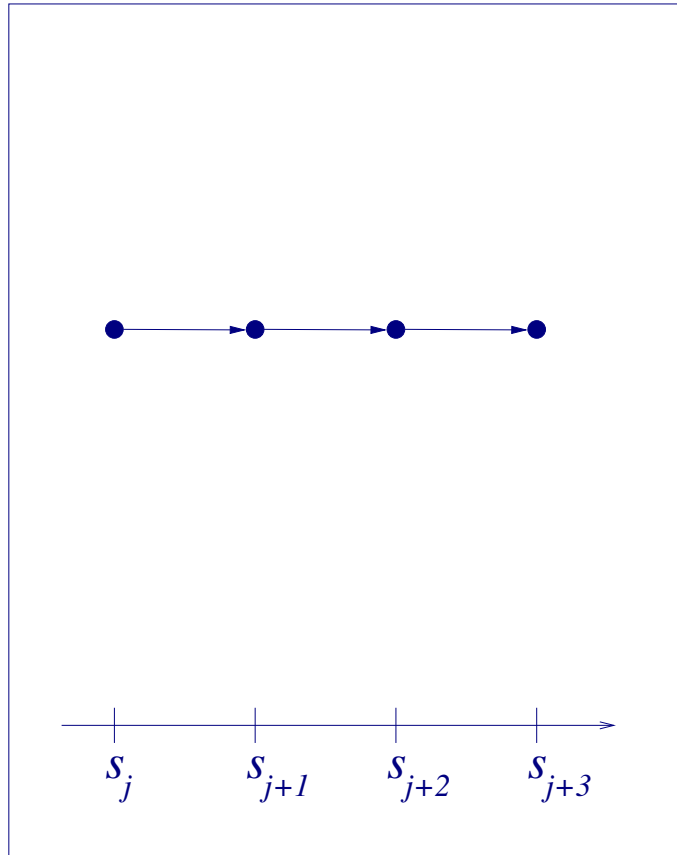
☞ LTL and CTL* overview

☐ The proposal: π -CTL*

☐ Related work

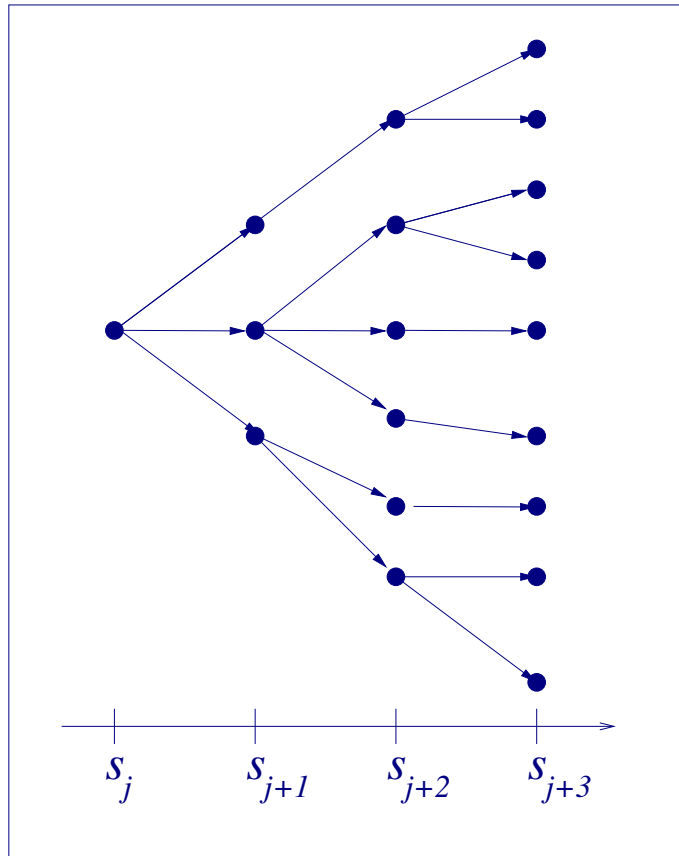
☐ Conclusions

Linear Temporal Logic LTL



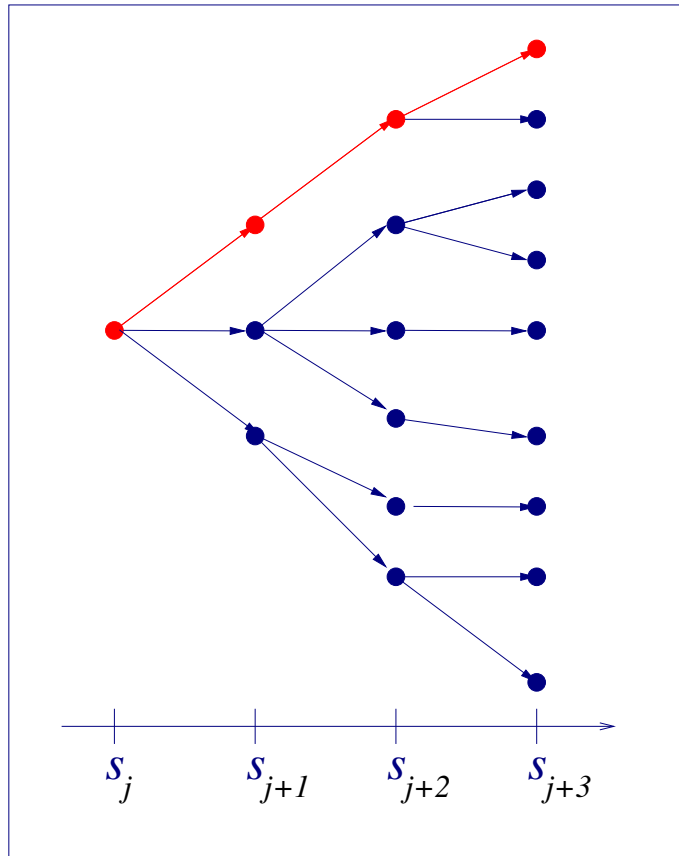
- Linear time: sequence of states
- Operators:
 - $\square p$ = always p
 - $\diamond p$ = eventually p
 - $\bigcirc p$ = next p
 - $p \text{ U } q$ = p true until q

Branching Temporal logic CTL*



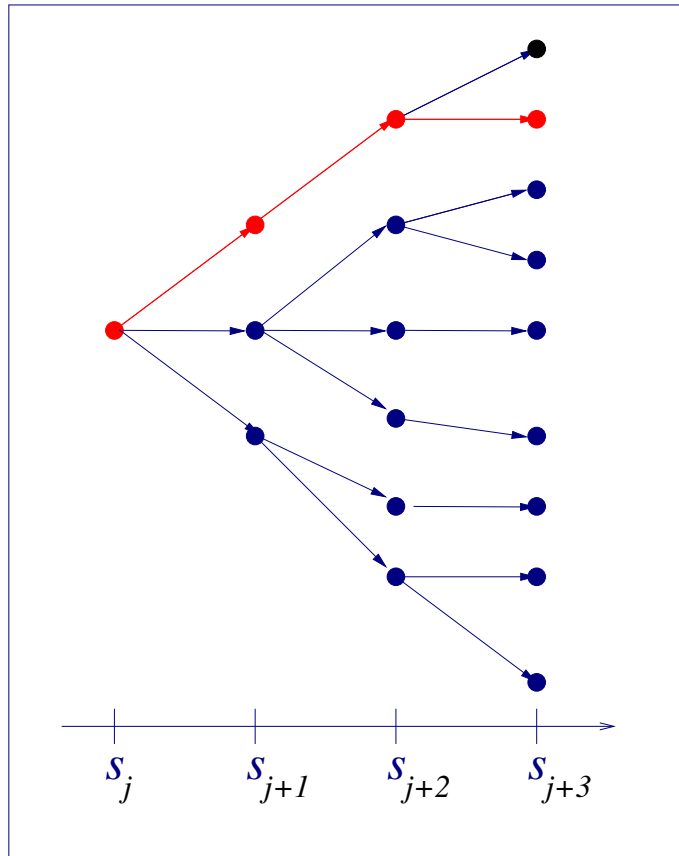
- Branching time
- New operators for **paths**

Branching Temporal logic CTL*



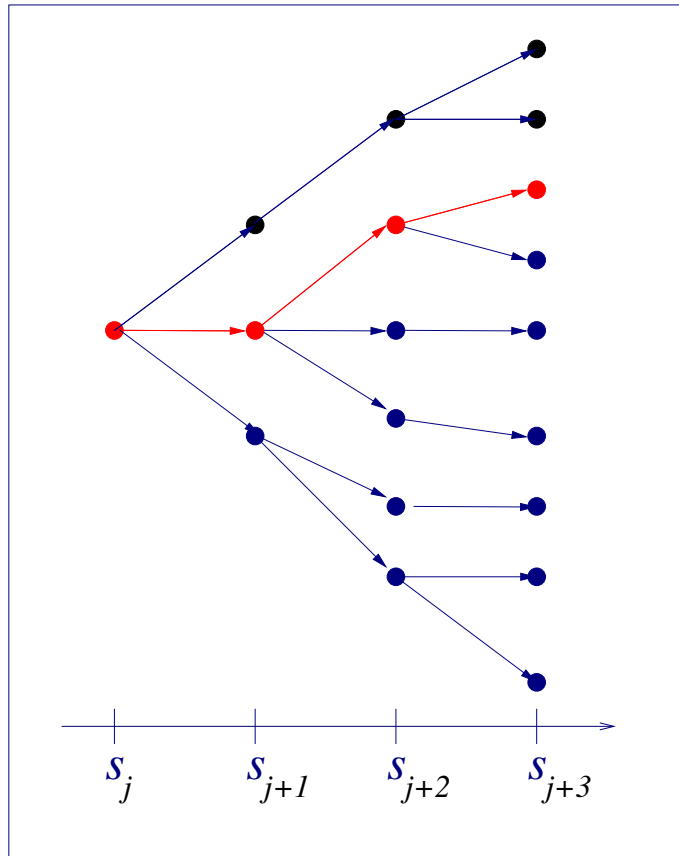
- Branching time
- New operators for **paths**

Branching Temporal logic CTL*



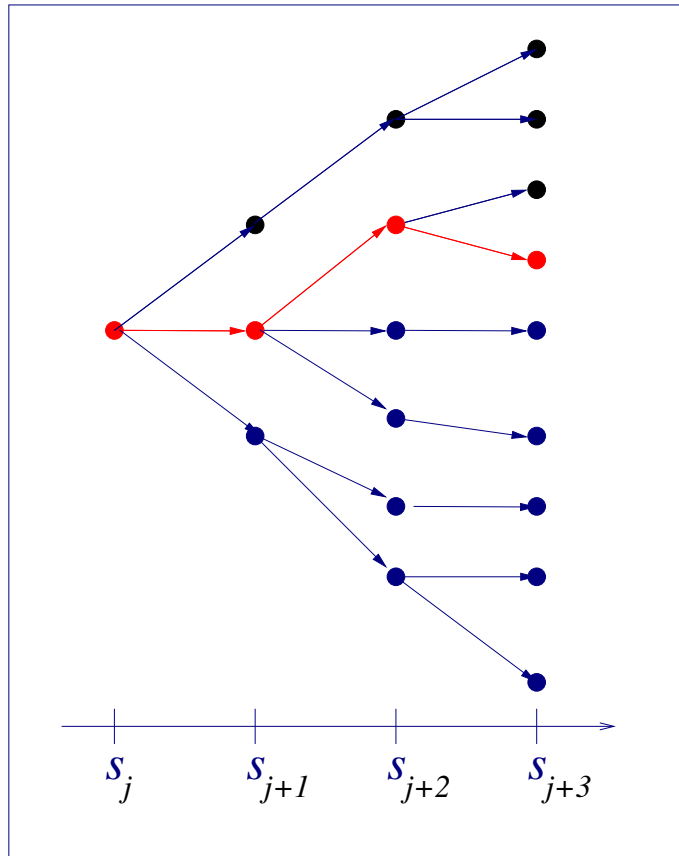
- Branching time
- New operators for **paths**

Branching Temporal logic CTL*



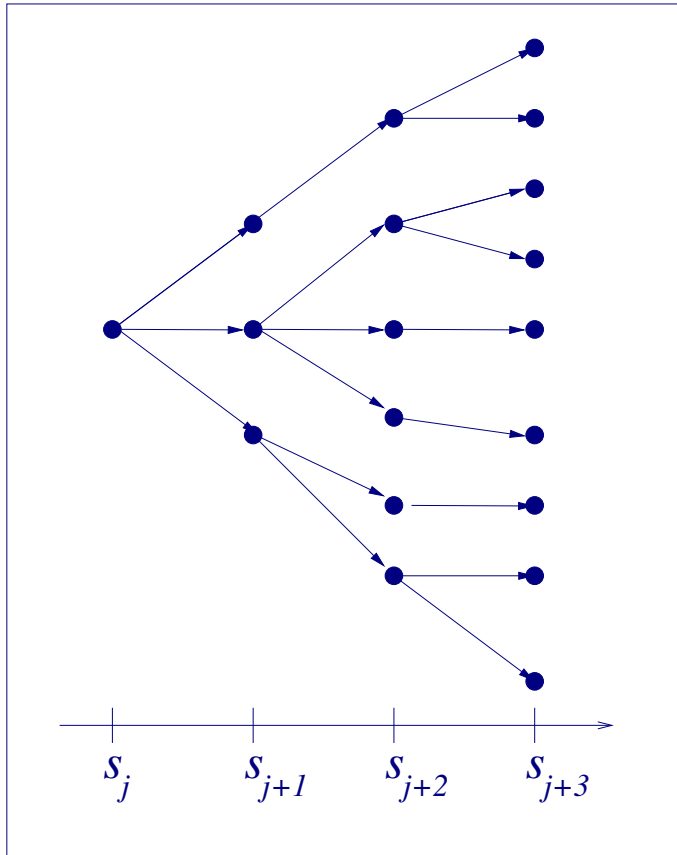
- Branching time
- New operators for **paths**

Branching Temporal logic CTL*



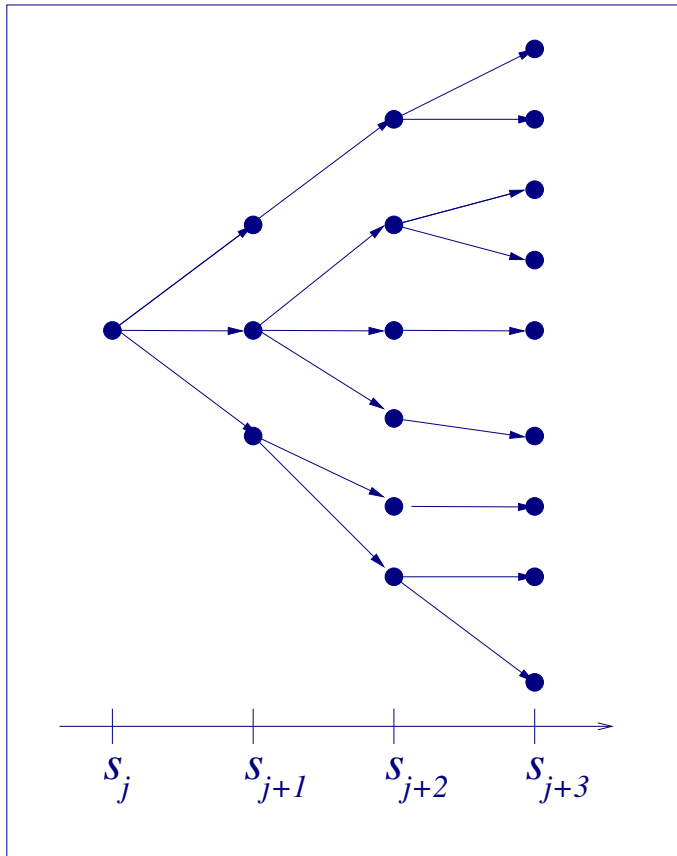
- Branching time
- New operators for **paths**

Branching Temporal logic CTL*



$A\phi$ = for any path, ϕ holds
 $E\phi$ = for some path, ϕ holds

Branching Temporal logic CTL*



Examples:

$A\Diamond p$ = all paths reach p

$E\Box p$ = in some path, always p

Branching Temporal logic CTL*

Syntax:

$\langle p \rangle$ = propositional formula;

$\langle sf \rangle$ = “state” formula;

$\langle pf \rangle$ = “path” formula

Branching Temporal logic CTL*

Syntax:

$\langle p \rangle$ = propositional formula;

$\langle sf \rangle$ = “state” formula;

$\langle pf \rangle$ = “path” formula

$\langle sf \rangle ::= \langle p \rangle \mid \langle sf \rangle \wedge \langle sf \rangle \mid \langle sf \rangle \vee \langle sf \rangle \mid \neg \langle sf \rangle \mid \mathbf{E} \langle pf \rangle \mid \mathbf{A} \langle pf \rangle$

Branching Temporal logic CTL*

Syntax:

$\langle p \rangle$ = propositional formula;

$\langle sf \rangle$ = “state” formula;

$\langle pf \rangle$ = “path” formula

$$\langle sf \rangle ::= \langle p \rangle \mid \langle sf \rangle \wedge \langle sf \rangle \mid \langle sf \rangle \vee \langle sf \rangle \mid \neg \langle sf \rangle \mid \mathbf{E} \langle pf \rangle \mid \mathbf{A} \langle pf \rangle$$
$$\langle pf \rangle ::= \langle sf \rangle \mid \langle pf \rangle \vee \langle pf \rangle \mid \neg \langle pf \rangle \mid \langle pf \rangle \wedge \langle pf \rangle \mid \\ \langle pf \rangle \mathbf{U} \langle pf \rangle \mid \mathbf{O} \langle pf \rangle \mid \mathbf{\diamond} \langle pf \rangle \mid \mathbf{\square} \langle pf \rangle$$

Branching Temporal logic CTL*

Semantics (state formulas). We use a pair (s_j, R) with s_j a state and R the transition relation:

- $(s_j, R) \models p$ iff p is true in s_j .
- \neg, \wedge, \vee as usual.
- $(s_j, R) \models E pf$ iff there exists a path σ in R starting from s_j such that $(s_j, R, \sigma) \models pf$.
- $(s_j, R) \models A pf$ iff for all paths σ in R starting from s_j we have that $(s_j, R, \sigma) \models pf$.

Branching Temporal logic CTL*

Semantics (path formulas). We use triplet (s, R, σ) where σ given by the sequence of states s_0, s_1, \dots , is a path.

- $(s_j, R, \sigma) \models sf$ iff $(s_j, R) \models sf$.
- \neg, \wedge, \vee as usual.
- $(s_j, R, \sigma) \models \bigcirc f$ iff $(s_{j+1}, R, \sigma) \models f$
- $(s_j, R, \sigma) \models \square f$ iff $(s_k, R, \sigma) \models f$, for all $k \geq j$.
- $(s_j, R, \sigma) \models \diamond f$ iff $(s_k, R, \sigma) \models f$, for some $k \geq j$.
- $(s_j, R, \sigma) \models f_1 \cup f_2$ iff there exists $k \geq j$ such that $(s_k, R, \sigma) \models f_2$ and for all $i, j \leq i < k, (s_i, R, \sigma) \models f_1$.

Branching Temporal logic CTL*

- A sequence of actions a_1, \dots, a_n is a **plan** with respect to the initial state s_0 and a CTL* goal G if
$$(s_0, R, \sigma) \models G,$$
where σ is the trajectory corresponding to s_0 and a_1, \dots, a_n .

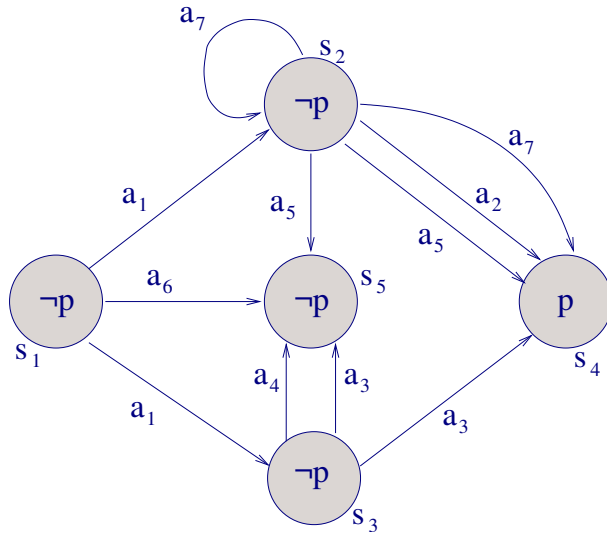
Outline

- ✓ Introduction
- ✓ LTL and CTL* overview
- The proposal: π -CTL*
- Related work
- Conclusions

Outline

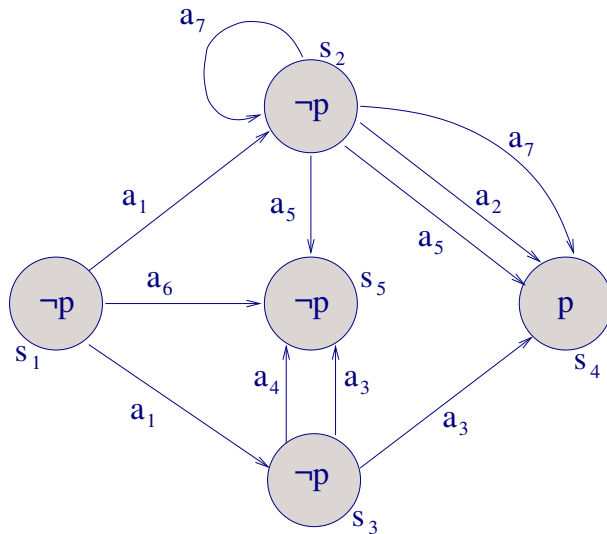
- ✓ Introduction
- ✓ LTL and CTL* overview
- ☞ The proposal: π -CTL*
- ☐ Related work
- ☐ Conclusions

When using CTL* for goal specification . . .



- Goals $E\Diamond p$ and $A\Diamond p$ in s_1 : policies are not distinguishable in them!

When using CTL* for goal specification . . .



- Goals $E\Diamond p$ and $A\Diamond p$ in s_1 : policies are not distinguishable in them!
- Besides A and E in capturing all paths in the plan, we may also need to consider those paths by following the policy.

The extension of CTL^{*}: π -CTL^{*}

- One natural extension of CTL^{*} is to group the set of paths from the initial state that all correspond to the same policy:

The extension of CTL*: π -CTL*

- One natural extension of CTL* is to group the set of paths from the initial state that all correspond to the same policy:
 - $A_\pi pf$: ‘for all paths that agree with the policy π , pf holds’;
 - $E_\pi pf$: ‘there exists a path that agrees with the policy π for which pf holds’.

The extension of CTL*: π -CTL*

- One natural extension of CTL* is to group the set of paths from the initial state that all correspond to the same policy:
 - $A_\pi pf$: ‘for all paths that agree with the policy π , pf holds’;
 - $E_\pi pf$: ‘there exists a path that agrees with the policy π for which pf holds’.
- Semantics: we add the policy accessibility relation R_π
 $(s_j, R, R_\pi) \models sf \quad (s_j, R, R_\pi, \sigma) \models pf$

The extension of CTL*: π -CTL*

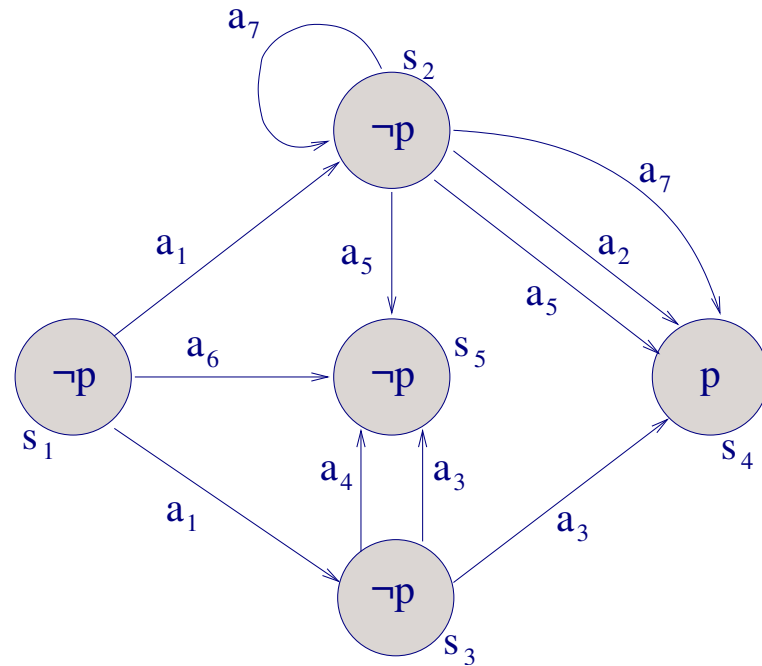
- Given an initial state s_0 , a policy π , a transition function Φ , and a goal G we say π is a policy for G from s_0 , iff $(s_0, R, R_\pi) \models G$.

The extension of CTL*: π -CTL*

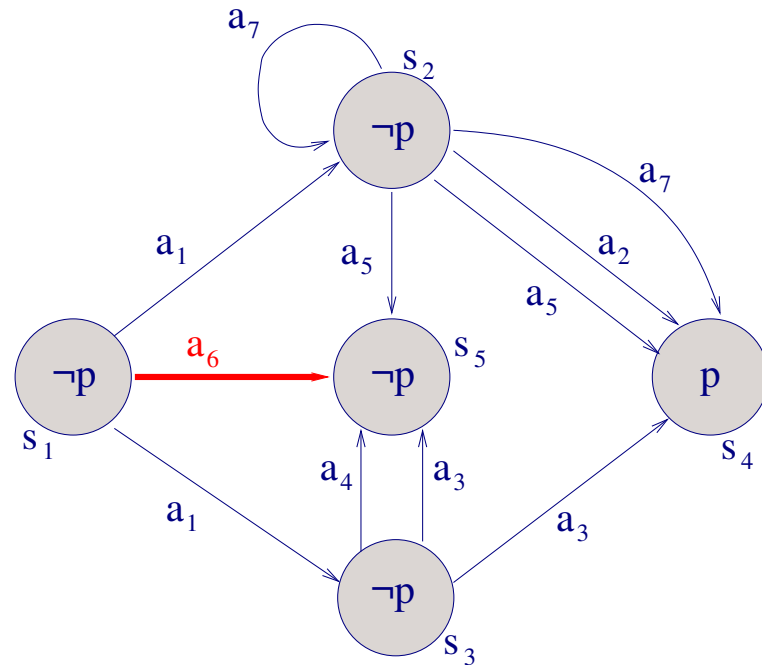
- Given an initial state s_0 , a policy π , a transition function Φ , and a goal G we say π is a policy for G from s_0 , iff $(s_0, R, R_\pi) \models G$.
- Note that goals must be state formulas rather than path formulas (in contrast to deterministic domains).

The extension of CTL*: π -CTL*

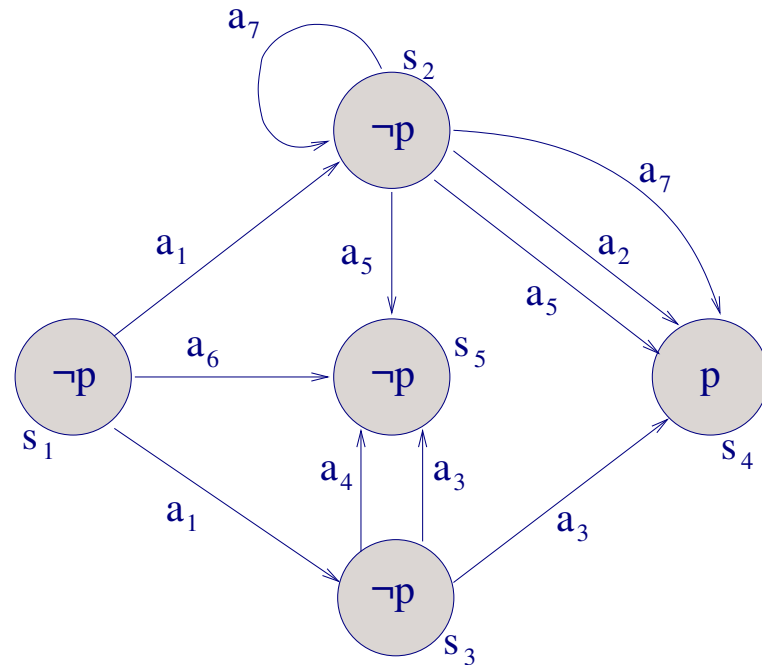
- Given an initial state s_0 , a policy π , a transition function Φ , and a goal G we say π is a policy for G from s_0 , iff $(s_0, R, R_\pi) \models G$.
- Note that goals must be state formulas rather than path formulas (in contrast to deterministic domains).
- Some examples of Goal Specification with π -CTL* . . .



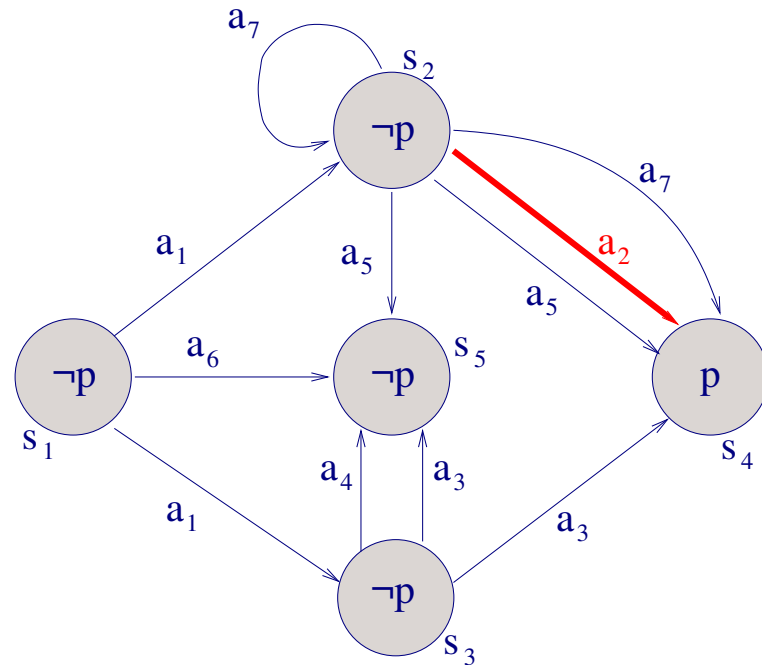
The weakest reachability goal “from the initial state there is a possibility that p can be reached” is expressed by $E_{\pi} \diamond p$.



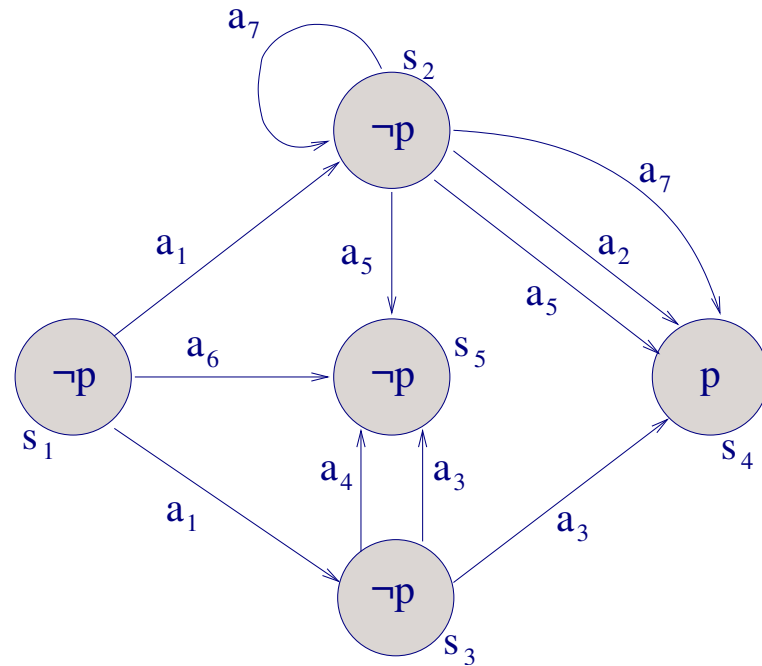
The weakest reachability goal “from the initial state there is a possibility that p can be reached” is expressed by $E_{\pi} \diamond p$. From s_1 , all policies but π_7 satisfy the goal.



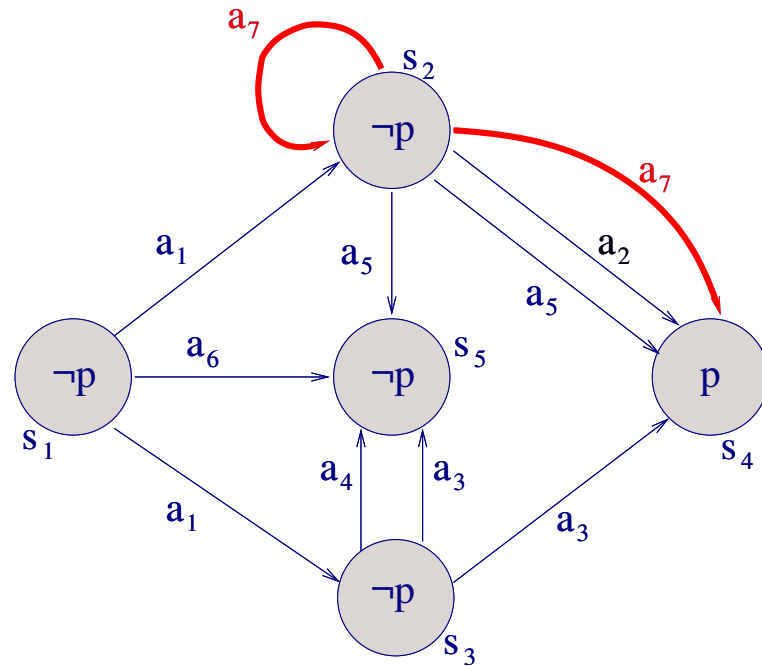
A stronger goal “from the initial state p must be reached” is expressed as $A_{\pi} \diamond p$. For s_1 , no policy makes it true.



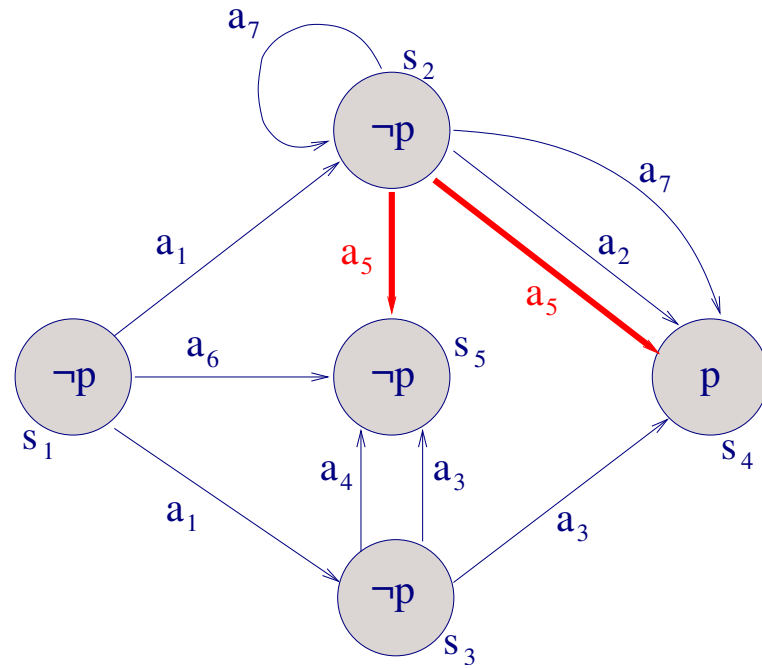
A stronger goal “from the initial state p must be reached” is expressed as $A_\pi \diamond p$. For s_1 , no policy makes it true. But, for instance, for s_2 the policy $\{(s_2, a_2)\}$ satisfies the goal.



“All along the trajectory there is always a possible path to p by following the policy” is expressed as $A_{\pi} \square (E_{\pi} \diamond p)$. For s_1 , no policy.



“All along the trajectory there is always a possible path to p by following the policy” is expressed as $A_\pi \square (E_\pi \diamond p)$. For s_1 , no policy. For s_2 , policies $\{(s_2, a_2)\}$ and $\{(s_2, a_7)\}$ satisfy this goal.



However, policy $\{(s_2, a_5)\}$ does not, (we could go to s_5 from where p can not be reached).

More examples

- $A_{\pi}(E \diamond p) =$ “All along the trajectory there is always a possible path to p , but this path is not necessary abide the policy the agent taken”.

More examples

- $A_{\pi}(E \diamond p)$ = “All along the trajectory there is always a possible path to p , but this path is not necessary abide the policy the agent taken”.
- $A(E_{\pi} \diamond p)$ = “For any state that is reachable from the initial state, there is always a path to p by following the policy.”

More examples

- $A_{\pi}(E \diamond p)$ = “All along the trajectory there is always a possible path to p , but this path is not necessary abide the policy the agent taken”.
- $A(E_{\pi} \diamond p)$ = “For any state that is reachable from the initial state, there is always a path to p by following the policy.”
- $E \diamond p \rightarrow E_{\pi} \diamond p$ = “from the initial state, if it is possible to reach p , the agent should possibly reach p ”. Useful to allow the agent to pursue an alternative goal when it realizes that its initial goal is no longer achievable.

More examples

- $A_\pi(E \diamond p)$ = “All along the trajectory there is always a possible path to p , but this path is not necessary abide the policy the agent taken”.
- $A(E_\pi \diamond p)$ = “For any state that is reachable from the initial state, there is always a path to p by following the policy.”
- $E \diamond p \rightarrow E_\pi \diamond p$ = “from the initial state, if it is possible to reach p , the agent should possibly reach p ”. Useful to allow the agent to pursue an alternative goal when it realizes that its initial goal is no longer achievable.
- $A_\pi \square(E \diamond p \rightarrow E_\pi \diamond p)$ = idem, but now from *any* state in the trajectory (not only initial one).

Outline

- ✓ Introduction
- ✓ LTL and CTL* overview
- ✓ The proposal: π -CTL*
- Related work
- Conclusions

Outline

- ✓ Introduction
- ✓ LTL and CTL* overview
- ✓ The proposal: π -CTL*
- ☞ Related work
- ☐ Conclusions

Related work

Dal Lago et al.'s formulation	π -CTL*
p	p
TryReach p	$A_{\pi} \square E_{\pi} \diamond p$
DoReach p	$A_{\pi} \diamond p$
TryMaint p	$A_{\pi} \square p$
DoMaint p	$A_{\pi} \square p$
g_1 And g_2	$g_1 \wedge g_2$

Related work

- Translation of other constructs still under study:

g_1 **Then** g_2

g_1 **Fail** g_2 (particularly interesting)

Repeat g_2

- Some features difficult (or impossible) to represent in Dal Lago et al's approach: E_π , \bigcirc , \diamond , nesting operators.

Outline

- ✓ Introduction
- ✓ LTL and CTL* overview
- ✓ The proposal: π -CTL*
- ✓ Related work
- Conclusions

Outline

- ✓ Introduction
- ✓ LTL and CTL* overview
- ✓ The proposal: π -CTL*
- ✓ Related work
- ☞ **Conclusions**

Conclusions

- We extended CTL* to handle non-deterministic actions;
- Compared to Dal Lago et. al.' formulation, our language does not define specialized syntax and semantics;
- More extensions are needed, some of them currently under study:
 - The comparison of different policies;
 - Using contexts: the agent can take different actions w.r.t. different contexts in a state.