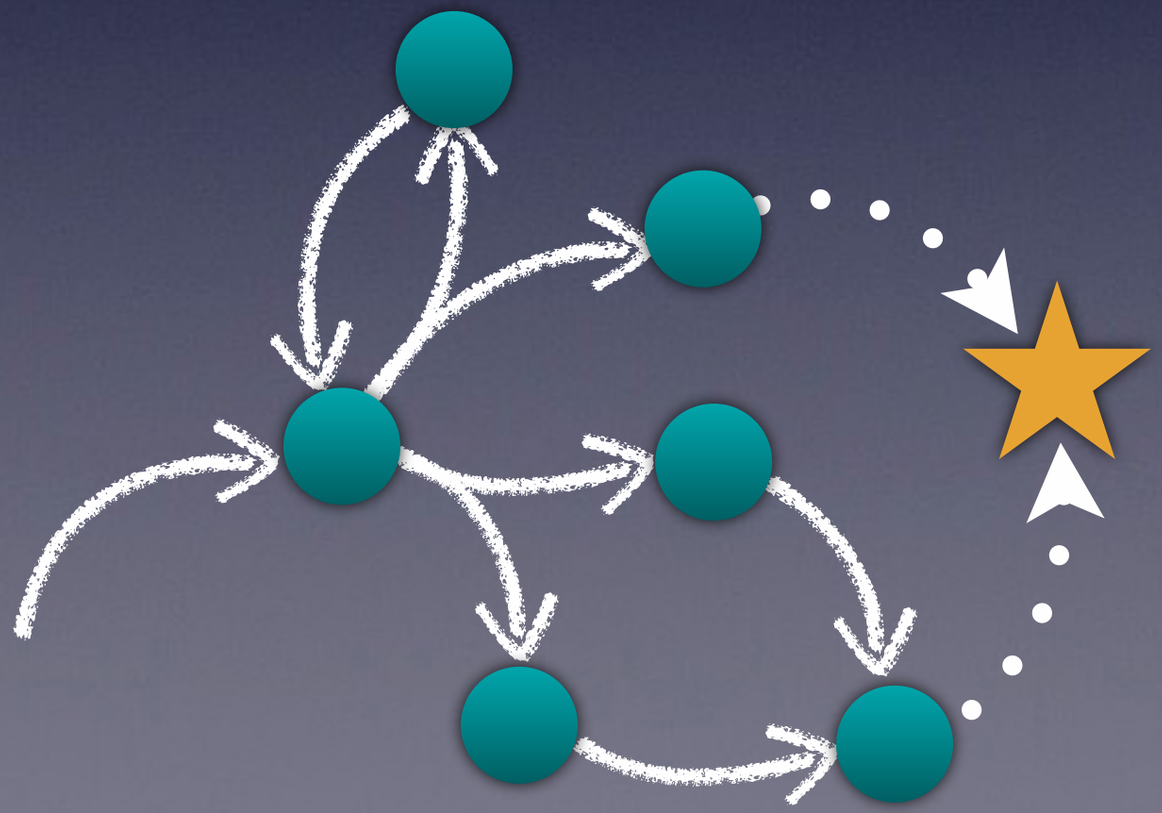
Improved Non-deterministic Planning by Exploiting State Relevance

Christian Muise

Sheila A. McIlraith

J. Christopher Beck



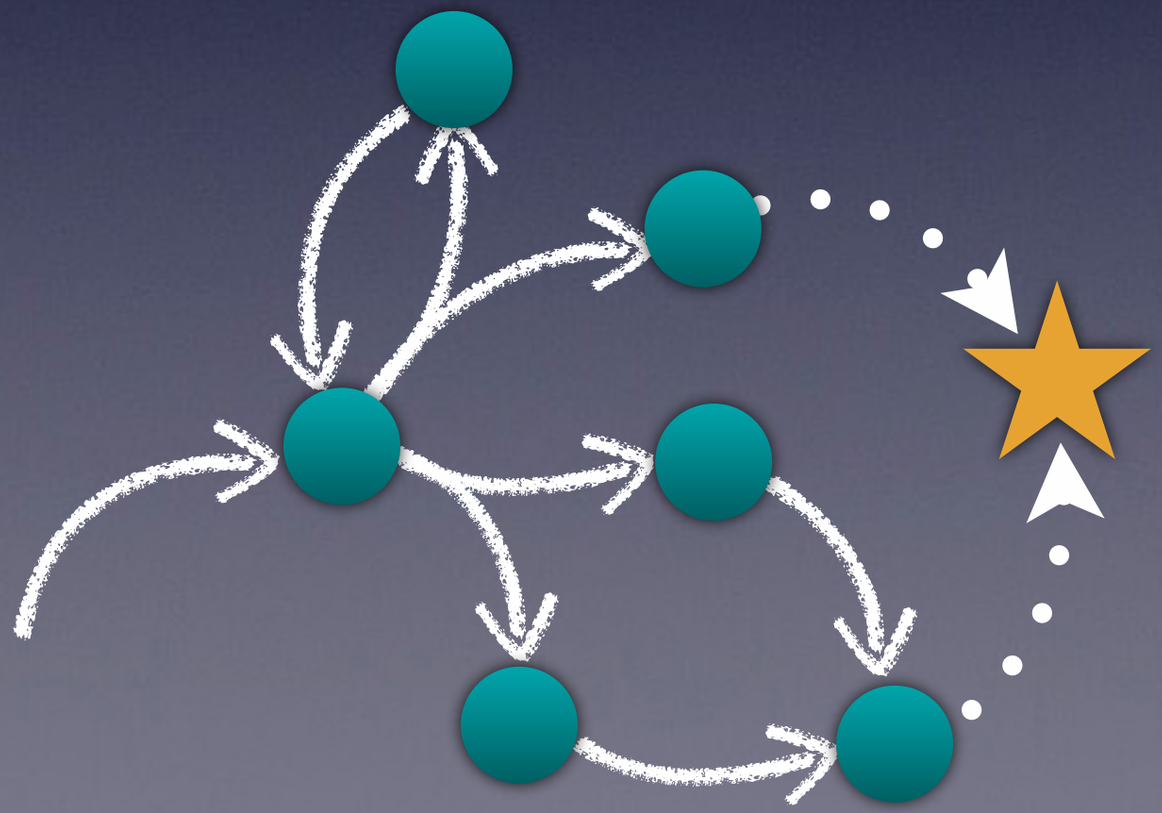
UNIVERSITY OF TORONTO

Improved Non-deterministic Planning by Exploiting State Relevance

Christian Muise

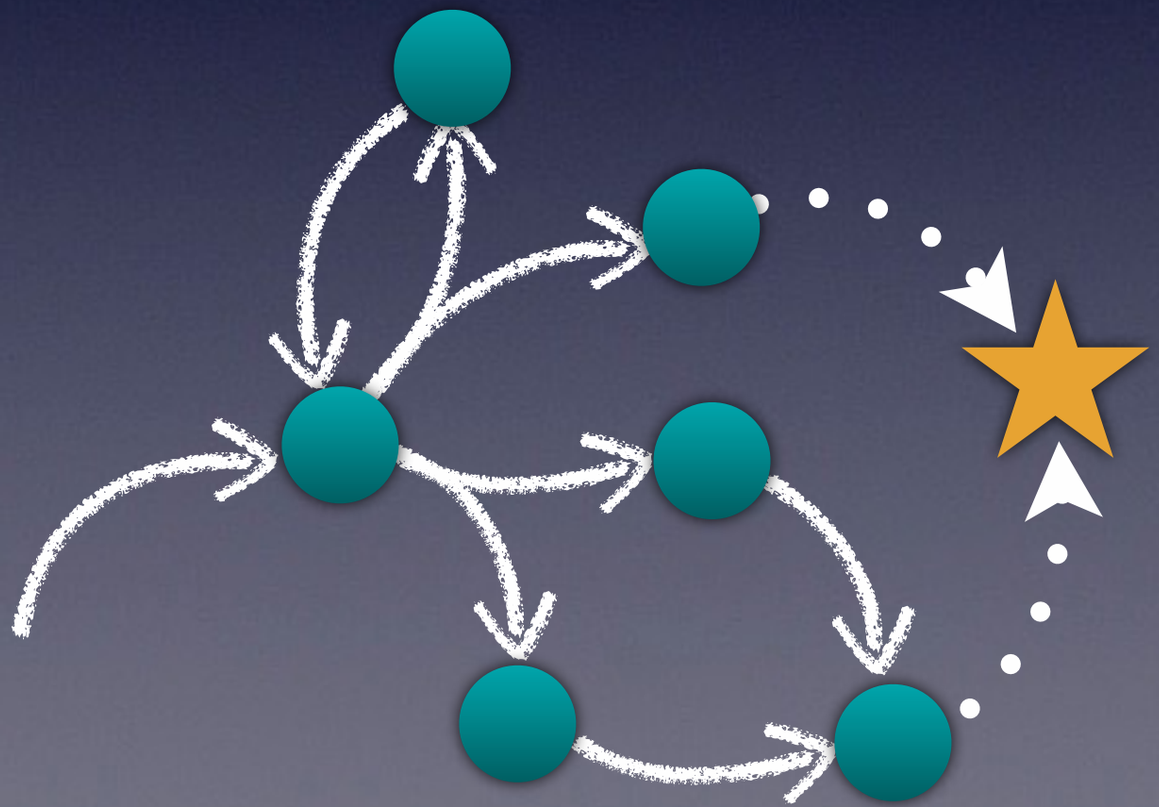
Sheila A. McIlraith

J. Christopher Beck



UNIVERSITY OF TORONTO

Fully **O**bservable
Non-**D**eterministic
Planning (FOND)



Christian Muise

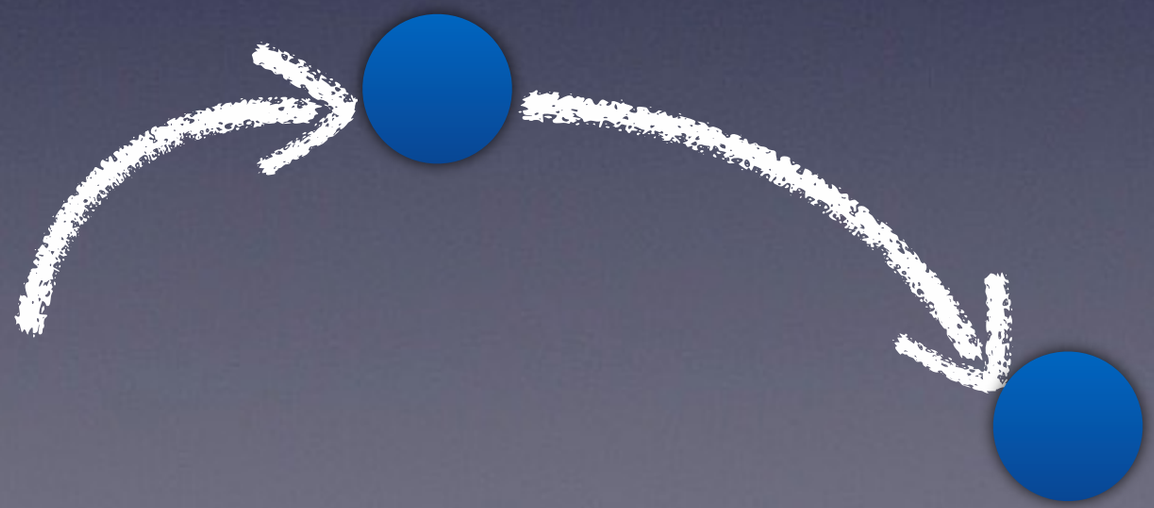
FOND

- **F**ully **O**bservable **N**on-**D**eterministic Planning
- Action outcomes are selected at random
- Typical approach uses *determinization*

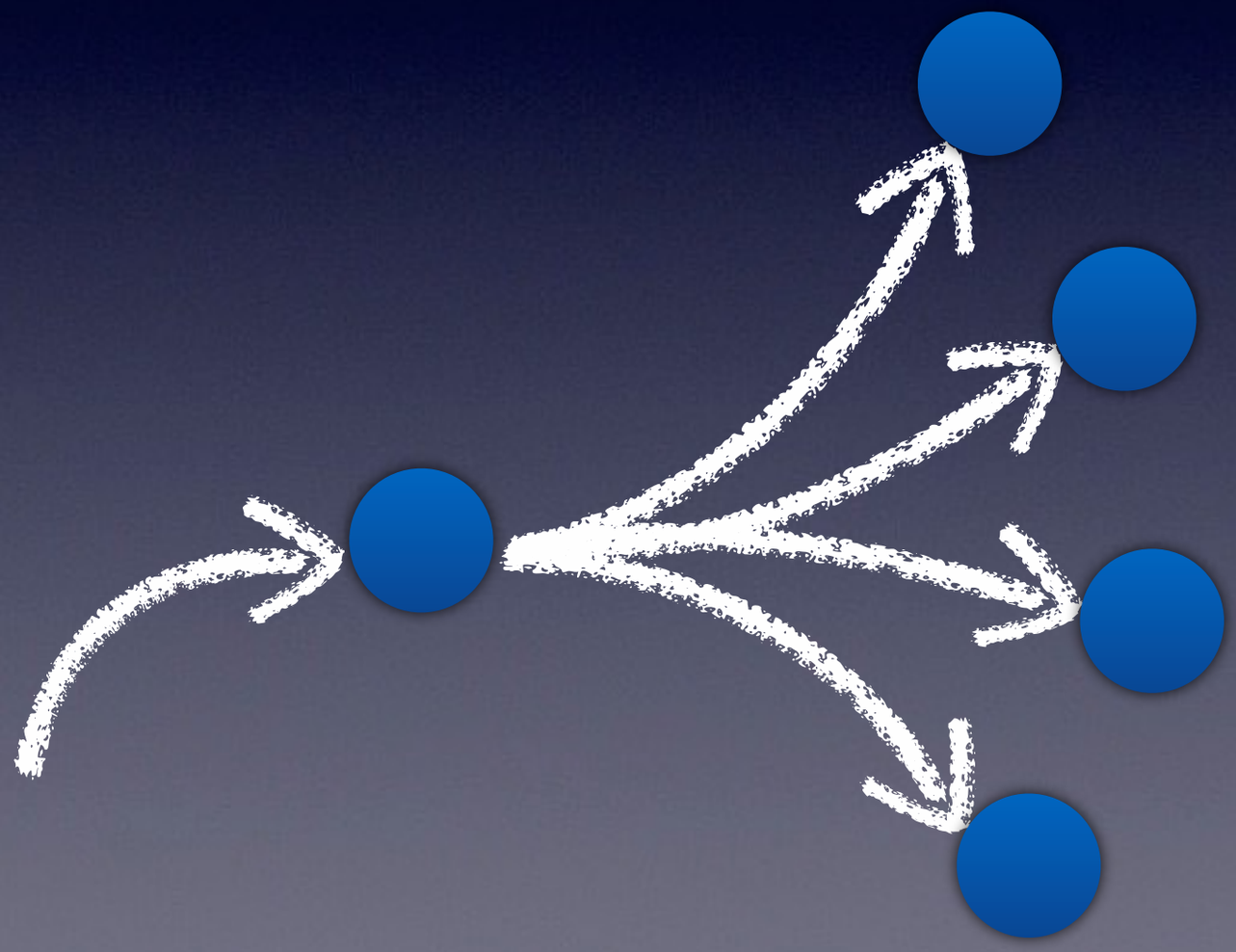
Non-Deterministic Action Outcomes

Non-Deterministic Action Outcomes

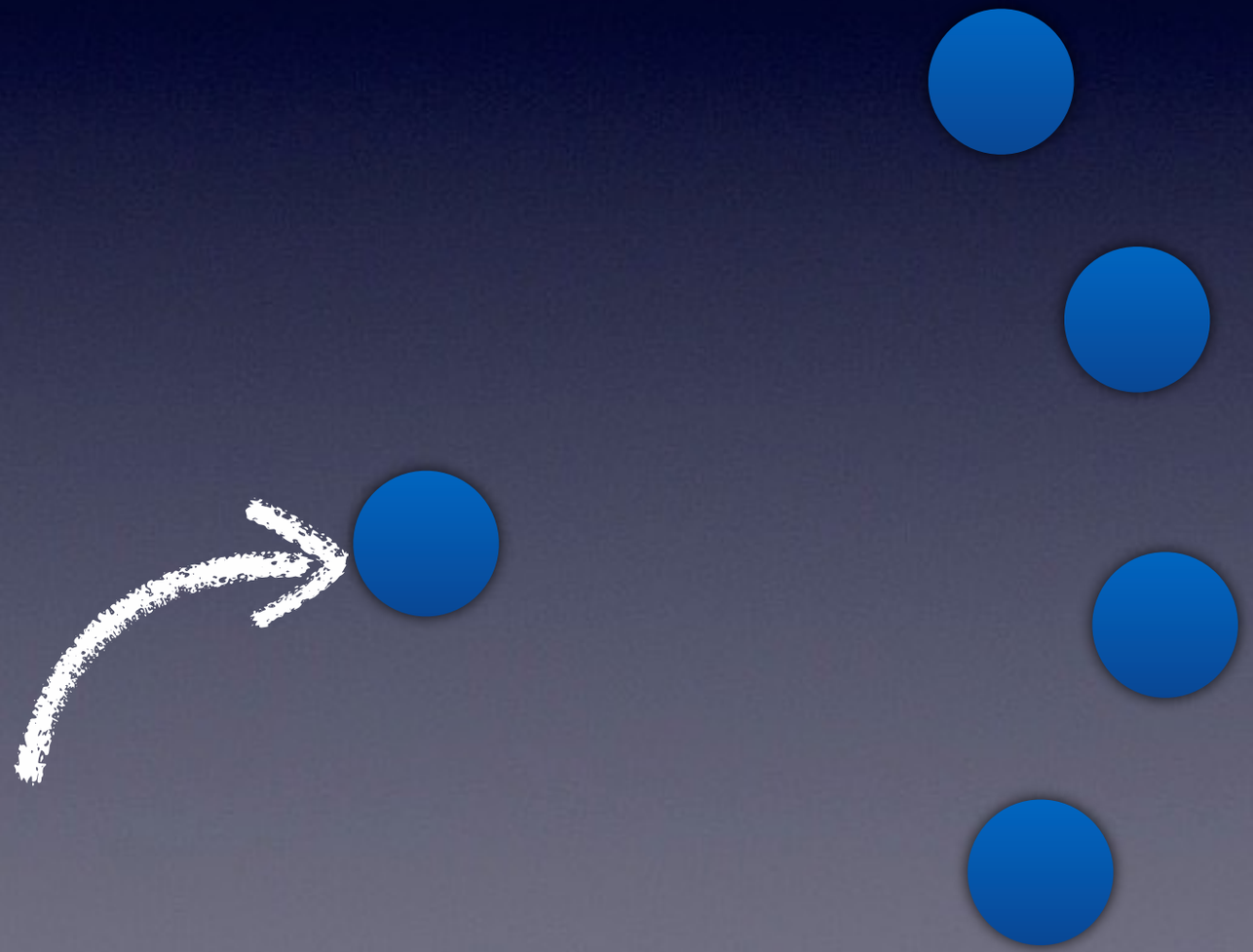
Non-Deterministic Action Outcomes



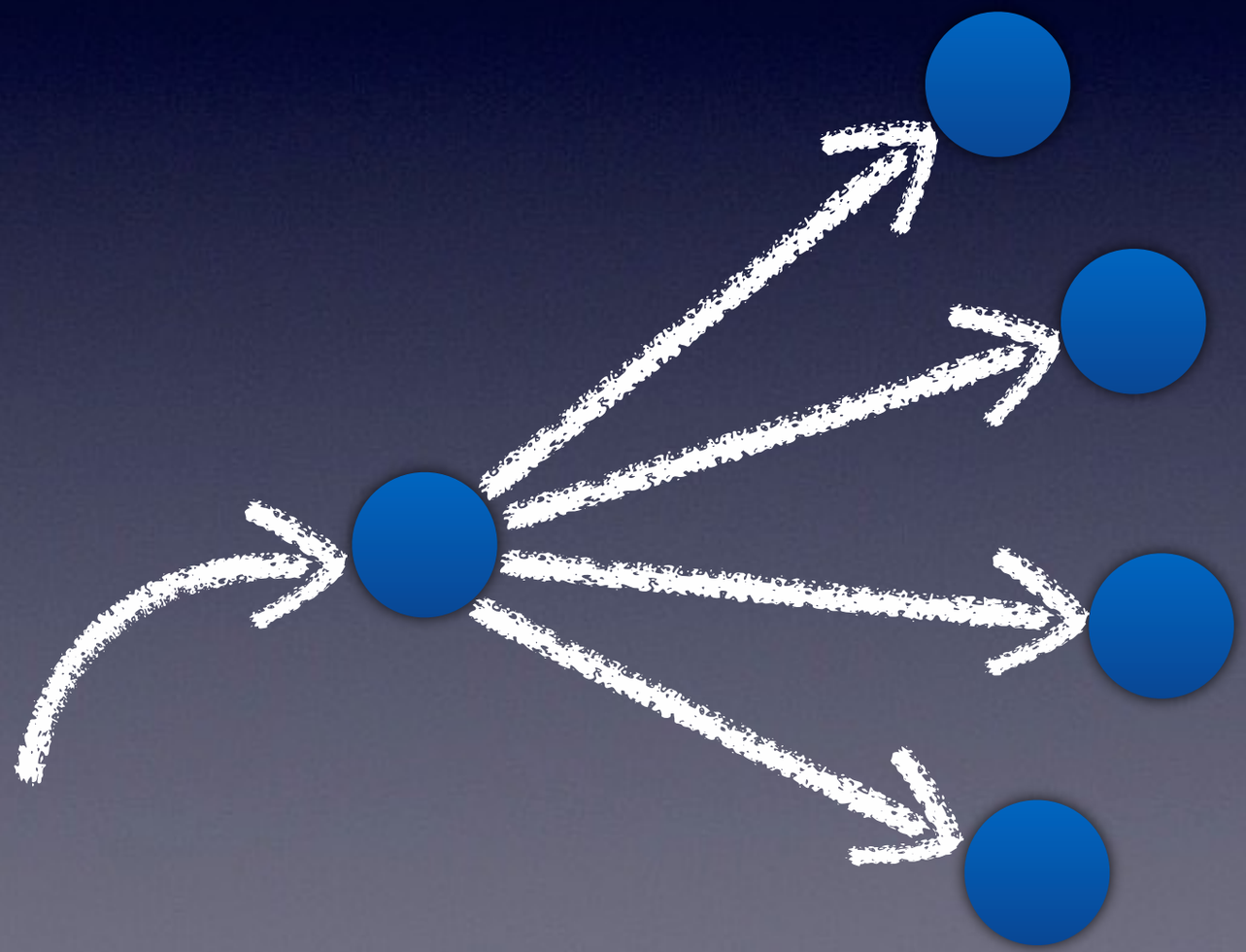
Non-Deterministic Action Outcomes



All-Outcomes Determinization



All-Outcomes Determinization



FOND Plans

- **Strong Cyclic Plan**: Policy of actions that achieves the goal, possibly revisiting a state

- **Weak Plan**: Works for at least one set of outcomes for the non-deterministic actions

- **Strong Plan**: Policy of actions that achieves the goal and never revisits the same state

SAS+ FOND Plans

- **Strong Cyclic Plan**: Policy of actions that achieves the goal, possibly revisiting a state

- **Weak Plan**: Works for at least one set of outcomes for the non-deterministic actions

- ~~**Strong Plan**: Policy of actions that achieves the goal and never revisits the same state~~

Approaches

- Symbolic: MBP, Grendel
- Value or Policy Iteration (VI / PI)
- Plan Aggregation: NDP, FIP, PRP

SAS+ FOND Task

$\mathcal{V}$: Variable

s_0 : Initial state

s_* : Goal state

$\mathcal{A}$: Actions

$\forall v \in \mathcal{V},$

$s(v) \in D_v \cup \{\bot\}$

$\forall a \in \mathcal{A},$

$a = \langle \mathrm{Pre}_a, \mathrm{Eff}_a \rangle$

SAS+ FOND Task

$\mathcal{V}$: Variable

s_0 : Initial state

s_* : Goal state

$\mathcal{A}$: Actions

$\forall v \in \mathcal{V},$

$s(v) \in D_v \cup \{\bot\}$

$\forall a \in \mathcal{A},$

$a = \langle \mathrm{Pre}_a, \mathrm{Eff}_a \rangle$

Set of possible outcomes

Triangle Tireworld

Drivable car

Road

Location with tire

Location without

Goal location

Triangle Tireworld

Drivable car

Road

E.g.,
at = B
flat = T
hasTireA = T

Location with tire

Location without

Goal location

Triangle Tireworld

Drivable car

Road

STRIPS
- atB
- hasFlat
- hasTireA

E.g.,
- at = B
- flat = T
- hasTireA = T

Location with tire

Location without

Goal location

Triangle Tireworld

Drivable car

Road

E.g.,
at = B
flat = T
hasTireA = T

***E.g.*, drive_A_B**

Pre: [at=A, flat=F]

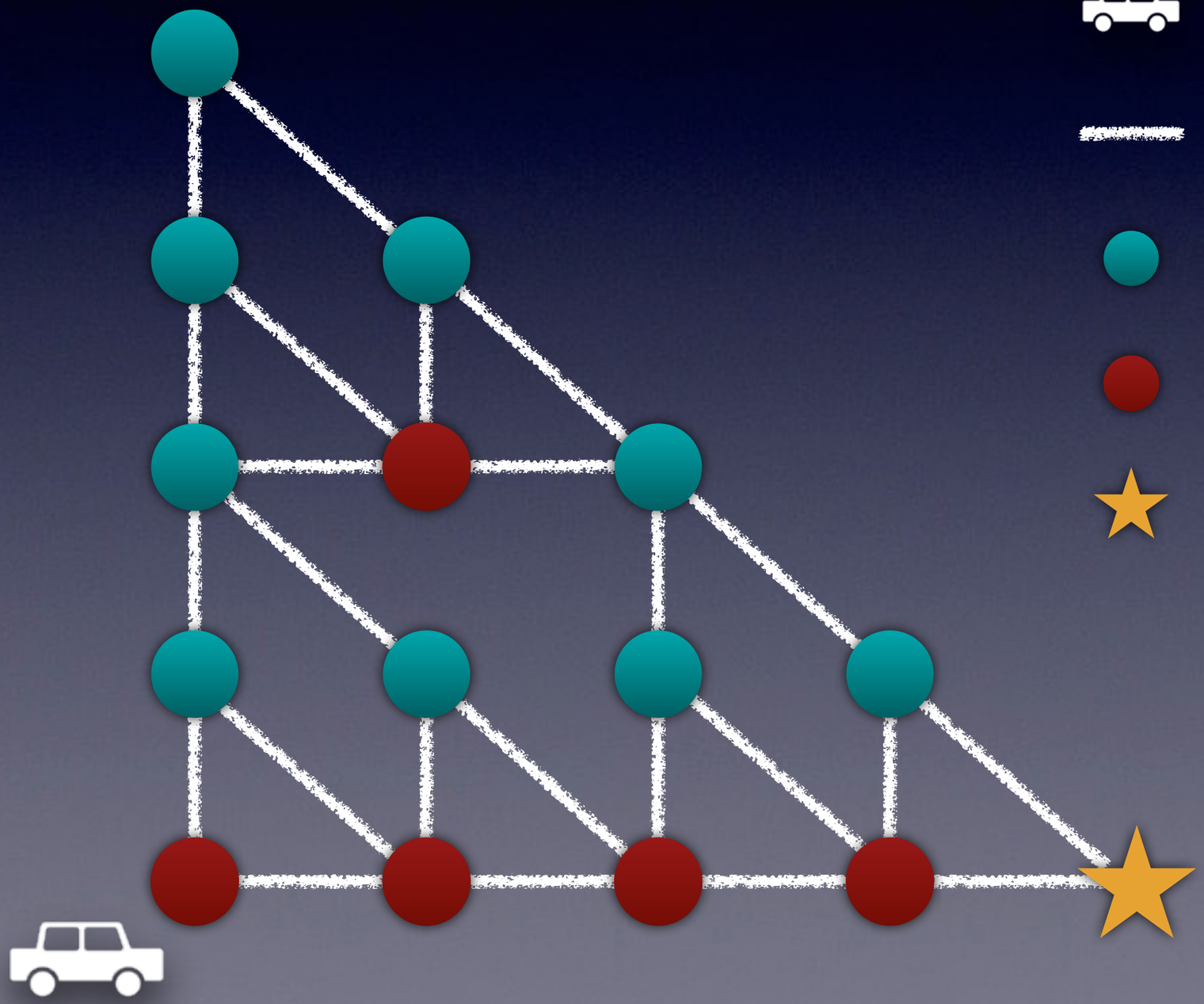
Eff: { [at=B],
[at=B, flat=T] }

Location with tire

Location without

Goal location

Triangle Tireworld



Drivable car

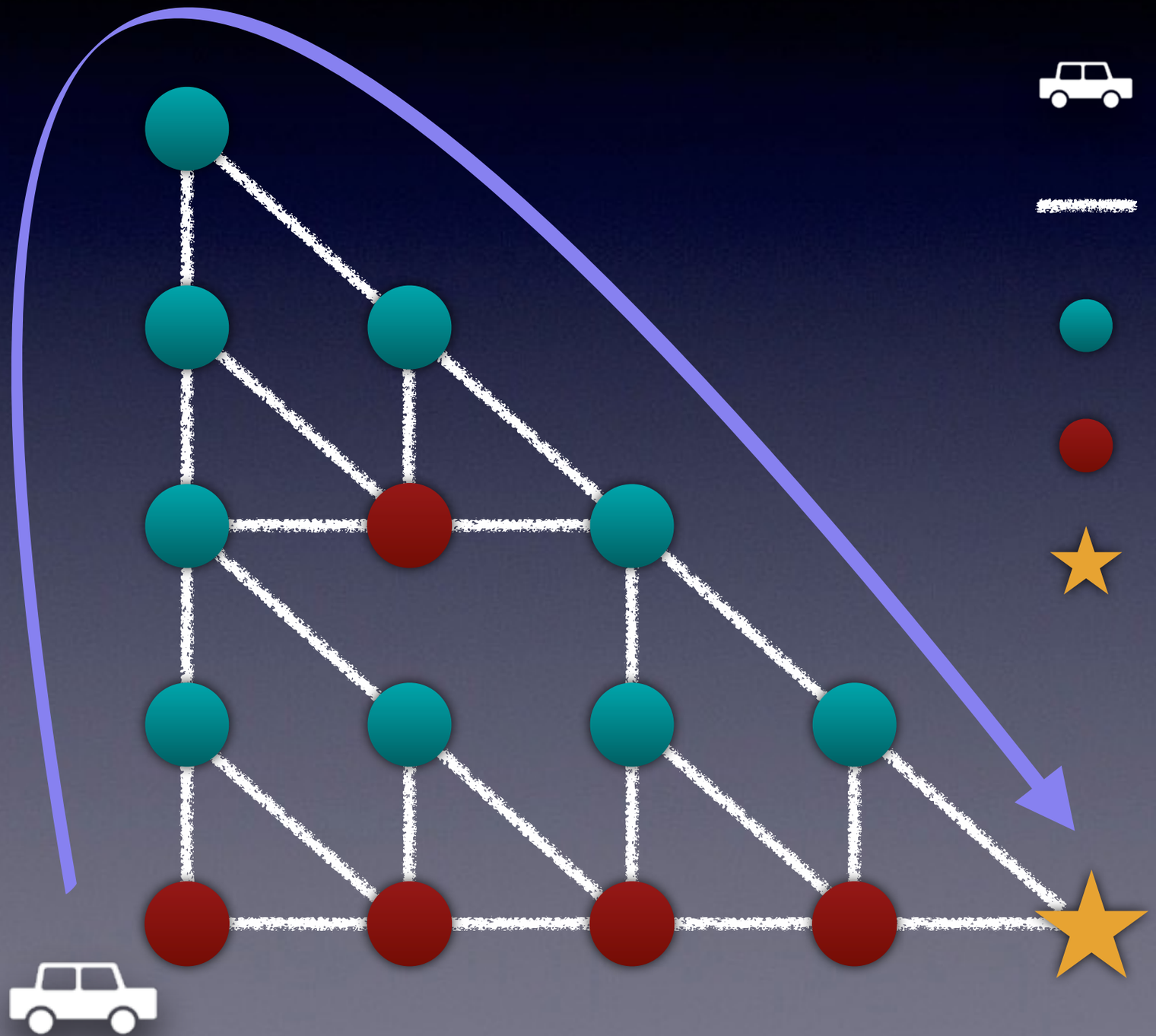
Road

Location with tire

Location without tire

Goal location

Triangle Tireworld



Drivable car

Road

Location with tire

Location without tire

Goal location

Contributions

- Introduce a state-of-the-art FOND planner: **PRP** (**P**lanner for **R**elevant **P**olicies)
- Develop a suite of principled methods to leverage the relevant parts of the state
- PRP is exponentially *faster*, produces exponentially more *succinct* policies, and is significantly more *robust*

Outline

- Approach
- Evaluation
- Conclusion

Outline

- **Approach**
  - Algorithm
  - Deadends
  - Planning locally
  - Strong cyclic confirmation

General Approach

- **Input**: SAS+ FOND Planning Task
- **Output**: Strong Cyclic Policy mapping states to actions
  - $P : S \rightarrow A$
    - Represented compactly as a set of pairs: (partial state, action)
    - Many partial states may be consistent with the current state, so the *best* pair is selected
    - We also record the *expected outcome* of the action

General Approach

Simulate Policy

General Approach

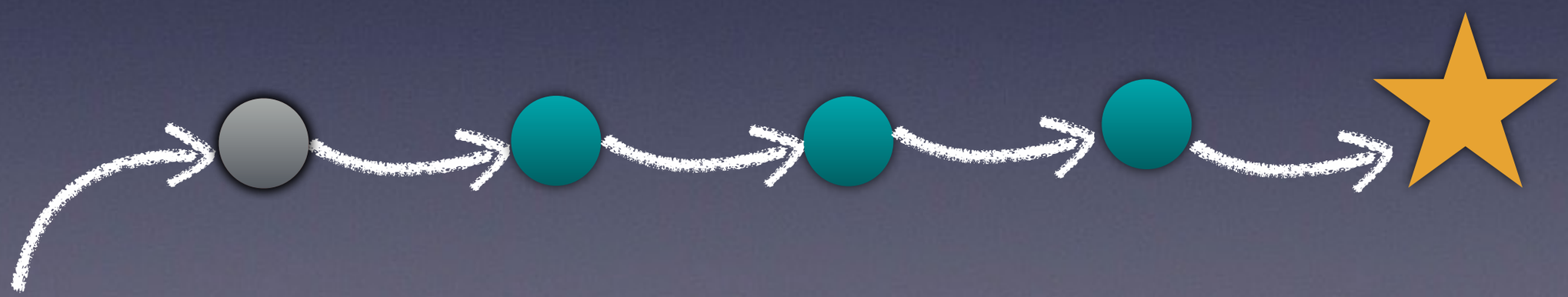
Simulate Policy

General Approach

Simulate Policy

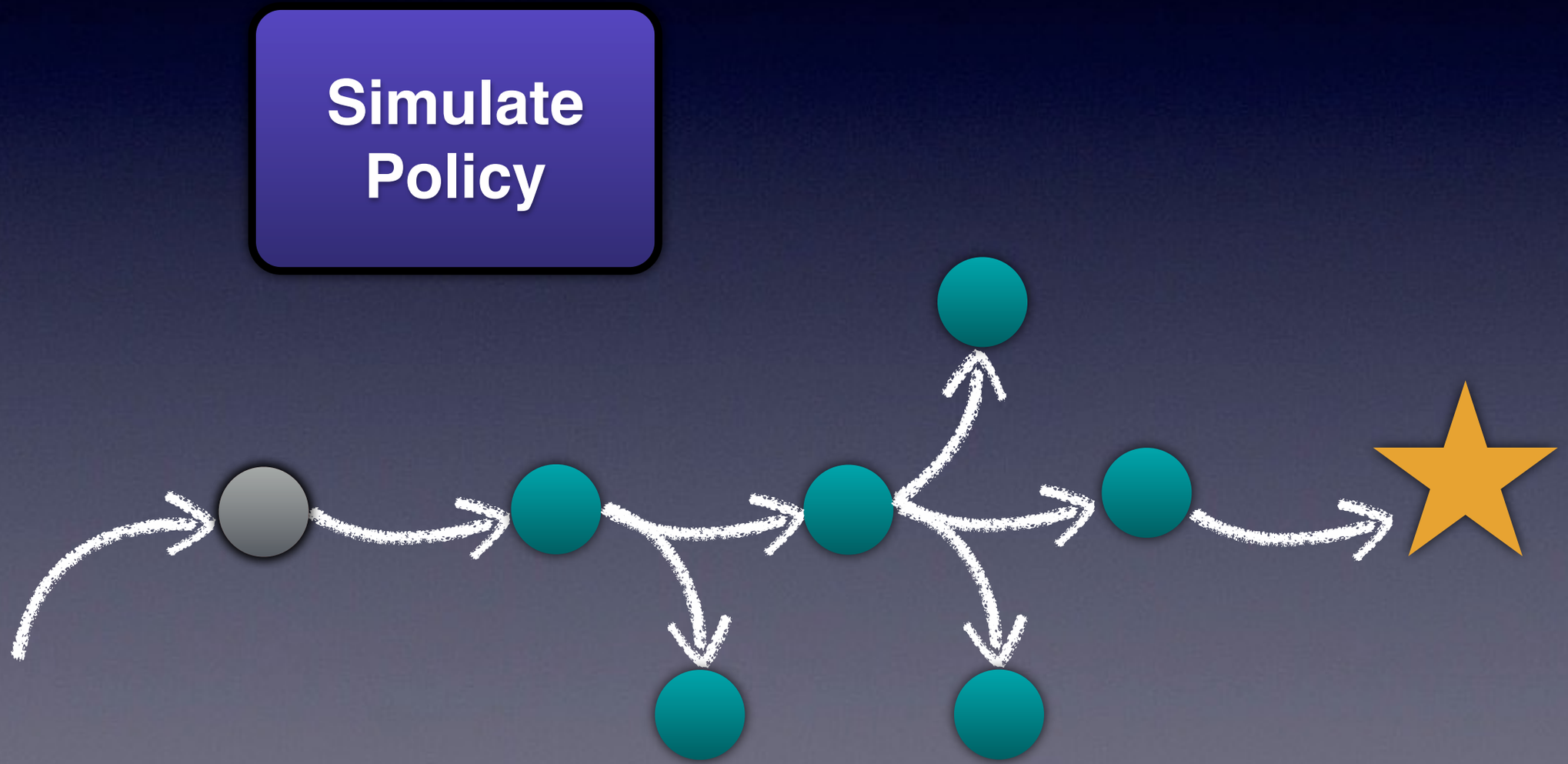
General Approach

Simulate Policy



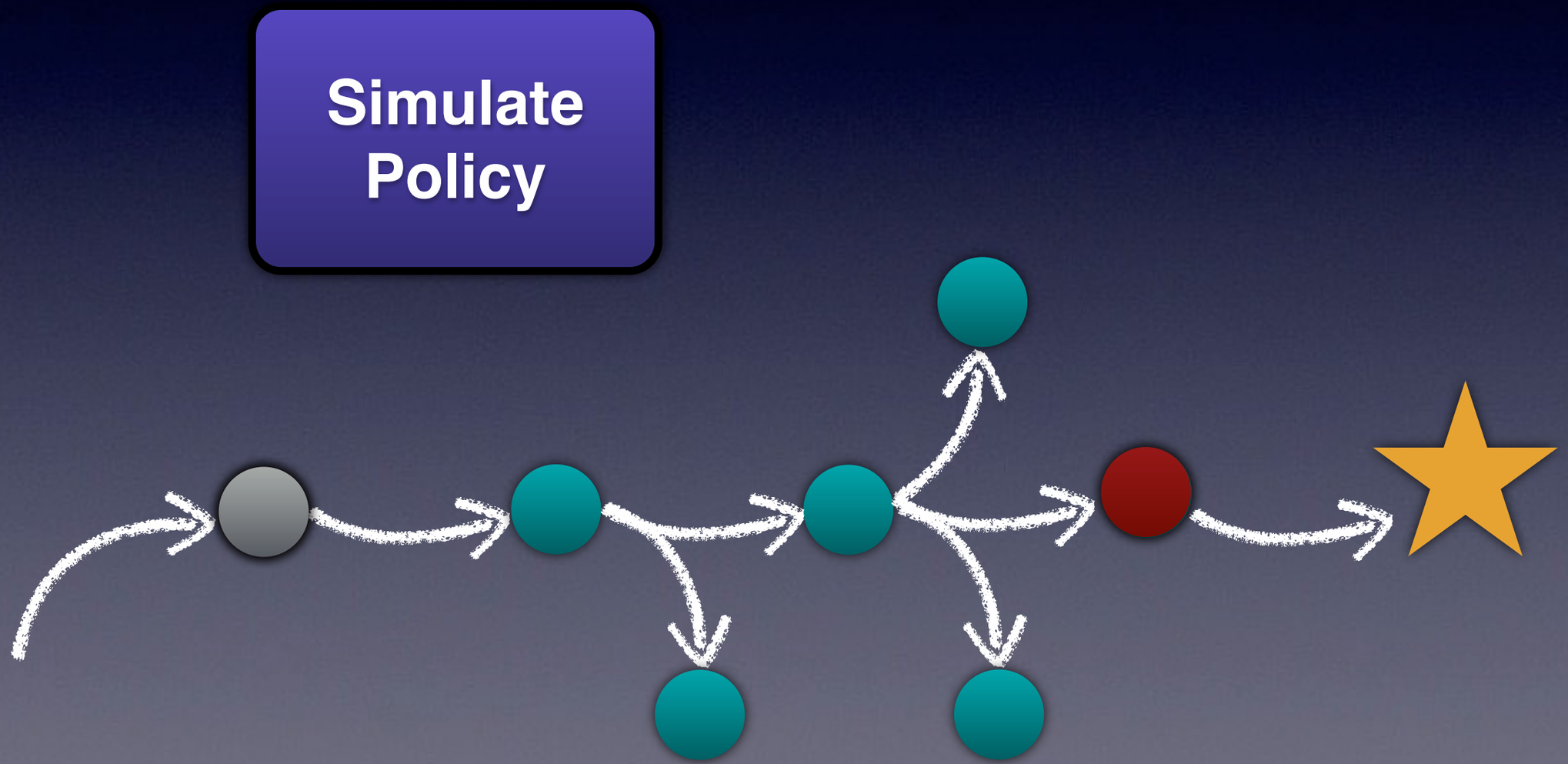
General Approach

Simulate Policy



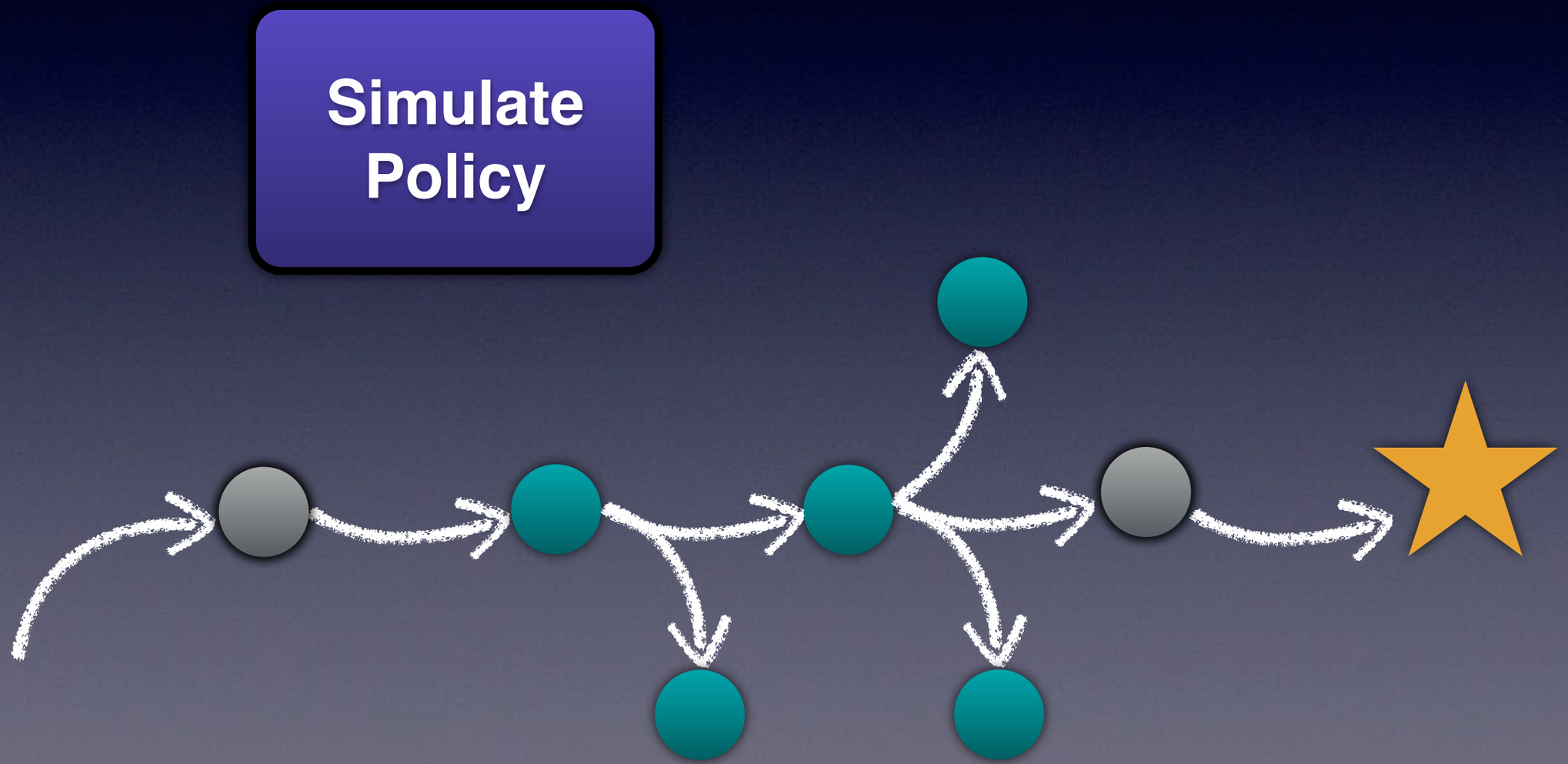
General Approach

Simulate Policy



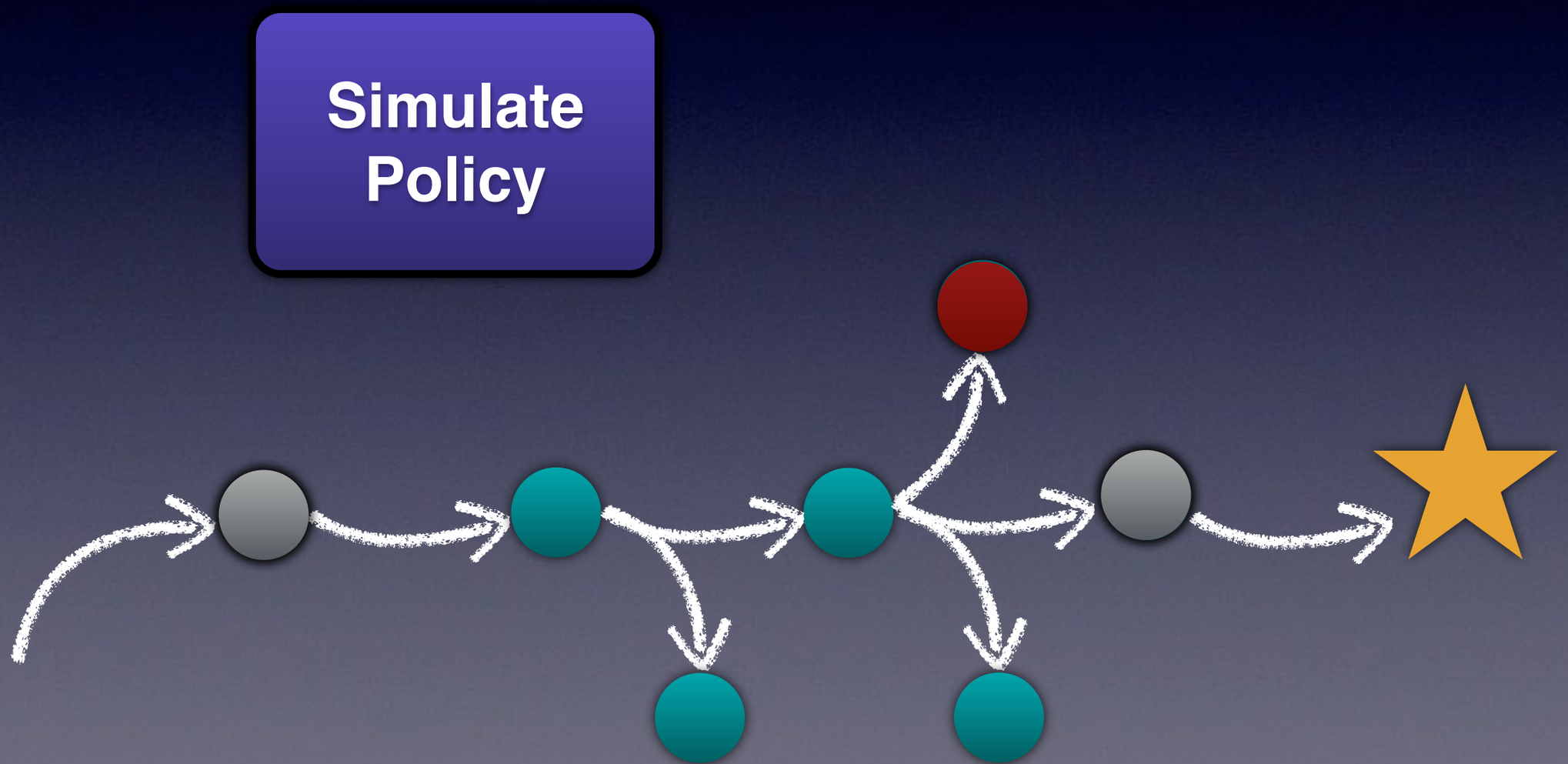
General Approach

Simulate Policy



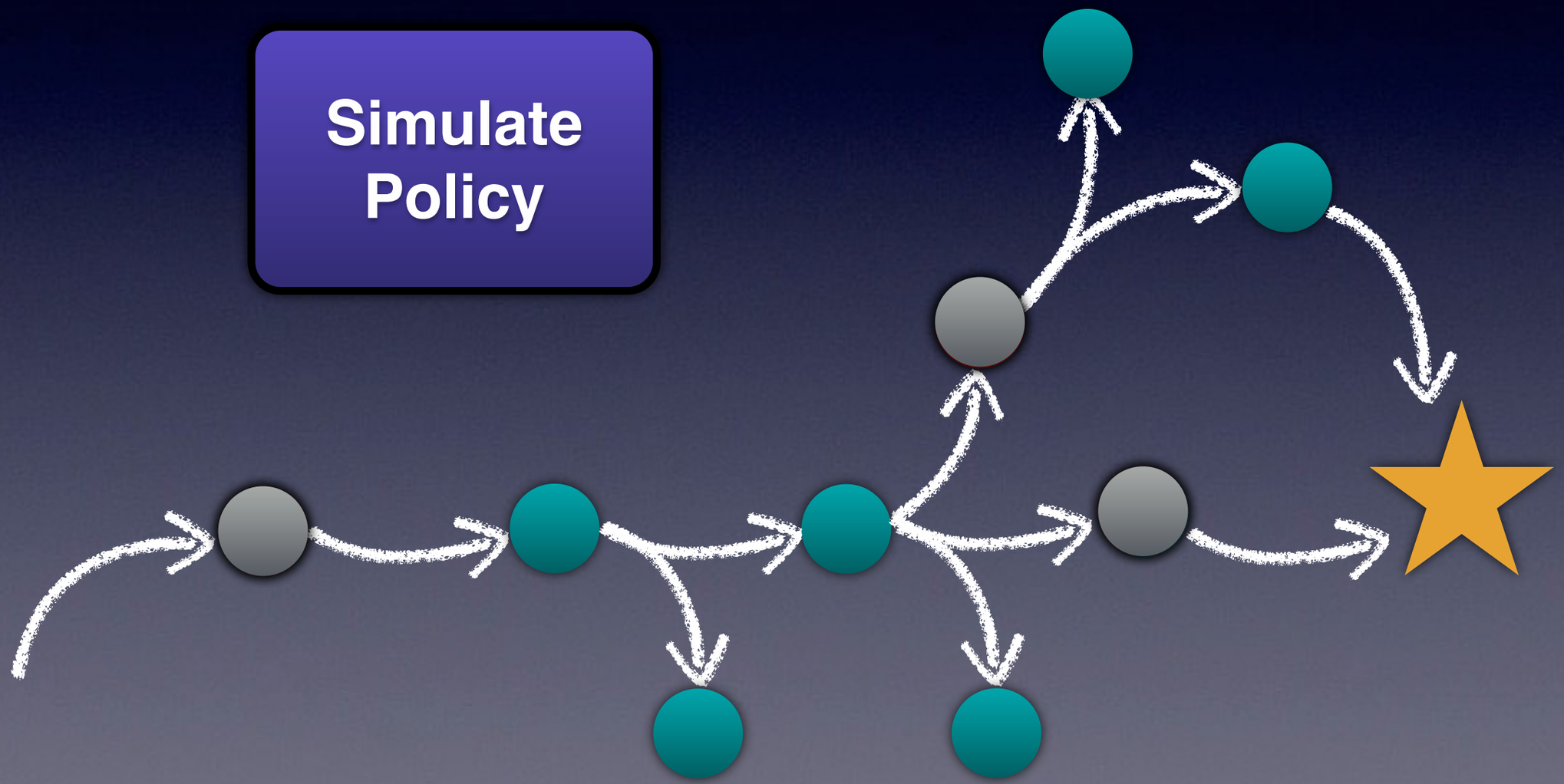
General Approach

Simulate Policy

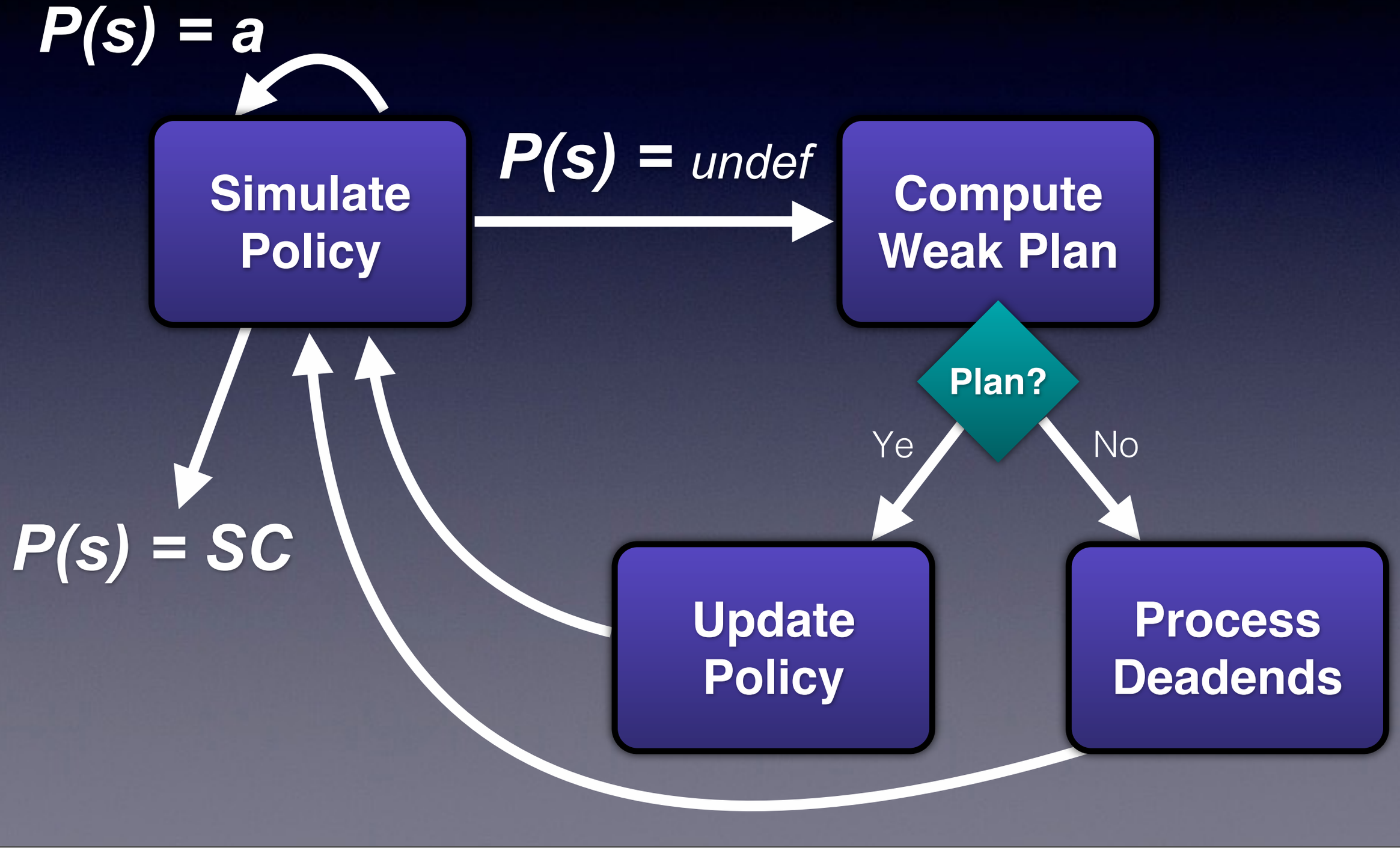


General Approach

Simulate Policy



General Approach



General Approach

Compute Weak Plan

General Approach

Compute Weak Plan

- Classical planner used on the *determinization*
- Forbidden state-action pairs are avoided
- Planning is halted when the policy matches a state in the search space

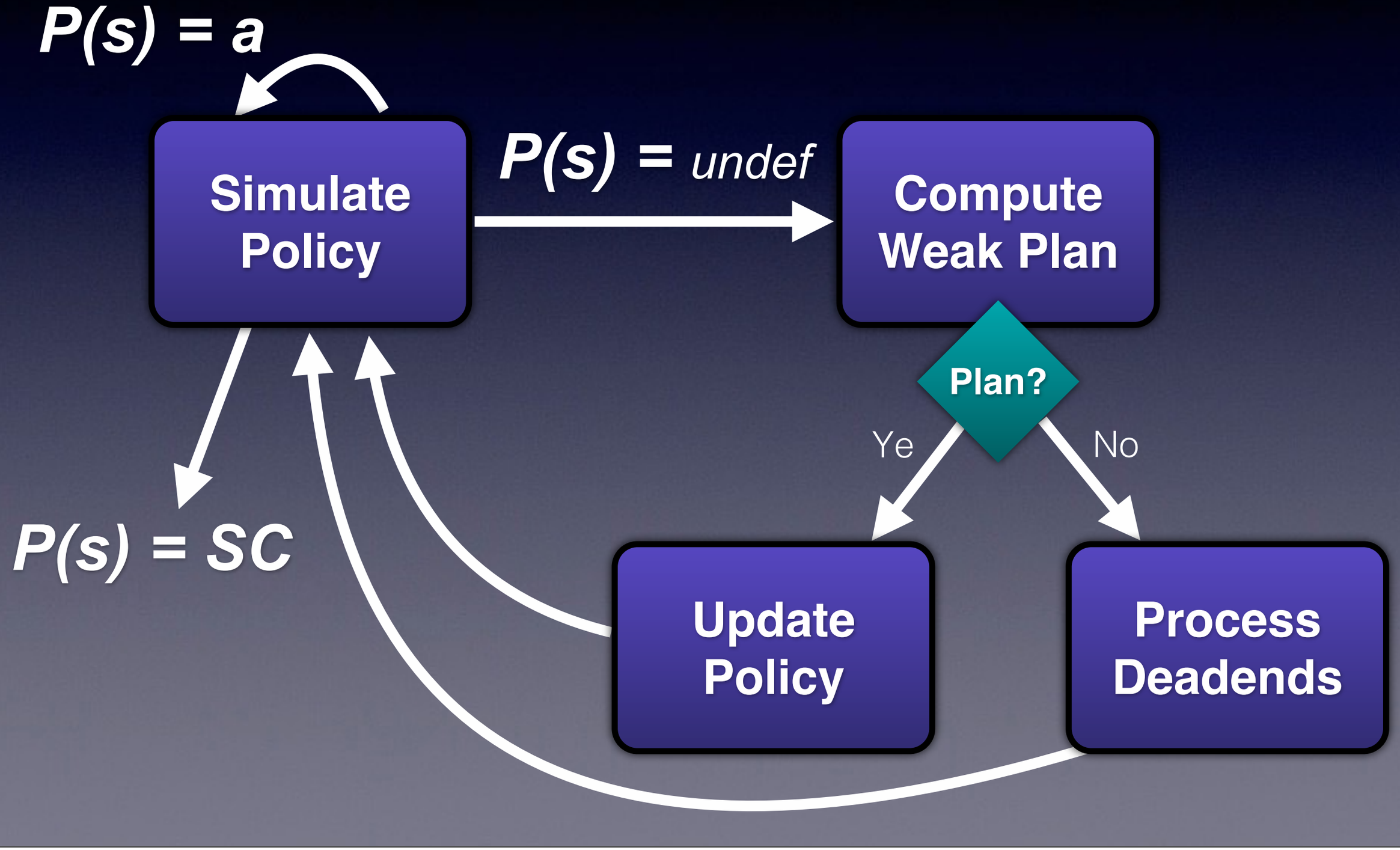
General Approach

Compute Weak Plan

Thm: If the policy returns an action for a state, the policy can be used to compute a weak plan.

- Classic *inization*
- Forbid ded
- Planning is halted when the policy matches a state in the search space

General Approach



General Approach

General Approach

1. Compute the **relevant** conditions (subset of the state) for every suffix of the weak plan to reach the goal using *regression*:

$$Regr(\phi, a, e) = (\phi - e) + Pre_a$$

Update Policy

General Approach

1. Compute the **relevant** conditions (subset of the state) for every suffix of the weak plan to reach the goal using *regression*:

$$Regr(\phi, a, e) = (\phi - e) + Pre_a$$

E.g.,

Regr([at=B, hasTireB=T],
drive_A_B,
[at=B])

= [at=A, flat=F, hasTireB=T]

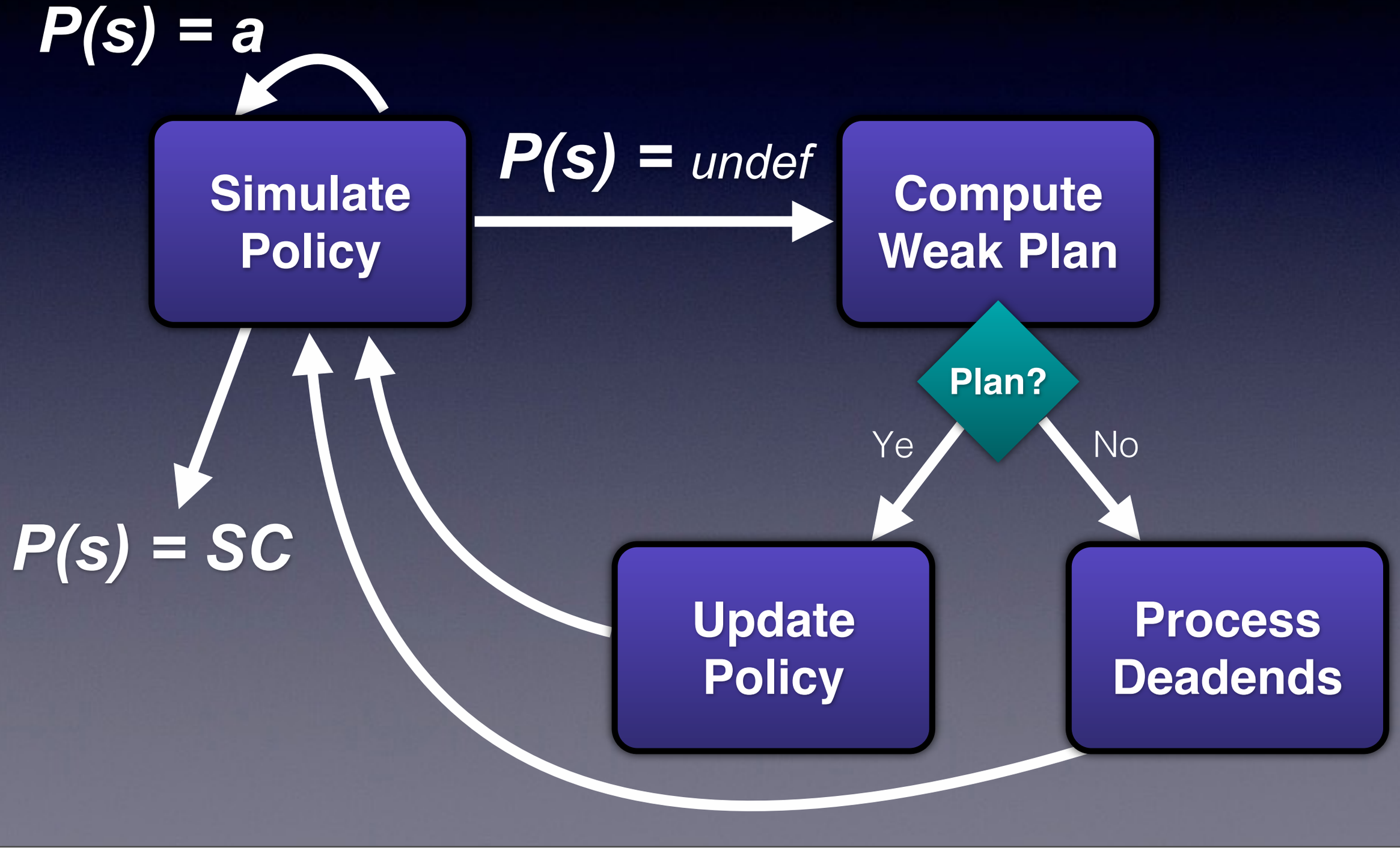
General Approach

1. Compute the **relevant** conditions (subset of the state) for every suffix of the weak plan to reach the goal using *regression*:
2. Add every condition and corresponding action to the policy:

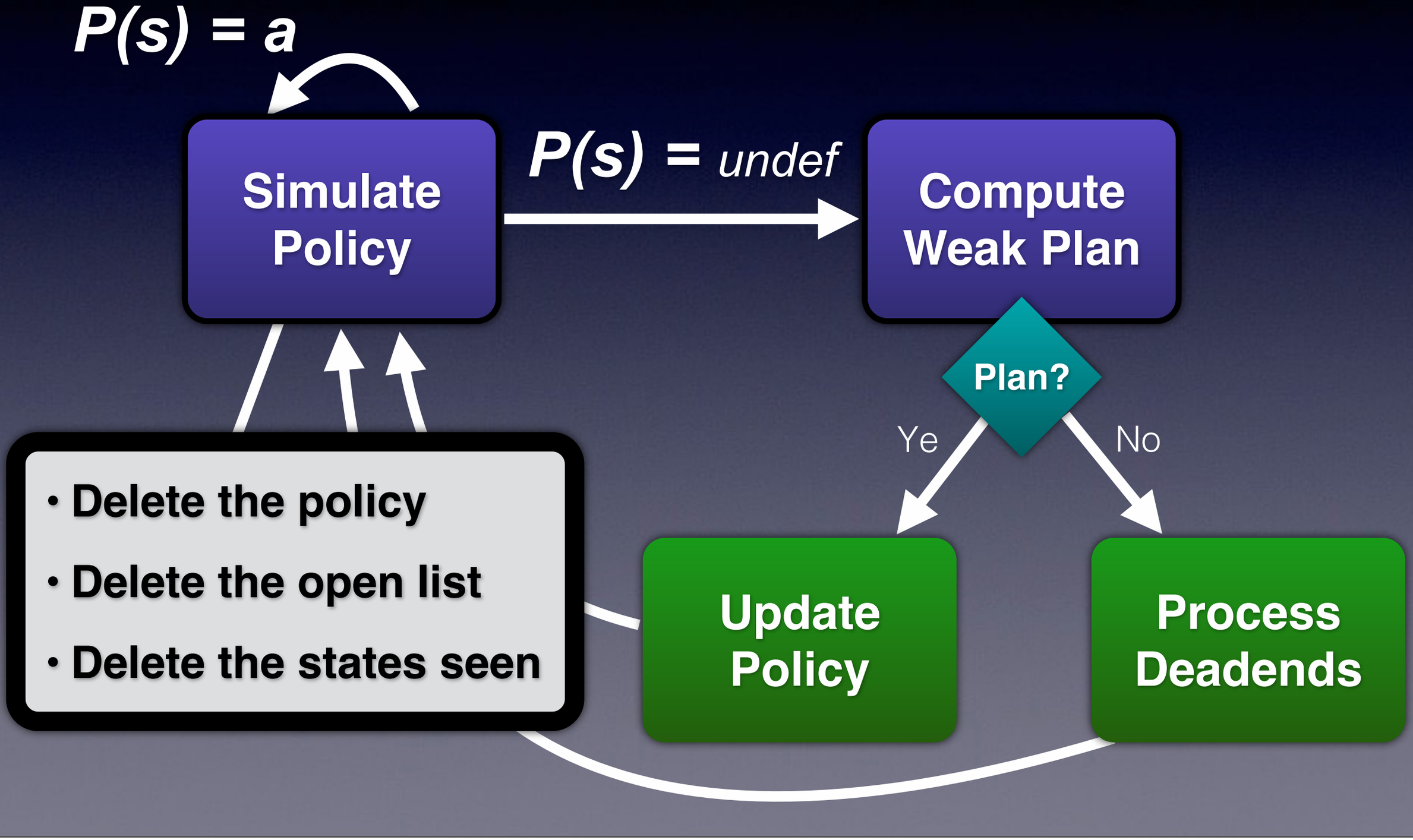
- Quality is measured as distance-to-goal

Update Policy

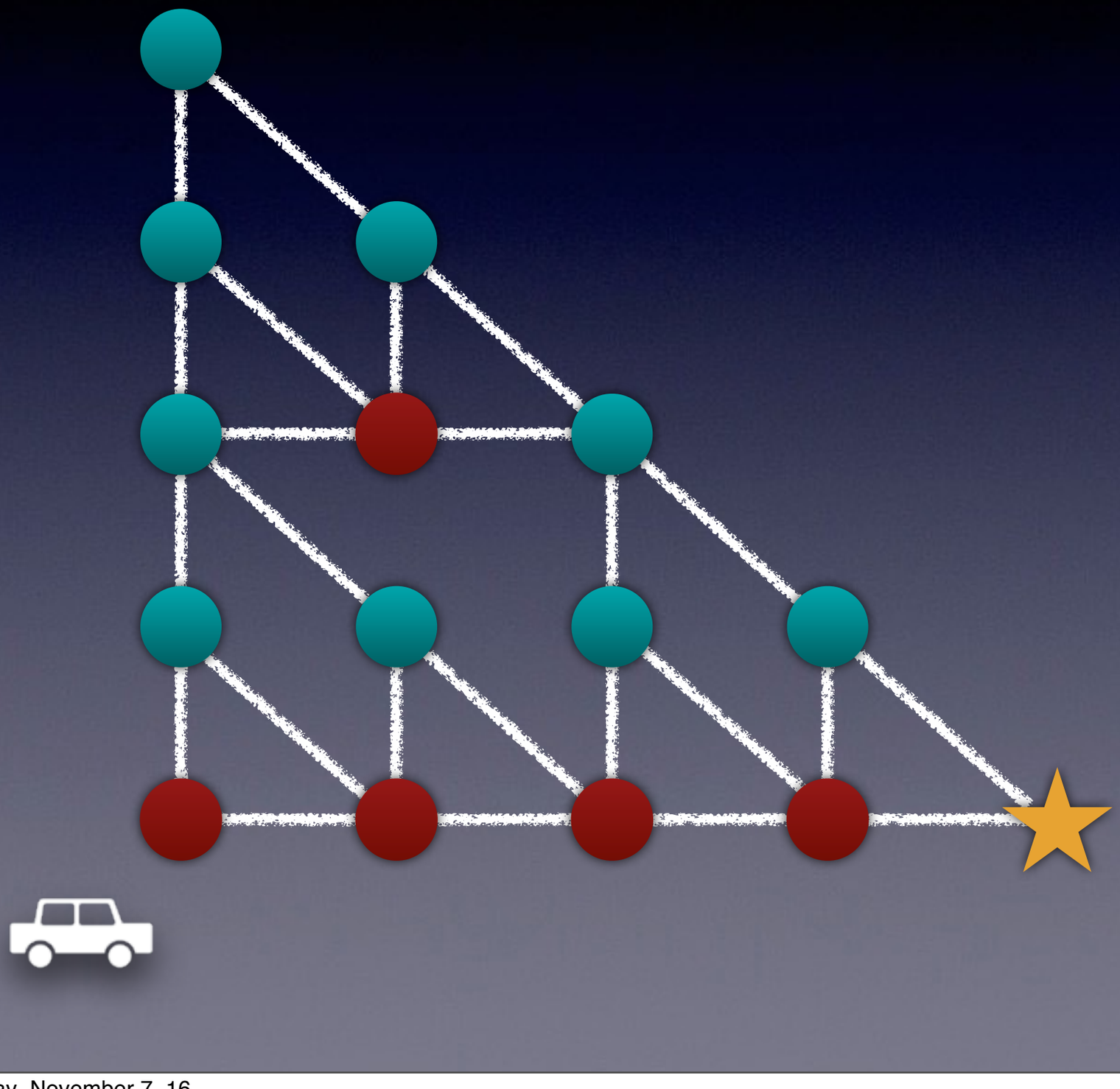
General Approach



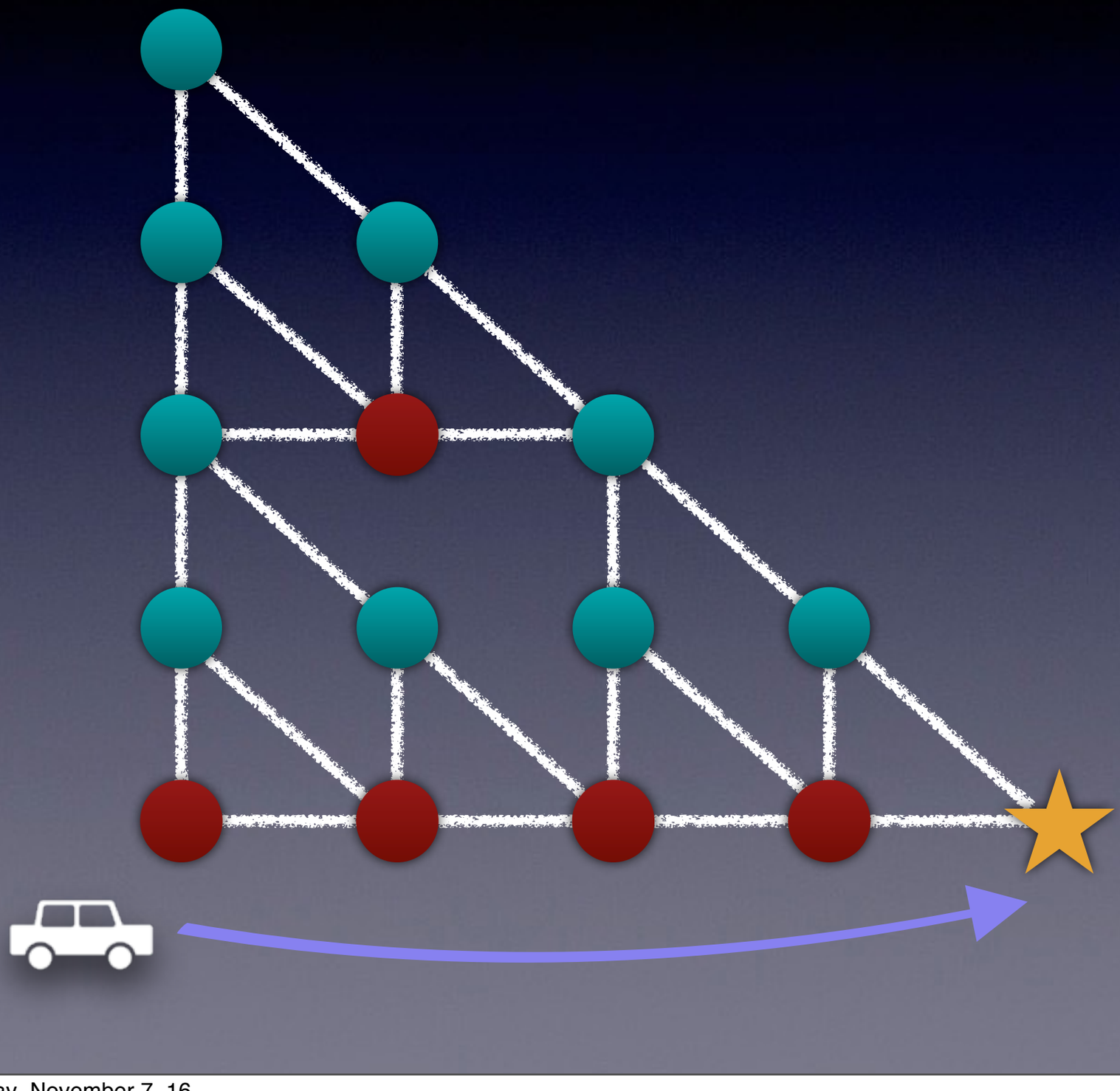
General Approach



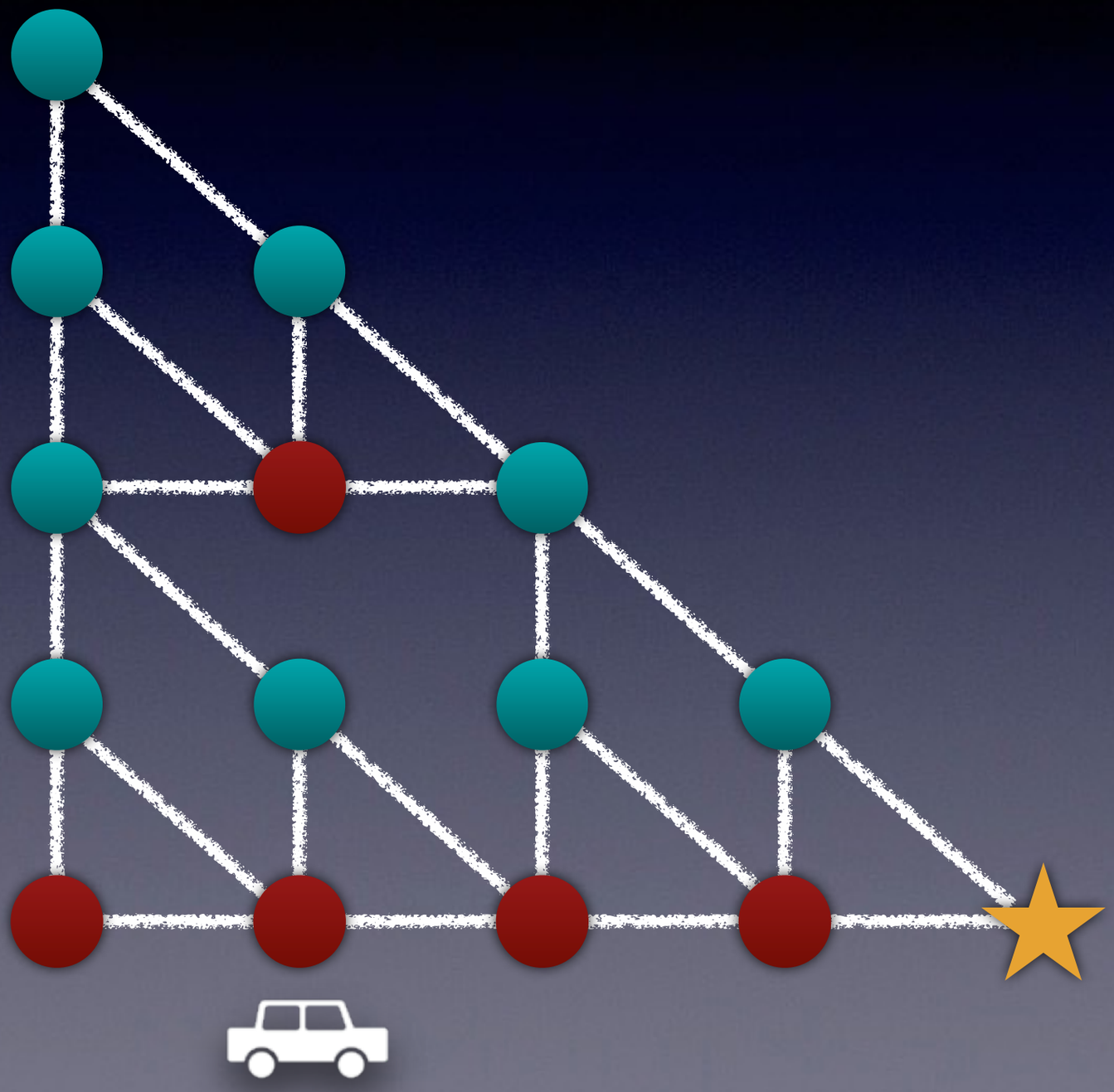
Deadends



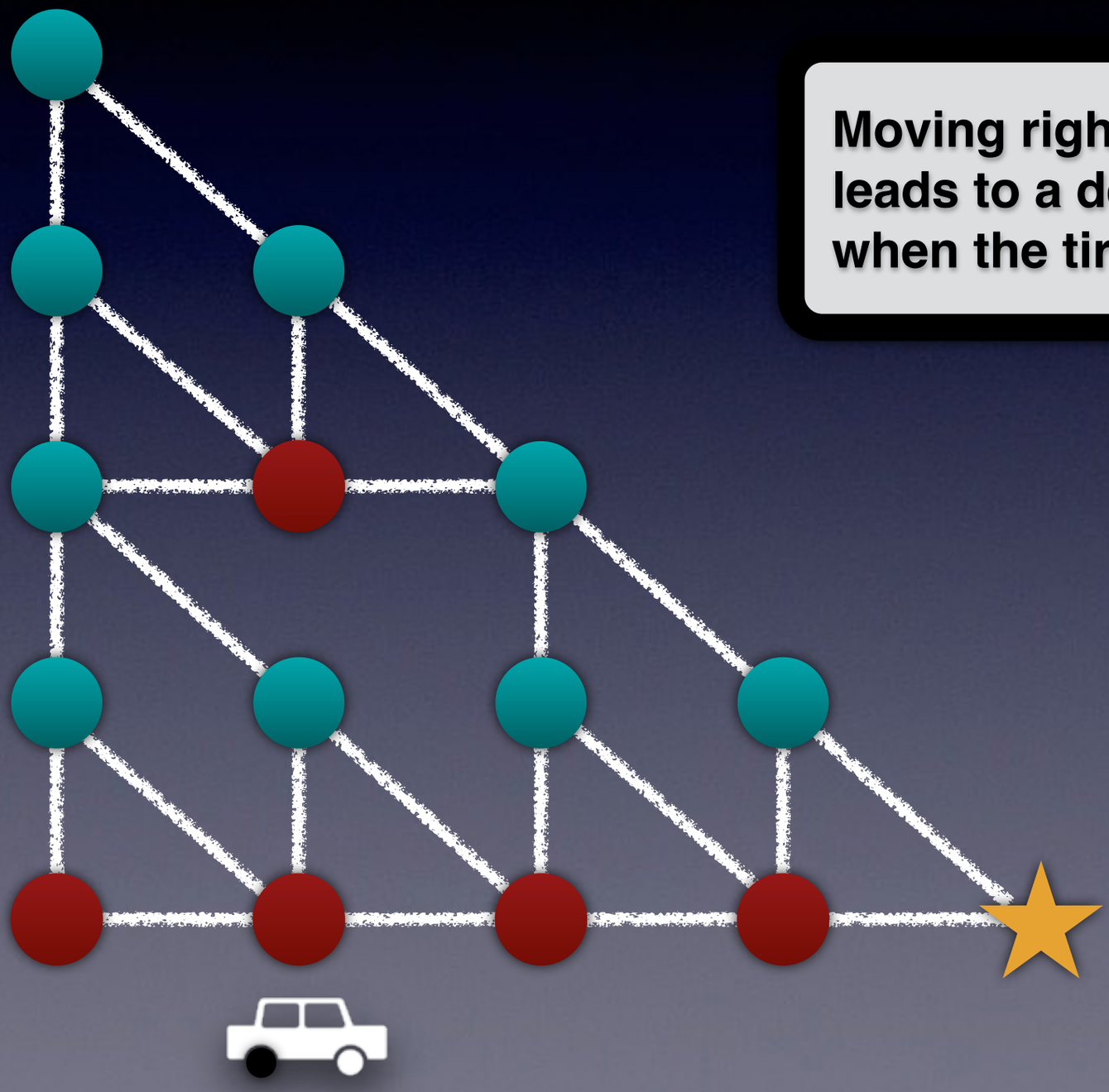
Deadends



Deadends

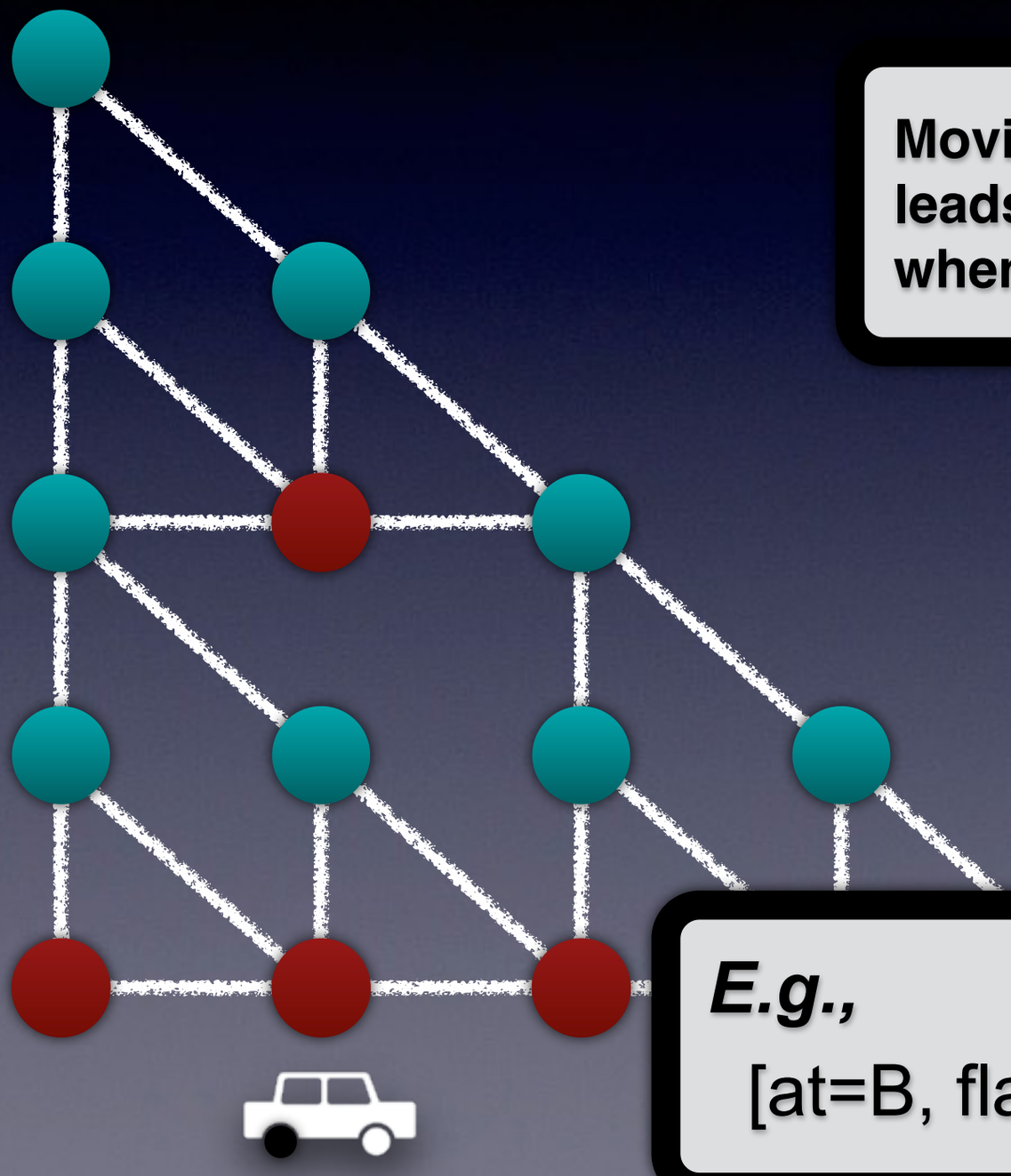


Deadends



Moving right once leads to a deadend when the tire blows

Deadends



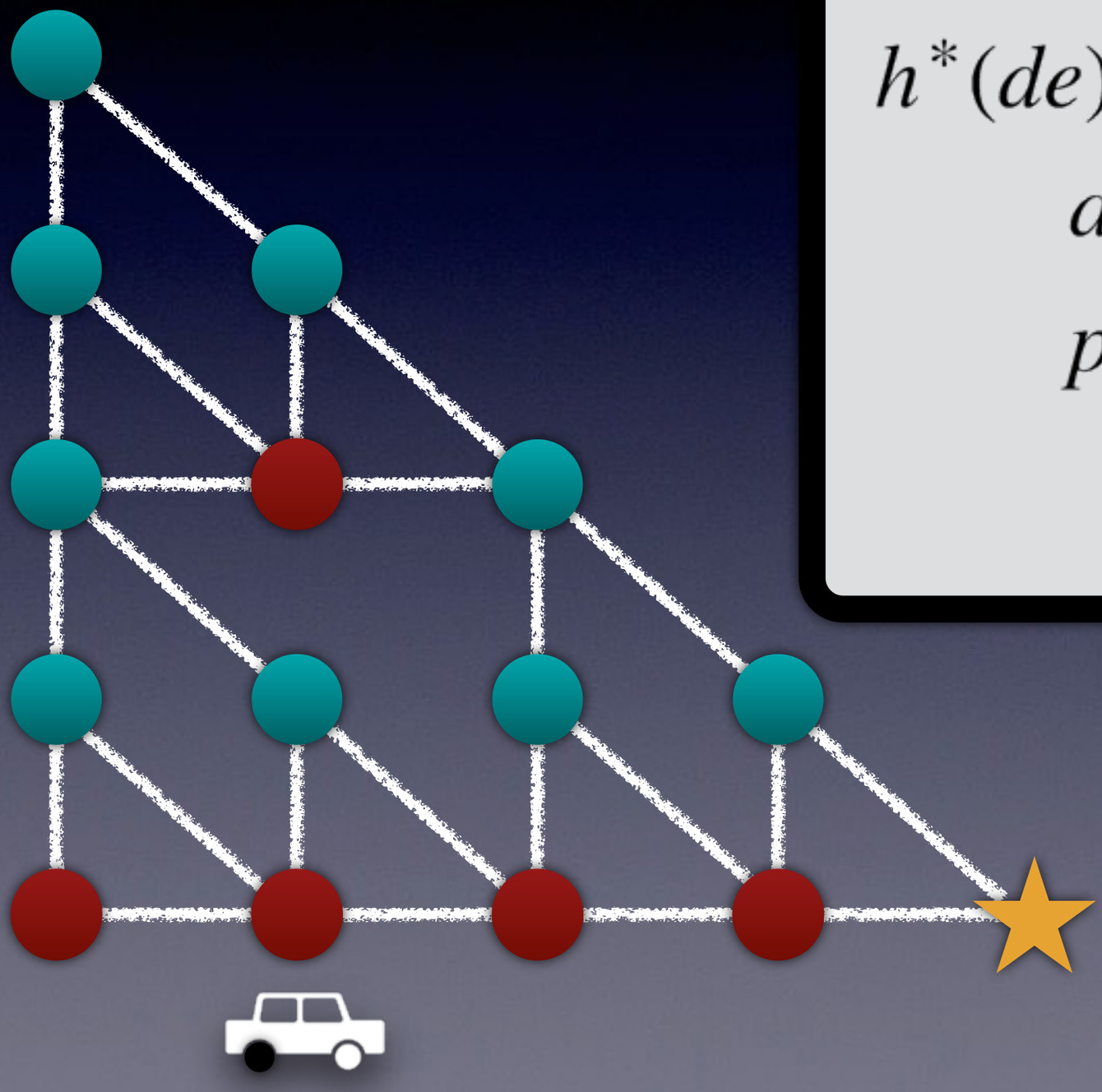
Moving right once leads to a deadend when the tire blows

Deadend includes the entire state, so we *generalize* to a relevant subset

E.g.,

[at=B, flat=T, hasTireB=F]

Deadends



Forbidden State-action Pair

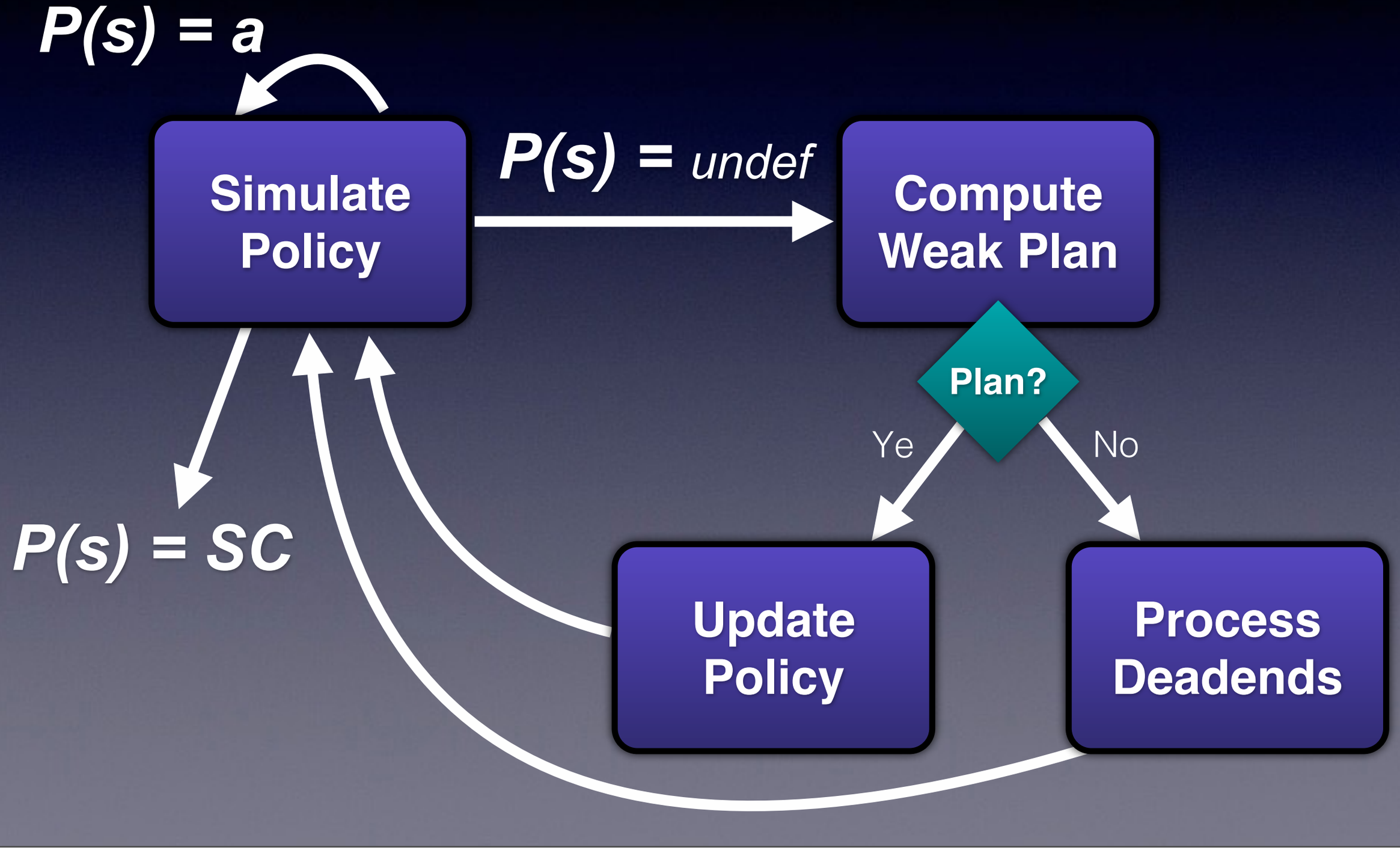
$$h^*(de) = \infty$$

$$a \in \mathcal{A}, e \in Eff_a$$

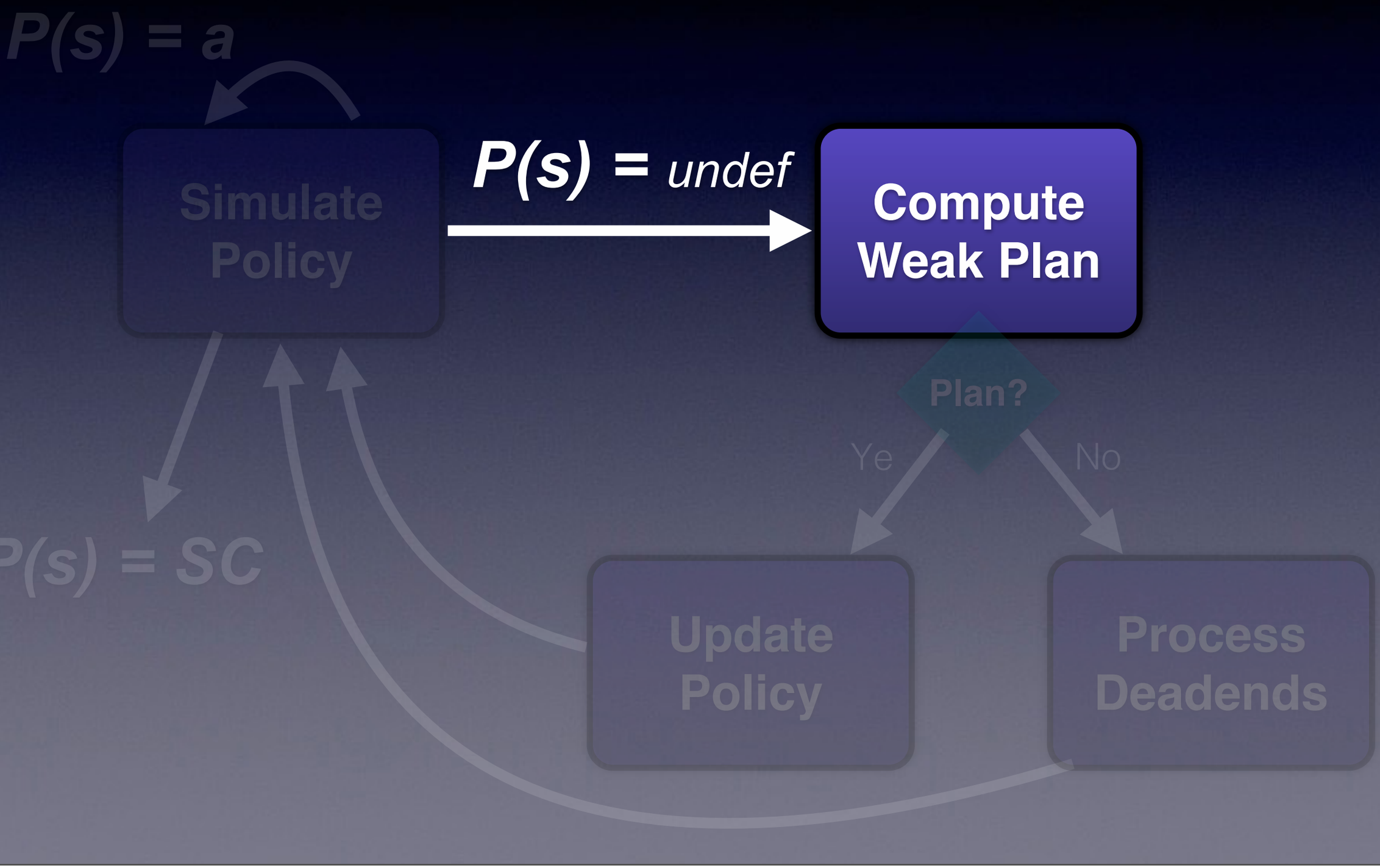
$$p = Regr(de, a, e)$$

$$\langle p, a \rangle$$

General Approach

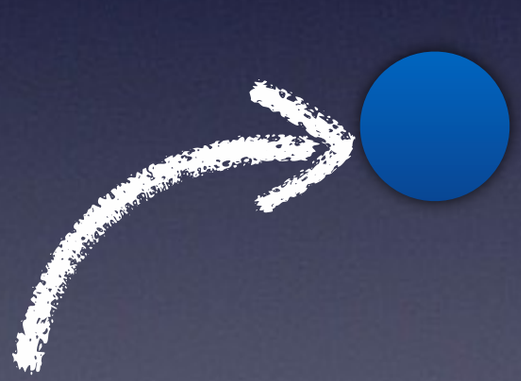


Planning Locally

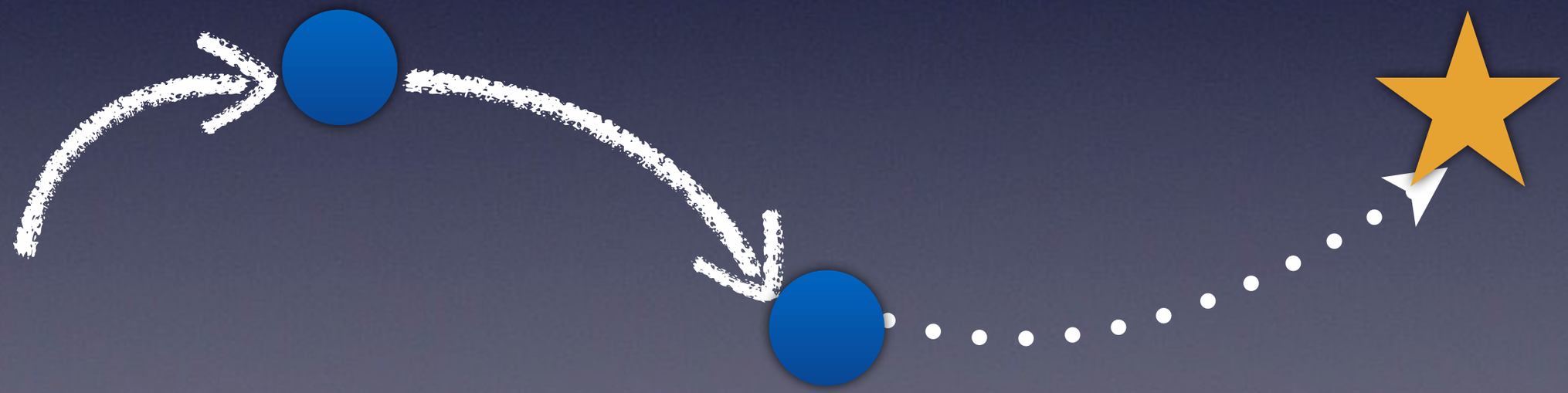


Planning Locally

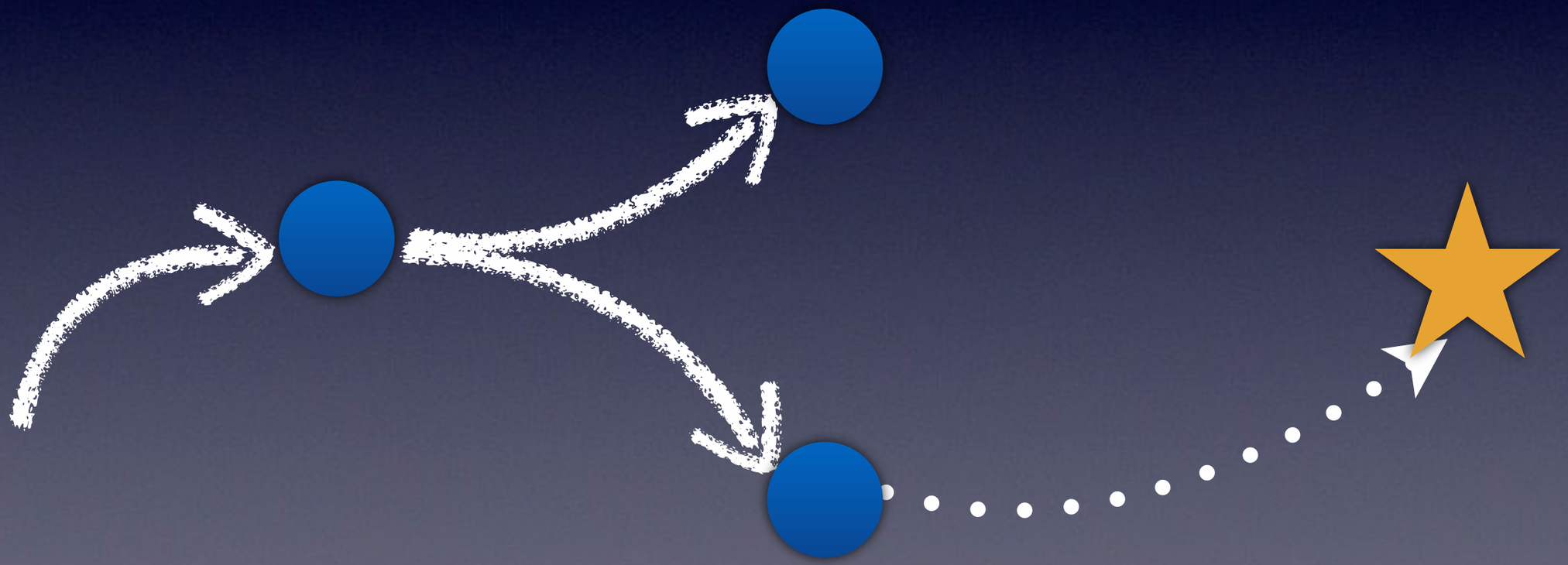
Planning Locally



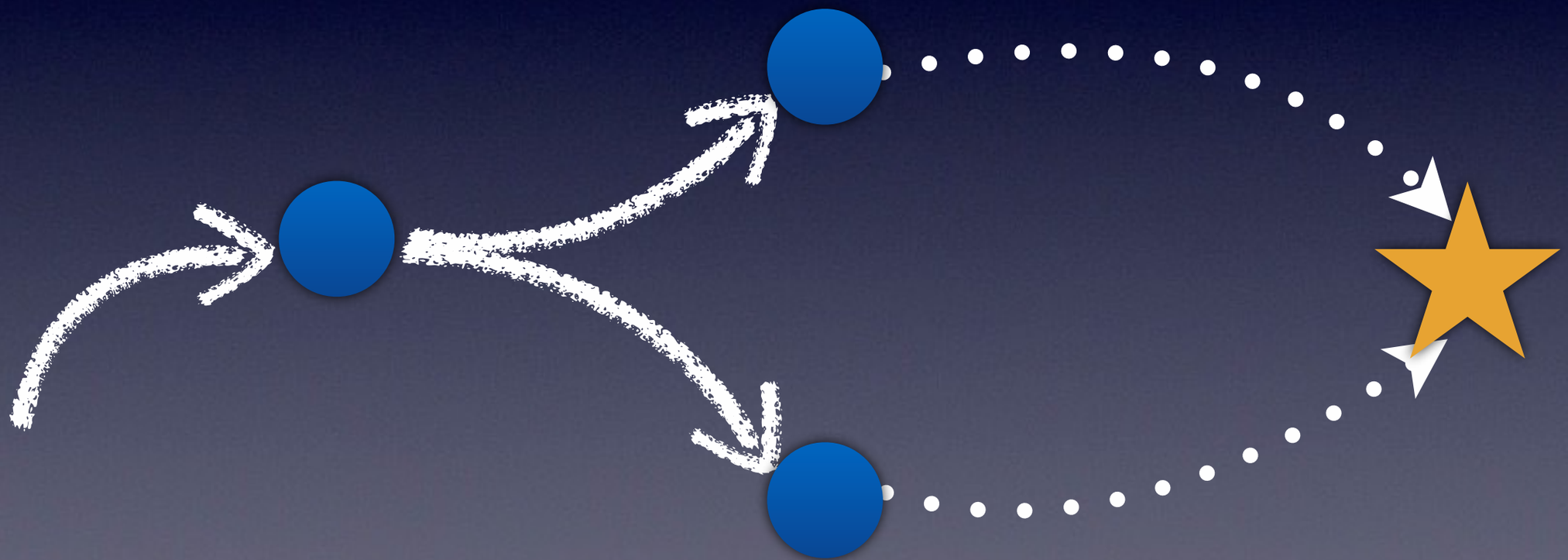
Planning Locally



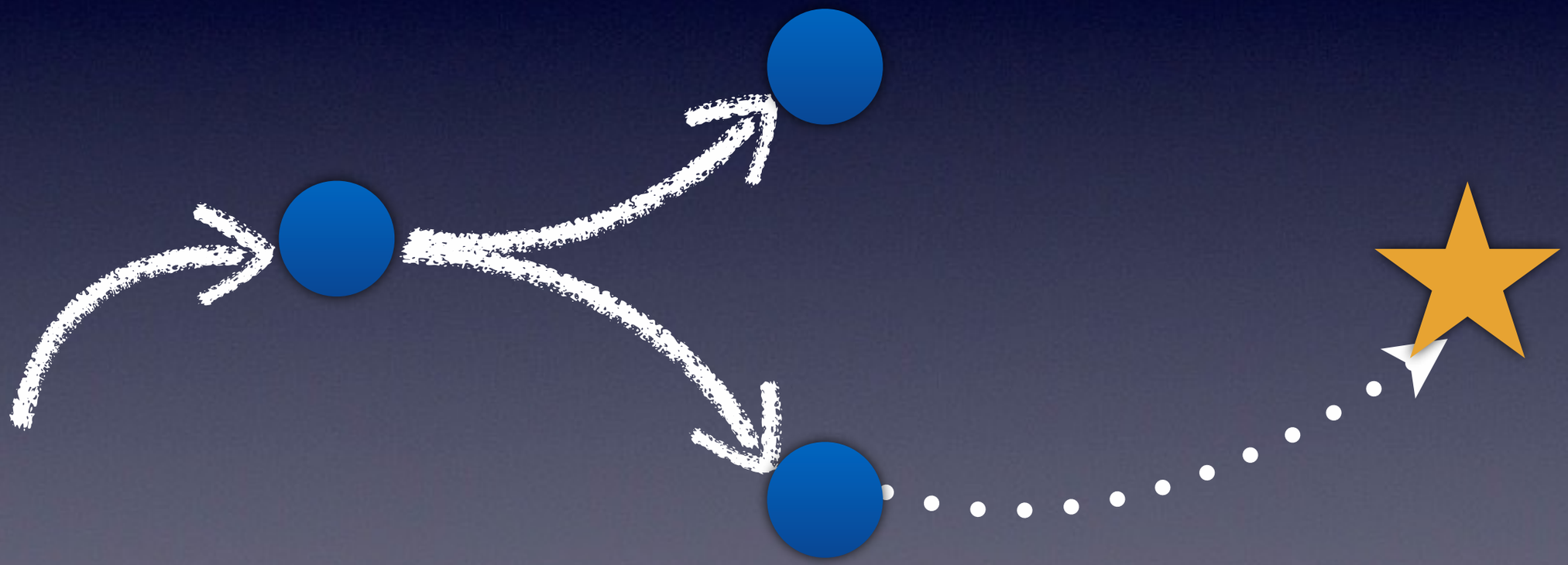
Planning Locally



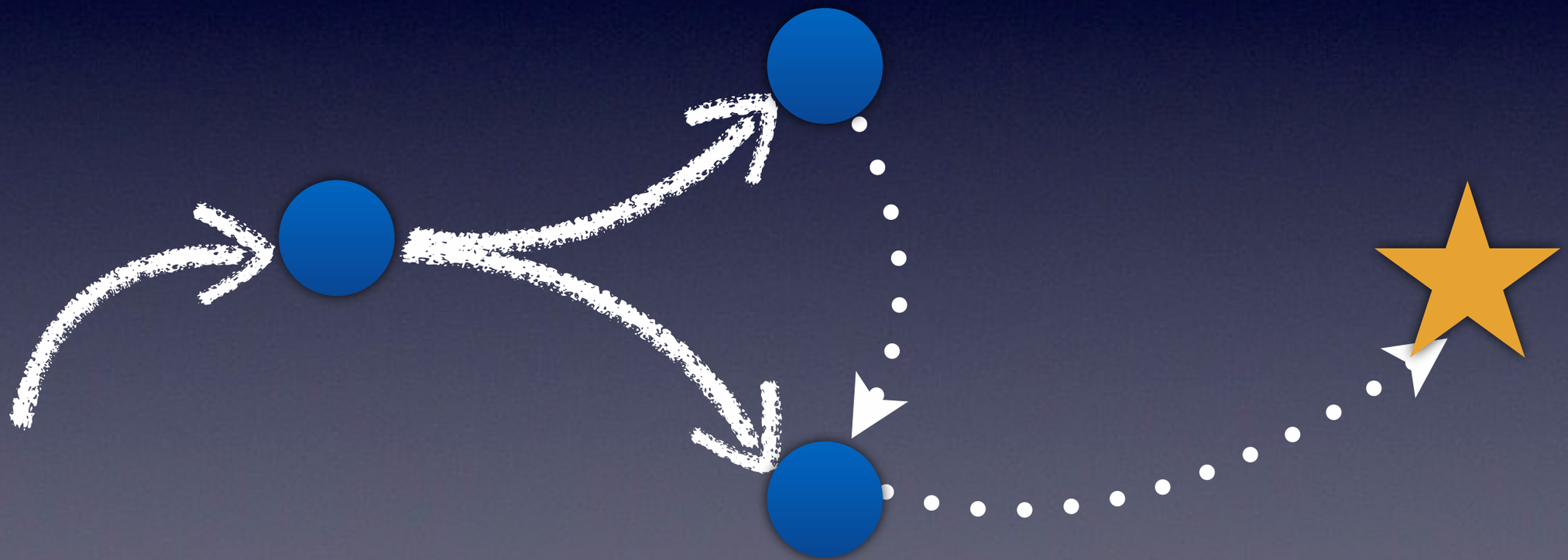
Planning Locally



Planning Locally

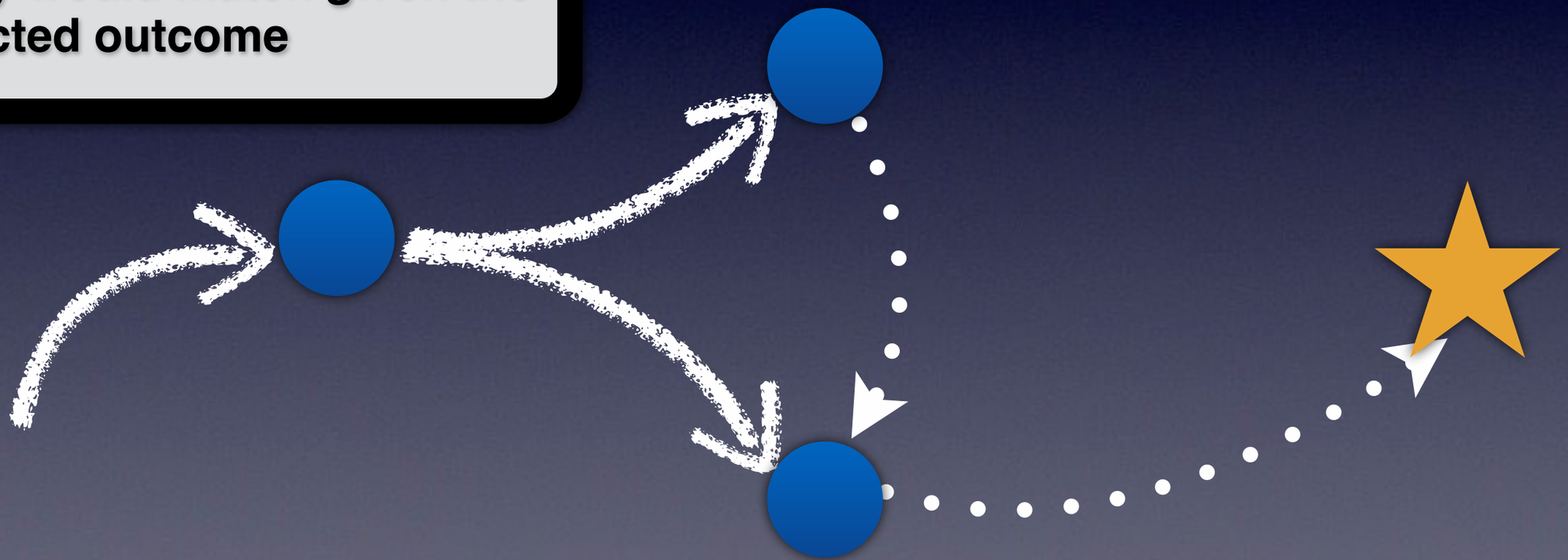


Planning Locally



Planning Locally

New goal is the *relevant* part of the expected state that the policy would match given the expected outcome



Planning Locally

New goal is the *relevant* part of the expected state that the policy would match given the expected outcome

E.g., ***drive_A_B***

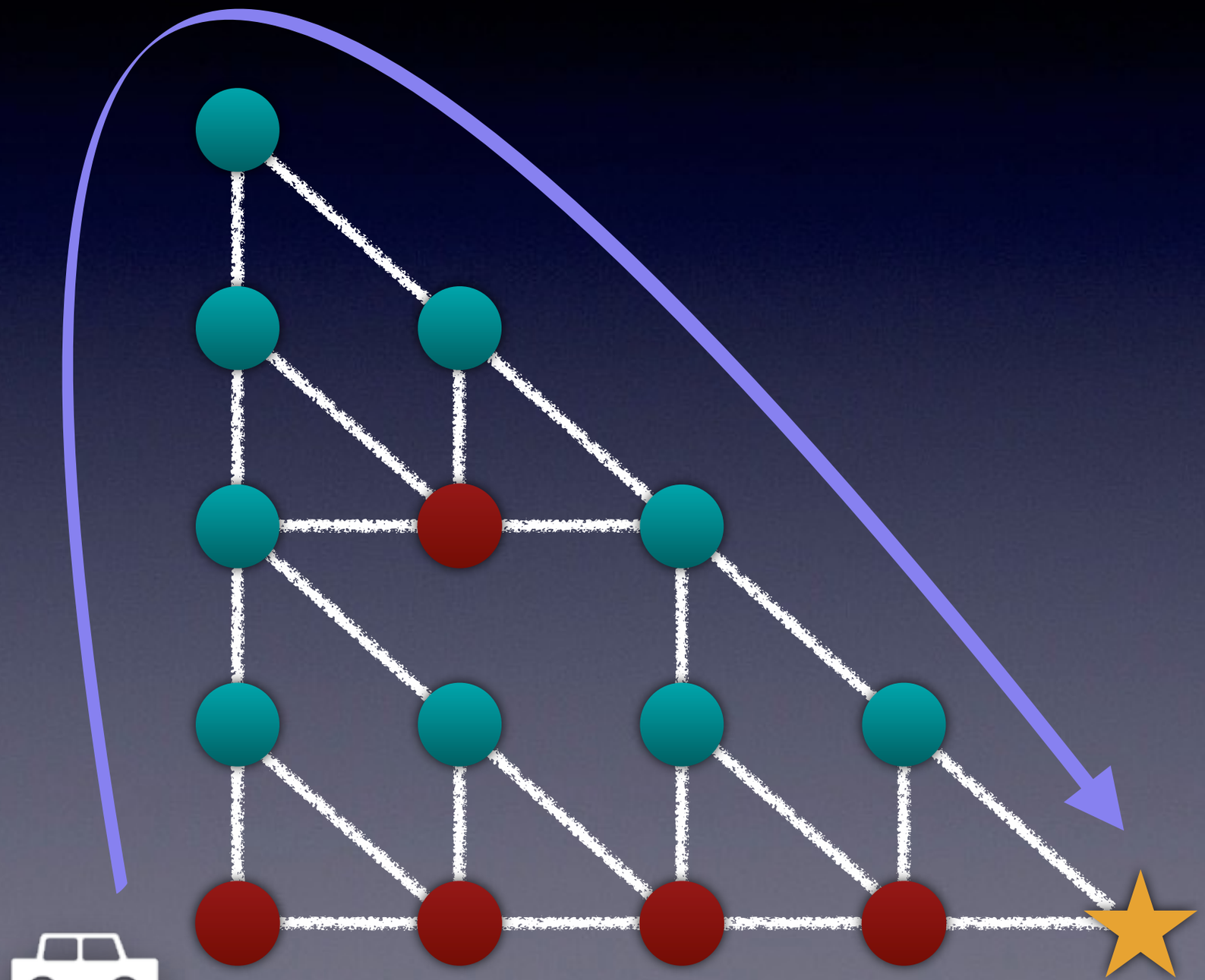
Expected state

[at=B, flat=F, hasTireA=F, hasTireB=T, hasTireC=T, ...]

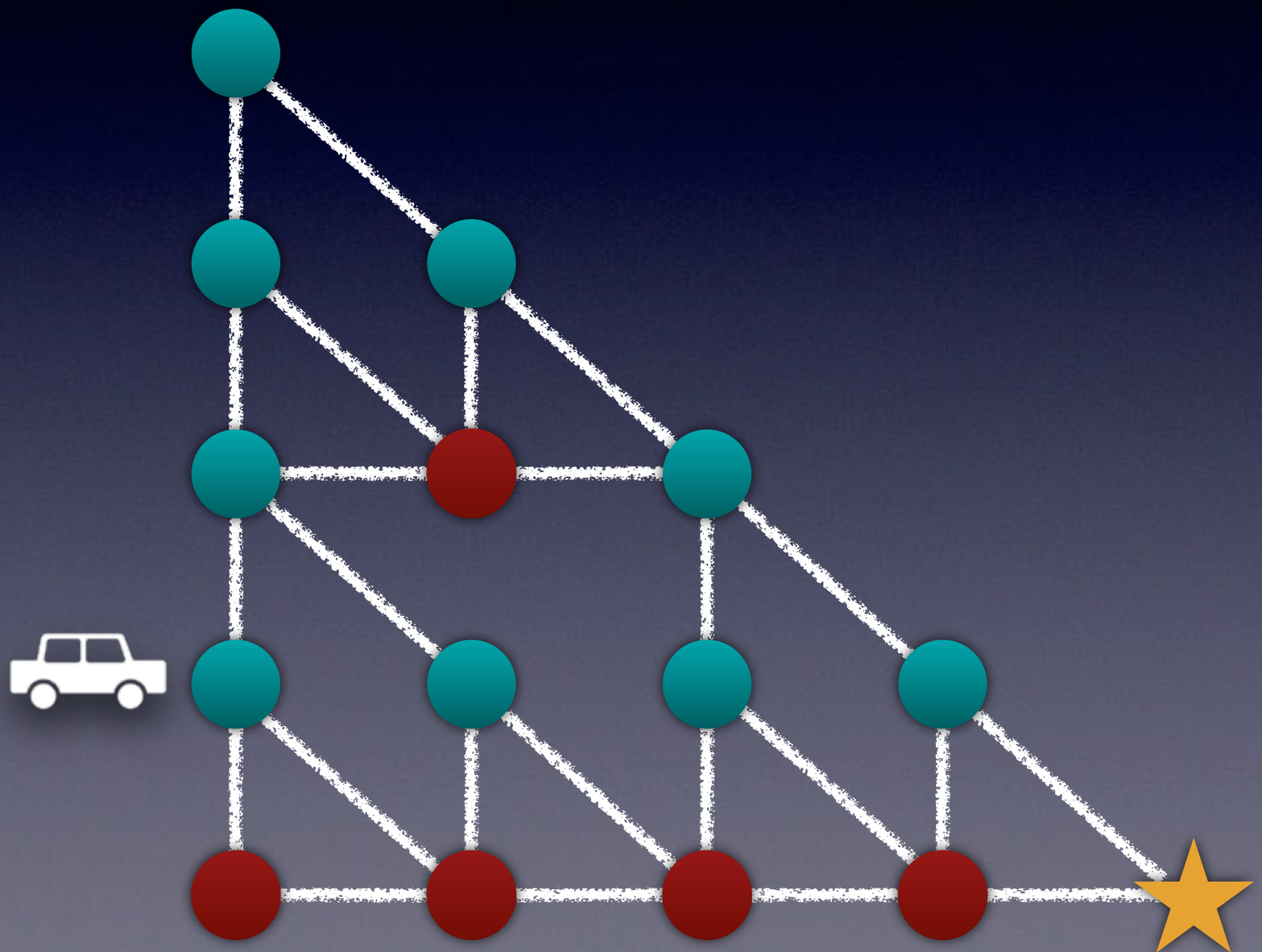
Expected partial state

[at=B, flat=F, ~~hasTireA=F~~, ~~hasTireB=T~~, hasTireC=T, ...]

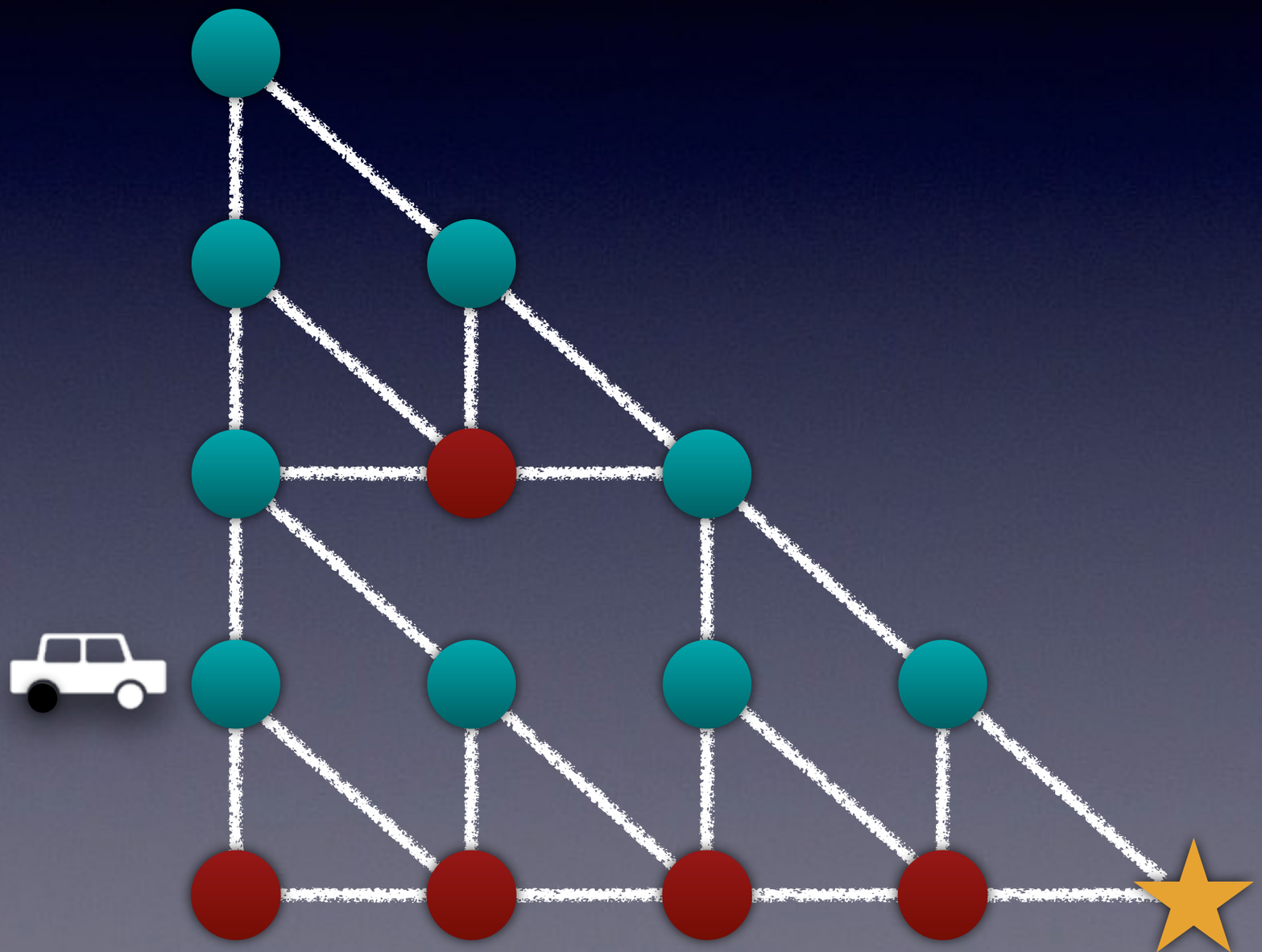
Planning Locally



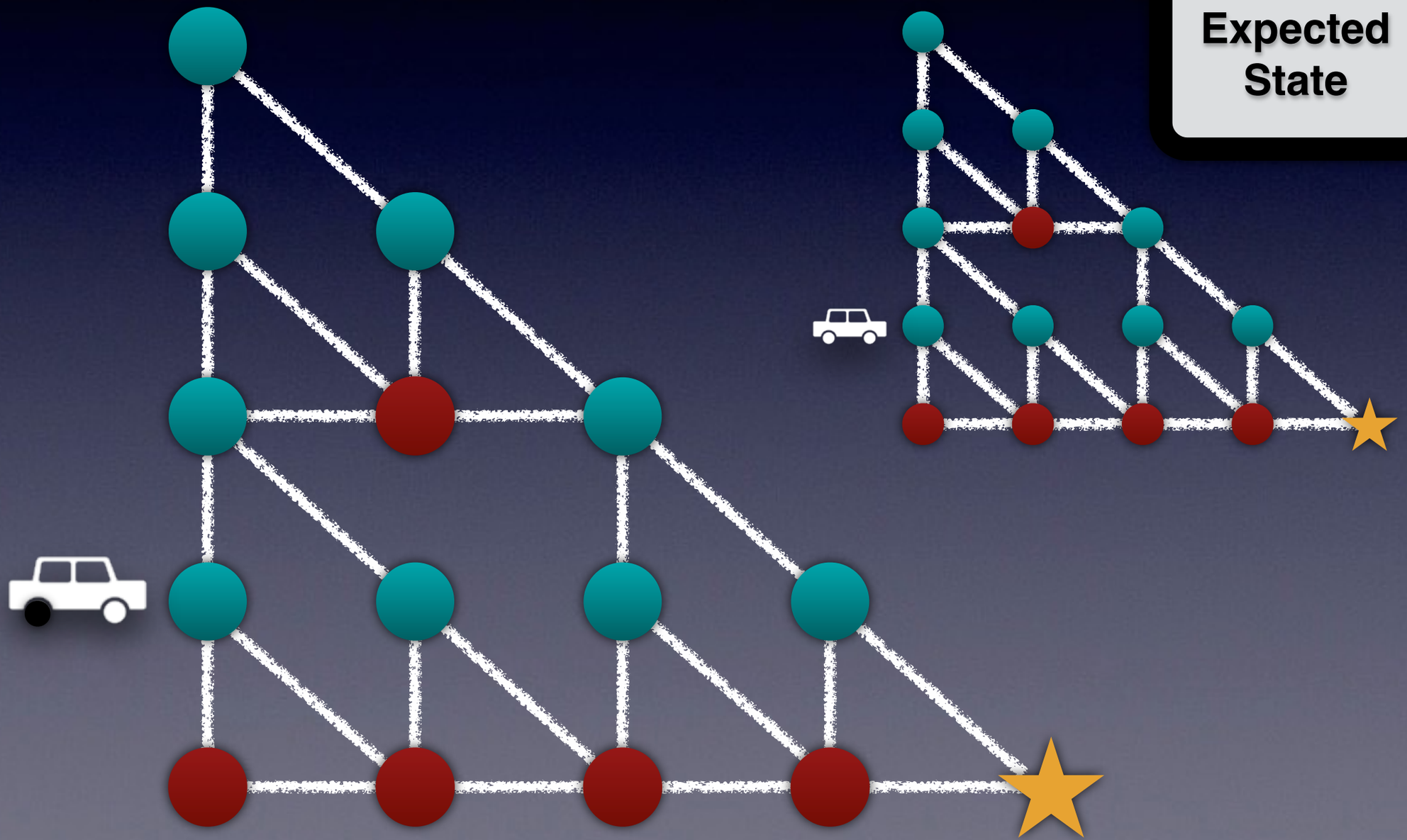
Planning Locally



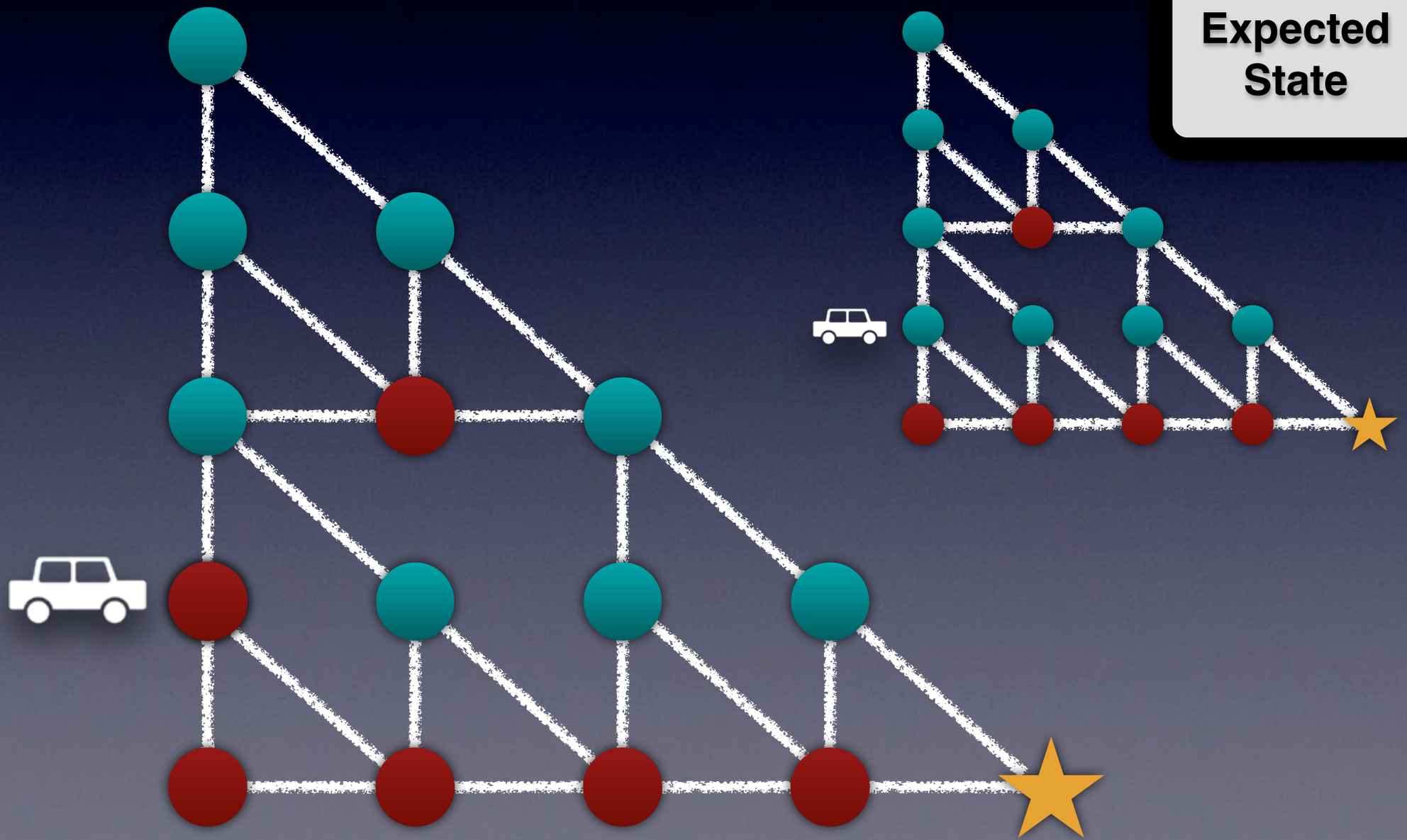
Planning Locally



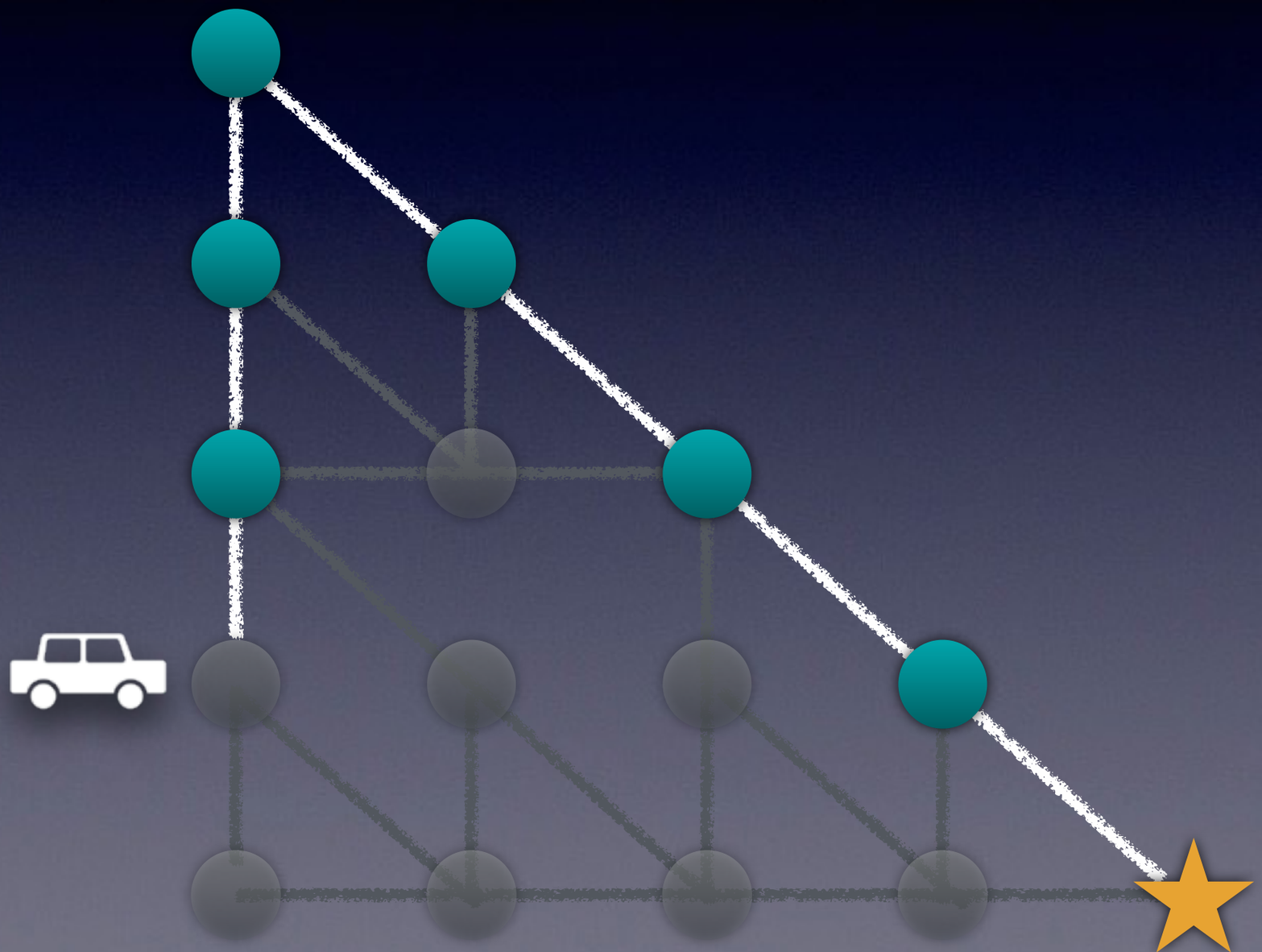
Planning Locally



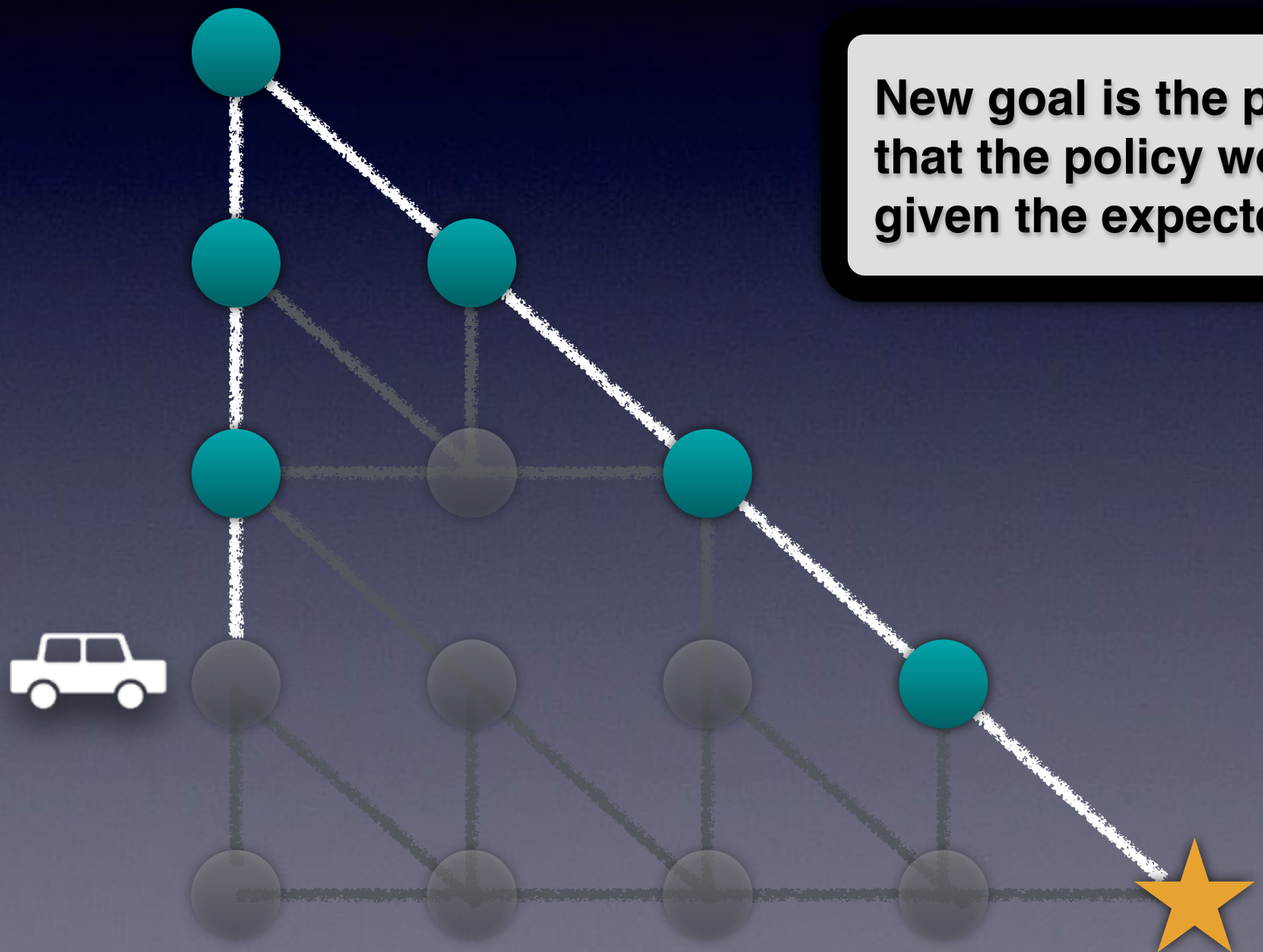
Planning Locally



Planning Locally

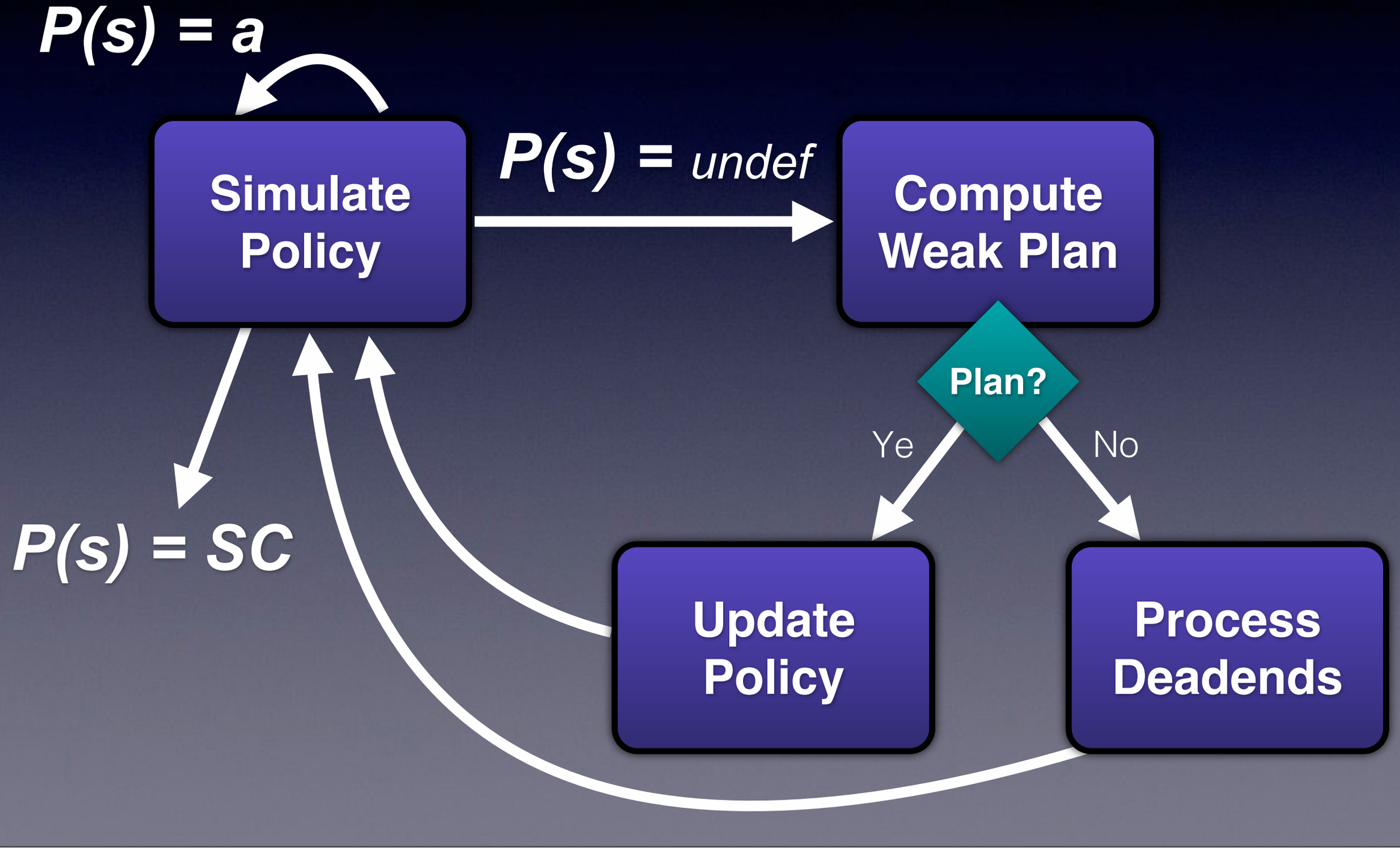


Planning Locally



New goal is the partial state that the policy would match given the expected outcome

General Approach

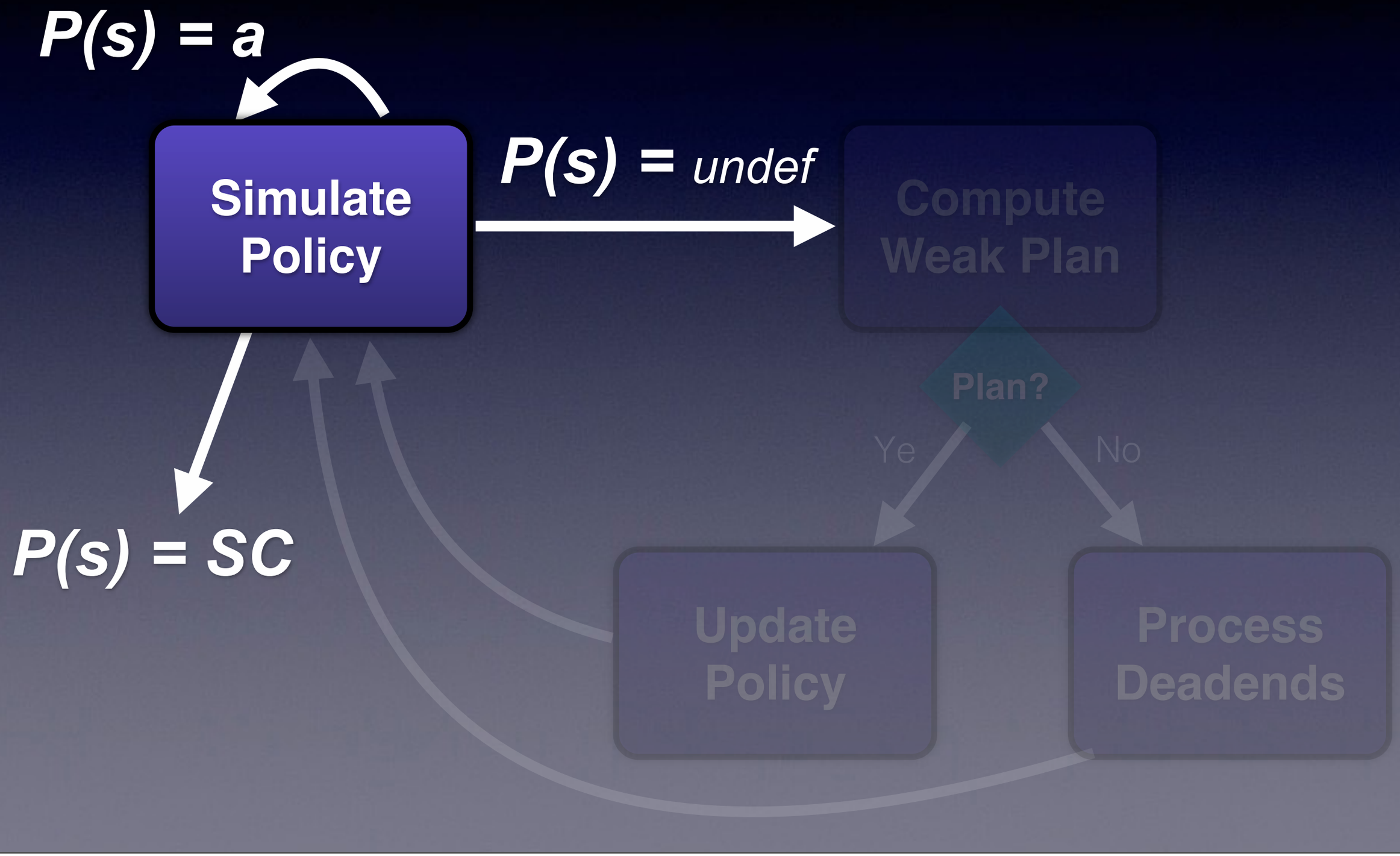


Algorithm 1: Generate Strong Cyclic Plan

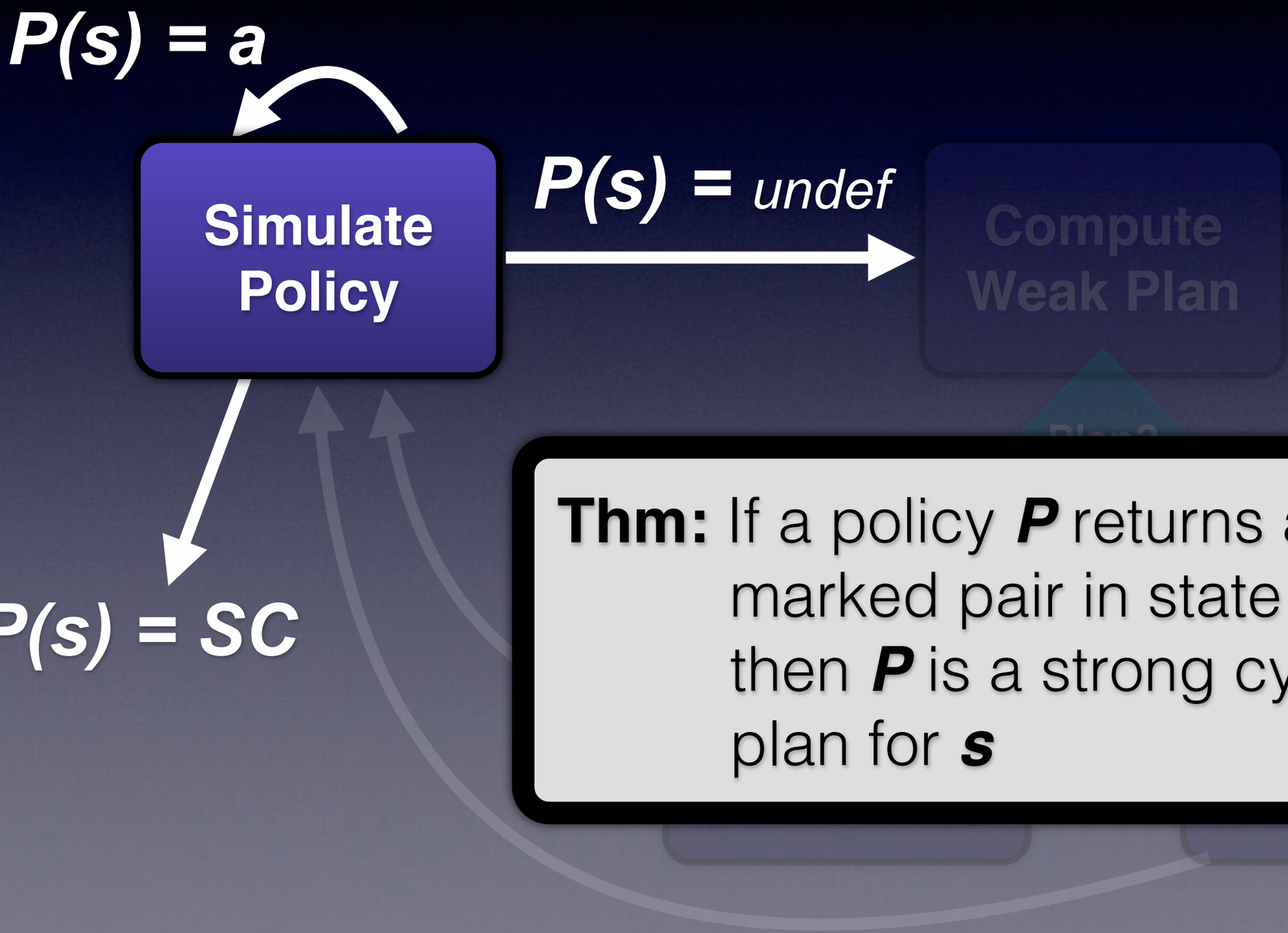
Input: FOND planning task $\Pi = \langle \mathcal{V}, s_0, s_*, \mathcal{A} \rangle$
Output: Partial policy P

Initialize policy P
while P *changes* **do**
  $Open = \{s_0\}$; $Seen = \{\}$;
  while $Open \neq \emptyset$ **do**
    $s = Open$.pop();
    if $s \not\models s_* \wedge s \notin Seen$ **then**
      $Seen$.add(s);
      if $P(s)$ *is undefined* **then**
        GENPLANPAIRS($\langle \mathcal{V}, s, s_*, \mathcal{A} \rangle, P$);
      if $P(s)$ *is defined* **then**
        $\langle p, a \rangle = P(s)$;
        for $e \in Eff_a$ **do**
          $Open$.add($Prog(s, a, e)$);
  PROCESSDEADENDS();
return P;

Strong Cyclic Confirmation



Strong Cyclic Confirmation



Thm: If a policy ***P*** returns a marked pair in state ***s***, then ***P*** is a strong cyclic plan for ***s***

Strong Cyclic Confirmation

- Still must fully simulate the partial policy
- While the (partial) policy may be small, the simulation can be quite expensive
- Want a sufficient condition for the policy to eventually achieve the goal

Algorithm 2: Mark State-Action Pairs

Input: Planning problem Π and rule set $\mathcal{R}$
Output: Annotated rule set $\mathcal{R}$

$\forall \langle p, a \rangle \in \mathcal{R}, \langle p, a \rangle.marked = True$;
while *Some pair becomes unmarked* **do**
  foreach $\langle p, a \rangle \in \mathcal{R}$ *s.t.* $\langle p, a \rangle.marked \wedge p \not\models s_*$ **do**
    foreach $e \in Eff_a$ **do**
      if $\nexists \langle p', a' \rangle \in \mathcal{R}$ *s.t.* $p' \models Prog(p, a, e)$ **then**
        $\langle p, a \rangle.marked = False$;
      else if $\exists \langle p', a' \rangle \in \mathcal{R}$ *s.t.*
          $\langle p', a' \rangle.marked = False \wedge$
          $p' \approx Prog(p, a, e)$ **then**
        $\langle p, a \rangle.marked = False$;
return $\mathcal{R}$;

Strong Cyclic Confirmation

1. **Mark** every state-action pair in the policy

2. While some pair becomes unmarked

   - For every pair that does not match the goal

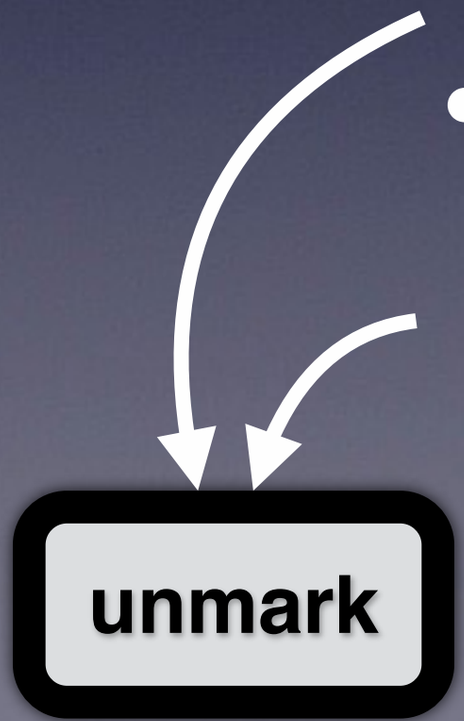
Strong Cyclic Confirmation

1. **Mark** every state-action pair in the policy

2. While some pair becomes unmarked

   - For every pair that does not match the goal

     - If there exists some effect of the action that leads to a state not handled by the policy



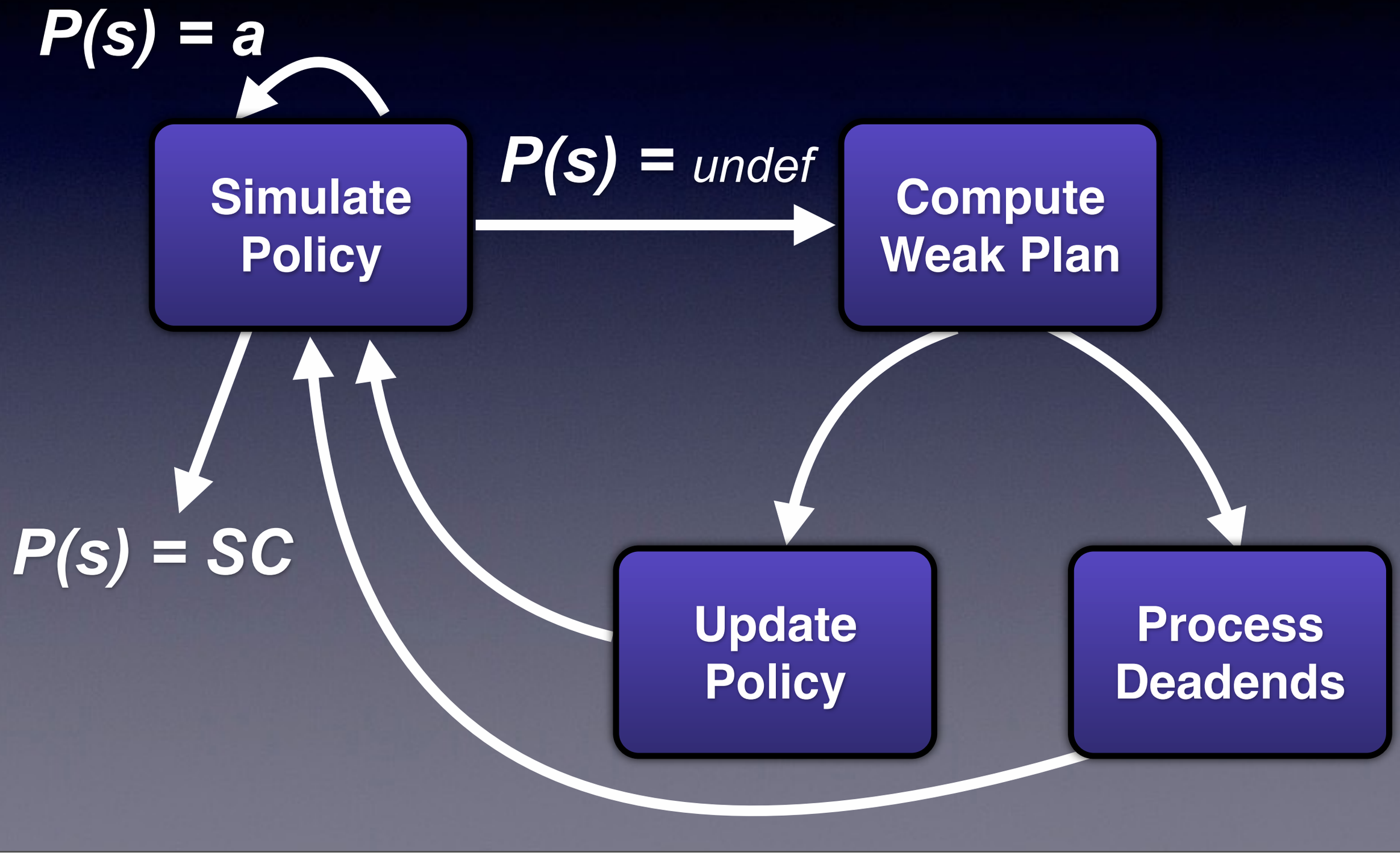
Strong Cyclic Confirmation

1. **Mark** every state-action pair in the policy
2. While
   - For
     - If there exists some effect of the action that leads to a state not handled by the policy

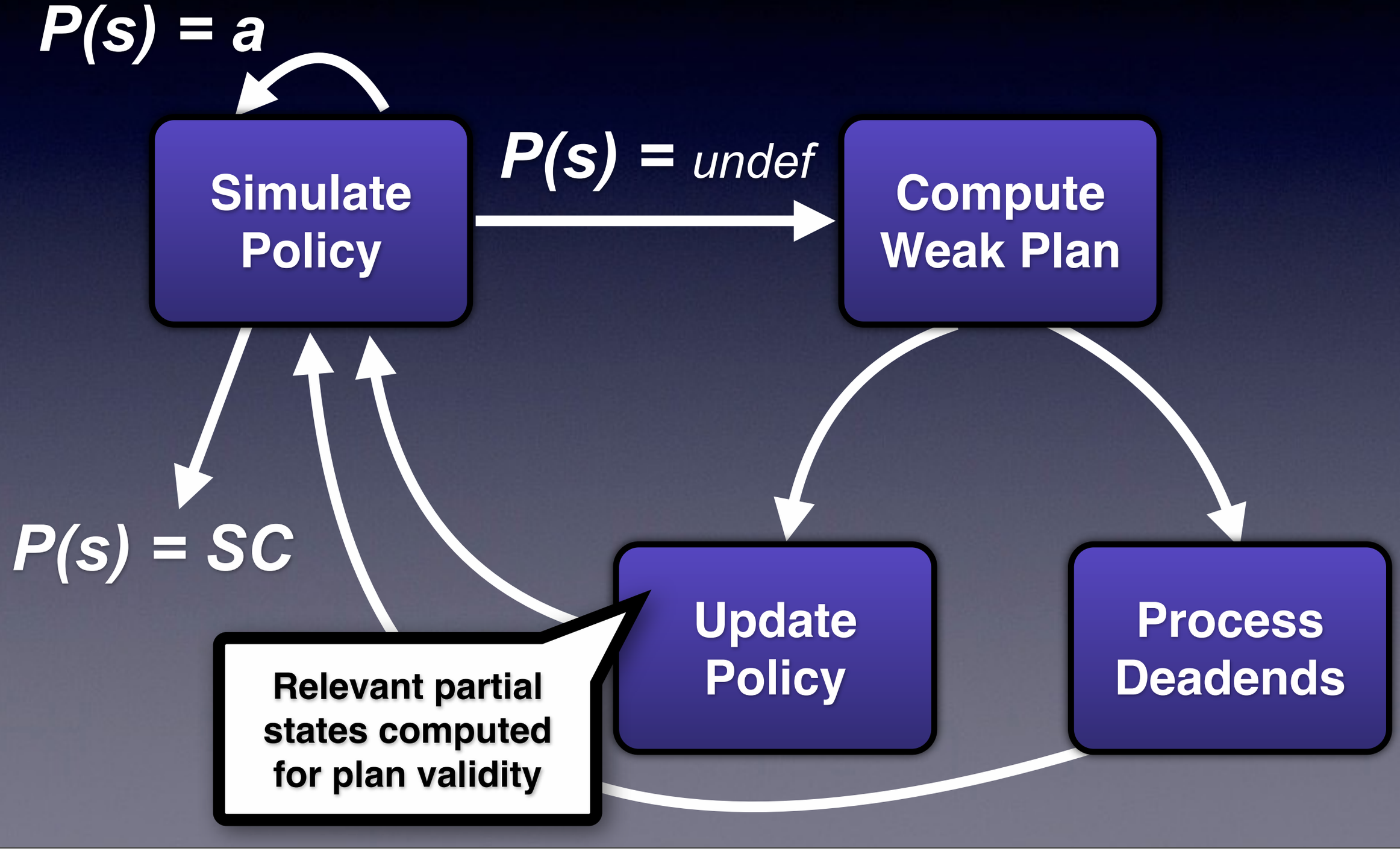
unmark

Thm: If a policy ***P*** returns a marked pair in state ***s***, then ***P*** is a strong cyclic plan for ***s***

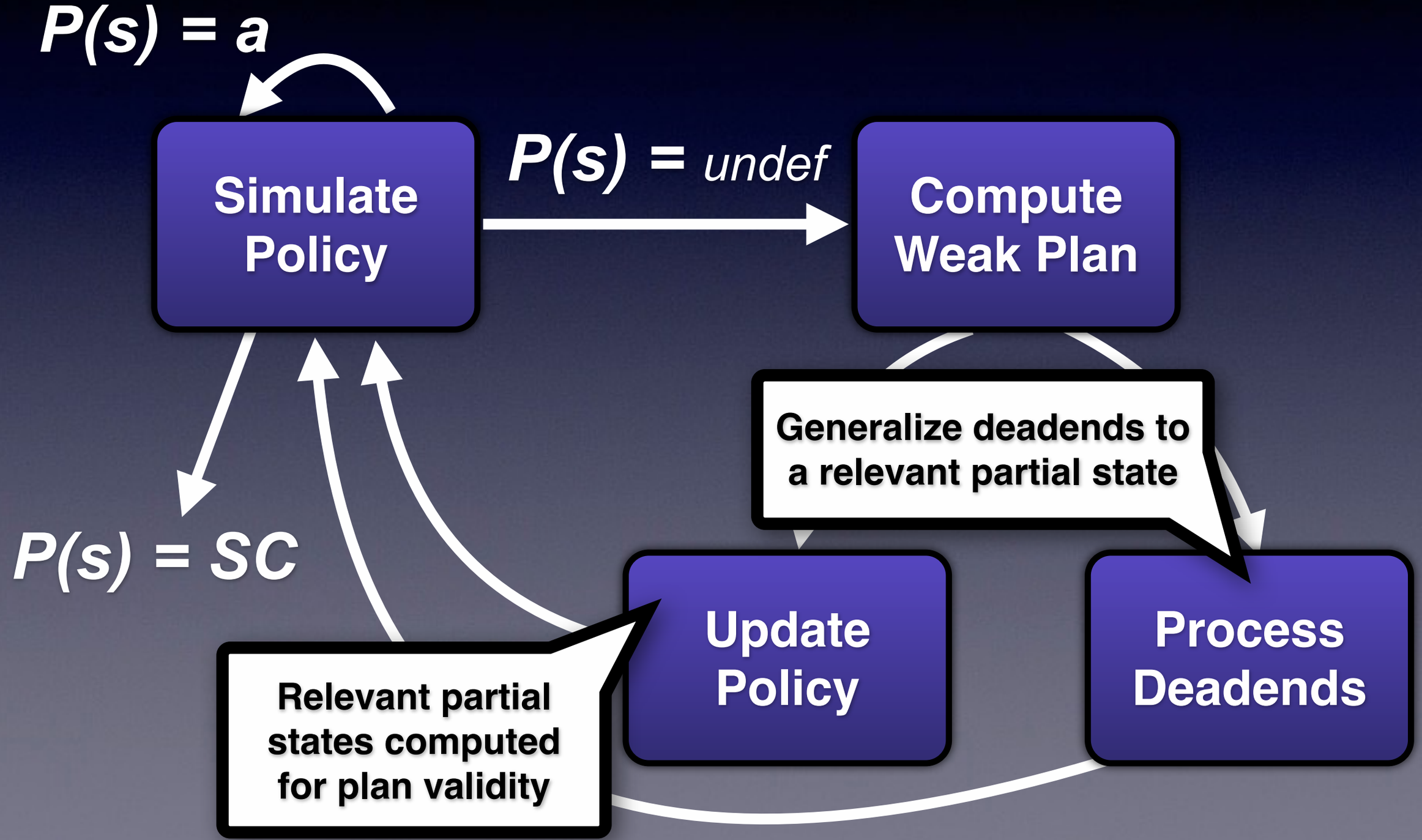
Where's the relevance?



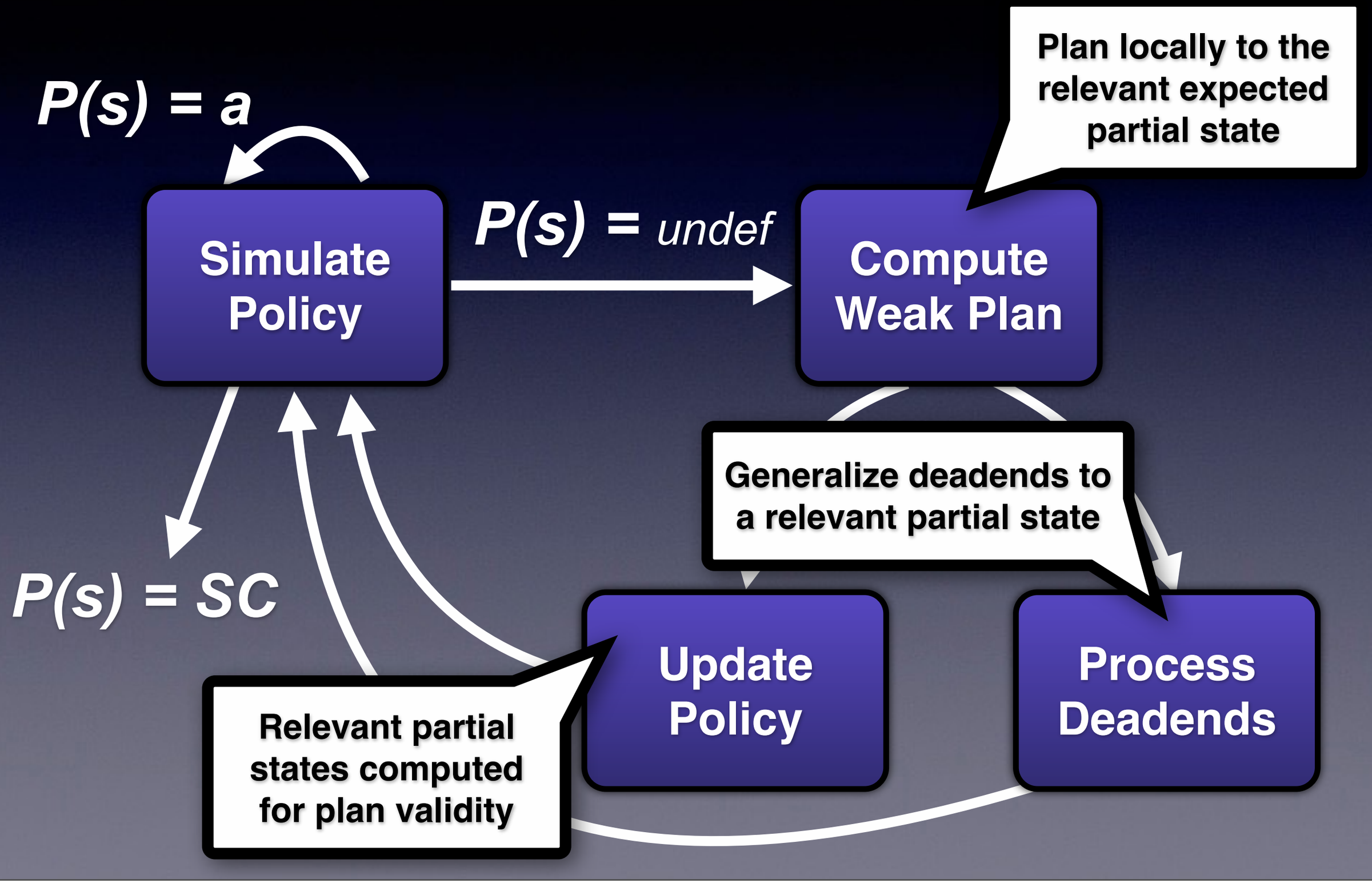
Where's the relevance?



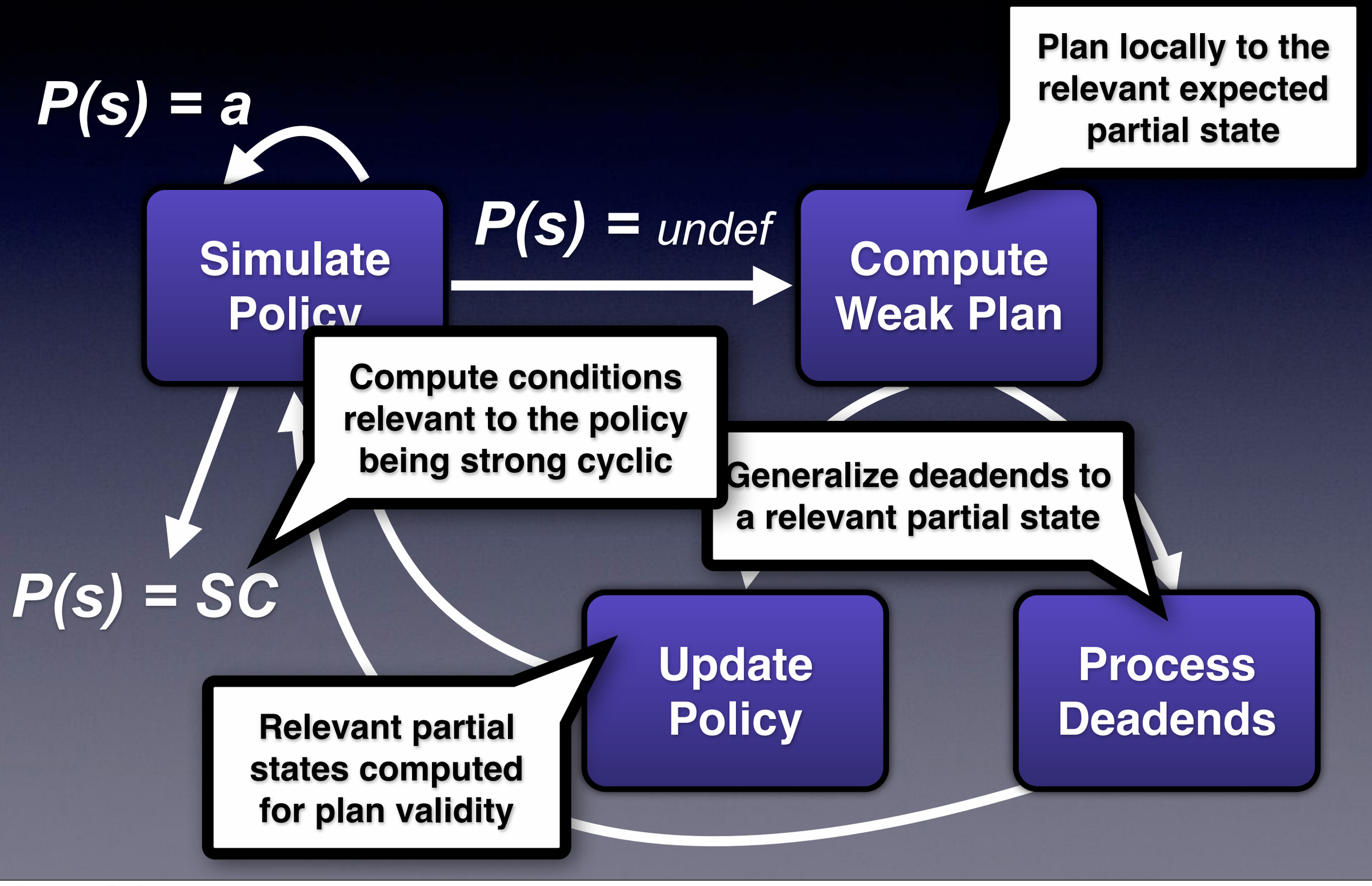
Where's the relevance?



Where's the relevance?



Where's the relevance?



Outline

- Approach
- **Evaluation**
- Conclusion

Evaluation

- Offline planning efficiency
- Impact of relevance
- Application to probabilistic domains
- Online replanning efficiency (see paper)

Experimental Setup

- **PRP**: Based on the Fast Downward planner
- Limited to 30 minutes and 2Gb of memory
- Time reported does not include translation

FOND Domains

Domain	No. of Problems	FIP Solved (unsat)	PRP Solved (unsat)

- **FIP**: Simple and fast strong cyclic planning for fully observable non-deterministic planning problems. (Fu *et al*. 2011)

FOND Domains

Domain	No. of Problems	FIP Solved (unsat)	PRP Solved (unsat)
blocks	30	**30** (0)	**30** (0)
faults	55	**55** (0)	**55** (0)
first	100	**100** (**25**)	**100** (**25**)

FOND Domains

Domain	No. of Problems	FIP Solved (unsat)	PRP Solved (unsat)
blocks	30	**30** (0)	**30** (0)
faults	55	**55** (0)	**55** (0)
first	100	**100** (**25**)	**100** (**25**)
forest	90	20 (11)	**66** (**48**)

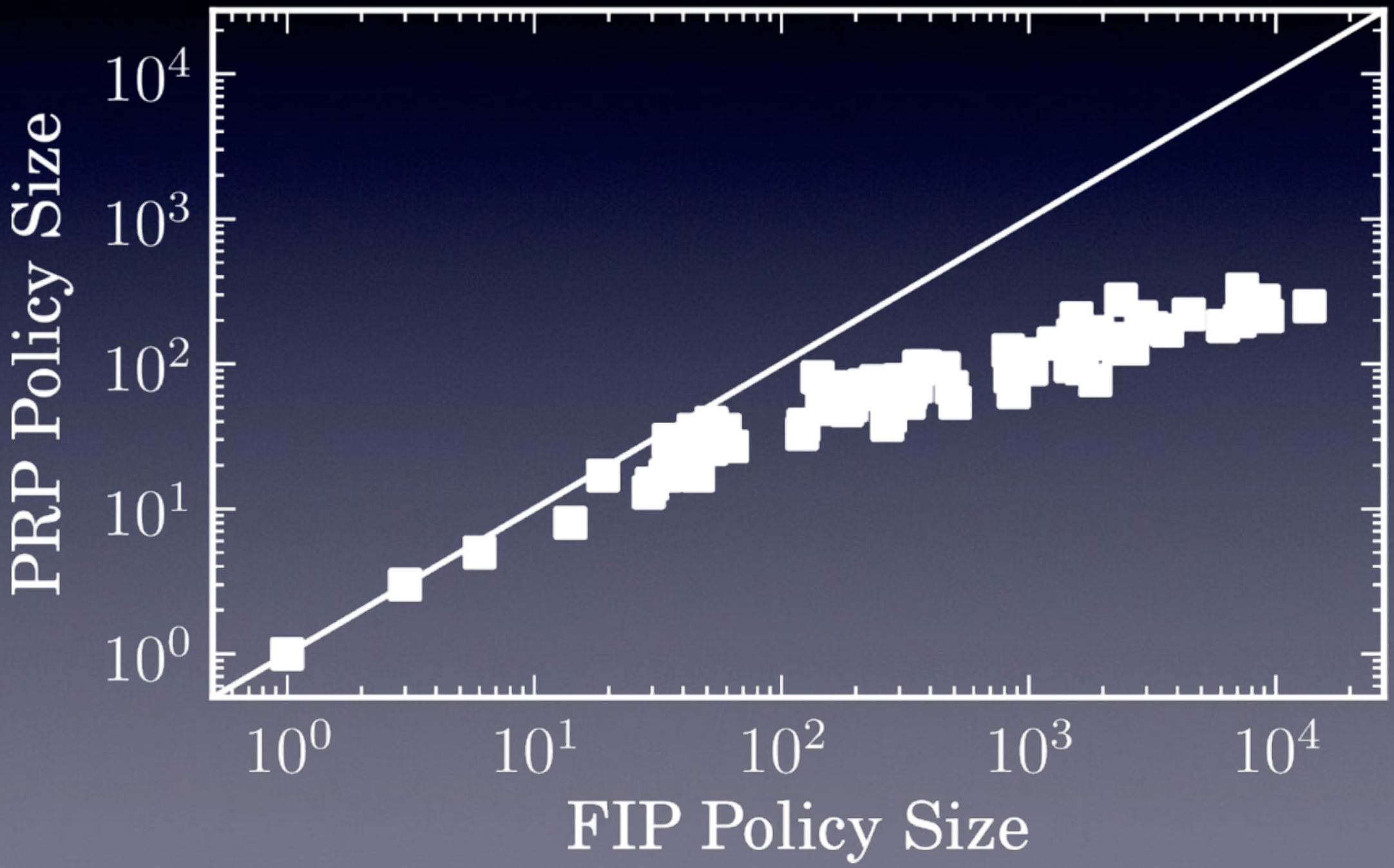
FOND Domains

Domain	No. of Problems	FIP Solved (unsat)	PRP Solved (unsat)
blocks	30	**30** (0)	**30** (0)
faults	55	**55** (0)	**55** (0)
first	100	**100** (**25**)	**100** (**25**)
forest	90	20 (11)	**66** (**48**)
blocks-new	50	33 (0)	**46** (0)
forest-new	90	51 (0)	**81** (0)

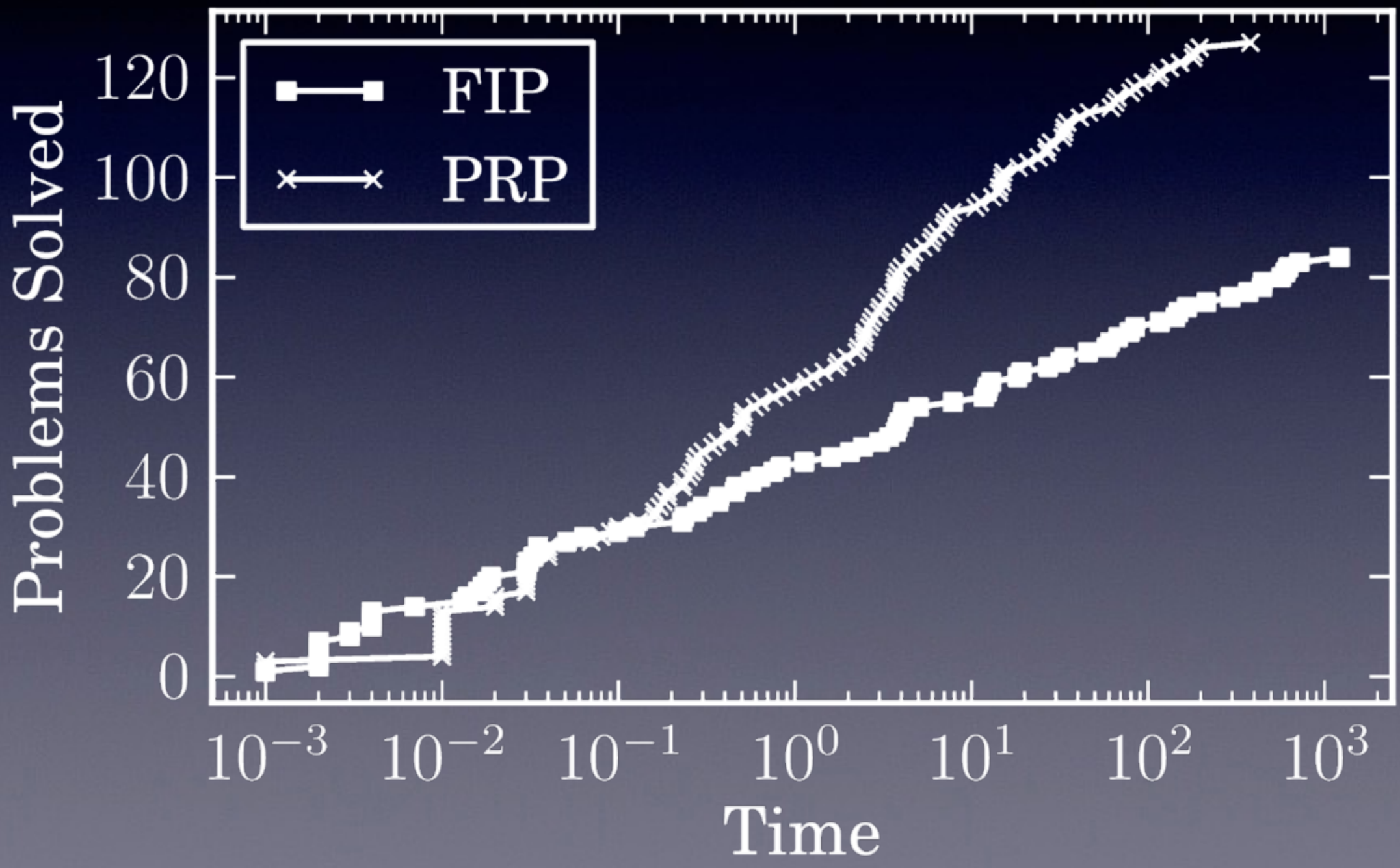
FOND Domains

Domain	No. of Problems	FIP Solved (unsat)	PRP Solved (unsat)
blocks	30	**30** (0)	**30** (0)
faults	55	**55** (0)	**55** (0)
first	100	**100** (**25**)	**100** (**25**)
forest	90	20 (11)	**66** (**48**)
blocks-new	50	33 (0)	**46** (0)
forest-new	90	51 (0)	**81** (0)
Total	415	289 (36)	**378** (**73**)

FOND Domains



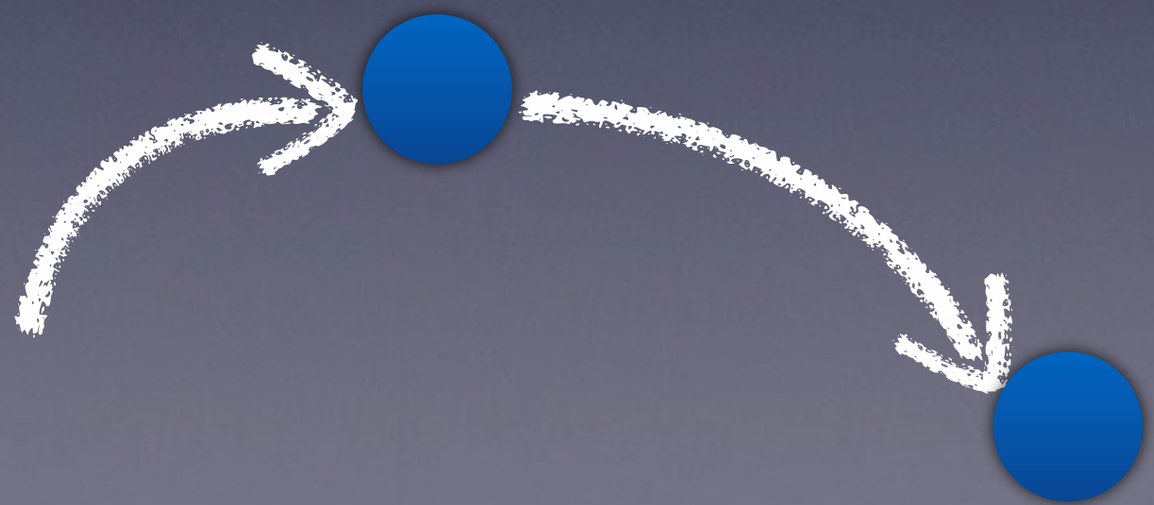
FOND Domains



Impact of Relevance

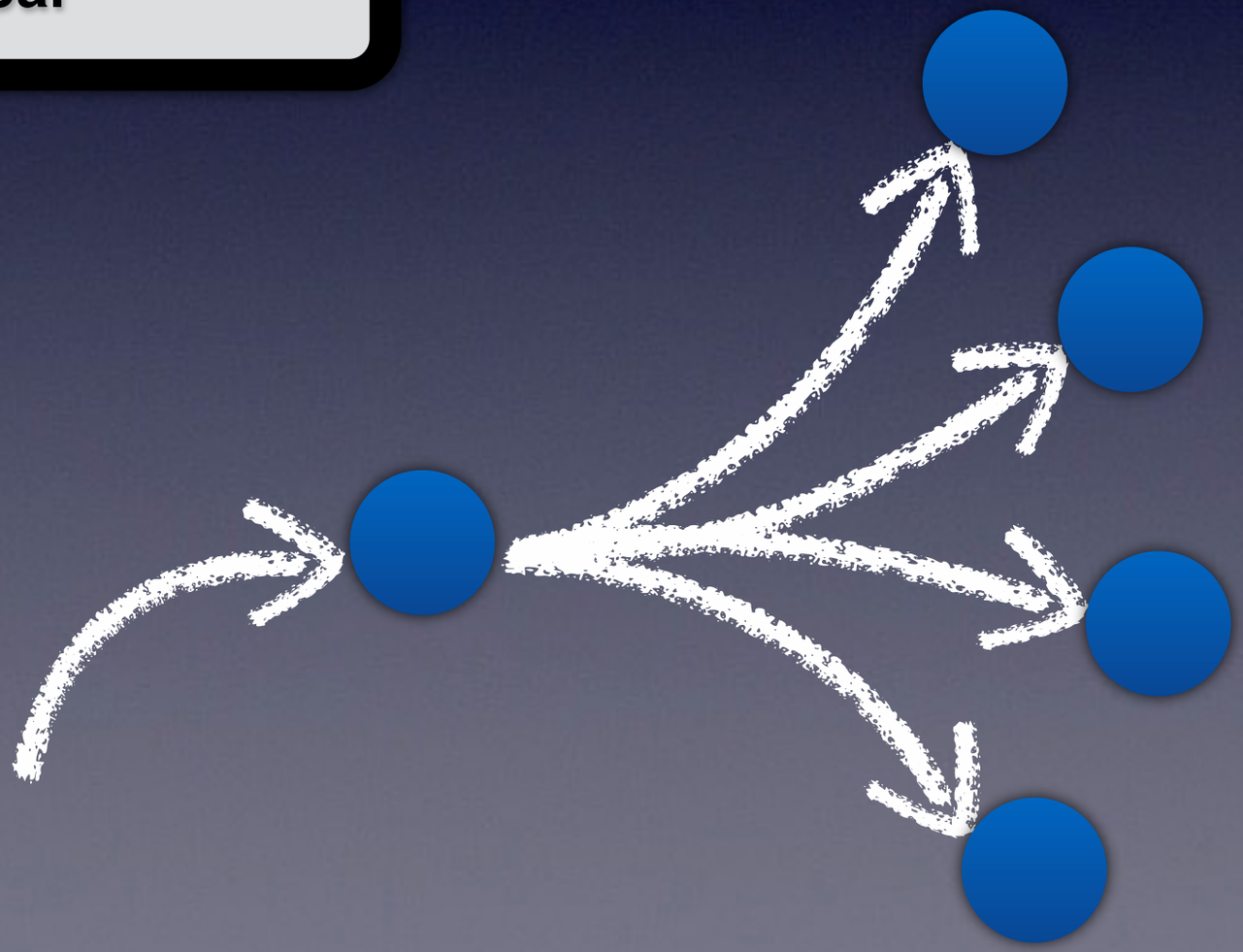
Impact of Relevance

Impact of Relevance

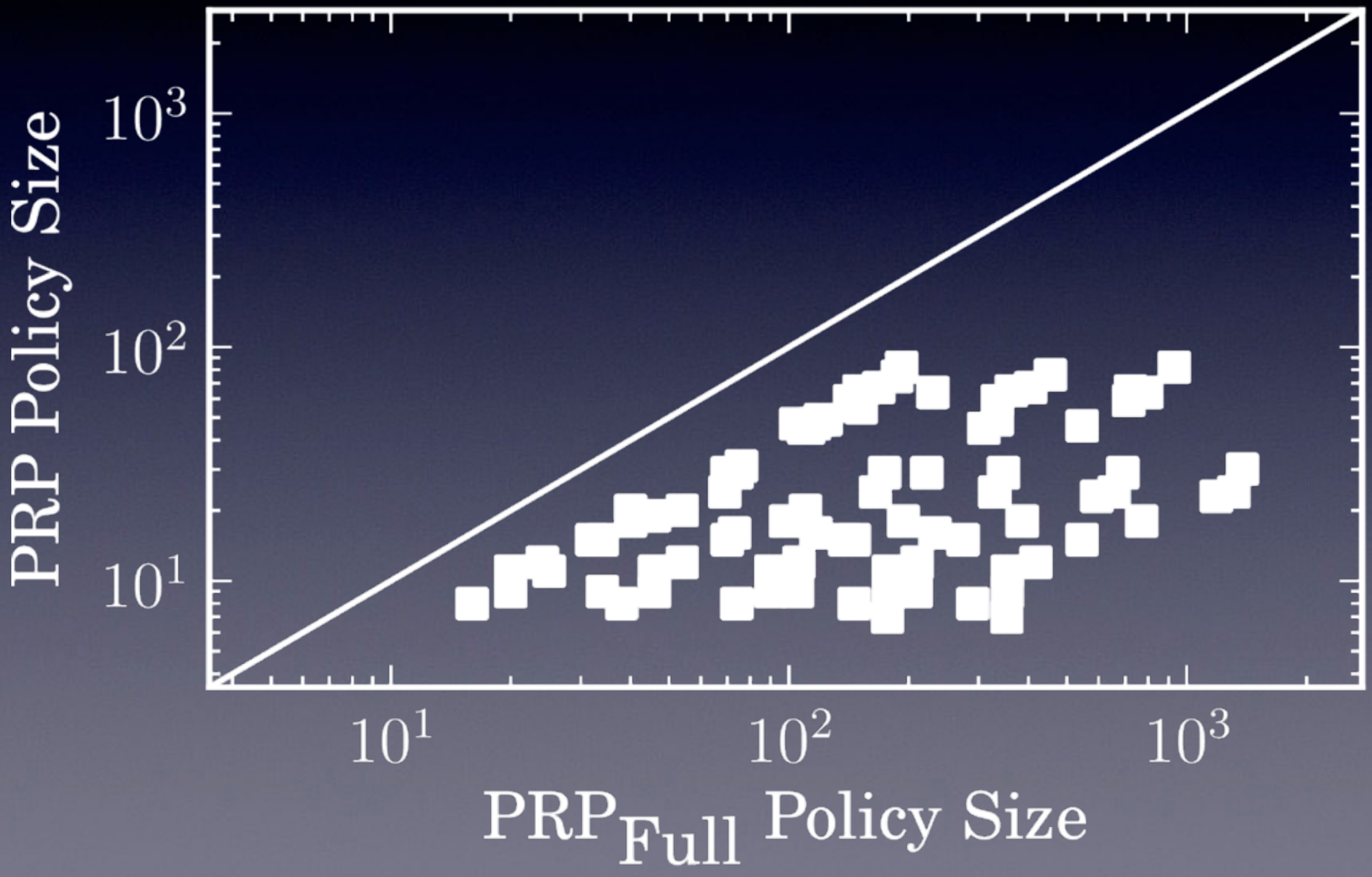


Impact of Relevance

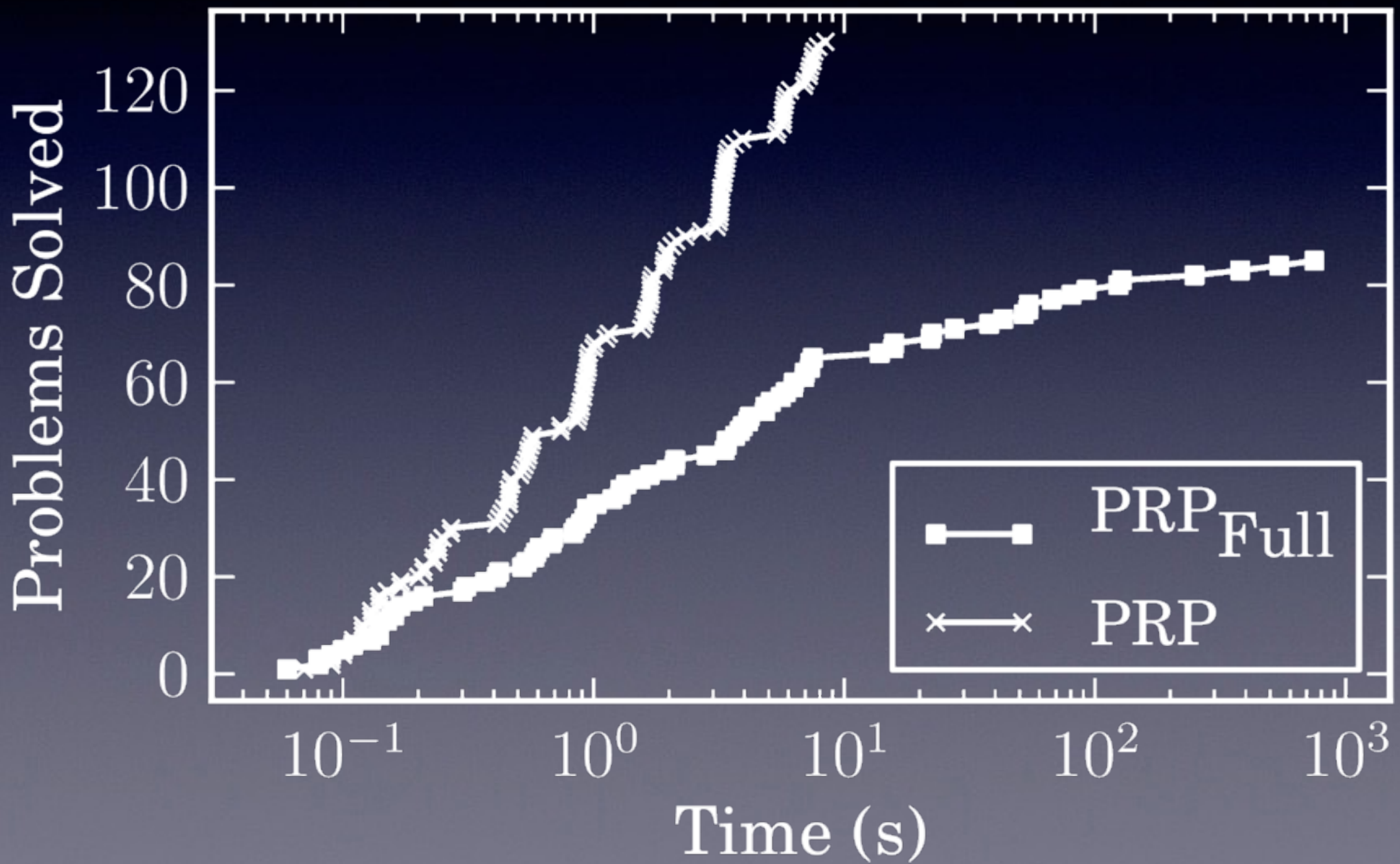
Non-deterministically flip *k* fluents that are irrelevant to achieving the goal



Impact of Relevance



Impact of Relevance



Probabilistic Domains

Domains	Success Rate (%)		Total Time (sec.)	
	FF-H+	PRP	FF-H+	PRP

- **FF-H+**: Improving determinization in hindsight for online probabilistic planning. (Yoon *et al*. 2011)

Probabilistic Domains

Domains (# probs)	Success Rate (%)		Total Time (sec.)	
	FF-H+	PRP	FF-H+	PRP
bw-2 (15)	74.4	**100**	900	**8.4**
elev (15)	64.9	**100**	1620	**1.7**
zeno (15)	68.9	**100**	1620	**98.7**

Probabilistic Domains

Domains (# probs)	Success Rate (%)		Total Time (sec.)	
	FF-H+	PRP	FF-H+	PRP
bw-2 (15)	74.4	**100**	900	**8.4**
elev (15)	64.9	**100**	1620	**1.7**
zeno (15)	68.9	**100**	1620	**98.7**
climber (1)	**100**	**100**	-	**0**
river (1)	**66.7**	**66.7**	-	**0**
bus-fare (1)	**100**	**100**	-	**0**

Probabilistic Domains

Domains (# probs)	Success Rate (%)		Total Time (sec.)	
	FF-H+	PRP	FF-H+	PRP
bw-2 (15)	74.4	**100**	900	**8.4**
elev (15)	64.9	**100**	1620	**1.7**
zeno (15)	68.9	**100**	1620	**98.7**
climber (1)	**100**	**100**	-	**0**
river (1)	**66.7**	**66.7**	-	**0**
bus-fare (1)	**100**	**100**	-	**0**
tire-1 (1)	**100**	**100**	-	**0**
tire-17 (1)	**100**	**100**	-	**18.5**
tire-35 (1)	x	**100**	-	**1519.5**

Outline

- Approach
- Evaluation
- **Conclusion**

Summary

- Introduced **PRP**: A state-of-the-art planner for non-deterministic planning
- Developed novel methods to leverage state relevance in building an efficient policy
- Demonstrated an exponential improvement in both speed and policy size for PRP

Thanks

- Thanks Christian Muise for sharing this wanderful slides.

- Thanks Sheila for connecting me with Christian.

- Thanks everyone in this room for listening.

Question?