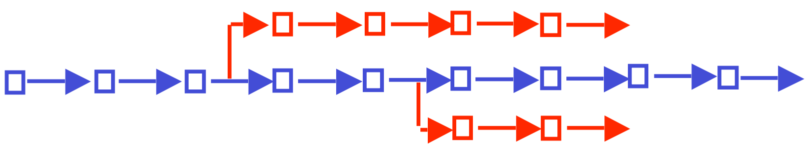
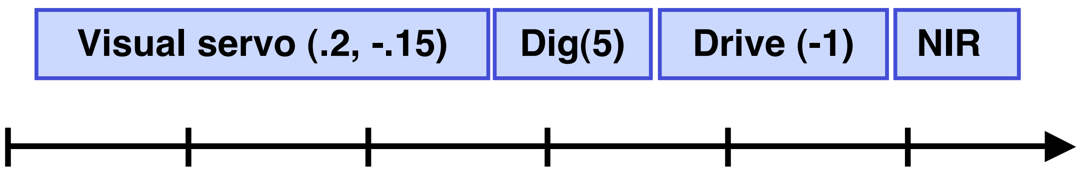
Probabilistic Temporal Planning

PART I: The Problem

Mausam
David E. Smith
Sylvie Thiébaux



Motivation



K9

Reality Bites

?

power

power

[10 ,14:30]

window

Visual servo (.2, -.15)	Dig(5)	Drive (-1)	NIR

Discrete failures

- Tracking failure
- Instrument placement failure
- Hardware faults and failures

Time & Energy

- Wheel slippage
- Obstacle avoidance
- Feature tracking

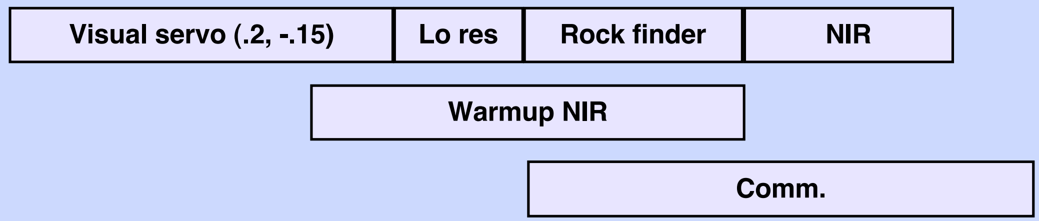
Alternative Approaches

Replanning		processing power
		safety
		lost opportunities
		dead ends
Improving robustness		
	Conservatism	wasteful
	Flexibility	useful but limited
	Conformance	difficult & limited
	Conditionality	very difficult

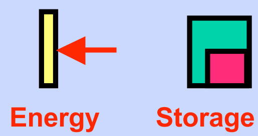
Technical Challenges

Durative actions

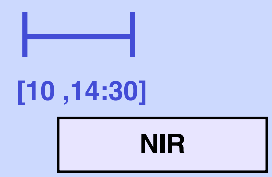
Concurrency



Continuous resources



Time constraints and resource bounds



Oversubscription

G_1, G_2, G_3, G_4, …

V_1, V_2, V_3, V_4, …

Problem Dimensions

Static vs. Dynamic

Predictable vs. Unpredictable

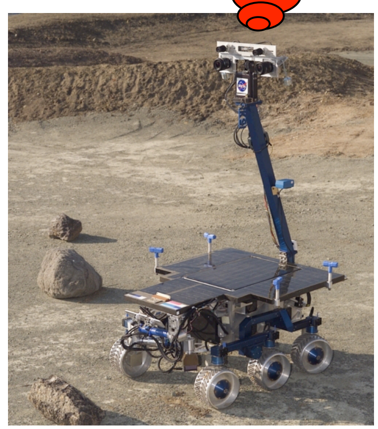
Environment

Fully vs. Partially Observable

Perfect vs. Noisy

Percepts

What action next?



Actions

Discrete vs. Continuous Outcomes

Deterministic vs. Stochastic

Sequential vs. Concurrent

Instantaneous vs. Durative

Full vs. Partial satisfaction

Assumptions

World:

- Static ✔

Actions:

- Durative ✔
- Concurrency ✔
- Stochastic ✔
- Discrete Outcomes ✖
- Complete model ✔

Percepts:

- Fully observable ✔
- Perfect ✔
- Free ✔

Objective:

- Goals ✖

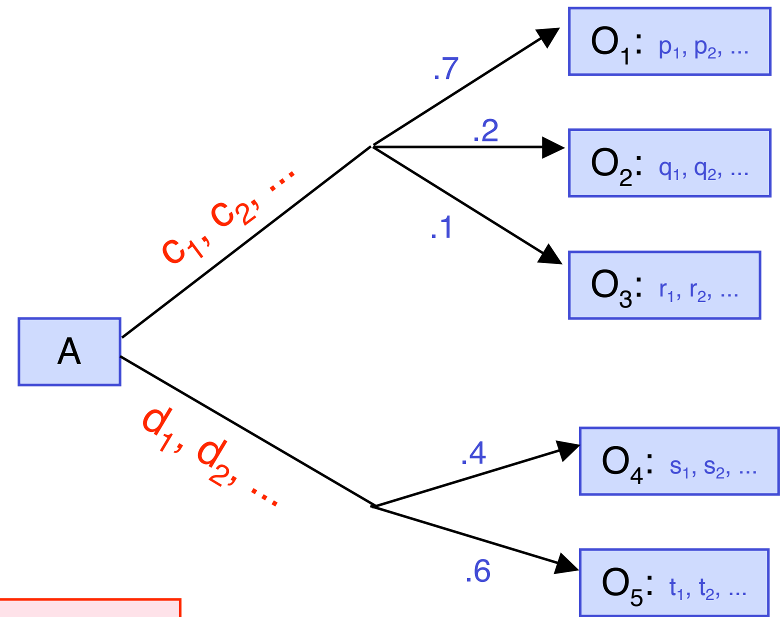
Probabilistic POCL Approaches

C-Buridan

DTPOP

Mahinur

Probapop



PPDDL-like model of action

- no concurrency
- no time
- no resources

Discrete action outcomes

Fixable

but:

- lack good heuristic guidance
- no guarantees of optimality

Outline

1. Introduction
2. Basics of probabilistic planning (Mausam)
3. Durative actions w/o concurrency (Mausam)
4. Concurrency w/o durative actions (Sylvie)
5. Durative actions w/concurrency (Sylvie)
6. Practical considerations

References

Bresina, J.; Dearden, R.; Meuleau, N.; Ramakrishnan, S.; Smith, D.; and Washington, R. Planning under continuous time and resource uncertainty: A challenge for AI. *UAI-02*.

Draper, D.; Hanks, S.; and Weld, D. Probabilistic planning with information gathering and contingent execution. *AIPS-94*.

Onder, N., and Pollack, M. Conditional, probabilistic planning: A unifying algorithm and effective search control mechanisms. *AAAI-99*.

Onder, N.; Whelan, G. C.; and Li, L. Engineering a conformant probabilistic planner. *JAIR 25*.

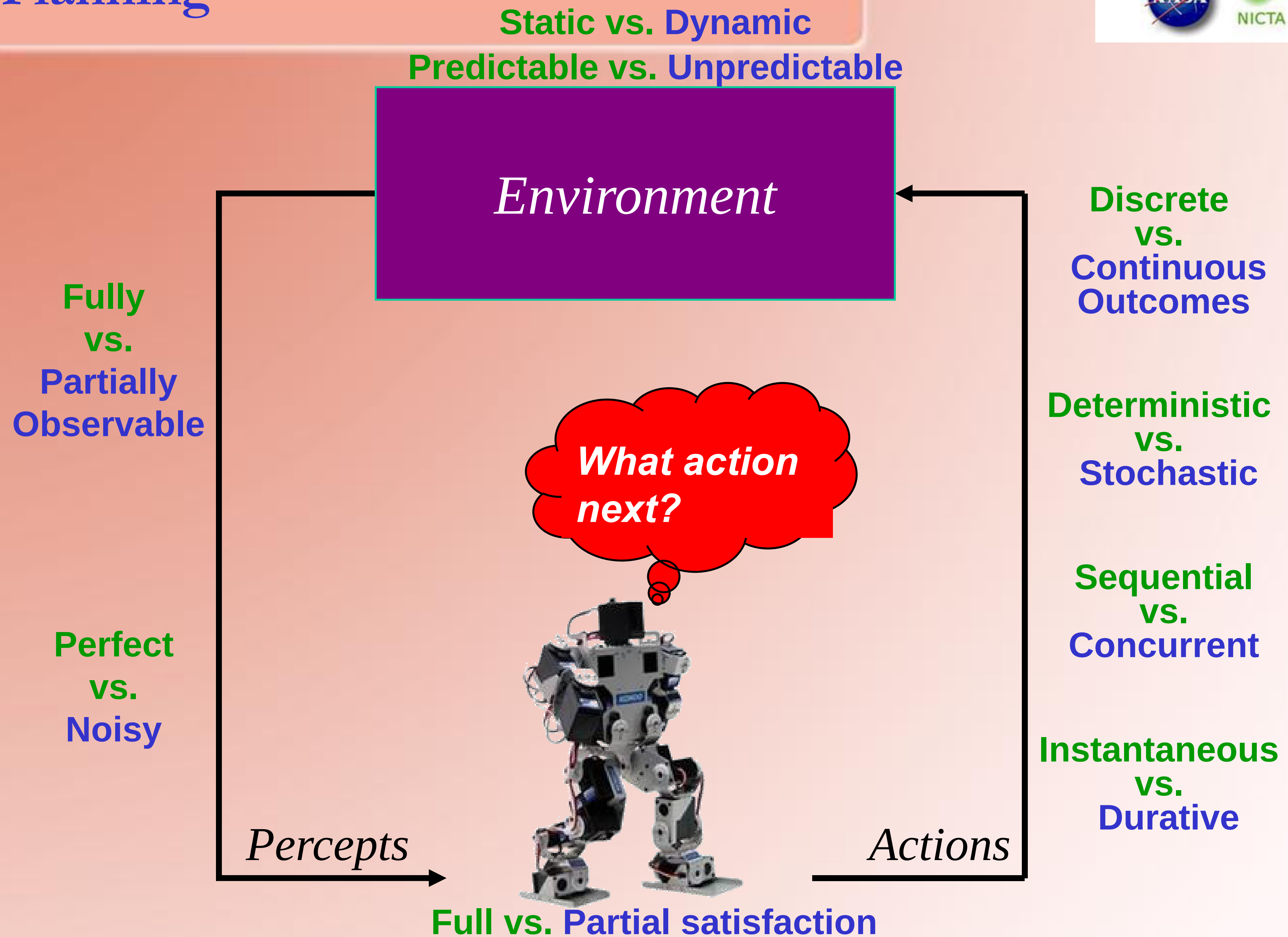
Peot, M. Decision-Theoretic Planning. Ph.D. Dissertation, Dept of Engineering Economic Systems, Stanford University, 1998.

Probabilistic Temporal Planning

PART II: Introduction to Probabilistic Planning Algorithms

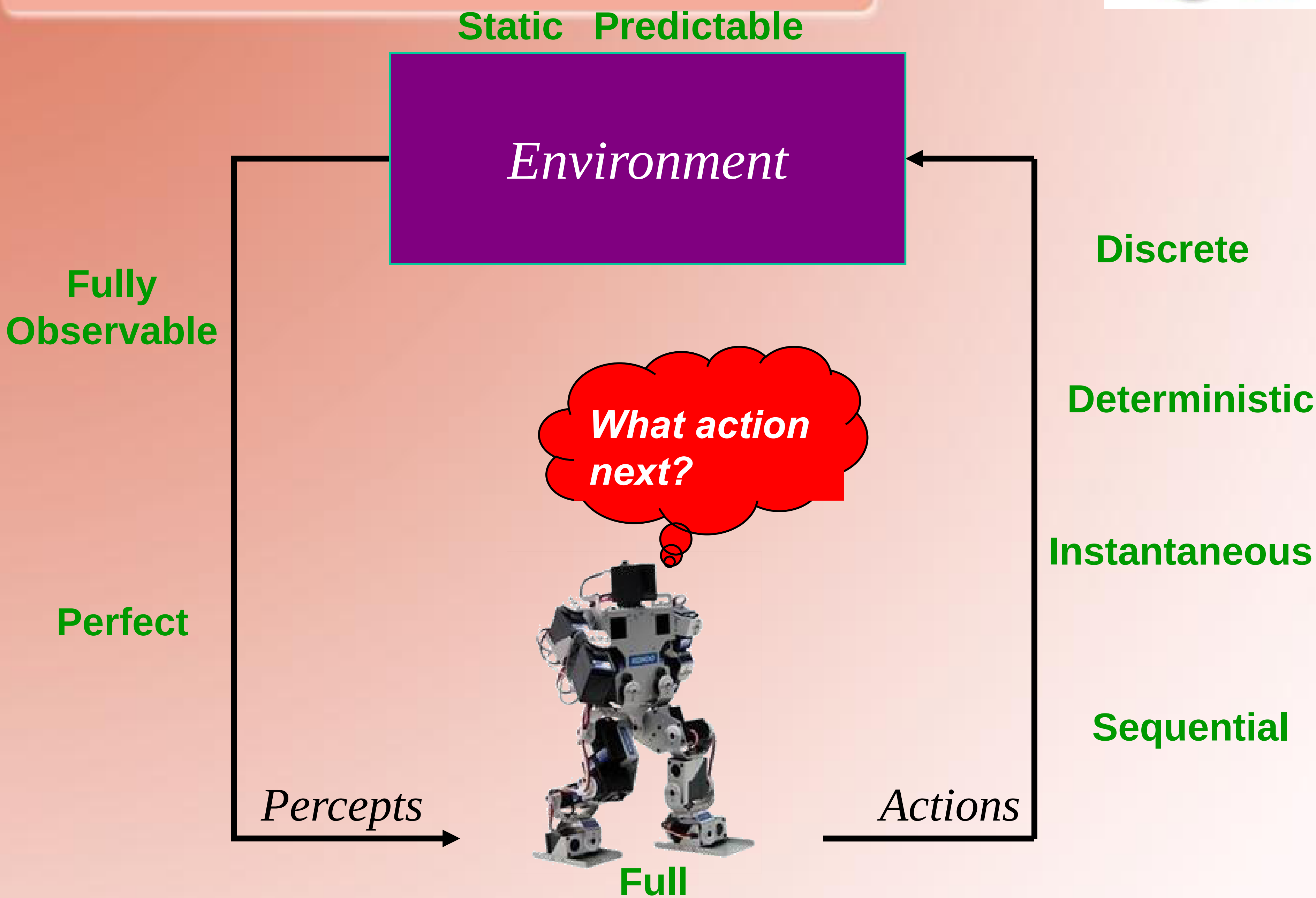
Mausam
David E. Smith
Sylvie Thiébaux

Planning



Static vs. Dynamic
Predictable vs. Unpredictable
Environment
Fully vs. Partially Observable
Perfect vs. Noisy
Discrete vs. Continuous Outcomes
Deterministic vs. Stochastic
Sequential vs. Concurrent
Instantaneous vs. Durative
What action next?
Percepts
Actions
Full vs. Partial satisfaction

Classical Planning



Stochastic Planning

Static Unpredictable

Environment

Fully Observable

Perfect

Discrete

Stochastic

Instantaneous

Sequential

What action next?

Percepts

Actions

Full

Markov Decision Process (MDP)

- $\mathcal{S}$: A set of states
- $\mathcal{A}$: A set of actions
- $\mathcal{P}r(s'|s,a)$: transition model
- $\mathcal{C}(s,a,s')$: cost model
- $\mathcal{G}$: set of goals
- s_0: start state
- γ: discount factor
- $\mathcal{R}(s,a,s')$: reward model

factored

Factored MDP

$\mathcal{C}(a)$ / $\mathcal{C}(s,a)$

absorbing/
non-absorbing

$\mathcal{R}(s)$ / $\mathcal{R}(s,a)$

Objective of a Fully Observable MDP

- Find a policy $\pi: \mathcal{S} \rightarrow \mathcal{A}$

- which optimises
  - minimises (discounted expected cost to reach a goal
  - maximises or expected reward
  - maximises undiscount.) expected (reward-cost)

- given a ____ horizon
  - finite
  - infinite
  - indefinite

- assuming full observability

Role of Discount Factor (γ)

- Keep the total reward/total cost finite
  - useful for infinite horizon problems
  - sometimes indefinite horizon: if there are deadends
- Intuition (economics):
  - Money today is worth more than money tomorrow.

- Total reward: $r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots$
- Total cost: $c_1 + \gamma c_2 + \gamma^2 c_3 + \ldots$

Examples of MDPs

- Goal-directed, Indefinite Horizon, Cost Minimisation MDP
  - $<\mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{C}, \mathcal{G}, s_0>$
  - Most often studied in planning community
- Infinite Horizon, Discounted Reward Maximisation MDP
  - $<\mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{R}, \gamma>$
  - Most often studied in reinforcement learning
- Goal-directed, Finite Horizon, Prob. Maximisation MDP
  - $<\mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{G}, s_0, T>$
  - Also studied in planning community
- Oversubscription Planning: Non absorbing goals, Reward Max. MDP
  - $<\mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{G}, \mathcal{R}, s_0>$
  - Relatively recent model

Bellman Equations for MDP_1

- $<\mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{C}, \mathcal{G}, s_0>$
- Define J*(s) {optimal cost} as the minimum expected cost to reach a goal from this state.
- J* should satisfy the following equation:
-

$$\boxed{\begin{aligned} J^*(s) &= 0 \ \ \mathit{if} \ \ s \in \mathcal{G} \\ J^*(s) &= \min_{a \in Ap(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|s,a)\left[\mathcal{C}(s,a,s') + J^*(s')\right] \end{aligned}}$$

Bellman Equations for MDP_2

- $<\mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{R}, s_0, \gamma>$
- Define **V*(s)** {optimal **value**} as the **maximum** expected **discounted reward** from this state.
- V* should satisfy the following equation:

$$\boxed{V^*(s) = \max_{a \in Ap(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|s,a) \left[\mathcal{R}(s,a,s') + \gamma V^*(s') \right]}$$

Bellman Equations for MDP_3

- <$\mathcal{S}$, $\mathcal{A}$, $\mathcal{P}$r, $\mathcal{G}$, s_0, T>
- Define **J*(s,t)** {optimal cost} as the minimum expected cost to reach a goal from this state **at t^{th} timestep**.
- J* should satisfy the following equation:

$$
\begin{aligned}
P^*(s,t) &= 1 \ \mathit{if} \ \ s \in \mathcal{G} \\
P^*(s,T) &= 0 \ \mathit{if} \ \ s \in \mathcal{G} \\
P^*(s,t) &= \max_{a \in Ap(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|s,a) P^*(s',t+1)
\end{aligned}
$$

Bellman Backup

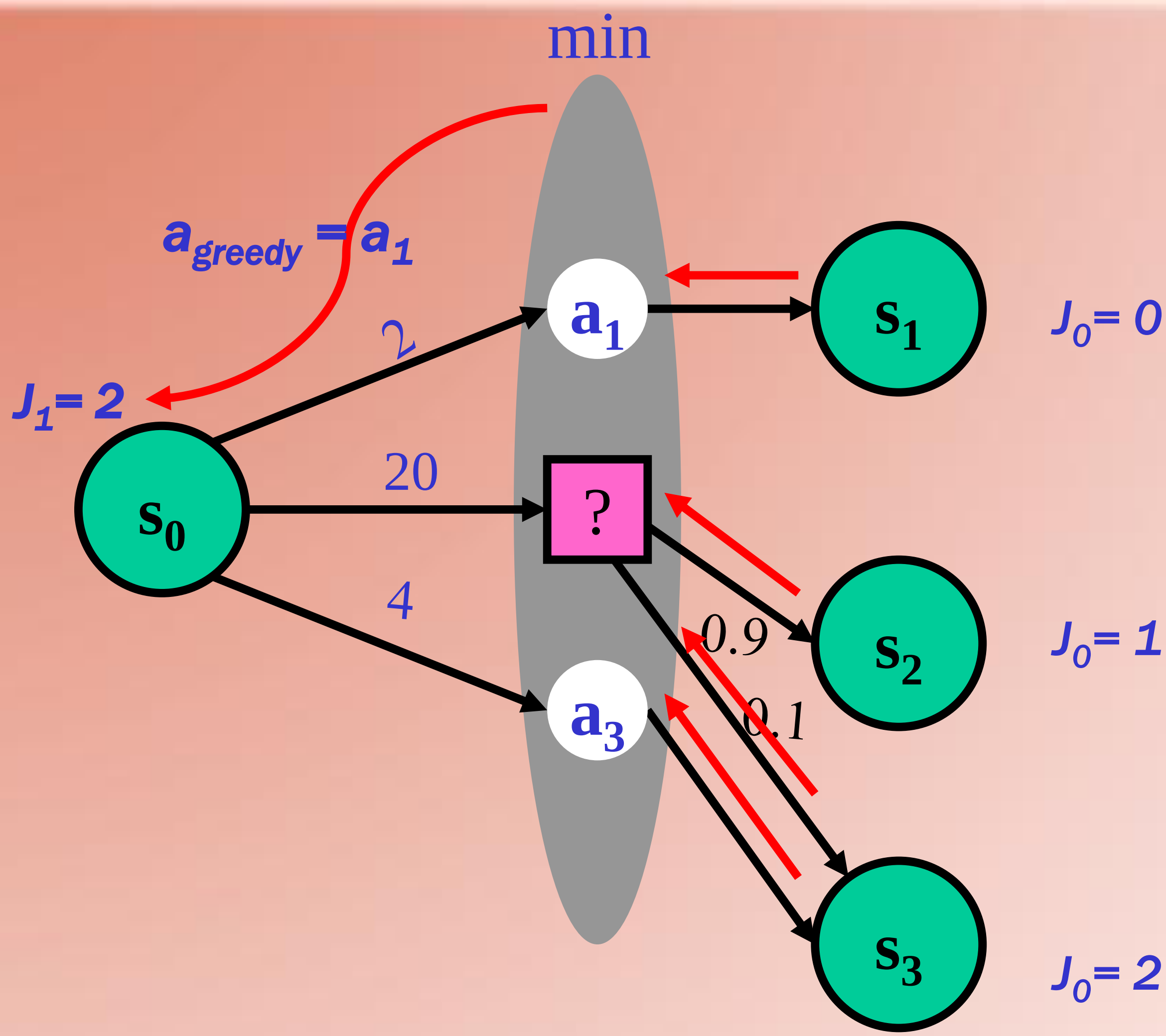
- Given an estimate of J* function (say J_n)
- Backup J_n function at state s
  - calculate a new estimate (J_{n+1}) :

$$Q_{n+1}(s,a) = \sum_{s' \in \mathcal{S}} Pr(s'|s,a)\left[\mathcal{C}(s,a,s') + J_n(s')\right]$$

$$J_{n+1}(s) = \min_{a \in Ap(s)} \left[Q_{n+1}(s,a)\right]$$

- $Q_{n+1}(s,a)$: value/cost of the strategy:
  - execute action a in s, execute π_n subsequently
  - $\pi_n = \text{argmin}_{a \in Ap(s)} Q_n(s,a)$

Bellman Backup



$Q_1(s,a_1) = 2 + 0$

$Q_1(s,a_2) = 20 + 0.9 \times 1$
$\quad + 0.1 \times 2$

$Q_1(s,a_3) = 4 + 2$

Value iteration [Bellman'57]

- assign an arbitrary assignment of J_0 to each state.

- repeat
  - for all states s
    - compute $J_{n+1}(s)$ by Bellman backup at s.
- until $\max_s |J_{n+1}(s) - J_n(s)| < \varepsilon$

Iteration n+1

Residual(s)

ε-convergence

Comments

- Decision-theoretic Algorithm
- Dynamic Programming
- Fixed Point Computation
- Probabilistic version of Bellman-Ford Algorithm
  - for shortest path computation
  - MDP_1 : Stochastic Shortest Path Problem

- $J_n \rightarrow J^*$ in the limit as $n \rightarrow \infty$
- ε-convergence : J_n function is within ε of J^*
  - works only when no state is a dead-end (J^* is finite)
- Monotonicity
  - $J_0 \leq_p J^* \Rightarrow J_n \leq_p J^*$ (J_n monotonic from below)
  - $J_0 \geq_p J^* \Rightarrow J_n \geq_p J^*$ (J_n monotonic from above)
  - otherwise J_n non-monotonic

Policy Computation

$$\begin{aligned}\pi^*(s) &= \operatorname*{argmin}_{a \in Ap(s)} Q^*(s,a) \\ &= \operatorname*{argmin}_{a \in Ap(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|s,a)\left[\mathcal{C}(s,a,s') + J^*(s')\right]\end{aligned}$$

Optimal policy is stationary and time-independent.

- for infinite/indefinite horizon problems

Policy Evaluation

$$J_\pi(s) = \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|s,\pi(s))\left[\mathcal{C}(s,\pi(s),s') + J_\pi(s')\right]$$

A system of linear equations in $|\mathcal{S}|$ variables.

Changing the Search Space

- Value Iteration
  - Search in value space
  - Compute the resulting policy
- Policy Iteration
  - Search in policy space
  - Compute the resulting value

Policy iteration [Howard'60]

- assign an arbitrary assignment of π_0 to each state.
- repeat
  - compute J_{n+1}: the evaluation of π_n
  - for all states s
    - compute $\pi_{n+1}(s)$: $\text{argmin}_{a \in Ap(s)} Q_{n+1}(s,a)$
- until $\pi_{n+1} = \pi_n$

costly: $O(n^3)$

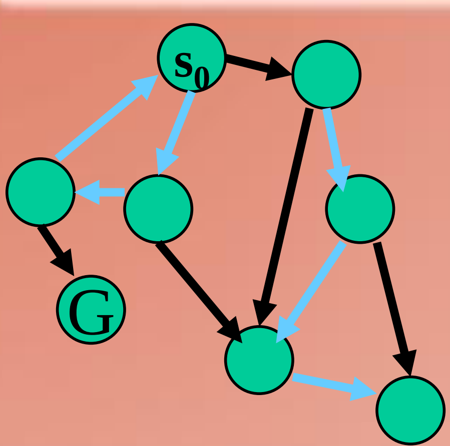
Modified Policy Iteration

approximate by value iteration using fixed policy

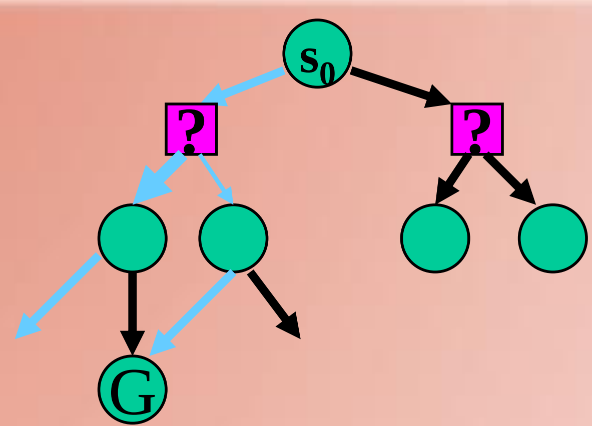
Advantage

- searching in a finite (policy) space as opposed to uncountably infinite (value) space $\Rightarrow$ convergence faster.
- all other properties follow!

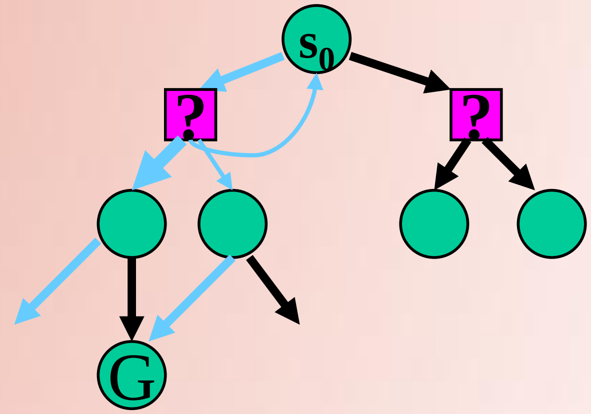
Connection with Heuristic Search



regular graph

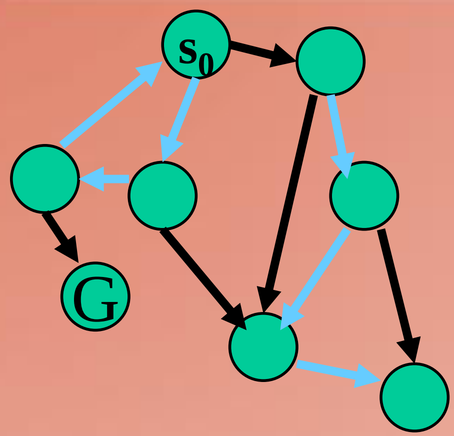


acyclic AND/OR graph



cyclic AND/OR graph

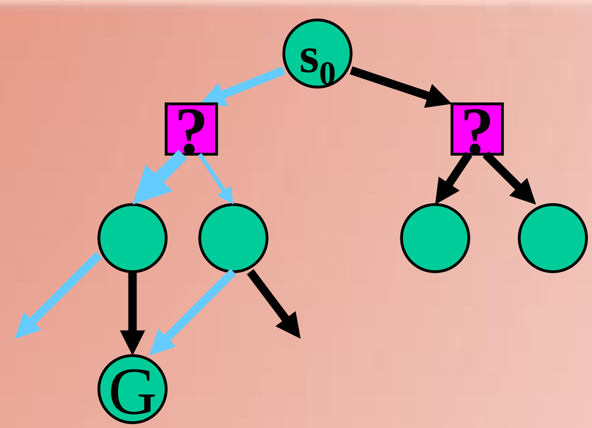
Connection with Heuristic Search



regular graph

soln:(shortest) path

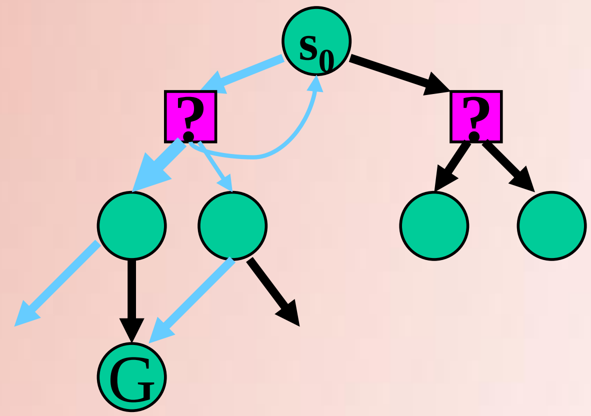
A*



acyclic AND/OR graph

soln:(expected shortest) acyclic graph

AO* [Nilsson'71]



cyclic AND/OR graph

soln:(expected shortest) cyclic graph

LAO* [Hansen&Zil.'98]

All algorithms able to make effective use of reachability information!

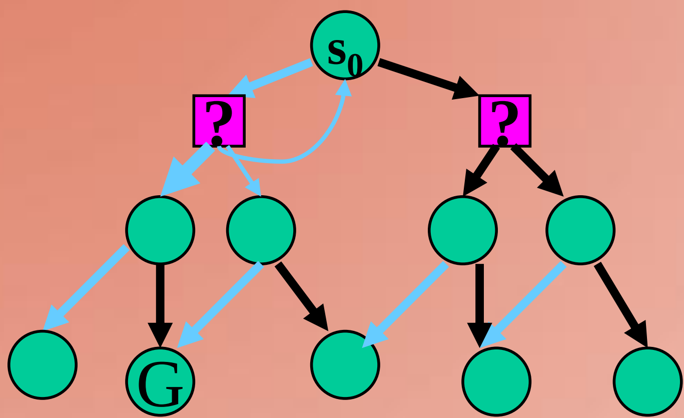
LAO* [Hansen&Zilberstein'98]

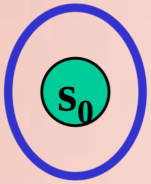
1. add s_0 in the fringe and in greedy graph

2. repeat
   - expand a state on the fringe (in greedy graph)
   - initialize all new states by their heuristic value
   - perform value iteration for all expanded states
   - recompute the greedy graph
3. until greedy graph is free of fringe states

4. output the greedy graph as the final policy

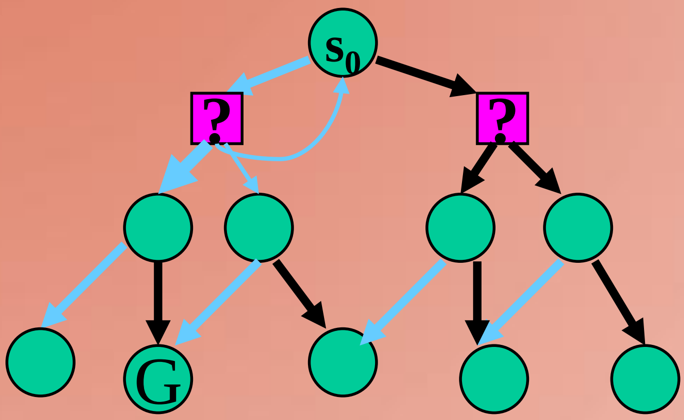
LAO* [Iteration 1]



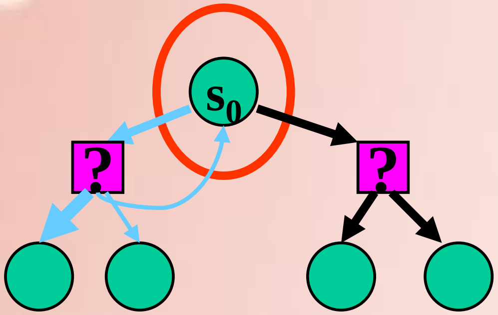


add s_0 in the fringe and in greedy graph

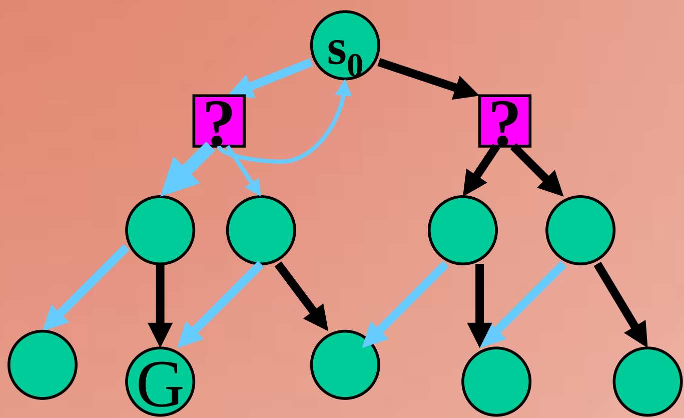
LAO* [Iteration 1]

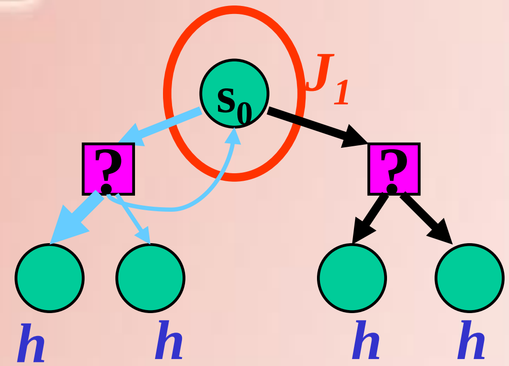


expand a state on fringe in greedy graph



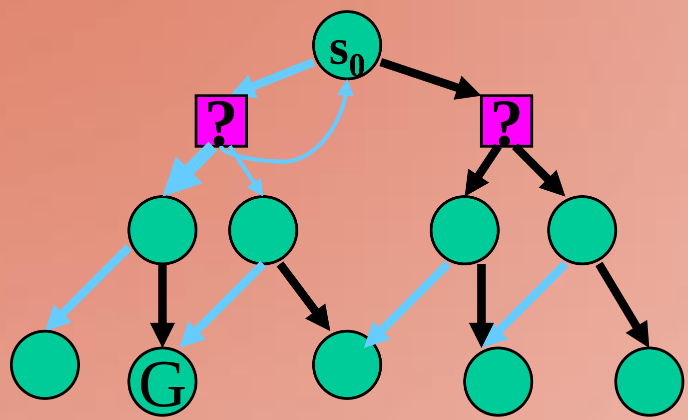
LAO* [Iteration 1]



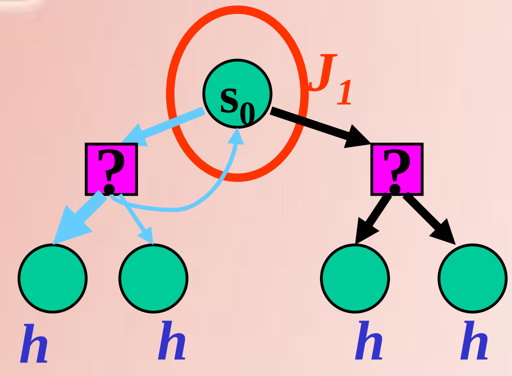


- initialise all new states by their heuristic values
- perform VI on expanded states

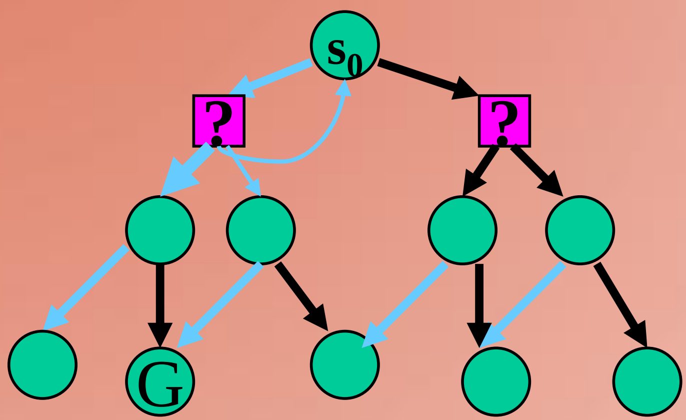
LAO* [Iteration 1]



recompute the greedy graph

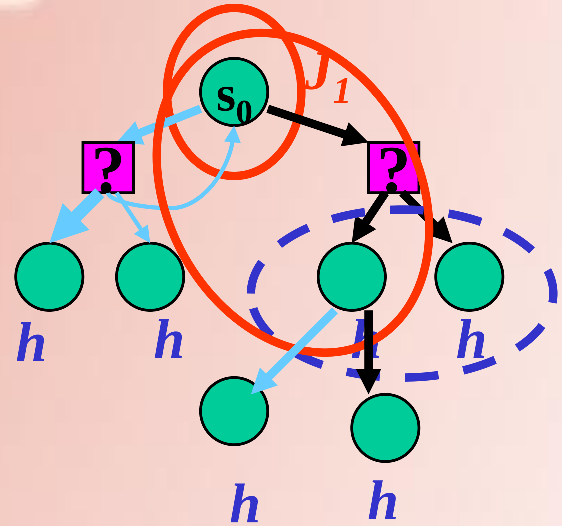


LAO* [Iteration 2]

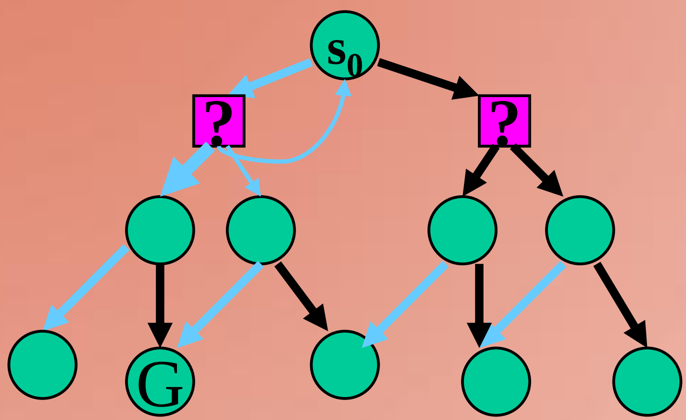


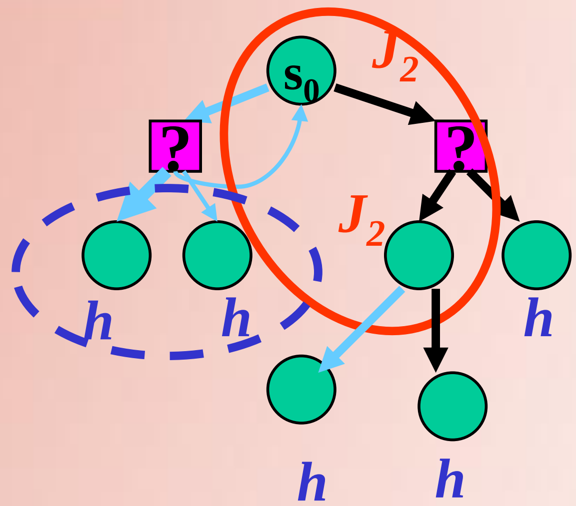
expand a state on the fringe

initialise new states



LAO* [Iteration 2]

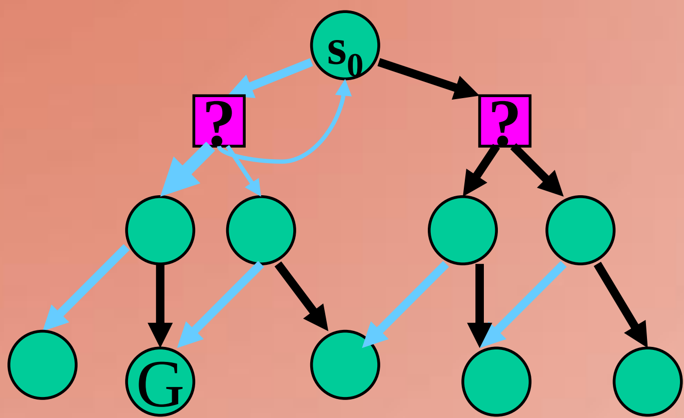




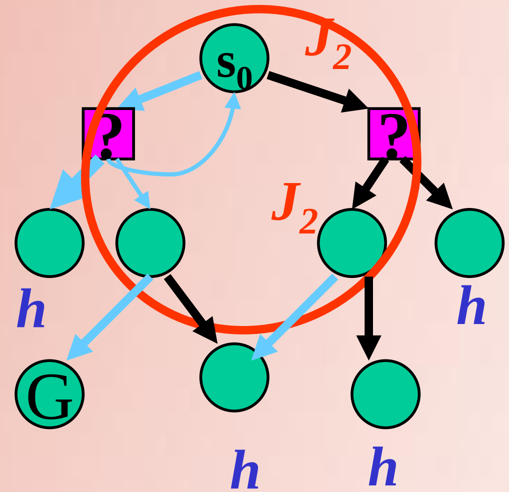
perform VI

compute greedy policy

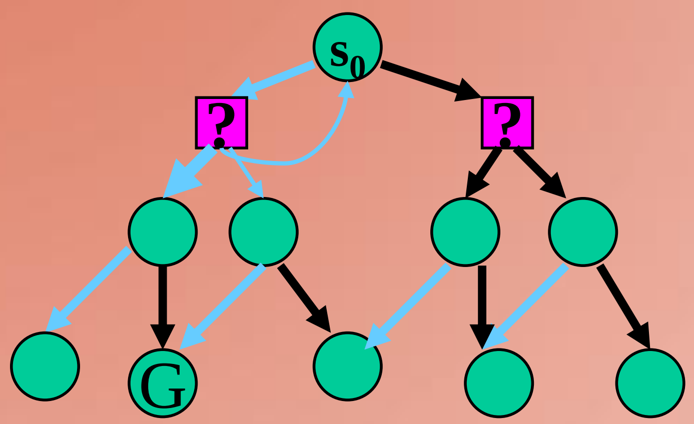
LAO* [Iteration 3]



expand fringe state

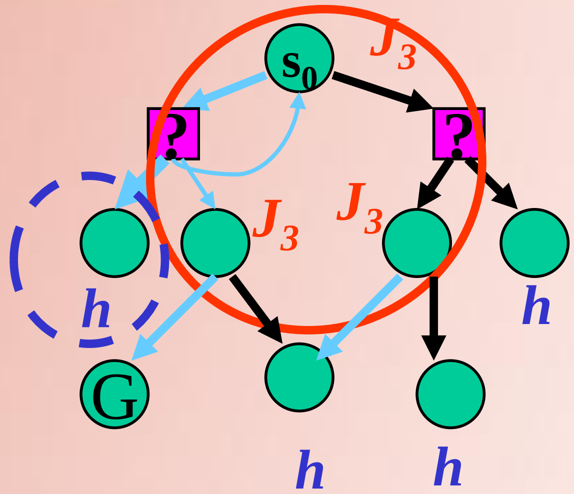


LAO* [Iteration 3]

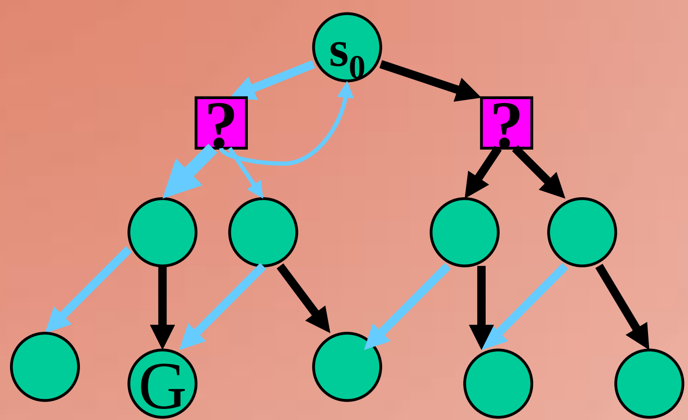


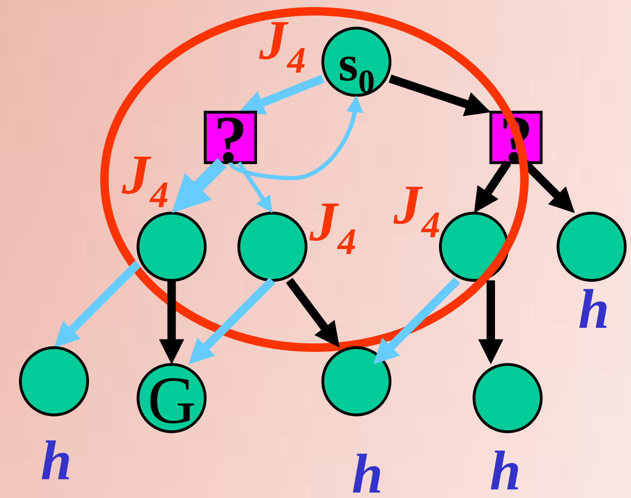
perform VI

recompute greedy graph

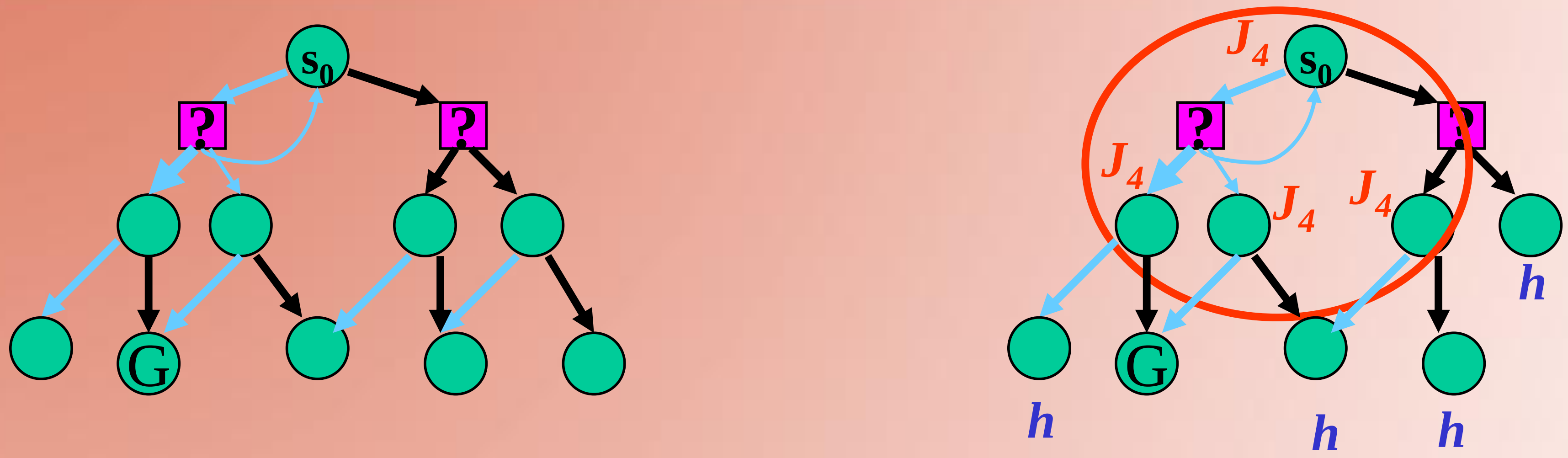


LAO* [Iteration 4]





LAO* [Iteration 4]



Stops when all nodes in greedy graph have been expanded

Comments

- Dynamic Programming + Heuristic Search
- admissible heuristic ⇒ optimal policy
- expands only part of the reachable state space
- outputs a partial policy
  - one that is closed w.r.t. to $\mathcal{P}$r and s_0

Speedups

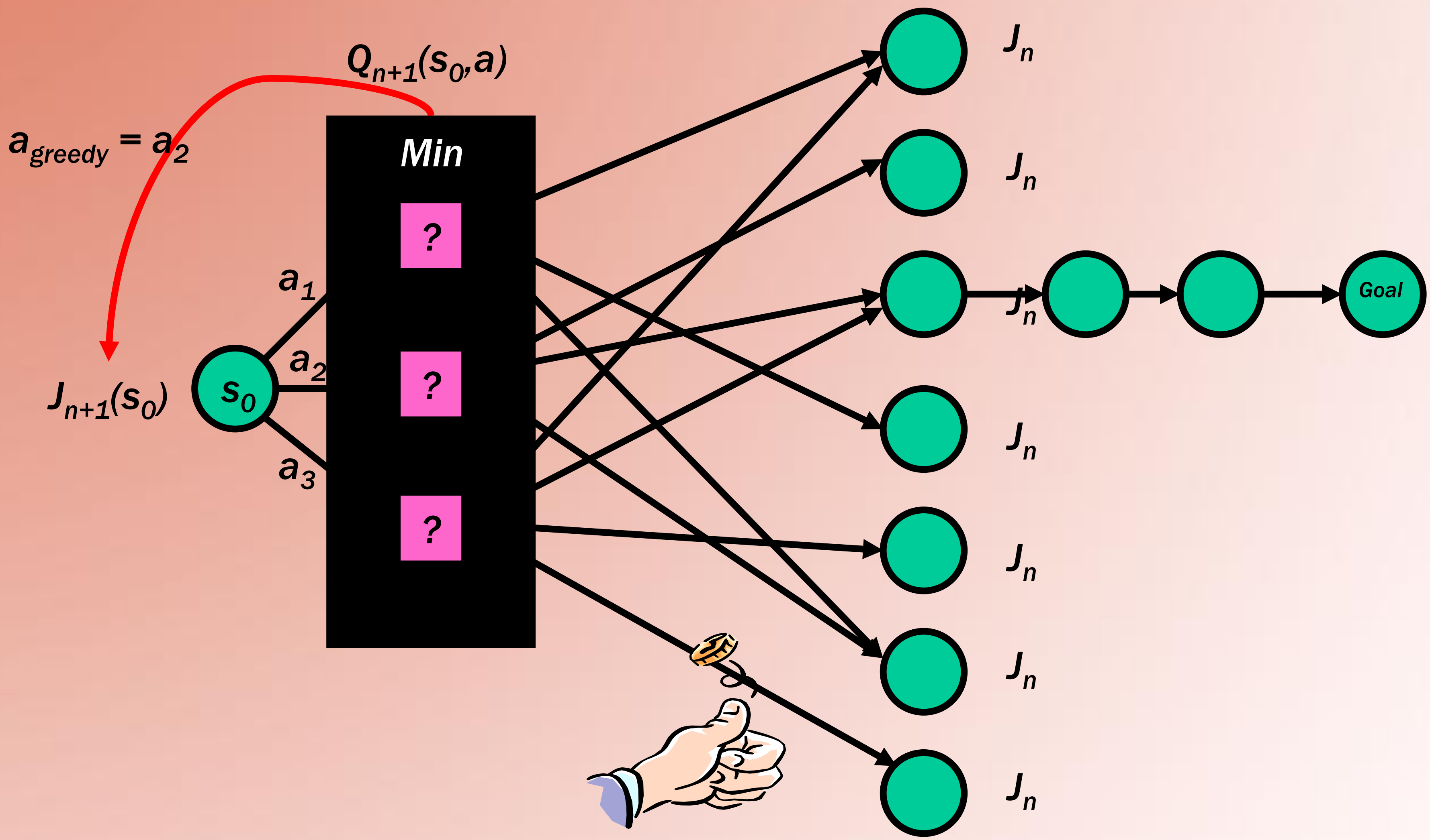
- expand all states in fringe at once
- perform policy iteration instead of value iteration
- perform partial value/policy iteration
- weighted heuristic: $f = (1-w).g + w.h$
- ADD based symbolic techniques (symbolic LAO*)

Real Time Dynamic Programming
[Barto, Bradtke, Singh'95]

- Trial: simulate greedy policy starting from start state;
  perform Bellman backup on visited states

- RTDP: repeat Trials until cost function converges

RTDP Trial

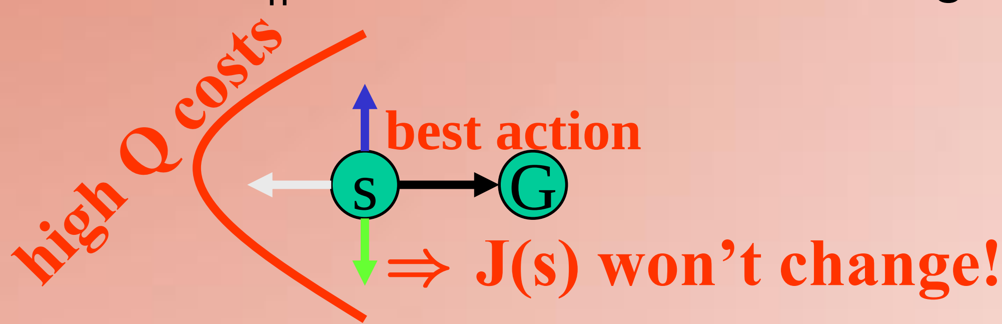


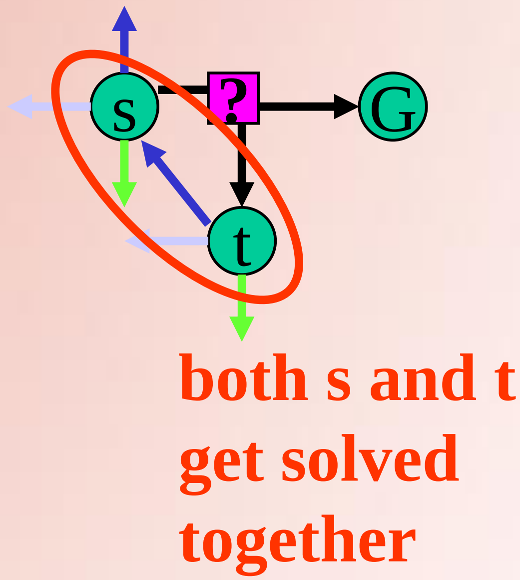
Comments

- Properties
  - if all states are visited infinitely often then $J_n \rightarrow J^*$
- Advantages
  - Anytime: more probable states explored quickly
- Disadvantages
  - complete convergence is slow!
  - no termination condition

Labeled RTDP [Bonet&Geffner'03]

- Initialise J_0 with an admissible heuristic
  - $\Rightarrow J_n$ monotonically increases
- Label a state as solved
  - if the J_n for that state has converged





- Backpropagate 'solved' labeling
- Stop trials when they reach any solved state
- Terminate with s_0 is solved

Properties

- admissible $J_0 \Rightarrow$ optimal J^*
- heuristic-guided
  - explores a subset of reachable state space
- anytime
  - focusses attention on more probable states
- fast convergence
  - focusses attention on unconverged states
- terminates in finite time

Recent Advances: Bounded RTDP
[McMahan, Likhachev & Gordon'05]

- Associate with each state
  - Lower bound (lb): for simulation
  - Upper bound (ub): for policy computation
  - gap(s) = ub(s) - lb(s)
- Terminate trial when gap(s) < ε
- Bias sampling towards unconverged states
  - proportional to $\mathcal{P}$r(s'|s,a).gap(s')
- Perform backups in reverse order for current trajectory.

Recent Advances: Focused RTDP [Smith&Simmons'06]

Similar to Bounded RTDP except

- a more sophisticated definition of priority that combines gap and prob. of reaching the state
- adaptively increasing the max-trial length

Recent Advances: Learning DFS [Bonet&Geffner'06]

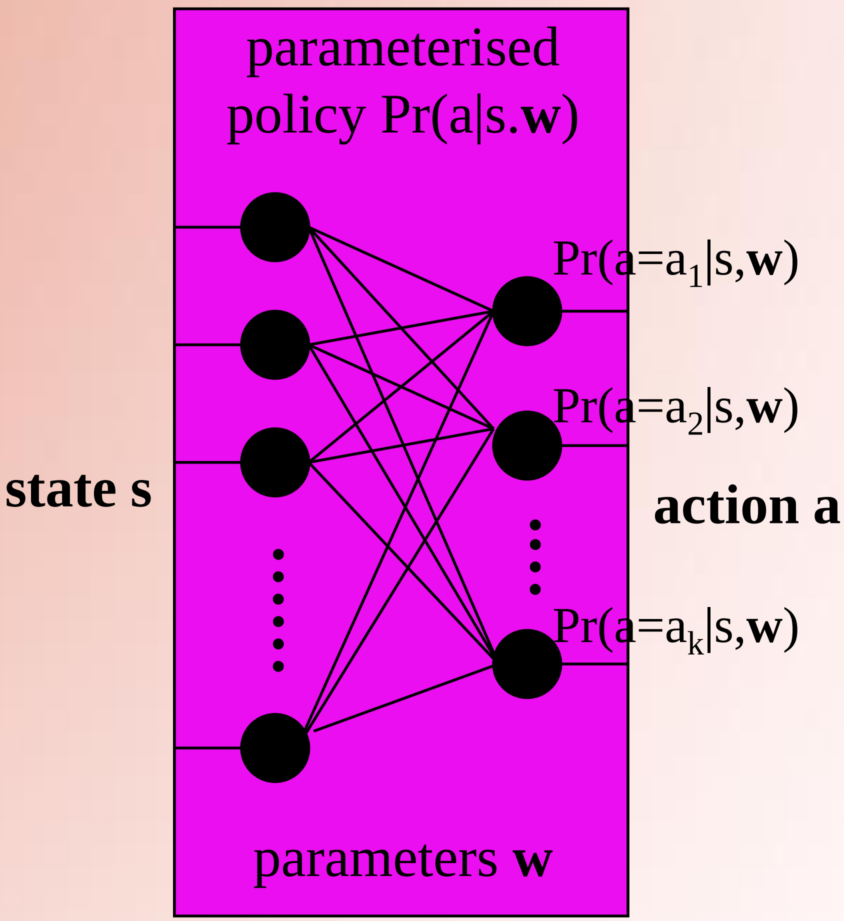
- Iterative Deepening A* equivalent for MDPs
- Find strongly connected components to check for a state being solved.

Other Advances

- Ordering the Bellman backups to maximise information flow.
  - [Wingate & Seppi'05]
  - [Dai & Hansen'07]
- Partition the state space and combine value iterations from different partitions.
  - [Wingate & Seppi'05]
  - [Dai & Goldsmith'07]
- External memory version of value iteration
  - [Edelkamp, Jabbar & Bonet'07]
- …

Policy Gradient Approaches [Williams'92]

- direct policy search
  - parameterised policy Pr(a|**s**,**w**)
  - no value function
  - flexible memory requirements
- policy gradient
  - $J(\mathbf{w})=E_{\mathbf{w}}[\sum_{t=0..\infty}\gamma^t c_t]$
  - gradient descent (wrt **w**)
  - reaches a local optimum
  - continuous/discrete spaces



Policy Gradient Algorithm

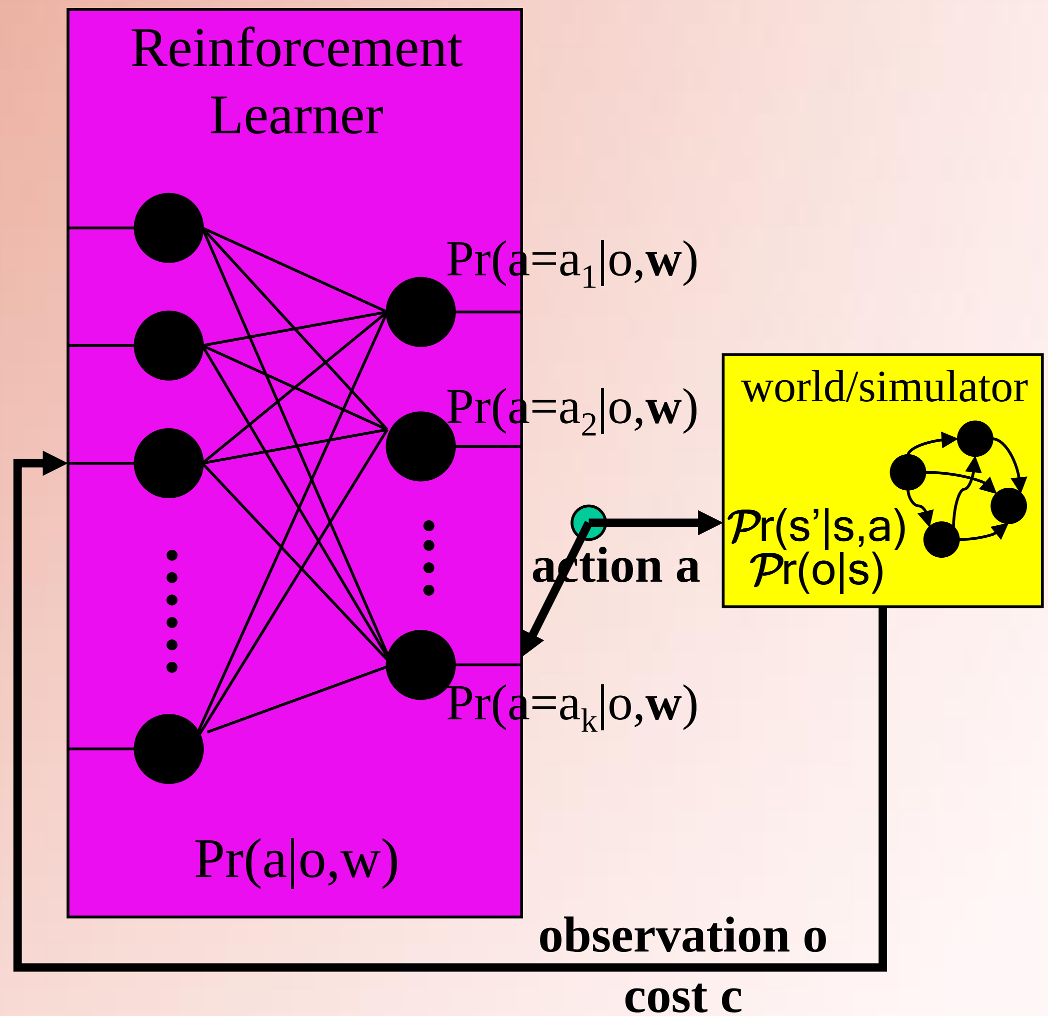
- $J(\mathbf{w})=E_{\mathbf{w}}[\sum_{t=0..\infty}\gamma^t c_t]$ (failure prob.,makespan, …)
- minimise J by
  - computing gradient $\nabla J(\mathbf{w}) = \left[\frac{\partial J}{\partial \mathbf{w}_1}, \frac{\partial J}{\partial \mathbf{w}_2}, \ldots, \frac{\partial J}{\partial \mathbf{w}_k}\right]$
  - stepping the parameters away $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\nabla J(\mathbf{w})$
- until convergence

- Gradient Estimate [Sutton et.al.'99, Baxter & Bartlett'01]
- Monte Carlo estimate from trace $s_1, a_1, c_1, \ldots, s_T, a_T, C_T$
  - $\mathbf{e}_{t+1} = \mathbf{e}_t + \nabla_{\mathbf{w}} \log \Pr(a_{t+1}|s_t,\mathbf{w}_t)$
  - $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\gamma^t c_t \mathbf{e}_{t+1}$

Policy Gradient Approaches

- often used in reinforcement learning
  - partial observability
  - model free ($\mathcal{P}r(s'|s,a)$, $\mathcal{P}r(\mathbf{o}|s)$ are unknown)

to learn a policy from observations and costs



LP Formulation of MDPs

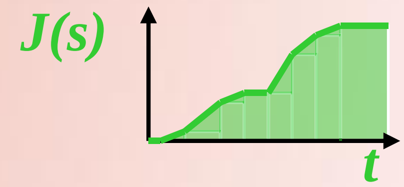
maximise $\Sigma_{s \in \mathcal{S}} \alpha(s) J^*(s)$

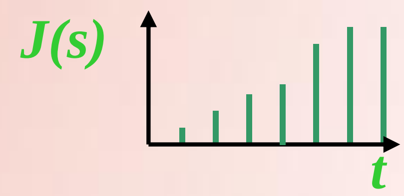
under constraints

- for $s \in \mathcal{G}$: $J^*(s) = 0$
- for every s, a:
  $J^*(s) \leq \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|a,s)[\mathcal{C}(s,a,s') + J^*(s')]$
- $\alpha(s) > 0$

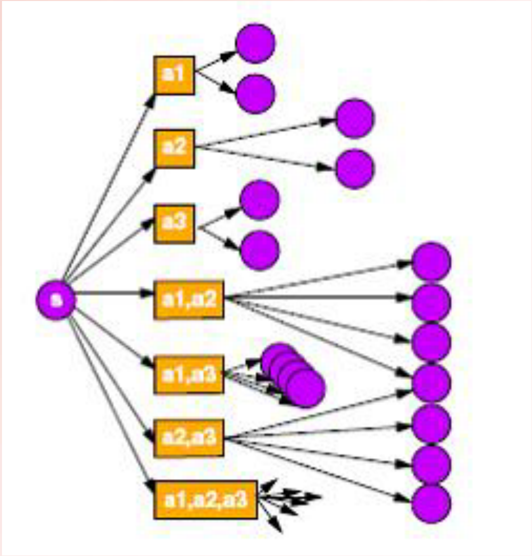
Modeling Complex Problems

- Modeling time
  - continuous variable in the state space
  - discretisation issues
  - large state space





- Modeling concurrency
  - many actions may execute at once
  - large action space
- Modeling time and concurrency
  - large state and action space!!



References

- Simple statistical gradient following algorithms for connectionist reinforcement learning. R. J. Williams. Machine Learning, 1992.
- Learning to Act using Real-Time Dynamic Programming. Andrew G. Barto, Steven J. Bradtke, Satinder P. Singh. Artificial Intelligence, 1995.
- Policy Gradient Methods for Reinforcement Learning with Function Approximation. Richard S. Sutton, David A. McAllester, Satinder P. Singh, Yishay Mansour. NIPS 1999.
- Infinite-Horizon Policy-Gradient Estimation. Jonathan Baxter and Peter L. Bartlett. JAIR 2001.
- LAO*: A Heuristic Search Algorithm that Finds Solutions with Loops. E.A. Hansen and S. Zilberstein. Artificial Intelligence, 2001.
- Labeled RTDP: Improving the Convergence of Real-Time Dynamic Programming. Blai Bonet and Héctor Geffner. ICAPS 2003.

References

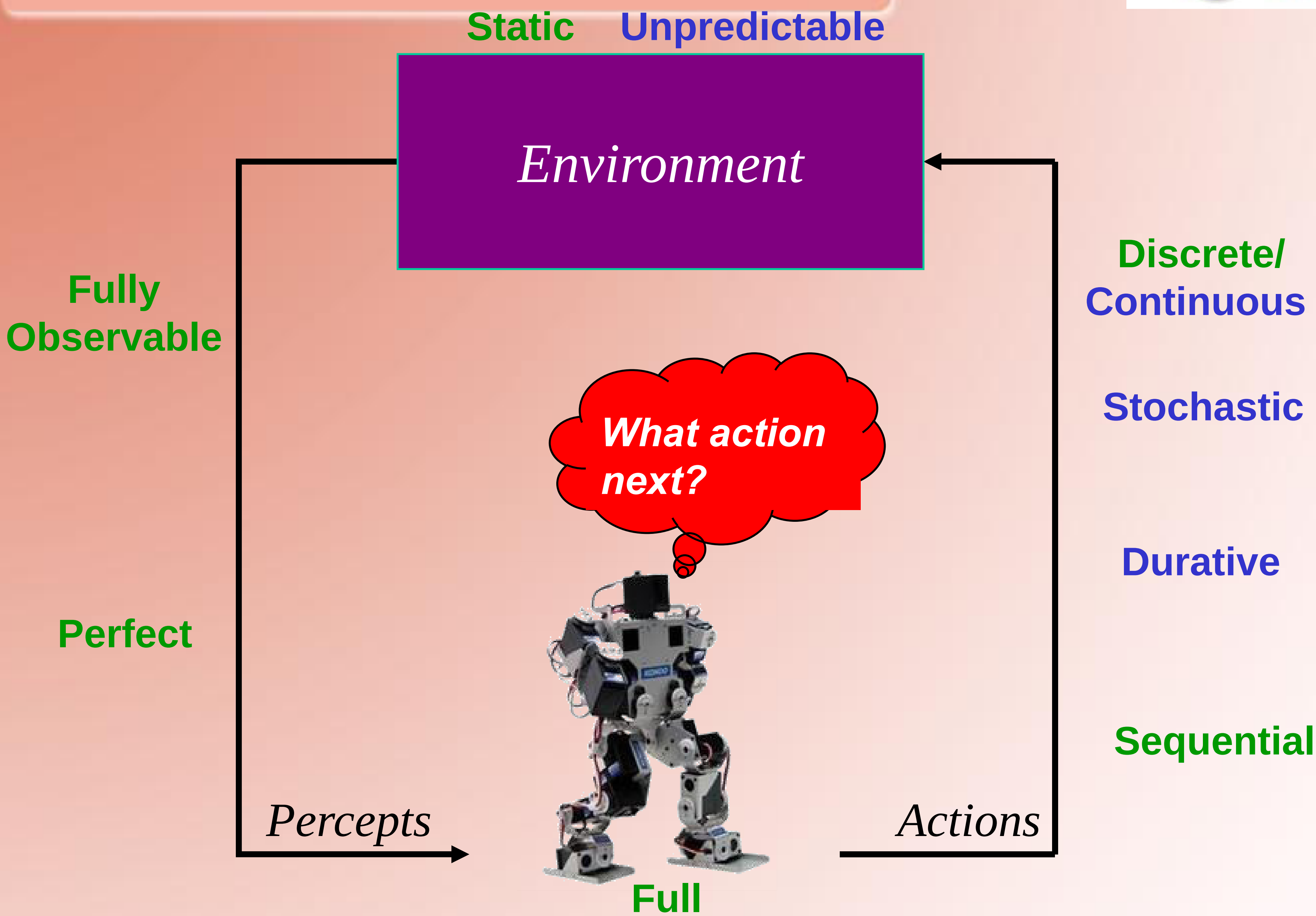
- Bounded Real-Time Dynamic Programming: RTDP with monotone upper bounds and performance guarantees. H. Brendan McMahan, Maxim Likhachev, and Geoffrey Gordon. ICML 2005.
- Learning Depth-First Search: A Unified Approach to Heuristic Search in Deterministic and Non-Deterministic Settings, and its application to MDPs. Blai Bonet and Héctor Geffner. ICAPS 2006.
- Focused Real-Time Dynamic Programming for MDPs: Squeezing More Out of a Heuristic, Trey Smith and Reid Simmons. AAAI 2006.
- Prioritization Methods for Accelerating MDP Solvers. David Wingate, Kevin Seppi. JMLR 2005.
- Topological Value Iteration Algorithm for Markov Decision Processes. Peng Dai, Judy Goldsmith. IJCAI 2007.
- Prioritizing Bellman Backups Without a Priority Queue. Peng Dai and Eric Hansen. ICAPS 2007.
- External Memory Value Iteration. Stefan Edelkamp, Shahid Jabbar and Blai Bonet. ICAPS 2007.

Probabilistic Temporal Planning

PART III: Durative Actions without Concurrency

Mausam
David E. Smith
Sylvie Thiébaux

Stochastic Planning w/ Durative Actions



Motivation

- Why are durative actions important?
  - Race against time: deadlines
  - Increase reward (single goal): time dependent reward
  - Increase reward (many non-absorbing goals)
    - oversubscription Planning
    - achieve as many goals as possible in the given time
- Why is uncertainty important?
  - durations could be uncertain
  - we may decide the next action based on the time taken by the previous ones.

Different Related Models

MDP < SMDP
TMDP < HMDP

- MDP
  no explicit action durations
- Semi-MDP
  continuous/discrete action durations
  discounted/undiscounted
- Time-dependent MDP
  discrete MDP + one continuous variable – time
  undiscounted
- Continuous MDP
  MDP with only continuous variables
- Hybrid MDP
  MDP with many discrete and continuous variables

discounting w/ action durations

undiscounted deadline problems.

Undiscounted/Discrete-time/No-deadline

- Embed the duration information in $\mathcal{C}$ or $\mathcal{R}$
- Minimise make-span

  initialise $\mathcal{C}$ by its duration

$$\begin{aligned} J^*(s) &= 0 \;\; if \;\; s \in \mathcal{G} \\ J^*(s) &= \min_{a \in Ap(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}r(s'|s,a) \left[\mathcal{C}(s,a,s') + J^*(s') \right] \end{aligned}$$

$$\begin{aligned} J^*(s) &= 0 \;\; if \;\; s \in \mathcal{G} \\ J^*(s) &= \min_{a \in Ap(s)} \sum_{s',N} \mathcal{P}r(s',N|s,a) \left[N + J^*(s') \right] \end{aligned}$$

Discounted/Discrete-time/No-deadline

- A single γ won't describe the dynamics accurately!

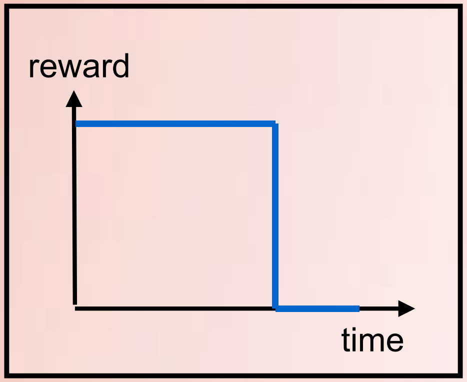
$$V^*(s) = \max_{a \in Ap(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}r(\cancel{s'}|s,a)\left[\mathcal{R}(s,a,s') + \cancel{\gamma} V^*(s')\right]$$

$$V^*(s) = \max_{a \in Ap(s)} \sum_{s',N} \mathcal{P}r(s',N|s,a)\left[\mathcal{R}(s,a,s') + \gamma^N V^*(s')\right]$$

Semi-MDP [Howard'71]

Undiscounted/Discrete-time/Deadline

- V* depends on
  - current state
  - current time



$$V^*(s,t) = \max_{a \in Ap(s)} \sum_{s',N} \mathcal{P}r(s',N|s,a)\left[\mathcal{R}(s,t,a,s',t+N) + V^*(s',t+N)\right]$$

Time-dependent MDP **[Boyan and Littman'00]**

Undiscounted/Continuous-time/Deadline

- Summation is now integral!

$$V^*(s,t) \;=\; \max_{a\in Ap(s)} \cancel{\sum_{s',N}} \mathcal{P}r(s',N|s,a)\left[\mathcal{R}(s,t,a,s',t+N)+V^*(s',t+N)\right]$$

$$\begin{aligned} V^*(s,t) &= \max_{a\in Ap(s)} \sum_{s'} \int_0^{\infty} \mathcal{P}r(s',N|s,a)\left[\mathcal{R}(s,t,a,s',t+N)+V^*(s',t+N)\right]dN \\ &= \max_{a\in Ap(s)} \sum_{s'} \mathcal{P}r(s'|s,a) \int_0^{\infty} \mathcal{P}r(N|s,a,s')\left[\mathcal{R}(s,t,a,s',t+N)+V^*(s',t+N)\right]dN \end{aligned}$$

Discounted/Continuous-time/No-deadline

$$
\begin{aligned}
V^*(s) &= \max_{a\in Ap(s)} \sum_{s'} \int_0^\infty \mathcal{P}r(s', N|s,a)\left[\mathcal{R}(s,a,s',N) + \gamma^N V^*(s')\right] dN \\
&= \max_{a\in Ap(s)} \sum_{s'} \mathcal{P}r(s'|s,a) \int_0^\infty \mathcal{P}r(N|s,a,s')\left[\mathcal{R}(s,a,s',N) + \gamma^N V^*(s')\right] dN
\end{aligned}
$$

convolutions

Algorithms

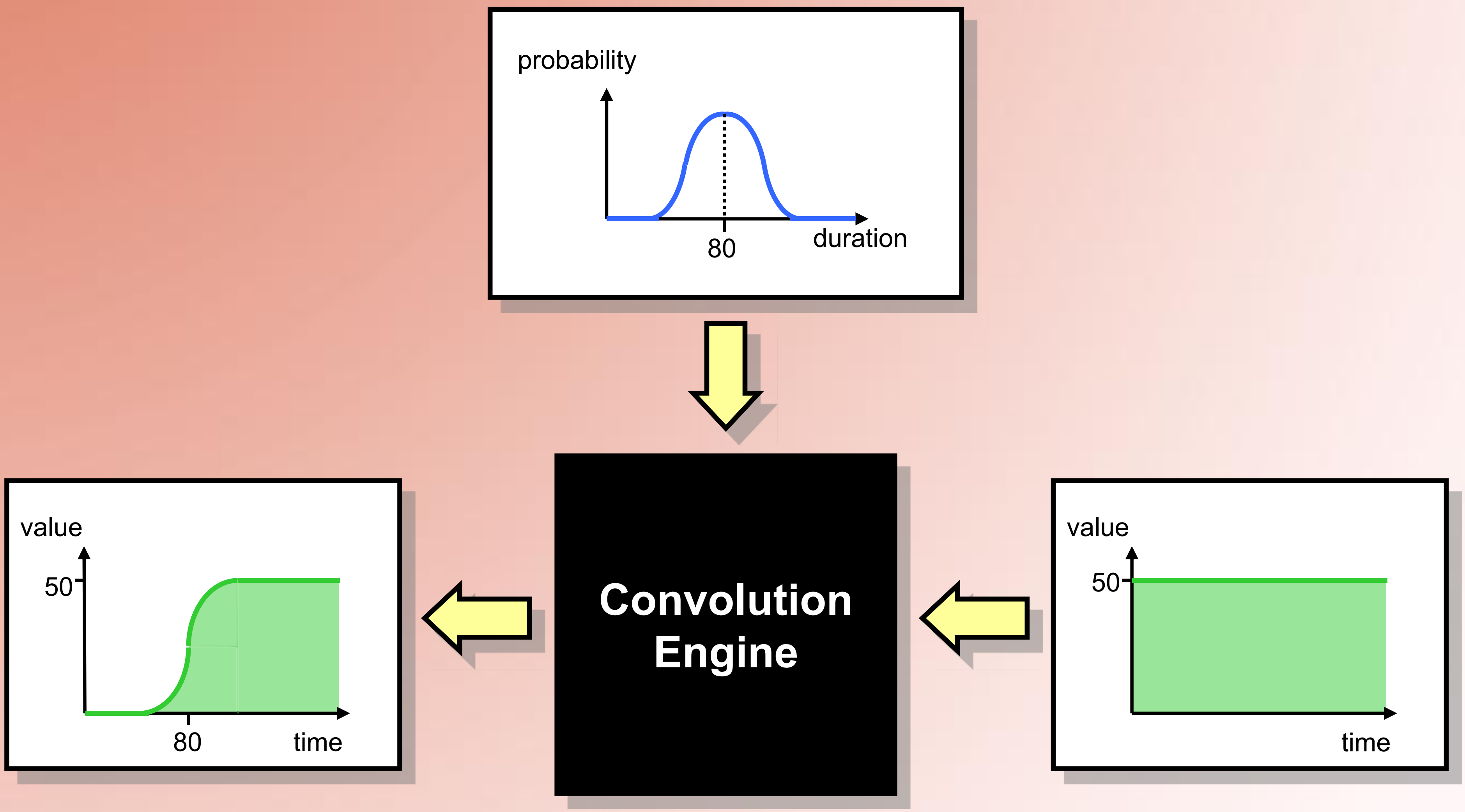
- All previous algorithms extend
  - **with new Bellman update rules**
  - e.g. value iteration, policy iteration, linear prog.
- Computational/representational challenges
  - efficient represent of continuous value functions
  - efficient computation of convolutions
- Algorithm extensions
  - reachability analysis in continuous space?

Representation of Continuous Functions

- flat discretisation
  - costly!
- piecewise constant
  - models deadline problems
- piecewise linear
  - models minimise make-span problems
- phase type distributions
  - approximates arbitrary probability density functions
- piecewise gamma function

Convolutions

$$\text{Convolution} = \int_0^\infty \mathcal{P}r(N|s,a,s')V^*(s',t+N)dN$$



Result of convolutions

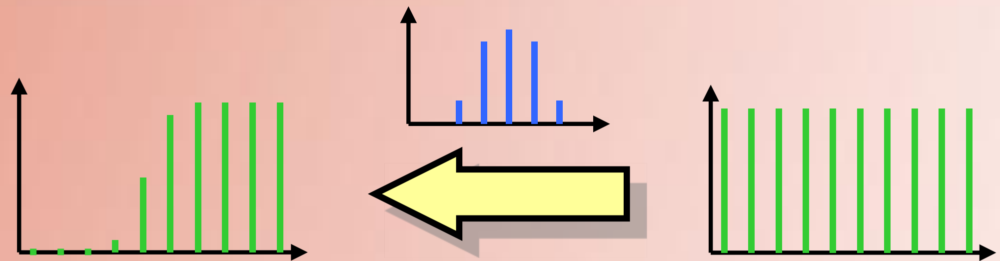
value function

probability density function

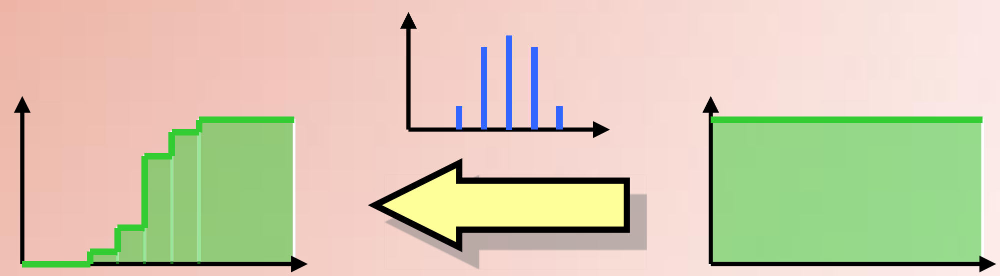
	discrete	constant	linear
discrete	discrete	constant	linear
constant	constant	linear	quadratic
linear	linear	quadratic	cubic

Convolutions

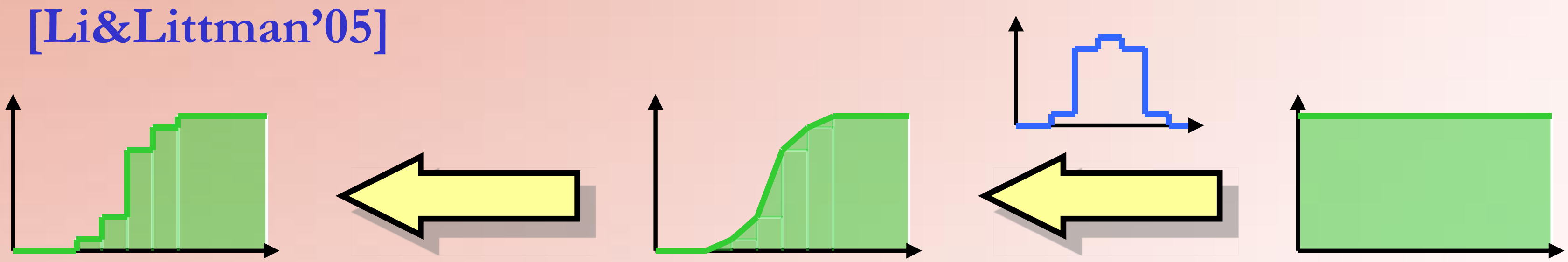
discrete-discrete



constant-discrete
[Feng et.al.'04]



constant-constant
[Li&Littman'05]



Analytical solution to convolutions
[Marecki, Koenig, Tambe'07]

UNIVERSITY OF WASHINGTON

NASA

NICTA

- probability function approximated one time
  - as phase-type distribution $p(N)= \lambda e^{-\lambda N}$
- value function is piecewise gamma

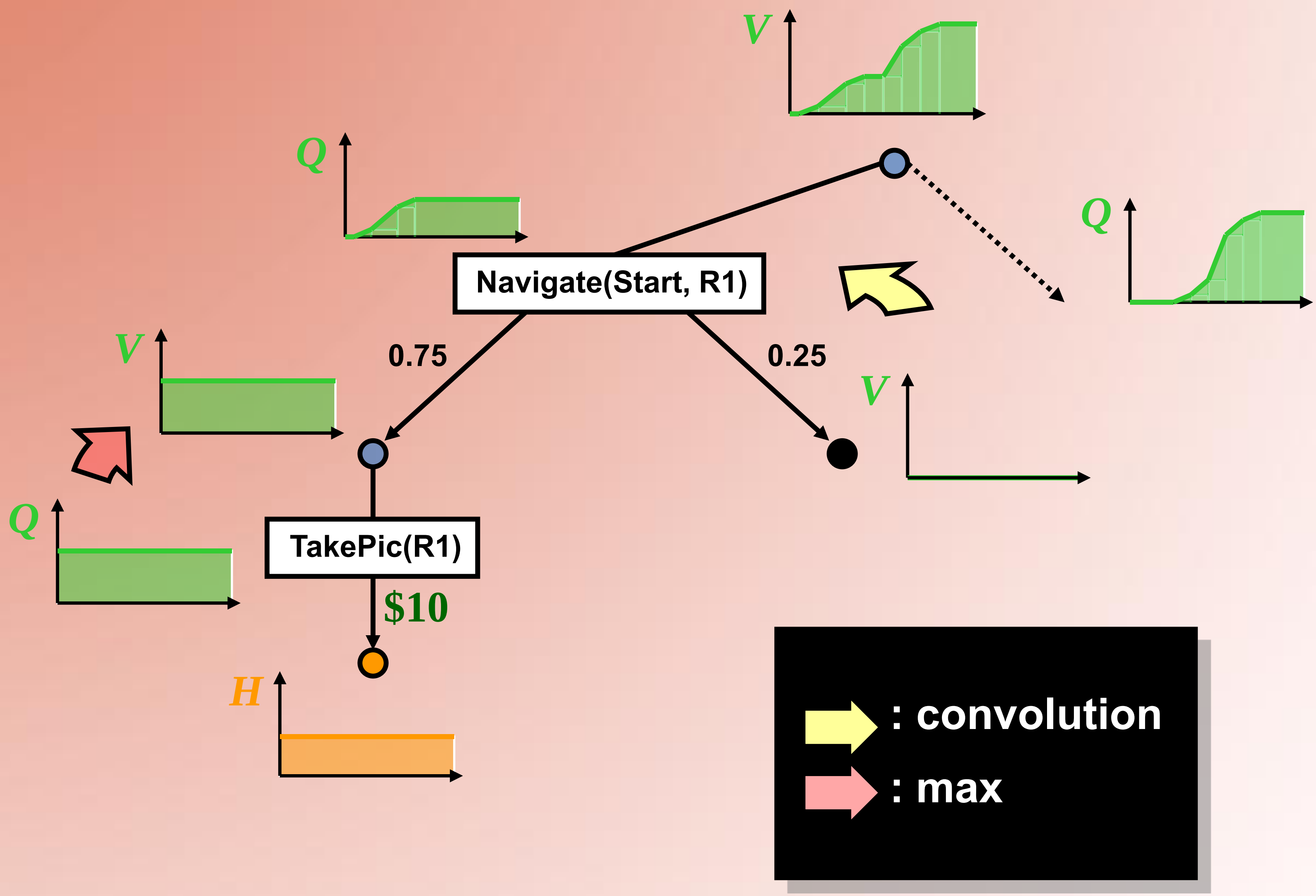
$$\text{gamma fn.} = c - e^{-\lambda t}\left(a_0 + a_1(\lambda t) + \ldots + \frac{a_n(\lambda t)^n}{n!}\right)$$

- convolutions can be computed analytically!

Hybrid AO* [Mausam et.al'05]

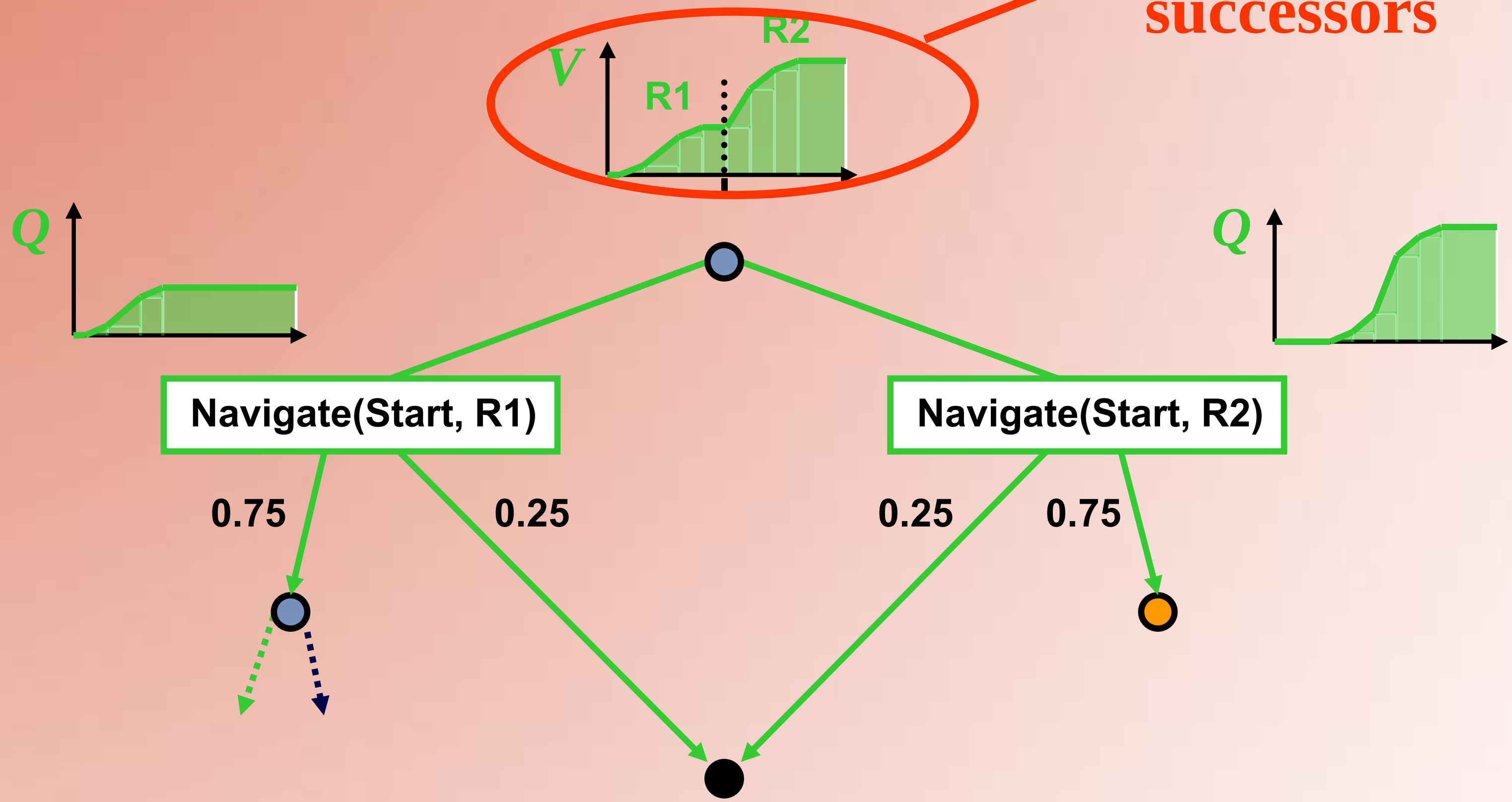
- search in discrete state space.
- associate piecewise constant value functions with each discrete node.
- employ sophisticated continuous reachability.

Hybrid AO*


V
Q
Q
Navigate(Start, R1)
0.75
0.25
V
V
Q
TakePic(R1)
$10
H
: convolution
: max

Hybrid AO*

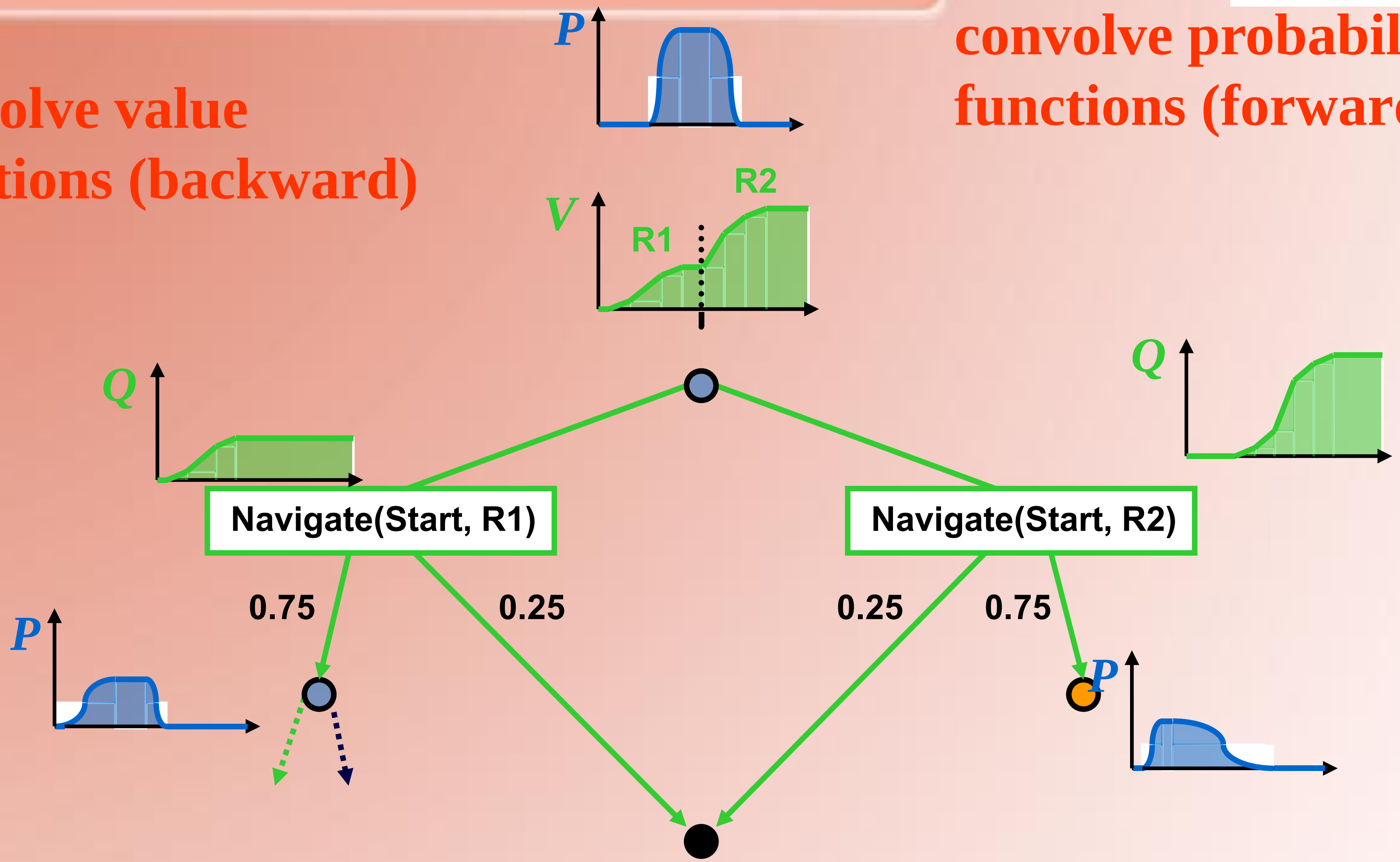
many greedy successors



Hybrid AO*

convolve value functions (backward)

convolve probability functions (forward)



References

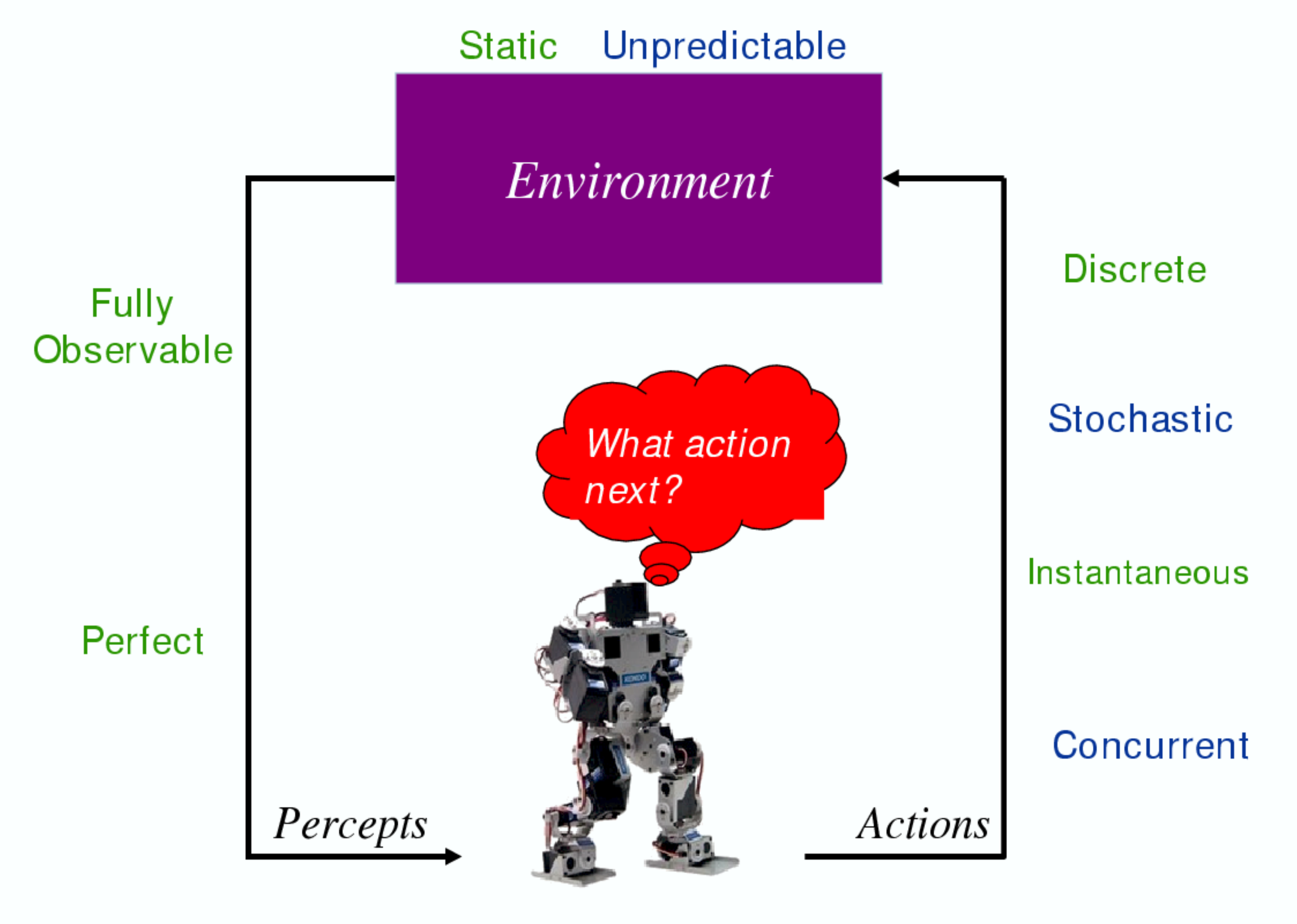
- Markov Decision Processes: Discrete Stochastic Dynamic Programming. Martin Puterman. John Wiley and Sons 1994.
- Dynamic Programming and Optimal Control. Dmitri Bertsekas. Athena Scientific 1995.
- Exact Solutions to Time-Dependent MDPs. Justin Boyan and Micheal Littman. NIPS 2000.
- Dynamic Programming for Structured Continuous Markov Decision Problems. Zhengzhu Feng, Richard Dearden, Nicolas Meuleau, and Richard Washington. UAI 2004.
- Lazy approximation for solving continuous finite-horizon MDPs. Lihong Li and Michael L. Littman. AAAI 2005.
- Planning with Continuous Resources in Stochastic Domains. Mausam, Emmanuelle Benazara, Ronen Brafman, Nicolas Meuleau, Eric Hansen. IJCAI 2005.
- A Fast Analytical Algorithm for Solving Markov Decision Processes with Real-Valued Resources. J. Marecki, Sven Koenig and Milind Tambe. IJCAI 2007.

Probabilistic Temporal Planning

PART IV: Concurrency w/o Durative Actions

Mausam, David E. Smith, Sylvie Thiébaux

Stochastic Planning



Plan for Part IV

- Concurrent MDP (CoMDP) Model
- Value-Based Algorithms
- Planning Graph Approaches
- Policy Gradient Approaches
- Related Models

Concurrent MDPs (CoMDPs)

- formally introduced by Mausam & Weld [AAAI-04]
- MDP that allows simultaneous execution of action sets
- $\neq$ semi-MDPs where time is explicit but concurrency is lacking
- cost of an action set accounts for time and resources

- notion of concurrency (mutex), generalising *independence* (deterministic actions a and b are independent iff $a; b \equiv b; a$):

restrictive: all executions of the actions are independent

permissive: some execution is independent; requires failure states

Concrete Independence Example

Probabilistic STRIPS:

- each action has a set of preconditions and a probability distribution over a set of outcomes
- each outcome has sets of positive and negative effects
- an outcome set is consistent when no outcome deletes a positive effect or the precondition of another('s action)
- a set of actions is independent when:

  restrictive: all joint outcomes of the actions are consistent

  permissive: at least one joint outcome is consistent

Concurrent MDPs (CoMDPs)

UNIVERSITY OF WASHINGTON

NASA

NICTA

MDP equivalent to a CoMDP

A CoMDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{C}, \mathcal{G}, s_0 \rangle$ translates into the MDP $\langle \mathcal{S}, \mathcal{A}_{||}, \mathcal{P}r_{||}, \mathcal{C}_{||}, \mathcal{G}, s_0 \rangle$:

- $\mathcal{A}_{||}(s)$: mutex-free subsets of actions $A = \{a_1, \ldots, a_k\} \subseteq \mathcal{A}(s)$
- due to independence
  $$\mathcal{P}r_{||}(s' \mid s, A) = \sum_{s_1 \in \mathcal{S}} \sum_{s_2 \in \mathcal{S}} \ldots \sum_{s_k \in \mathcal{S}} \mathcal{P}r(s_1 \mid s, a_1) \mathcal{P}r(s_2 \mid s1, a_2) \ldots \mathcal{P}r(s' \mid s_{k-1}, a_k)$$
- $\mathcal{C}_{||}(A) = \sum_{i=1}^{k} res(a_i) + \max_{i=1}^{k} dur(a_i)$

Plan for Part IV

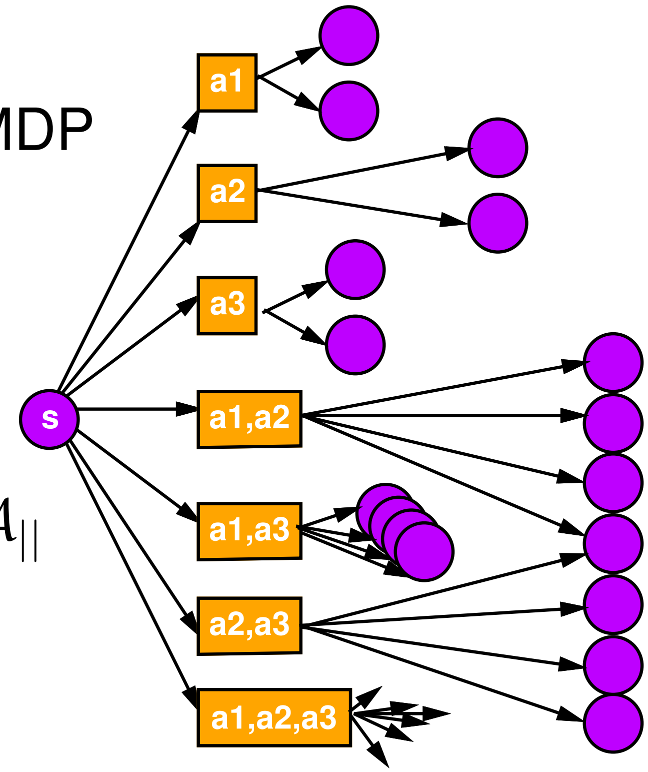
- Concurrent MDP (CoMDP) Model
- Value-Based Algorithms
- Planning Graph Approaches
- Policy Gradient Approaches
- Related Models

Value-Based Algorithms

- compute a proper optimal policy for the CoMDP
- dynamic programming, e.g., RTDP applies:

$$J_{||n}(s) = \min_{A \in \mathcal{A}_{||}(s)} Q_{||n}(s, A)$$

- need to mitigate the exponential blowup in $\mathcal{A}_{||}$
  1. pruning Bellman backups
  2. sampling Bellman backups



Pruning Bellman Backups

Theorem (Mausam & Weld AAAI-04)

Let U_n be an upper bound on $J_{||n}(s)$. If

$$U_n < \max_{i=1}^{k} Q_{||n}(s, \{a_i\}) + \mathcal{C}_{||}(A) - \sum_{i=1}^{k} \mathcal{C}_{||}(\{a_i\})$$

then combination A is not optimal for state s in this iteration.

Combo-skipping pruning rule:

1. compute $Q_{||n}(s, \{a\})$ for all applicable single actions
2. set $U_n \leftarrow Q_{||n}(s, A^*_{n-1})$, using the optimal combination A^*_{n-1} at the previous iteration
3. apply the theorem

Pruning Bellman Backups

Theorem (Bertsekas (1995))

Let L be a lower bound on $Q^*_{||}(s, A)$ and U be an upper bound on $J^*_{||}(s)$. If $L > U$ then A is not optimal for s.

Combo-elimination pruning rule:

1. initialise RTDP estimates with an admissible heuristic; $Q_{||n}(s, A)$ remain lower bounds
2. set U to the optimal cost of the serial MDP
3. apply the theorem

combo skipping: cheap but short-term benefits (try it first)

combo elimination: expensive but pruning is definitive

Sampling Bellman Backups

Backup random combinations

- bias towards action sets with previously best Q-values
- bias towards action sets built from best individual actions

Loss of optimality; $J_{||n}(s)$ might not monotonically increase

- do full backup when convergence is asserted for a state
- use (scaled down) result as heuristic to pruned RTDP

Plan for Part IV

- Concurrent MDP (CoMDP) Model
- Value-Based Algorithms
- Planning Graph Approaches
- Policy Gradient Approaches
- Related Models

Planning Graph Approaches

Motivated by the need to compress the state space

The planning graph data structure facilitates this by:

- exploiting a probabilistic STRIPS representation
- using problem relaxations to find cost lower bounds
- enabling goal-regression

History

- Graphplan [Blum & First IJCAI-95]
  - classical, concurrent, optimal
  - uses the graph as a heuristic and for goal regression search
- TGraphplan [Blum & Langford ECP-99]
  - replanner, concurrent, non-optimal
  - returns the most-likely trajectory to the goal
- PGraphplan [Blum & Langford ECP-99]
  - probabilistic contingent, non-concurrent, optimal
  - probabilistic graph yields a heuristic for DP
- Paragraph [Little & Thiébaux ICAPS-06]
  - probabilistic contingent, concurrent, optimal
  - extends the full Graphplan framework

Paragraph

- solves concurrent probabilistic STRIPS planning problems
- finds a concurrent contingency plan with smallest failure probability within a time horizon

$\Rightarrow$ goal-directed, finite horizon, prob. maximisation CoMDP

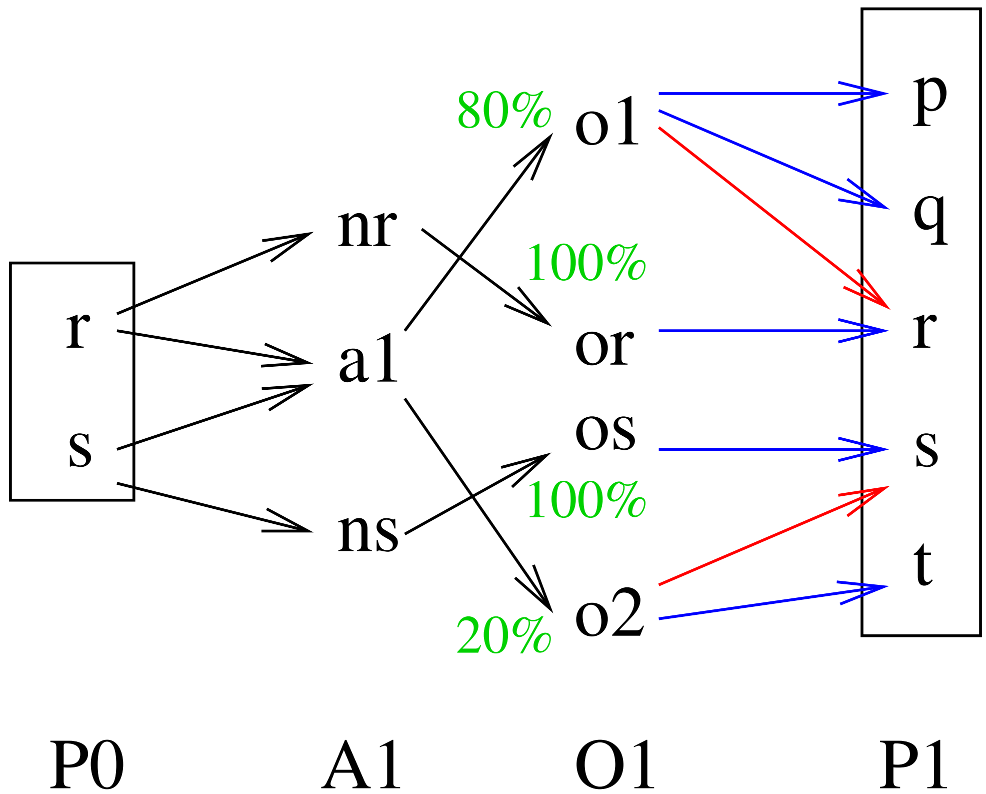
- has a cyclic version

Paragraph

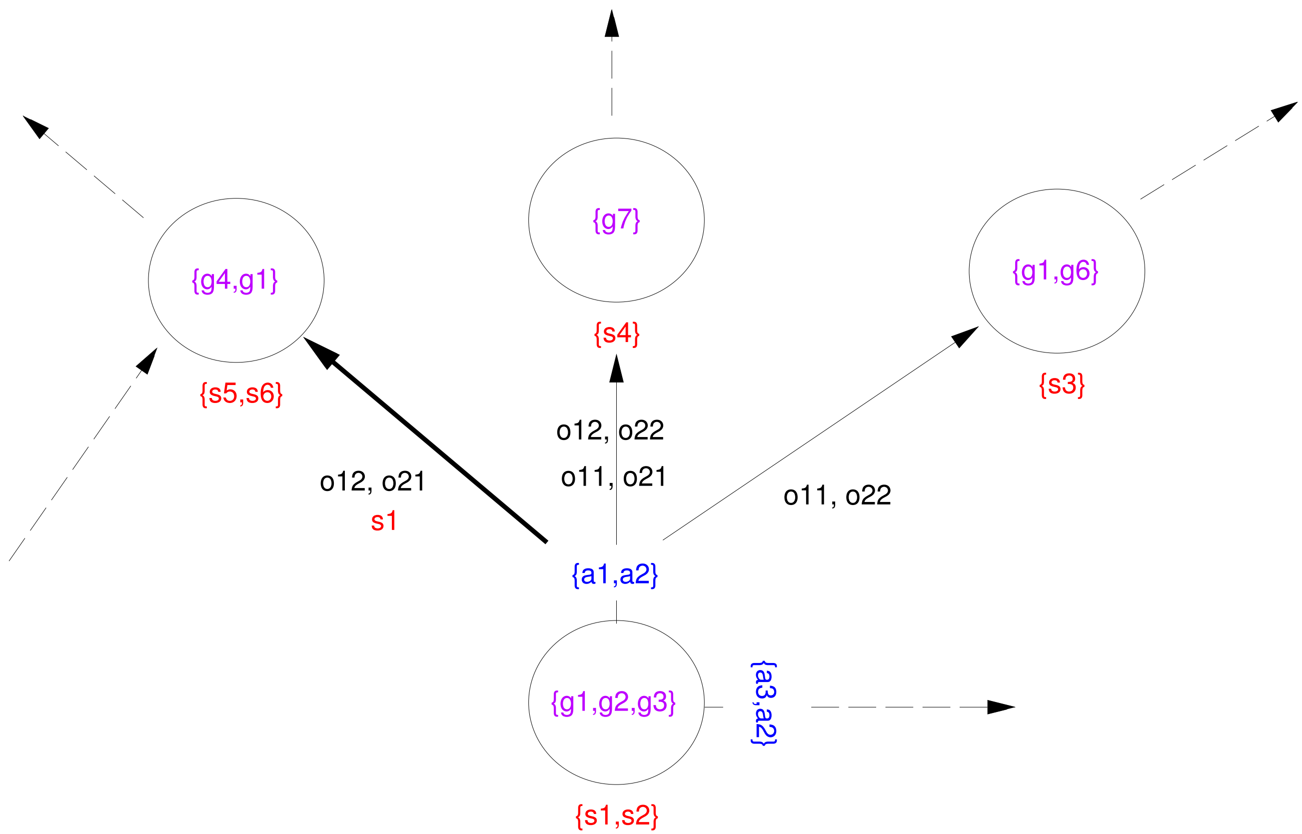
- Builds the probabilistic planning graph
  - until $G \subseteq P_i$ and G is mutex-free
- Attempts plan extraction
  - use goal regression search to find all trajectories that Graphplan would find
  - some of those will link naturally
  - additionally link other trajectories using forward simulation
- Alternates graph expansion and plan extraction
  - until the time horizon is exceeded or a plan of cost 0 is found (or goal unreachability can be proven)

Planning Graph (Probabilistic)

- action, propositions, and outcome levels and mutexes

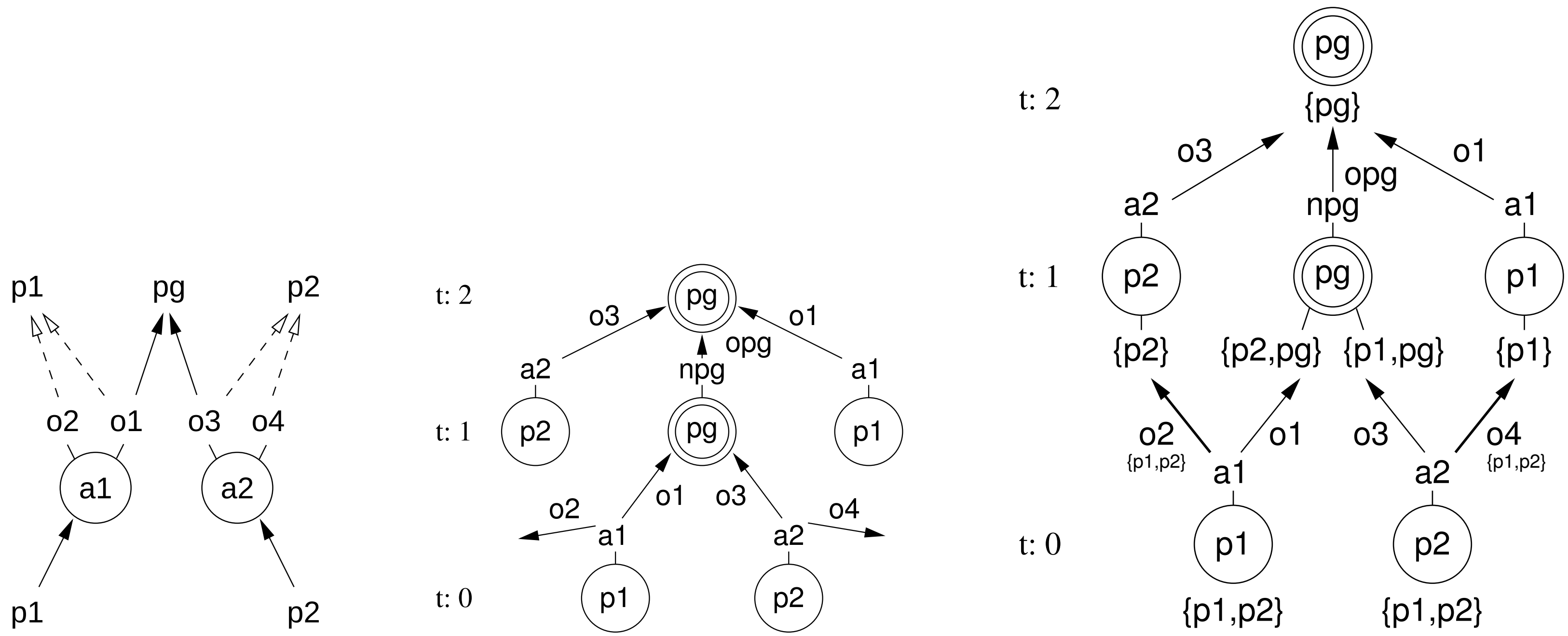


Goal-Regression Search (Probabilistic)



- nodes: goal set, action sets, world states set, cost, (time)
- arcs: joint outcome, (world state for conditional arcs)
- requires extra linking via forward simulation

Why do we need extra linking?



- $I = \{p1, p2\}$, $G = \{pg\}$
- optimal plan: execute one action; if it fails execute the other

Plan Extraction

- ends with forward simulation and backward cost update
- each node/world state pair yields a potential plan step
- select pairs and action sets with optimal cost

Cost (prob. failure) of a node/world state pair

$$C(n, s_n) =$$

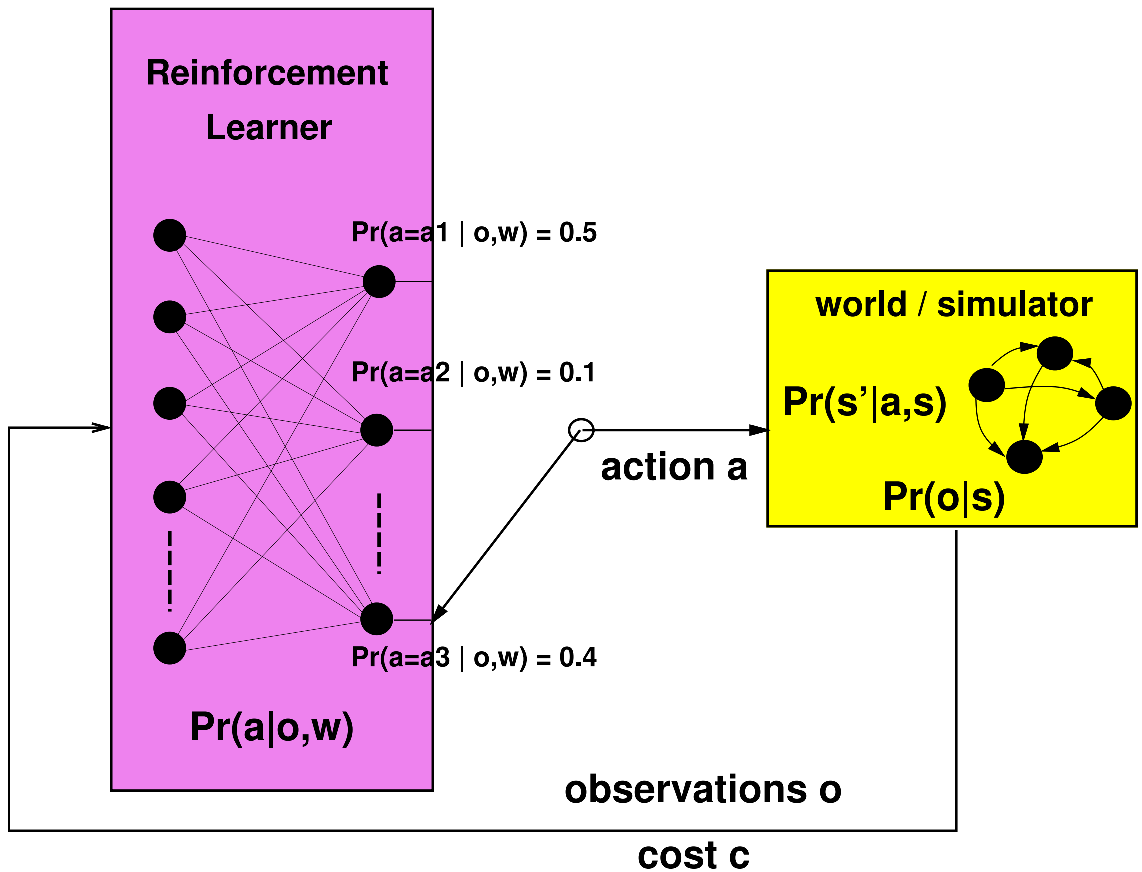
$$\begin{cases} 0 & \text{if } n \text{ is a goal node} \\ \min\limits_{A \in \mathrm{act}(n)} \sum\limits_{O \in \mathrm{Out}(A)} \Pr(O) \times \min\limits_{n' \in \mathrm{succ}(n, O, s_n)} C(n', res(O, s_n)) \end{cases}$$

Plan for Part IV

- Concurrent MDP (CoMDP) Model
- Value-Based Algorithms
- Planning Graph Approaches
- Policy Gradient Approaches
- Related Models

Policy Gradient Approaches

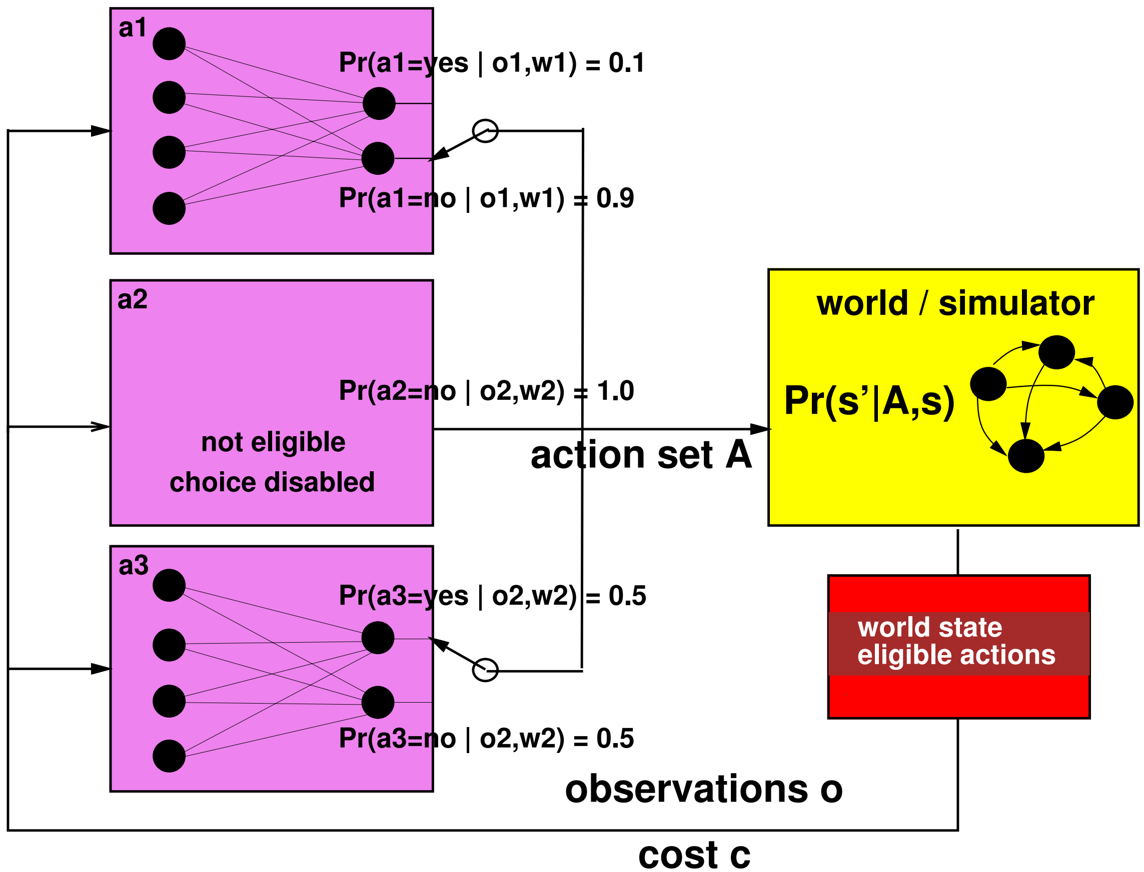
Minimise the expected cost of a parameterised policy by gradient descent in the parameters space.



Factored policy gradient

- need to mitigate the blowup caused by CoMDPs
- factorise the CoMDP policy into individual action policies

[Peshkin *et. al* UAI-00, Aberdeen & Buffet ICAPS-07]



Factored policy gradient

Theorem (Peshkin *et. al*, UAI-00)

- *For factored policies, factored policy gradient is equivalent to joint policy gradient.*
- *Every strict Nash equilibrium is a local optimum for policy gradient in the space of parameters of a factored policy, but not vice versa.*

FPG planner [Aberdeen & Buffet, 2007]

- did well in the probabilistic planning competition
- has a more efficient parallel version
- cost function favors reaching the goal as soon as possible
- individual policies are linear networks with prob. function:

$$\Pr(a_{it} = yes \mid \mathbf{o}_t, \mathbf{w_i}) = \frac{1}{\exp(\mathbf{o}_t^\top \mathbf{w}_i) + 1}$$

Plan for Part IV

- Concurrent MDP (CoMDP) Model
- Value-Based Algorithms
- Planning Graph Approaches
- Policy Gradient Approaches
- Related Models

Related Models

- range of decentralised MDP models [Goldman & Zilberstein AIJ-04]

Composite MDPs [Singh & Cohn NIPS-97]

- n component MDPs $\langle \mathcal{S}_i, \mathcal{A}_i, \mathcal{P}r_i, \mathcal{R}_i, s_{0\,i} \rangle$
- composite MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{R}, s_0 \rangle$ satisfies:
  - $\mathcal{S} = \prod_{i=1}^{n} \mathcal{S}_i, \quad s_0 = \prod_{i=1}^{n} s_{0\,i}$
  - $\mathcal{A}(s) \subseteq \prod_{i=1}^{n} \mathcal{A}_i(s)$ (constraints on simultaneous actions)
  - $\mathcal{P}r(s' \mid a, s) = \prod_{i=1}^{n} \mathcal{P}r_i(s'_i \mid a_i, s_i)$ (transition independence)
  - $\mathcal{R}(s, a, s') = \sum_{i=1}^{n} \mathcal{R}_i(s, a, s')$ (additive utility independence)

- useful for resource allocation [Meuleau *et. al* UAI-98]
- opt. solutions to component MDPs yield bounds for pruning composite MDPs (as in combo-elimination) [Singh & Cohn NIPS-97]
- composite value function can be approximated as a linear combination of component value functions [Guestrin *et. al* NIPS-01]

References

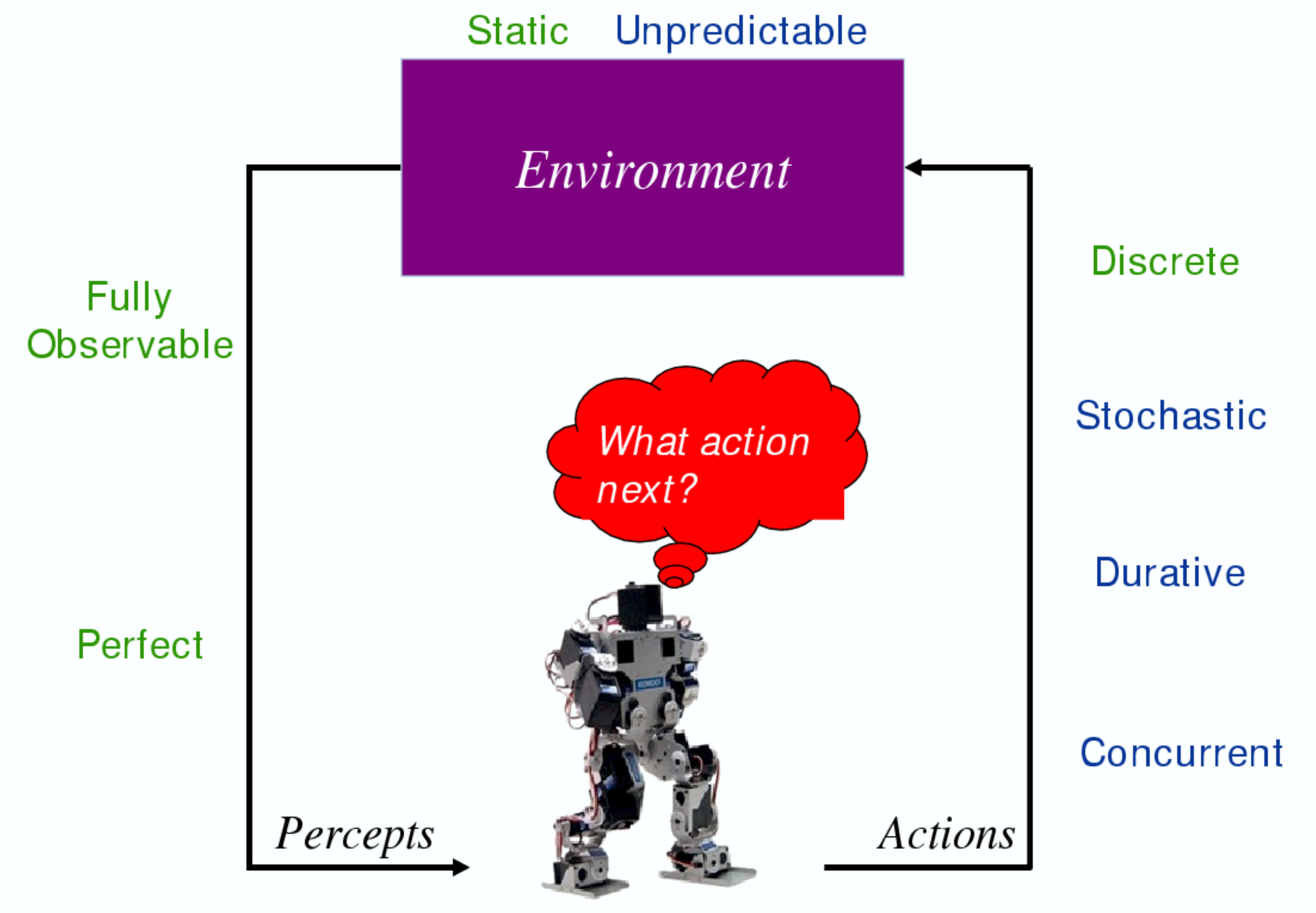
- How to Dynamically Merge Markov Decision Processes, S. Singh and D. Cohn. NIPS-97.
- Solving Very Large Weakly Coupled Markov Decision Processes, N. Meuleau, M. Hauskrecht, K.-E. Kim, L. Peshkin, L. Kaelbling, and T. Dean. UAI-98.
- Learning to Cooperate via Policy Search, L. Peshkin, K.-E. Kim, N. Meuleau, L. Kaelbling. UAI-00.
- Multi-Agent Planning with Factored MDPs, C. Guestrin, D Koller, and R. Parr. NIPS-01.
- Decentralized Control of Cooperative Systems: Categorization and Complexity Analysis, C.V. Goldman and S. Zilberstein. JAIR, 2004.
- Solving Concurrent Markov Decision Processes, Mausam and D. Weld. AAAI-04.
- Concurrent Probabilistic Planning in the Graphplan Framework, I. Little and S. Thiébaux. ICAPS-06.
- Concurrent Probabilistic Temporal Planning with Policy-Gradients, D. Aberdeen and O. Buffet. ICAPS-07.

Probabilistic Temporal Planning

PART V: Durative Actions w/ Concurrency

Mausam, David E. Smith, Sylvie Thiébaux

Stochastic Planning



Plan for Part V

- Concurrent Probabilistic Temporal Planning (CPTP)
- CoMDP Model
- Value-Based Algorithms
- AND-OR Search Formulation
- Policy Gradient Approach
- Related Models

Concurrent Probabilistic Temporal Planning

- **concurrency, time**
  - durative actions
  - timed effects
  - concurrency

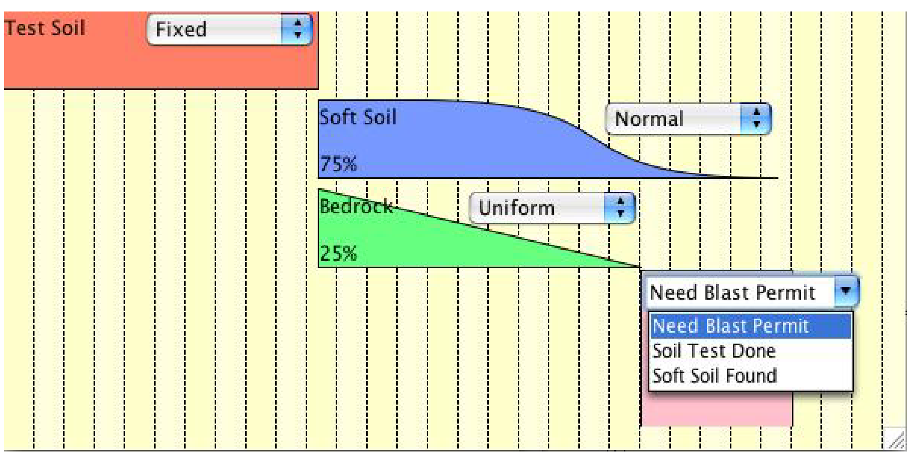
and

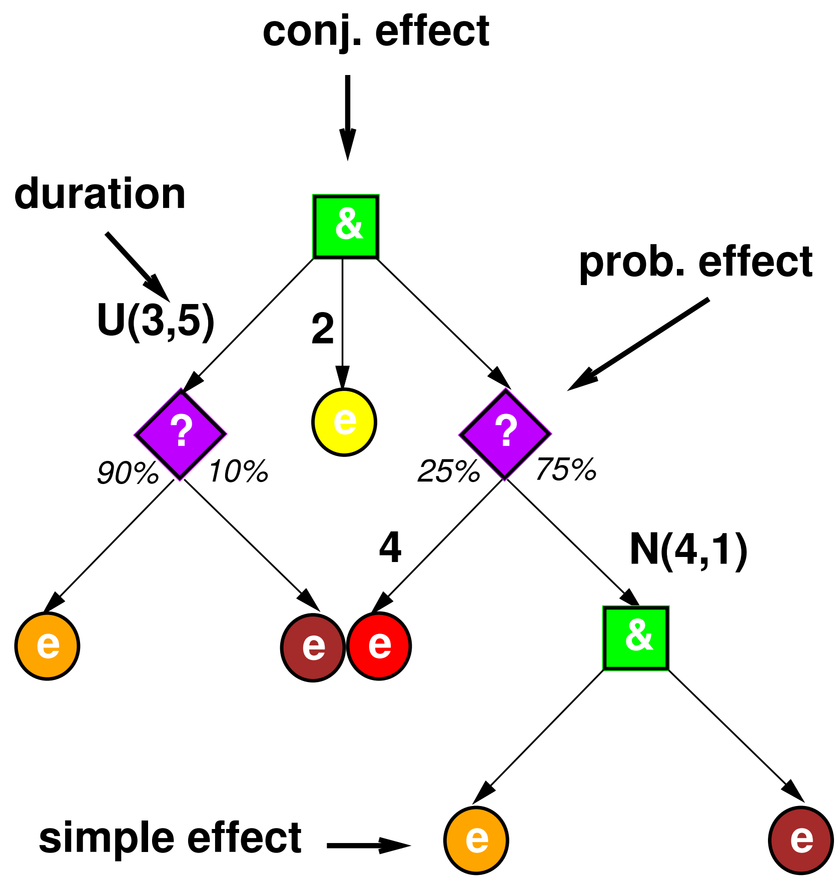
- **uncertainty**
  - about the effects
  - their timing
  - the action duration




Actions in CPTP

```
(:durative-action jump
      :parameters (?p - person ?c - parachute)
      :condition     (and (at start (and (alive ?p)
                                        (on ?p plane)
                                        (flying plane)
                                        (wearing ?p ?c)))
                          (over all (wearing ?p ?c)))
      :effect (and (at start (not (on ?p plane)))
                   (at end  (on ?p ground))
                   (at 5 (probabilistic
                          (0.8 (at 42 (standing ?p)))
                          (0.2 (at 13 (probabilistic
                                       (0.1 (at 14 (bruised ?p)))
                                       (0.9 (at 14 (not (alive ?p)))))))))))
```





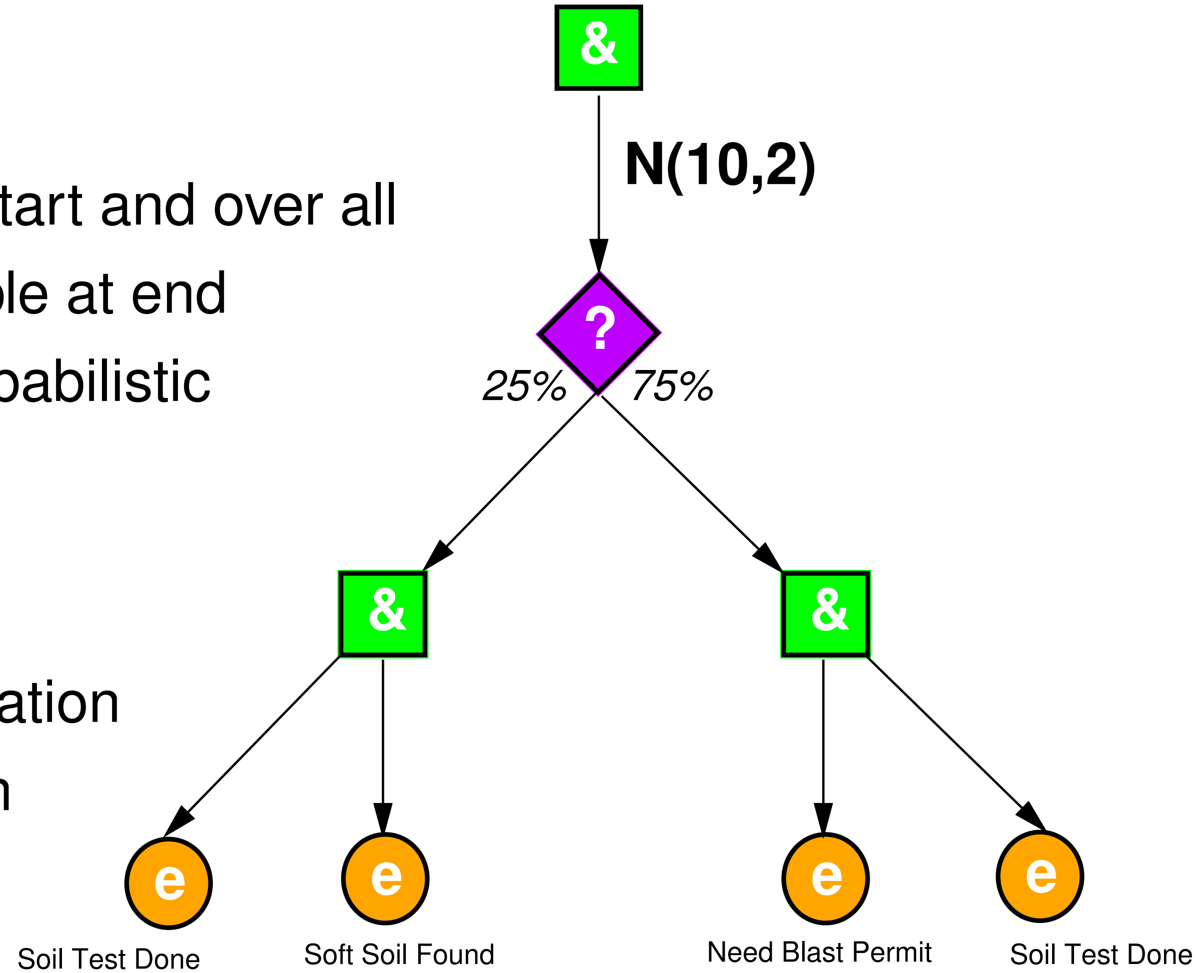
Actions in CPTP: The Simplest Case

TGP-style action:

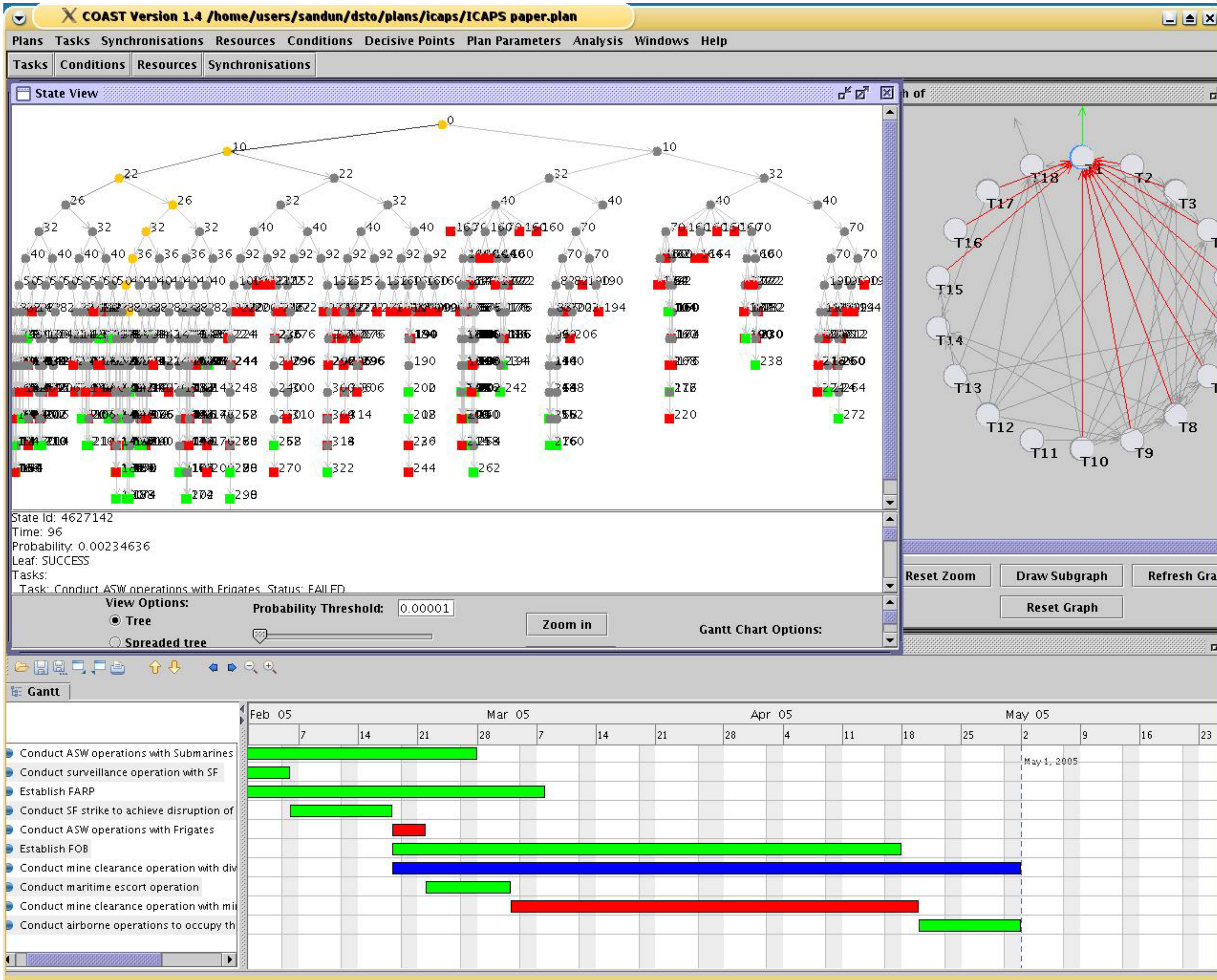
- preconditions hold at start and over all
- effects are only available at end
- duration is fixed or probabilistic

Additionally:

- effect-independent duration
- monotonic continuation (normal, uniform, exp.)



Plans in CPTP



Decision Points in CPTP

Definitions

Pivot: Time point at which an event might take place (effect, condition being needed).
Happening: Time point at which an event actually takes place.

Completeness/Optimality Results [Mausam & Weld, AAAI-06]

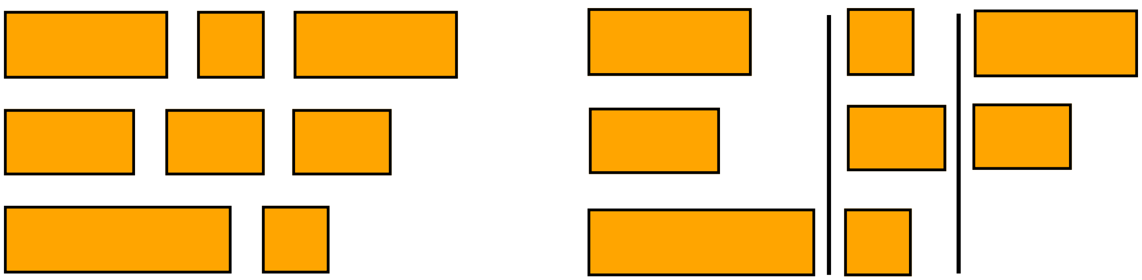
1. *With TGP actions, decision points may be restricted to pivots.*
2. *With TGP actions and deterministic durations, decision points may be restricted to happenings.*
3. Conjecture: idem with effect-independent durations and monotonic continuations.
4. *In general, restriction to pivots may cause incompleteness.*

Plan for Part V

- Concurrent Probabilistic Temporal Planning (CPTP)
- CoMDP Model
- Value-Based Algorithms
- AND-OR Search Formulation
- Policy Gradient Approach
- Related Models

CoMDP in Interwoven Epoch State Space

Why Interwoven?

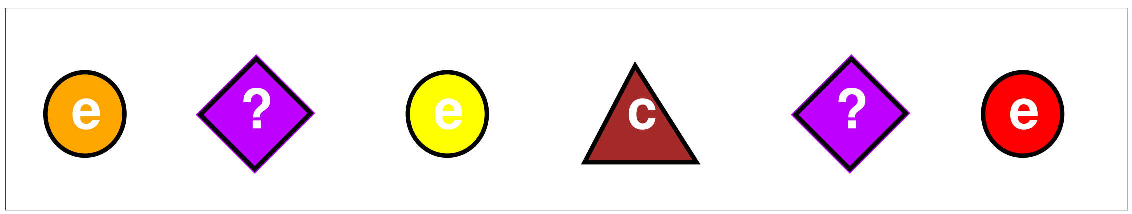


interwoven epochs aligned epochs

The traditional aligned CoMDP model is suboptimal for CPTP

CoMDP in Interwoven Epoch State Space

- **CoMDP state** contains:
  - current world state *w*
  - event queue *q*, records advancement of executing actions
  - inspired from SAPA, TLPlan, HSP, etc
- **Event queue** contains pairs:
  - event *e* (simple effect, prob effect, condition check . . .)
  - distribution for the duration remaining until *e* happens



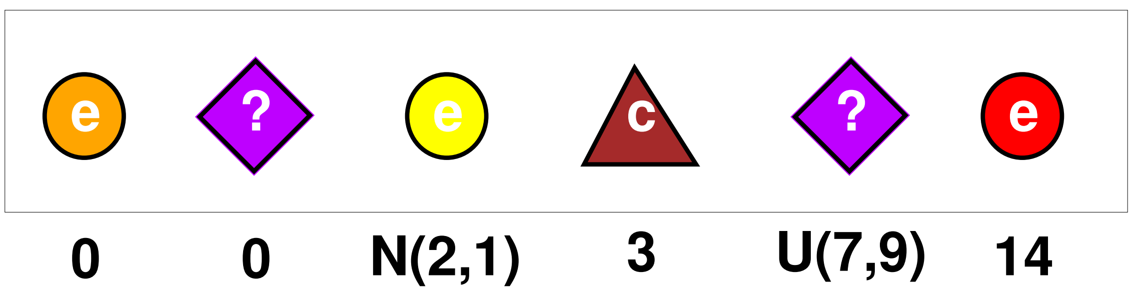
0 0 N(2,1) 3 U(7,9) 14

- **Queue for TGP actions with fixed durations:**
  $q = \{\langle a, \delta \rangle \mid a \text{ is executing and will terminate in } \delta \text{ time units}\}$

CoMDP in Interwoven Epoch State Space

- $\mathcal{A}(s)$: as in standard CoMDP, but includes the empty set (wait). Need to check interference with executing actions in the queue.

- $\mathcal{P}r$: tedious to formalise (even for restricted cases), see [Mausam & Weld, JAIR-07]. Considers all possible states at all pivots between the min. time an event could happen and the max. time one is guaranteed to happen. $\rightarrow$ motivates sampling!



- $\mathcal{C}(s, A, s')$: time elapsed between s and s'.

Plan for Part V

- Concurrent Probabilistic Temporal Planning (CPTP)
- CoMDP Model
- Value-Based Algorithms
- AND-OR Search Formulation
- Policy Gradient Approach
- Related Models

Value-Based Algorithms

- DUR family of planners [Mausam & Weld ICAPS-05, JAIR-07]
- assumptions (to start with):
  - TGP actions with fixed integer durations
  - $\Rightarrow$ decision points are happenings
  - $\Rightarrow$ event queue records remaining duration for each action
- sampled RTDP applies
- to cope with interwoven state space blow-up:

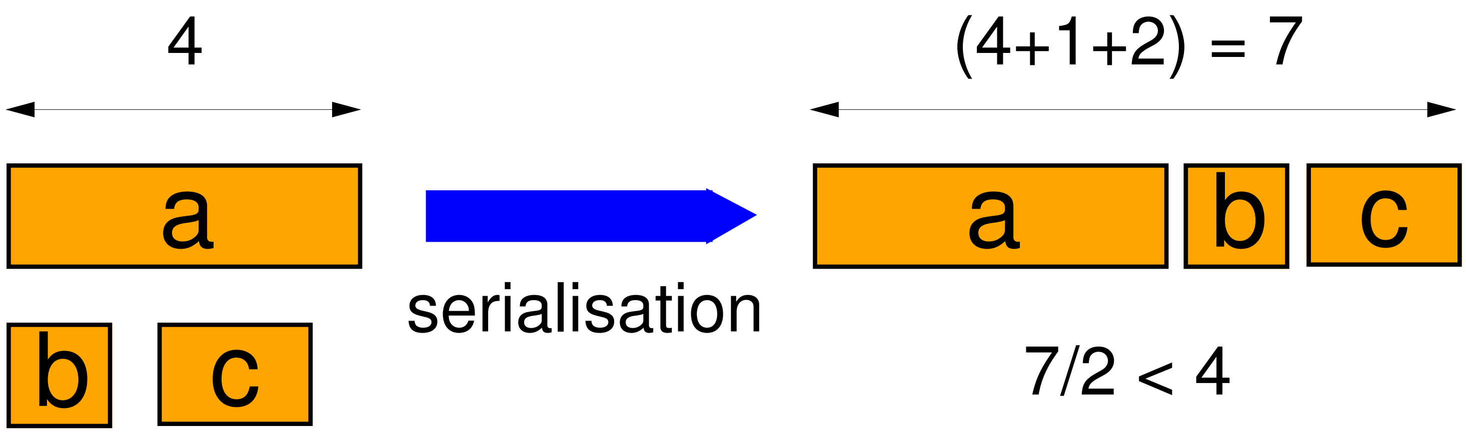
1. heuristics
2. hybridisation

Maximum Concurrency Heuristic

- divide the optimal serial MDP cost by
- max nb. actions executable concurrently in the domain

$$J^{*}_{\underline{-}}(\langle s, \emptyset \rangle) \geq \frac{J^{*}(s)}{m}$$

$$J^{*}_{\underline{-}}(\langle s, q \rangle) \geq \frac{Q^{*}(s, A_q)}{m}$$



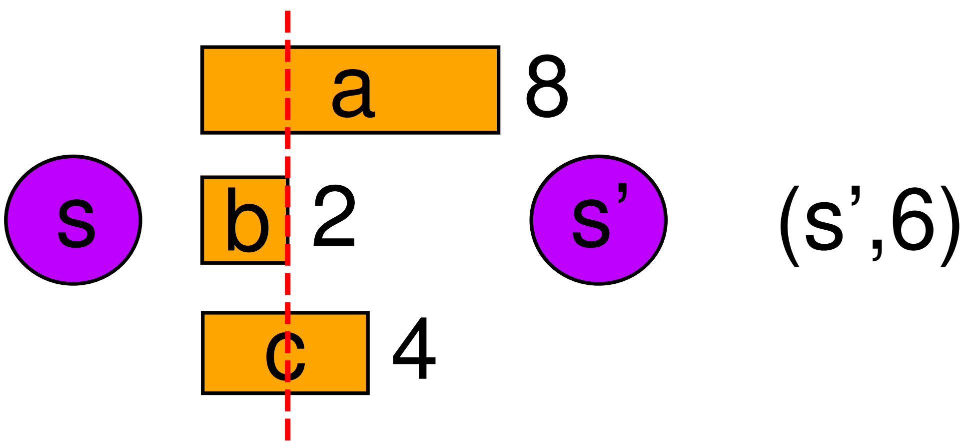
Eager Effects Heuristic

- effects realised when the fastest started actions ends
- time advances accordingly
- CoMDP state:

  ⟨ world state after effects, duration until last executing action ends ⟩
- relaxed problem:

– get information about effects ahead of time

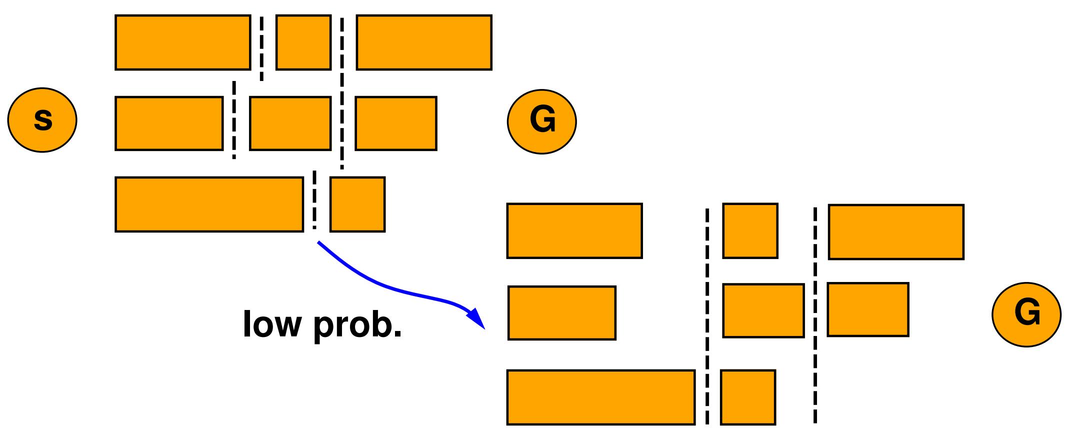
– mutex action combinations are allowed (lost track of time)



Hybridisation

Hybrid interwoven/aligned policy for probable/unprobable states

1. run RTDP interwoven for a number of trials
   $\longrightarrow$ yields lower bound $L = J(s_0)$
2. run RTDP aligned on low frequency states
3. clean up and evaluate hybrid policy π
   $\longrightarrow$ yields upper bound $u = J_\pi(s_0)$
4. repeat until performance ratio r reached ($\frac{(U-L)}{L} < r$)



Extensions of the DUR Planner

ΔDUR [Mausam & Weld, AAAI-06, JAIR-07] extends DUR to TGP actions with stochastic durations.

- **MC and hybrid:** apply with minor variations.

- **ΔDUR$_{exp}$, expected duration planner:**
  - effect-independent durations & monotonic continuations
  - assigns an action its (fixed) mean duration
  - use DUR to generate policy and execute:
  - if action terminates early, extend policy from current state
  - if action is late to terminate, update mean, then extend.

- **ΔDUR$_{arch}$, archetypal duration planner:**
  - extends ΔDUR$_{exp}$ to multimodal distributions
  - probabilistic outcomes with different mean durations

Plan for Part V

- Concurrent Probabilistic Temporal Planning (CPTP)
- CoMDP Model
- Value-Based Algorithms
- AND-OR Search Formulation
- Policy Gradient Approach
- Related Models

AND/OR Search Formulation

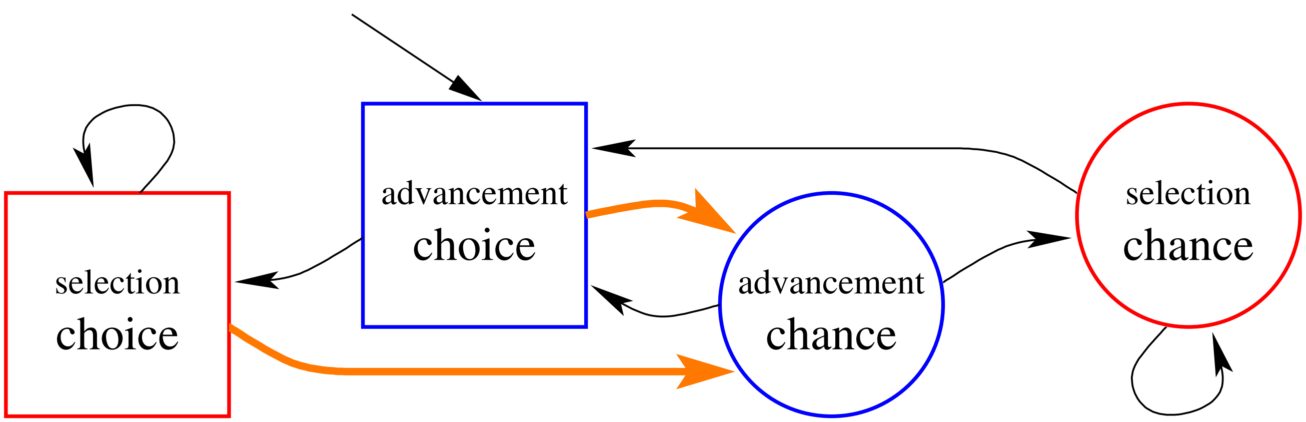
Prottle [Little *et. al* AAAI-05]

- forward search planner, solves CPTP over finite horizon
- not extremely different from DUR:
  - finer characterisation of the search space for CPTP
  - slightly different search algorithm (lower + upper bound)
  - planning graph heuristics
- current implementation:
  - handles general CPTP actions with fixed durations on arcs
  - incomplete: only considers pivots
  - takes cost to be the probability of failure

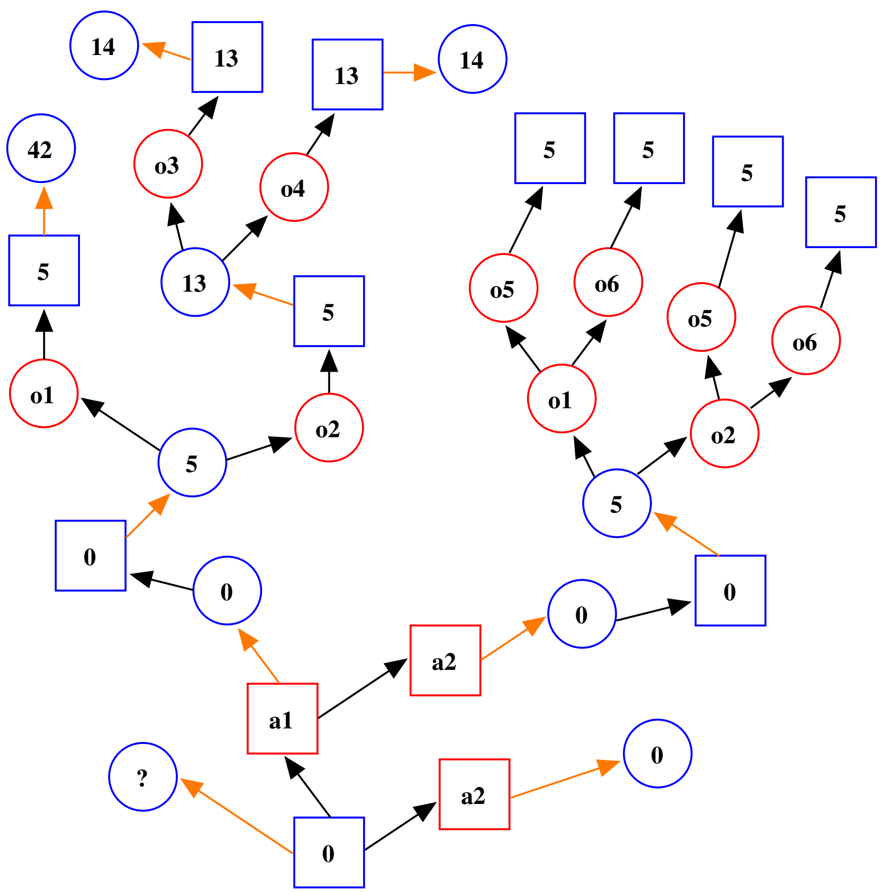
Prottle's Search Space

Interwoven epochs and-or graph

- **and-or graph:** and = chance, or = choice
- **node purposes:** action selection or time advancement
- **node contains:** current state, current time, event queue

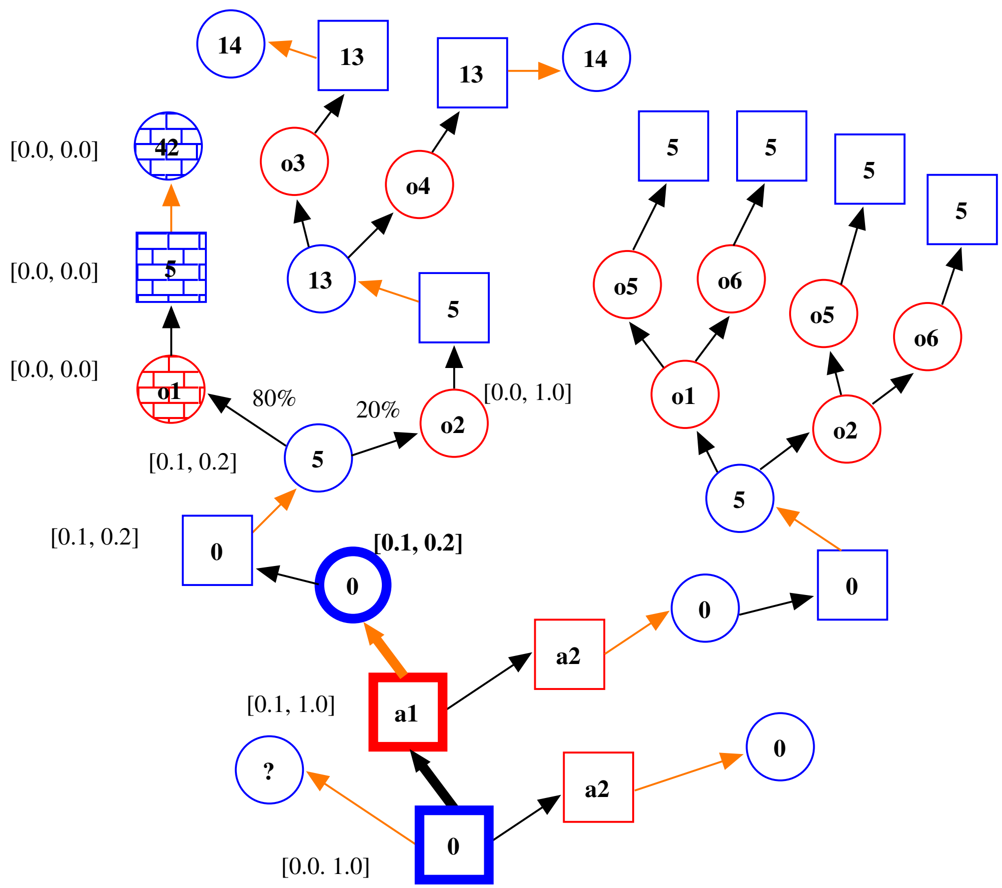


Prottle's Search Space



Prottle's Algorithm

Trial based with lower and upper bound (BRTDP and FRTDP are similar). Selection strategy quickly gets a likely path to the goal and robustifies known paths thereafter.



Prottle's Algorithm (details)

- **node lower/upper cost bounds**
  - cost = probability of failure
  - bounds initialised using heuristics

bound update rules

$$
\begin{aligned}
L_{\textbf{choice}}(n) &:= \max(L(n), \min_{n' \in S(n)} L(n')) \\
U_{\textbf{choice}}(n) &:= \min(U(n), \min_{n' \in S(n)} U(n')) \\
L_{\textbf{chance}}(n) &:= \max(L(n), \textstyle\sum_{n' \in S(n)} \Pr(n')\, L(n')) \\
U_{\textbf{chance}}(n) &:= \min(U(n), \textstyle\sum_{n' \in S(n)} \Pr(n')\, U(n'))
\end{aligned}
$$

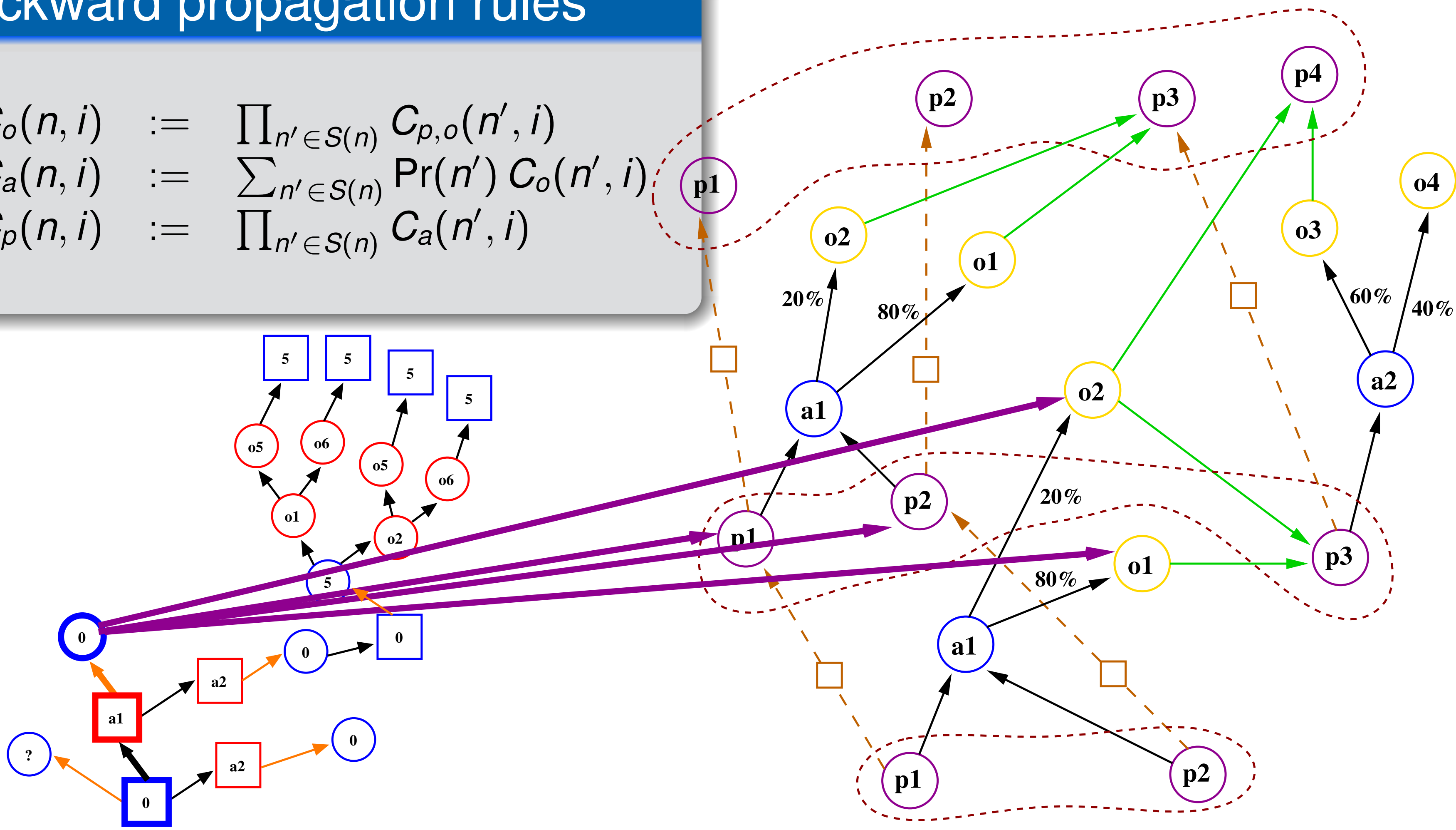
  - cost converges when $U(n) - L(n) \leq \epsilon$
- **node labels:** solved, failure (solved with cost 1), unsolved
- **node selection:** minimises $P(n)U(n)$, uses $P(n)L(n)$ to break ties

Prottle's Heuristic

Based on a probabilistic temporal planning graph

backward propagation rules

$$
\begin{aligned}
C_o(n,i) &:= \prod_{n' \in S(n)} C_{p,o}(n',i) \\
C_a(n,i) &:= \sum_{n' \in S(n)} \Pr(n')\, C_o(n',i) \\
C_p(n,i) &:= \prod_{n' \in S(n)} C_a(n',i)
\end{aligned}
$$

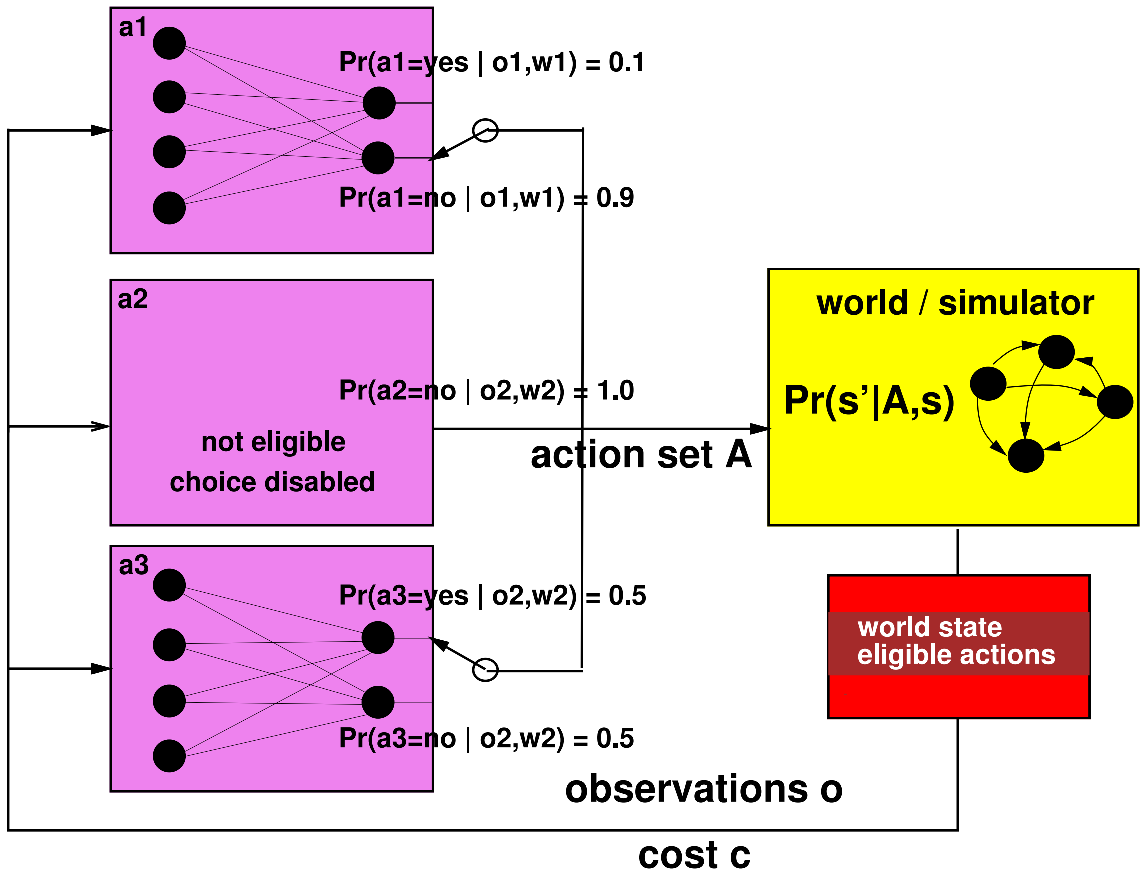


Plan for Part V

- Concurrent Probabilistic Temporal Planning (CPTP)
- CoMDP Model
- Value-Based Algorithms
- AND-OR Search Formulation
- Policy Gradient Approach
- Related Models

Policy Gradient Approach

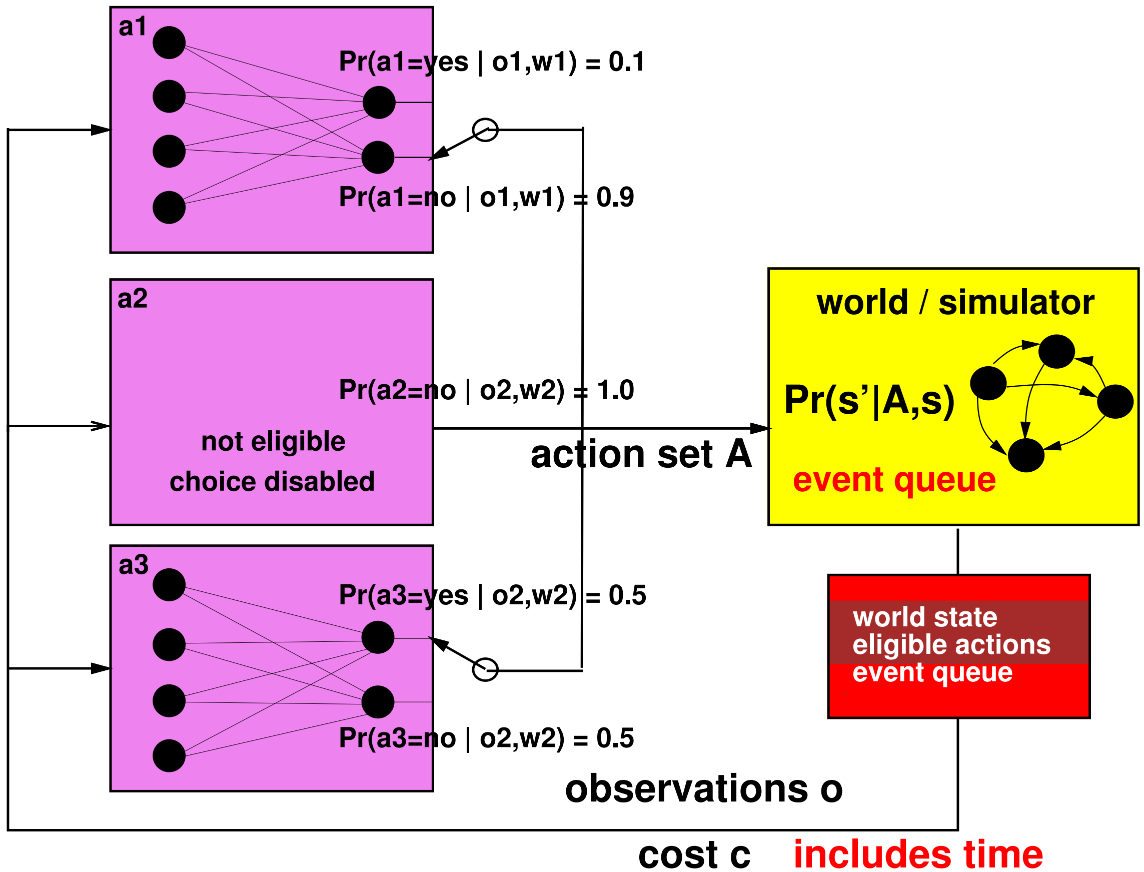
Minimises the expected cost of a factored parameterised policy by factored gradient descent in the parameters space.



Factored Policy Gradient for CPTP

FPG handles continuous time dist. [Aberdeen & Buffet ICAPS-07].

1. simulator manages an event queue
2. cost function takes durations into account



Plan for Part V

- Concurrent Probabilistic Temporal Planning (CPTP)
- CoMDP Model
- Value-Based Algorithms
- AND-OR Search Formulation
- Policy Gradient Approach
- Related Models

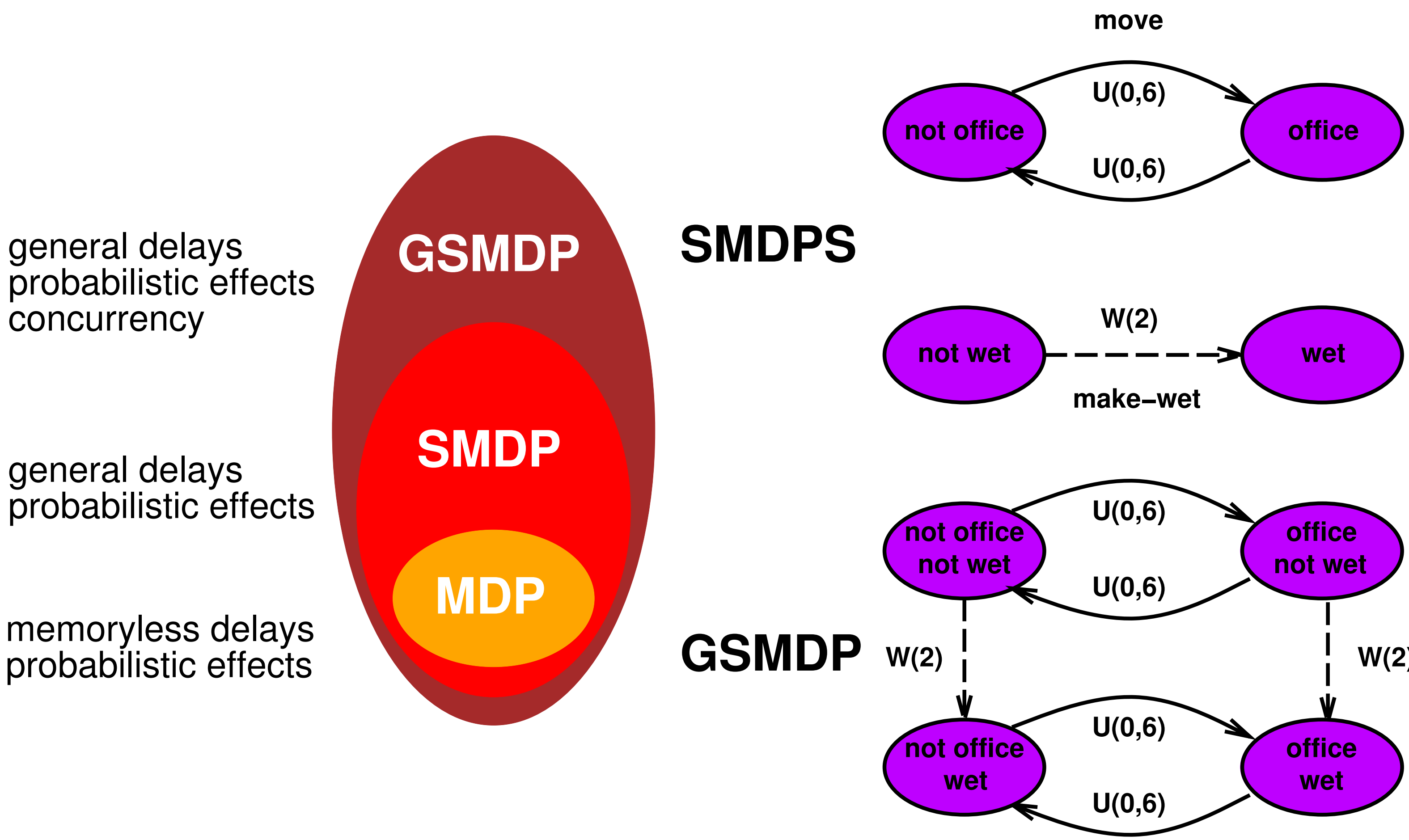
Related Models

Generalised Semi-MDP (GSMDP)[Younes & Simmons, AAAI-04]

- set of states S
- set of events E; each event e is associated with:
  - $\Phi_e(s)$: enabling condition
  - $G_e(t)$: probability that e remains enabled before it triggers
  - $\Pr(s' \mid e, s)$ transition probability when e triggers in s
- actions $A \subseteq E$ are controllable events
- rewards:
  - lump sum reward $k(s, e, s')$ for transitions
  - continuous reward rate $c(a, s)$ for $a \in A$ being enabled in s
  - disc. inf. horz. model; reward at time t counts as $e^{-\alpha t}$
- policy: maps timed histories to set of enabled actions

Generalised Semi-Markov Decision Process

Parallel (asynchronous) composition of SMDPs is a GSMDP: distribution of an enabled event may depend on history.



Generalised Semi-Markov Decision Process

UNIVERSITY OF WASHINGTON

NASA

NICTA

Specificities:

- synchronous systems
- discrete/continuous time

Solution methods:

- approximate distributions with phase-type distributions and solve the resulting MDP [younes & simmons AAAI-04]

$\longrightarrow$ to know more: attend Hakan's Dissertation Award talk!

- incremental generate - test (statistical sampling) - debug [younes & simmons ICAPS-04]

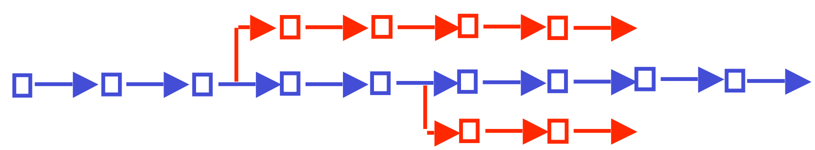
$\longrightarrow$ covered by David

References

- Policy Generation for Continuous Time Domains with Concurrency, H. Younes and R. Simmons. ICAPS-04.
- Solving Generalized Semi-Markov Processes using Continuous Phase-Type Distributions, H. Younes and R. Simmons. AAAI-04.
- Prottle: A Probabilistic Temporal Planner, I. Little, D. Aberdeen, and S. Thiébaux. AAAI-05.
- Concurrent Probabilistic Temporal Planning, Mausam and D. Weld. ICAPS-05.
- Probabilistic Temporal Planning with Uncertain Durations. Mausam and D. Weld. AAAI-06
- Concurrent Probabilistic Temporal Planning with Policy-Gradients, D. Aberdeen and O. Buffet. ICAPS-07.
- Planning with Durative Actions in Uncertain Domains, Mausam and D. Weld. JAIR, to appear, 2007.

Probabilistic Temporal Planning

PART 6: Practical Considerations



Outline

Incremental approaches

When is contingency planning really needed ?

Combining contingency planning & replanning

Applications

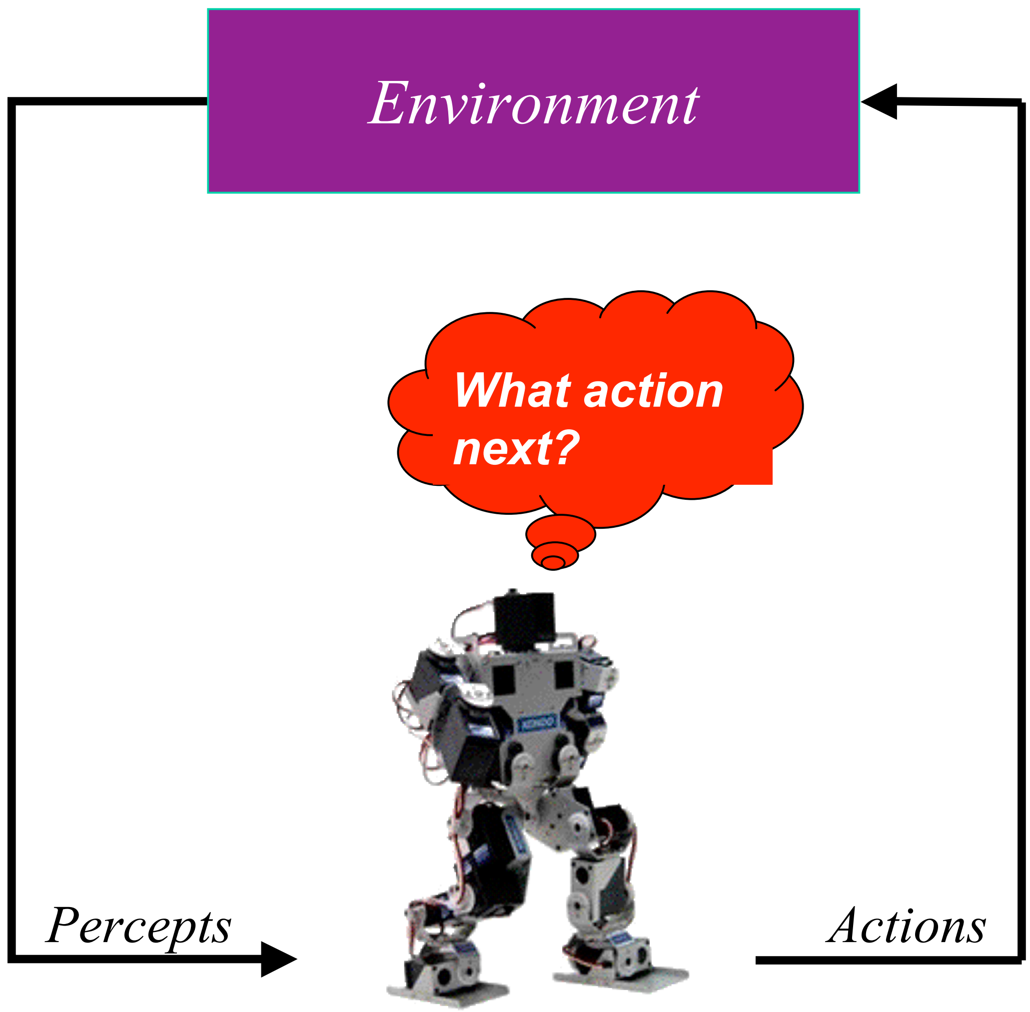
Problem Dimensions

Static vs. Dynamic

Predictable vs. Unpredictable

Fully vs. Partially Observable

Perfect vs. Noisy



Discrete vs. Continuous Outcomes

Deterministic vs. **Stochastic**

Sequential vs. **Concurrent**

Instantaneous vs. **Durative**

Full vs. Partial satisfaction

Problem Dimensions

Static vs. Dynamic
Predictable vs. Unpredictable

Environment

Fully
vs.
Partially
Observable

Perfect
vs.
Noisy

Discrete
vs.
Continuous Outcomes

Deterministic
vs.
Stochastic

Sequential
vs.
Concurrent

Instantaneous
vs.
Durative

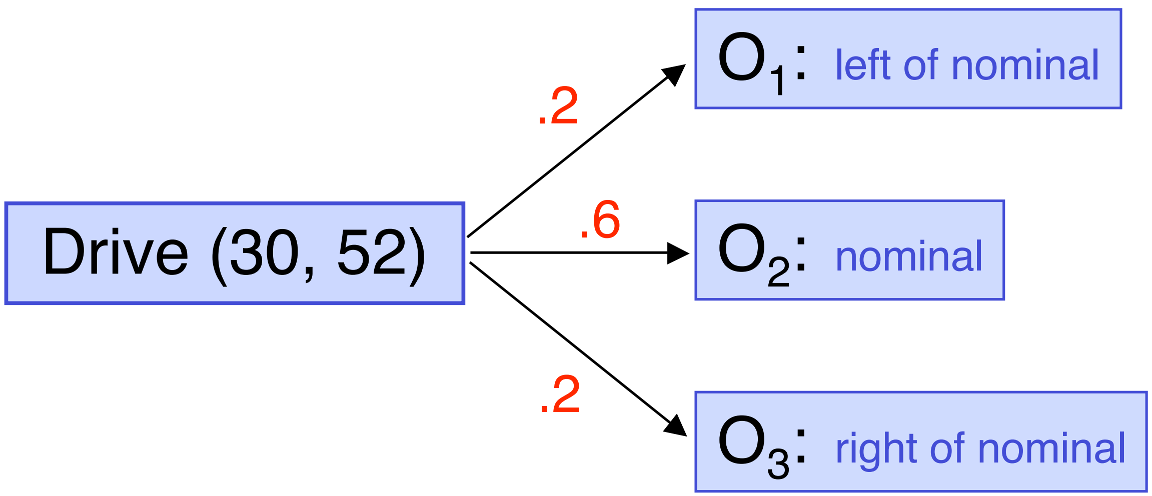
What action next?

Percepts

Actions

Full vs. **Partial satisfaction**

Can We Make it Discrete?



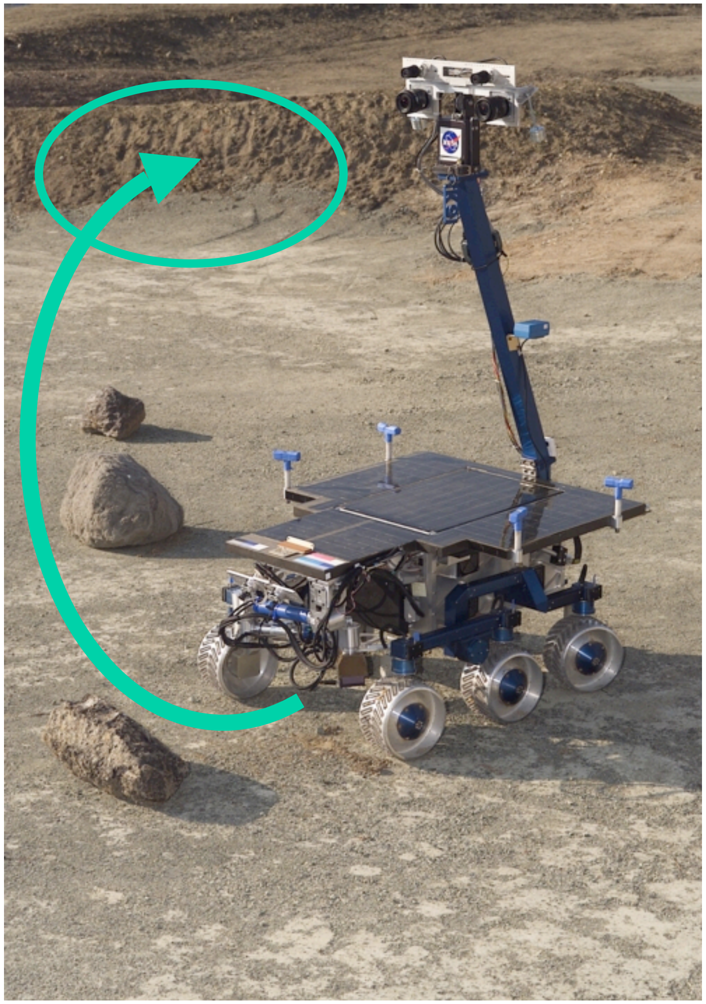
What does "nominal" mean?

Collect

Drive (30, 52)

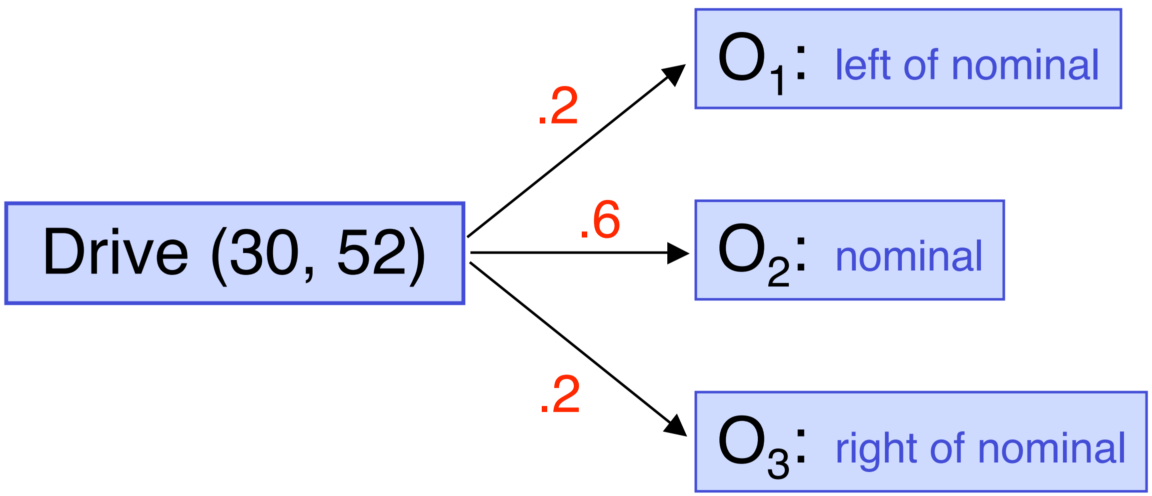
What does “nominal” mean?

Picture



Drive (30, 52)

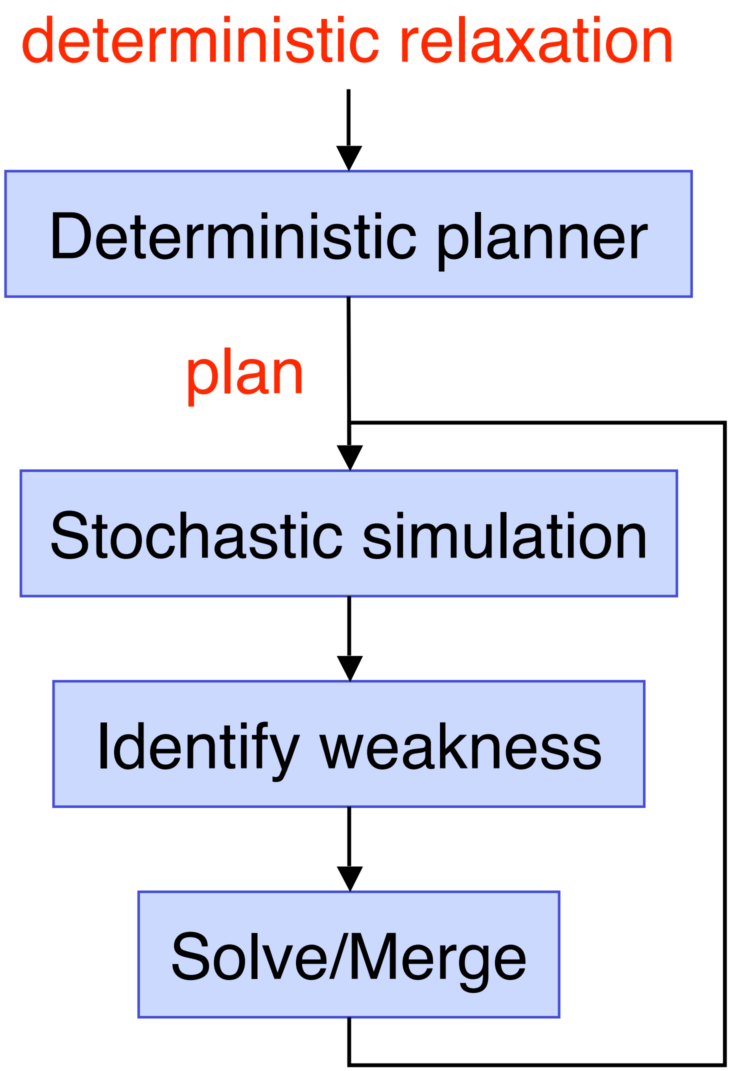
Depends on Objective



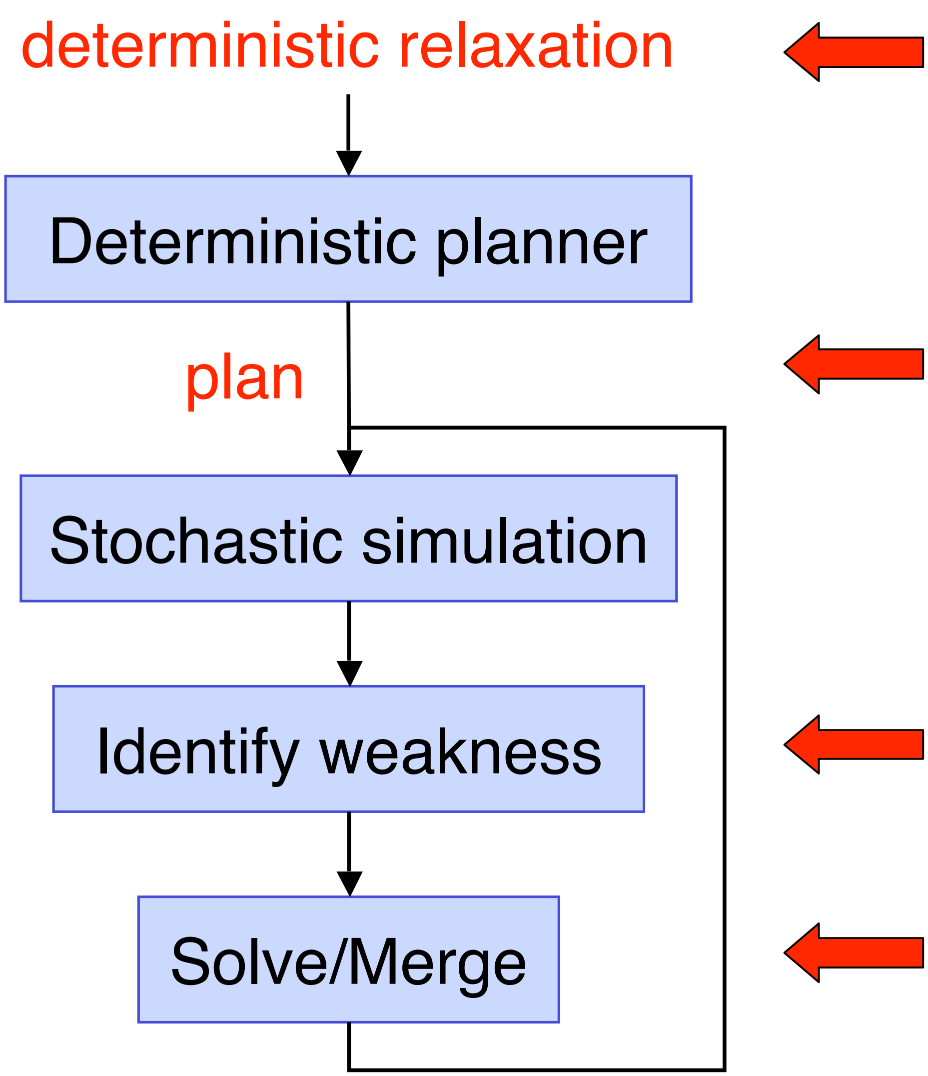
Outline

- Incremental approaches
  - JIC
  - ICP
  - Tempastic
- When is contingency planning needed ?
- Combining contingency planning & replanning
- Applications

Incremental Approaches



Differences



Differences, Differences, Differences, Differences

JIC

ICP

Tempastic

Opportunistic (Long/Fox)

TCP (Foss/Onder)

Outline

Incremental approaches

- JIC
- ICP
- Tempastic

When is contingency planning needed ?

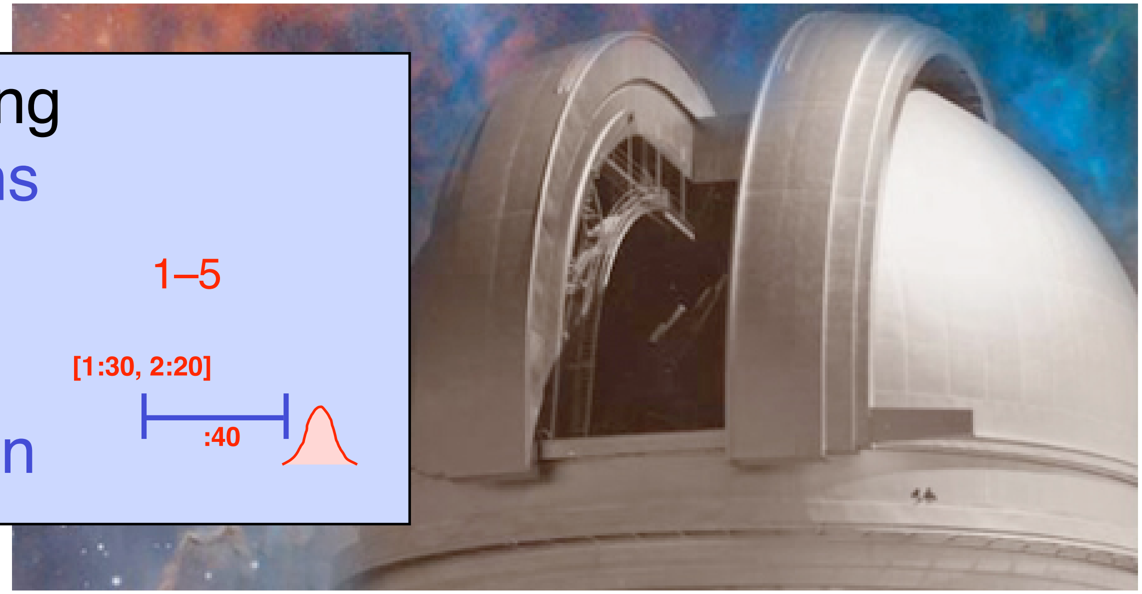
Combining contingency planning & replanning

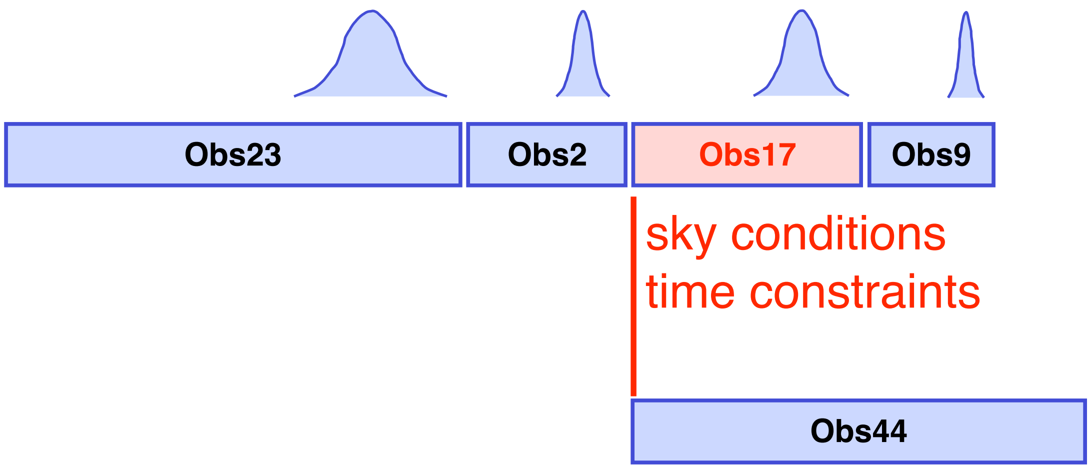
Applications

Just in Case (JIC) Scheduling

Observation Scheduling

- many observations
- priority 1–5
- time window [1:30, 2:20]
- stochastic duration :40

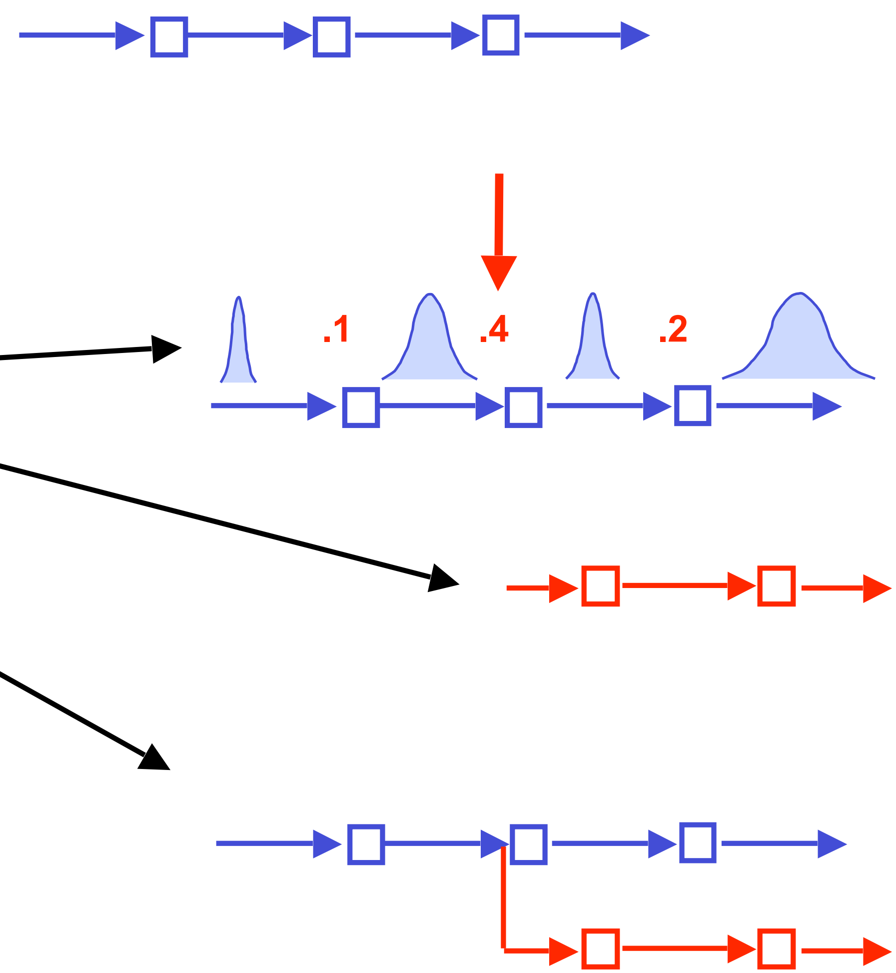




Ref: Drummond, Bresina, & Swanson, AAAI-94

The JIC Algorithm

1. Seed schedule
2. Identify most likely failure
3. Generate a contingency branch
4. Incorporate the branch



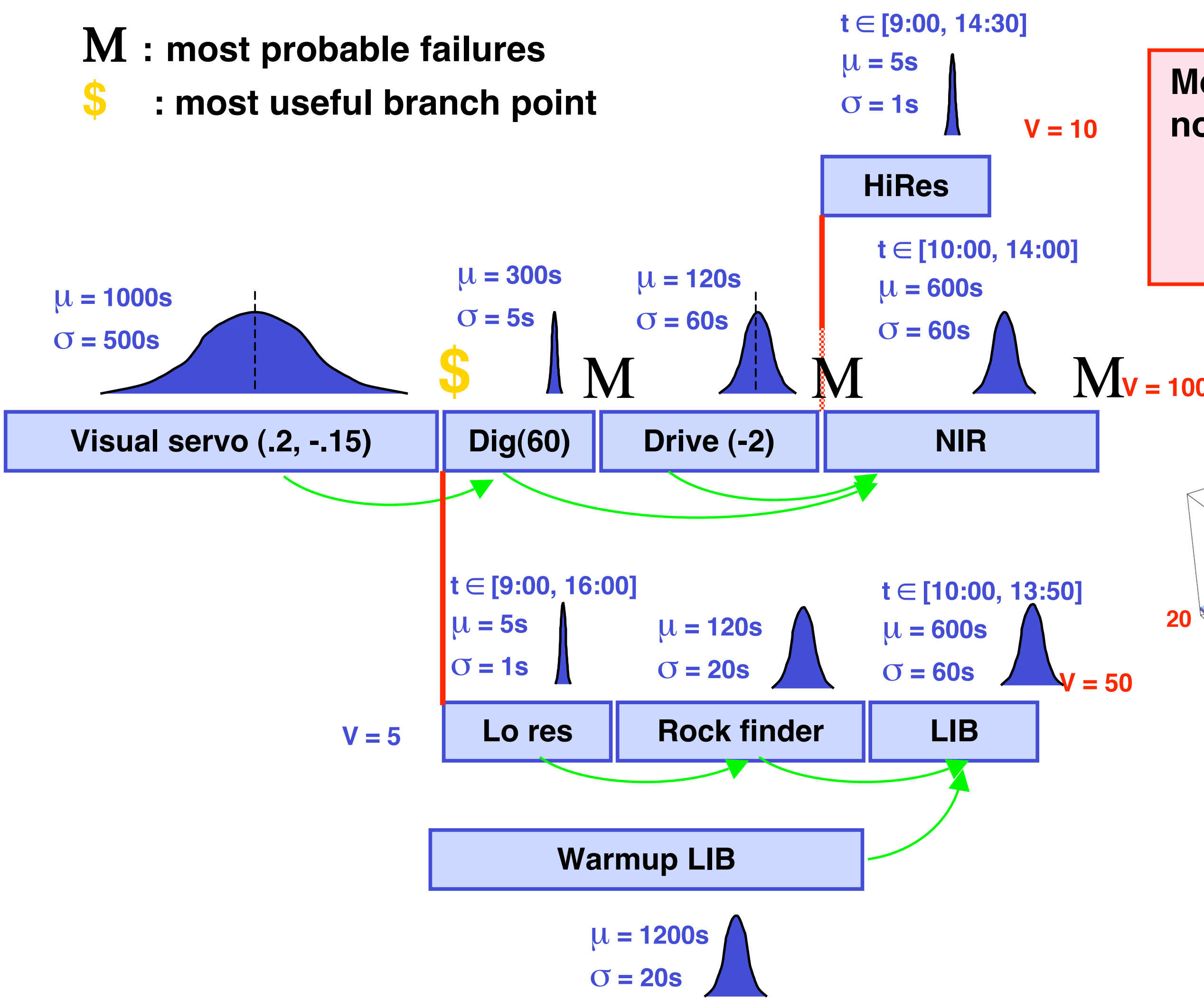
Advantages:
- **Tractability**
- **Simple schedules**
- **Anytime**

Ref: Drummond, Bresina, & Swanson, AAAI-94

Limits of JIC Heuristic

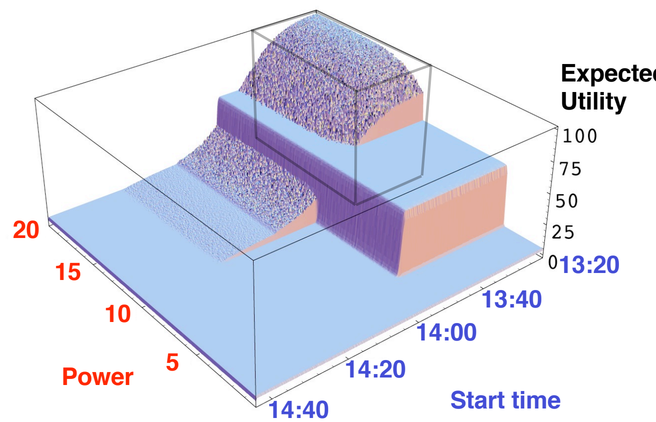
M : most probable failures

$: most useful branch point



Most probable failure points may not be the best branch-points:

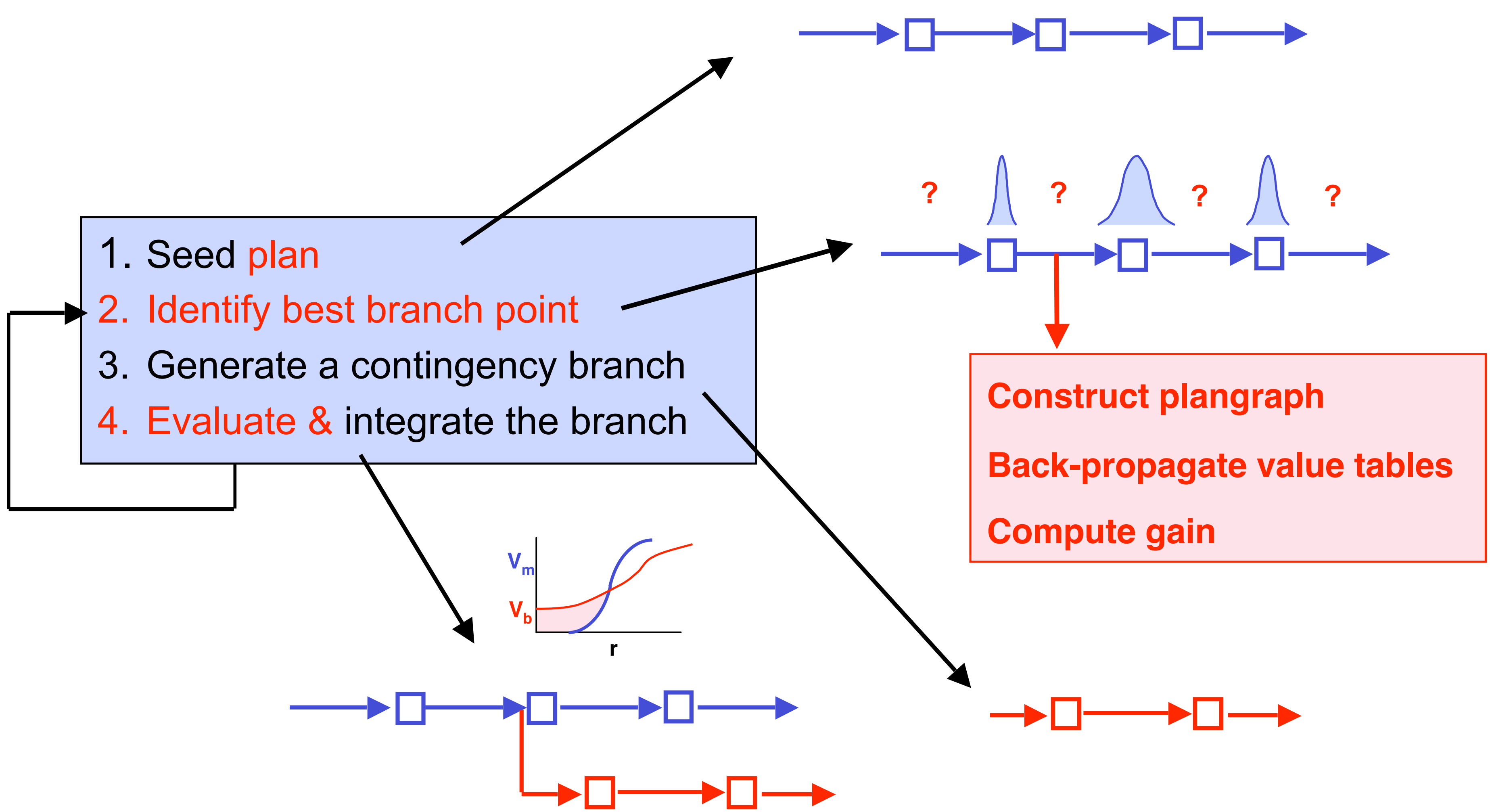
It is often too late to attempt other goals when the plan is about to fail.



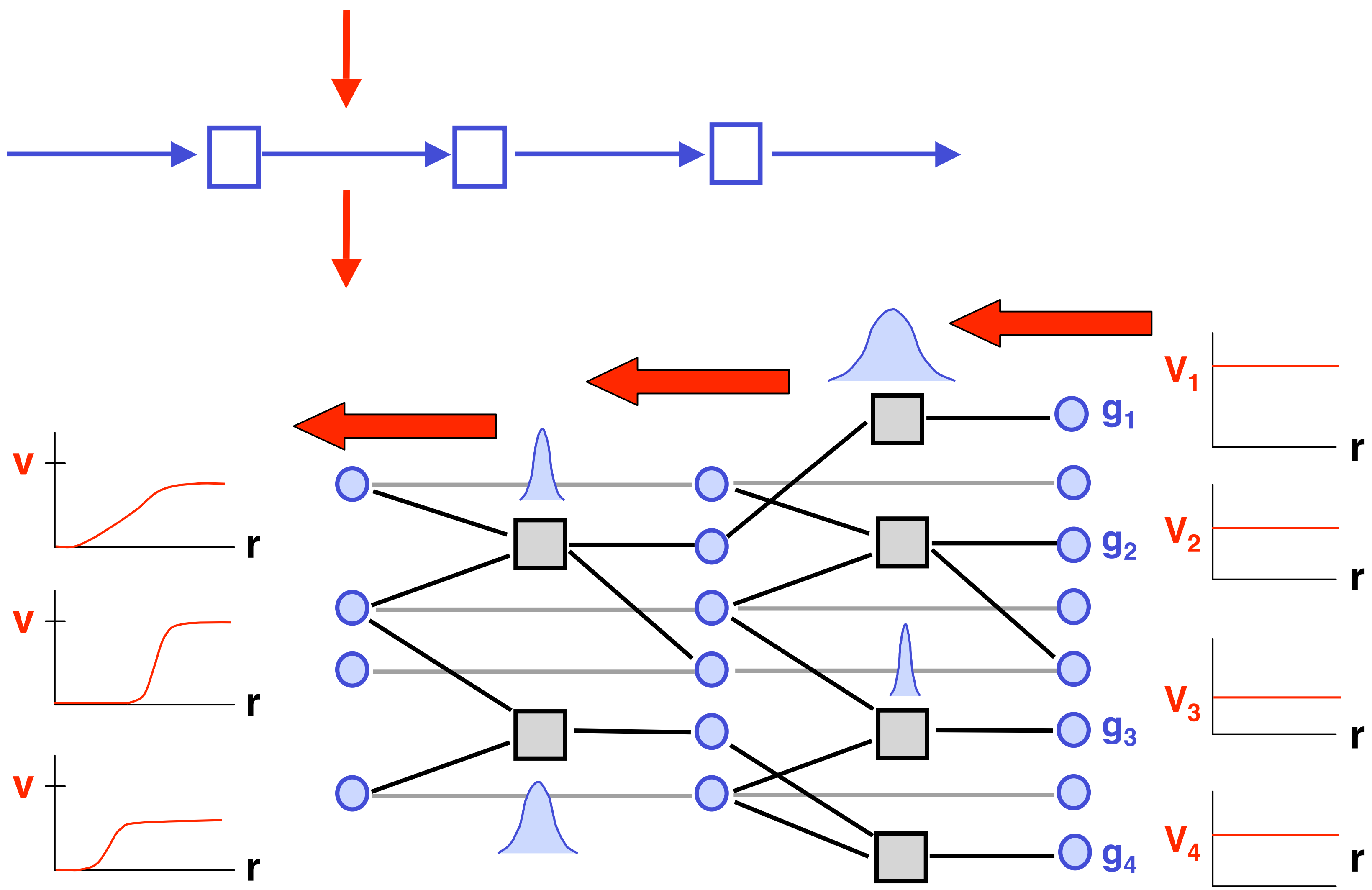
Outline

- Incremental approaches
  - JIC
  - ICP
  - Tempastic
- When is contingency planning needed ?
- Combining contingency planning & replanning
- Applications

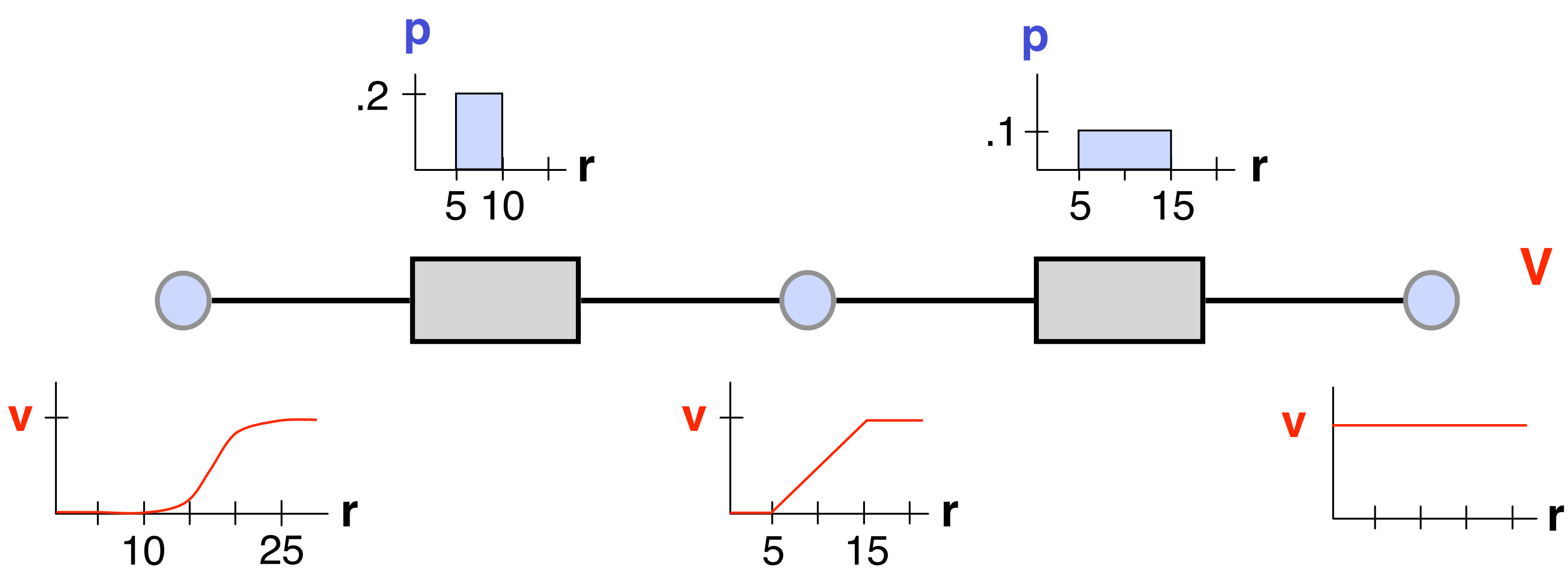
Incremental Contingency Planning



Back-Propagate Value Tables

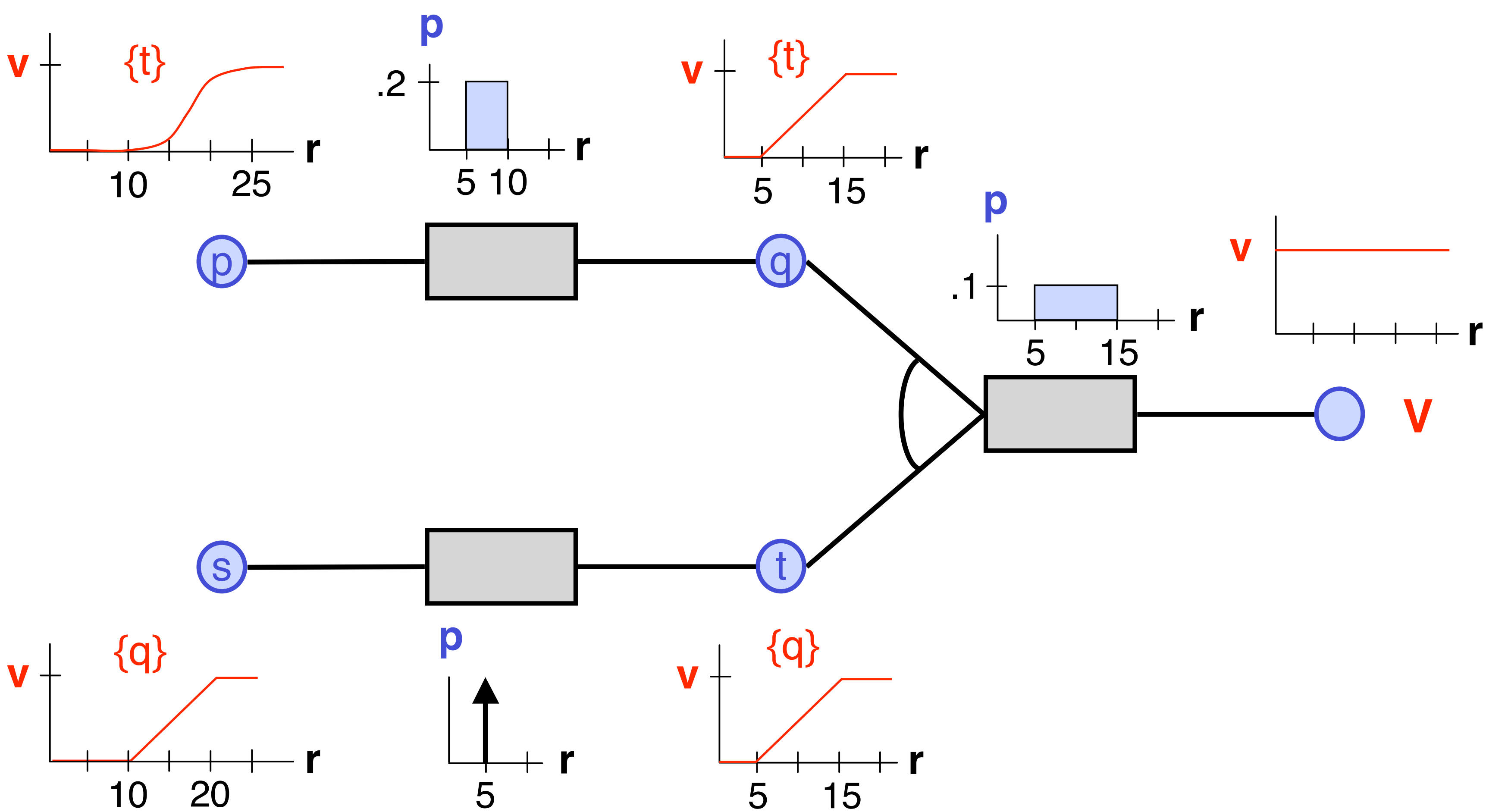


Simple Back-Propagation

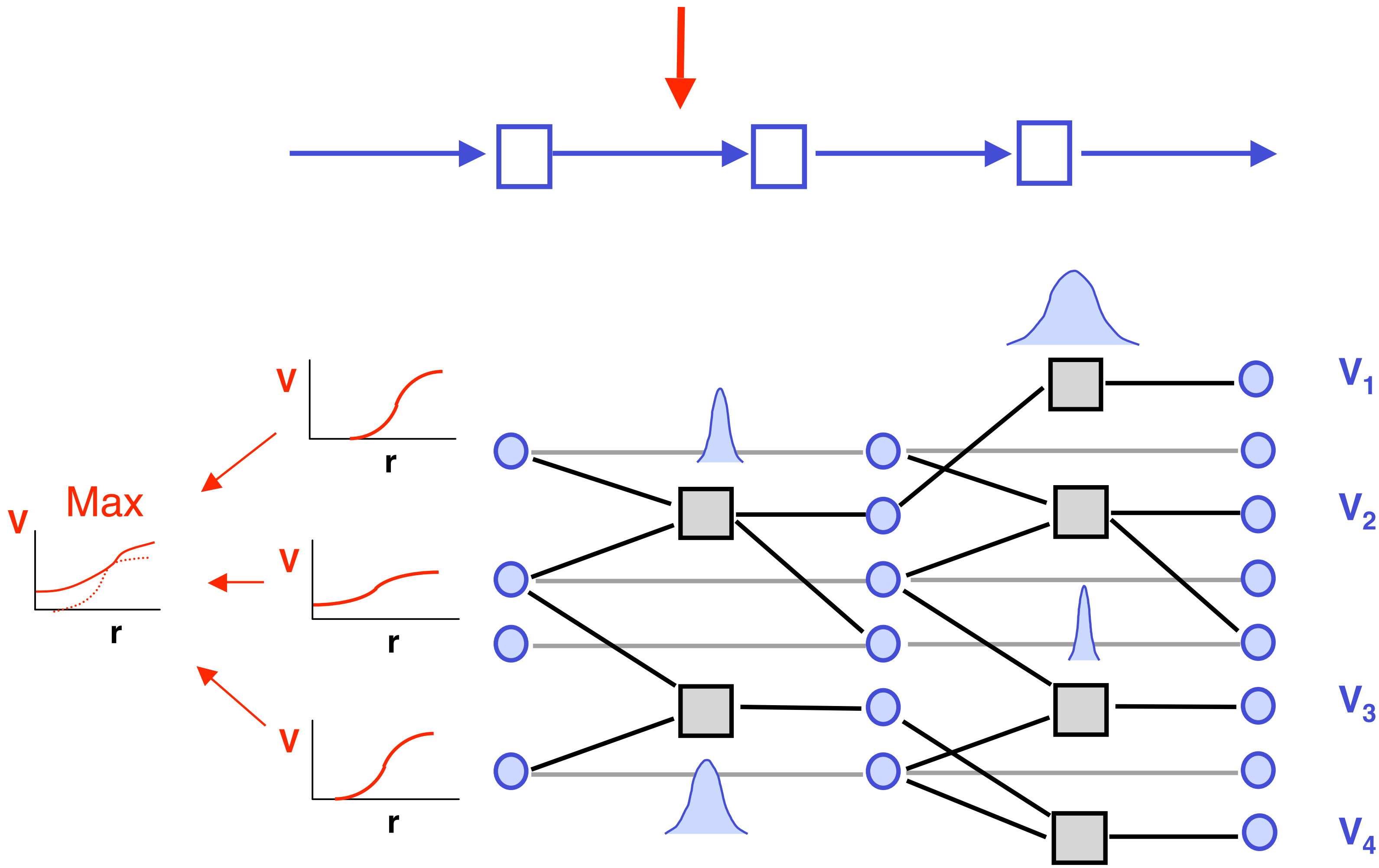


$$V(r') = \int_0^{\infty} P_c(r)\, V(r'-r)\, dr$$

Conjunctions

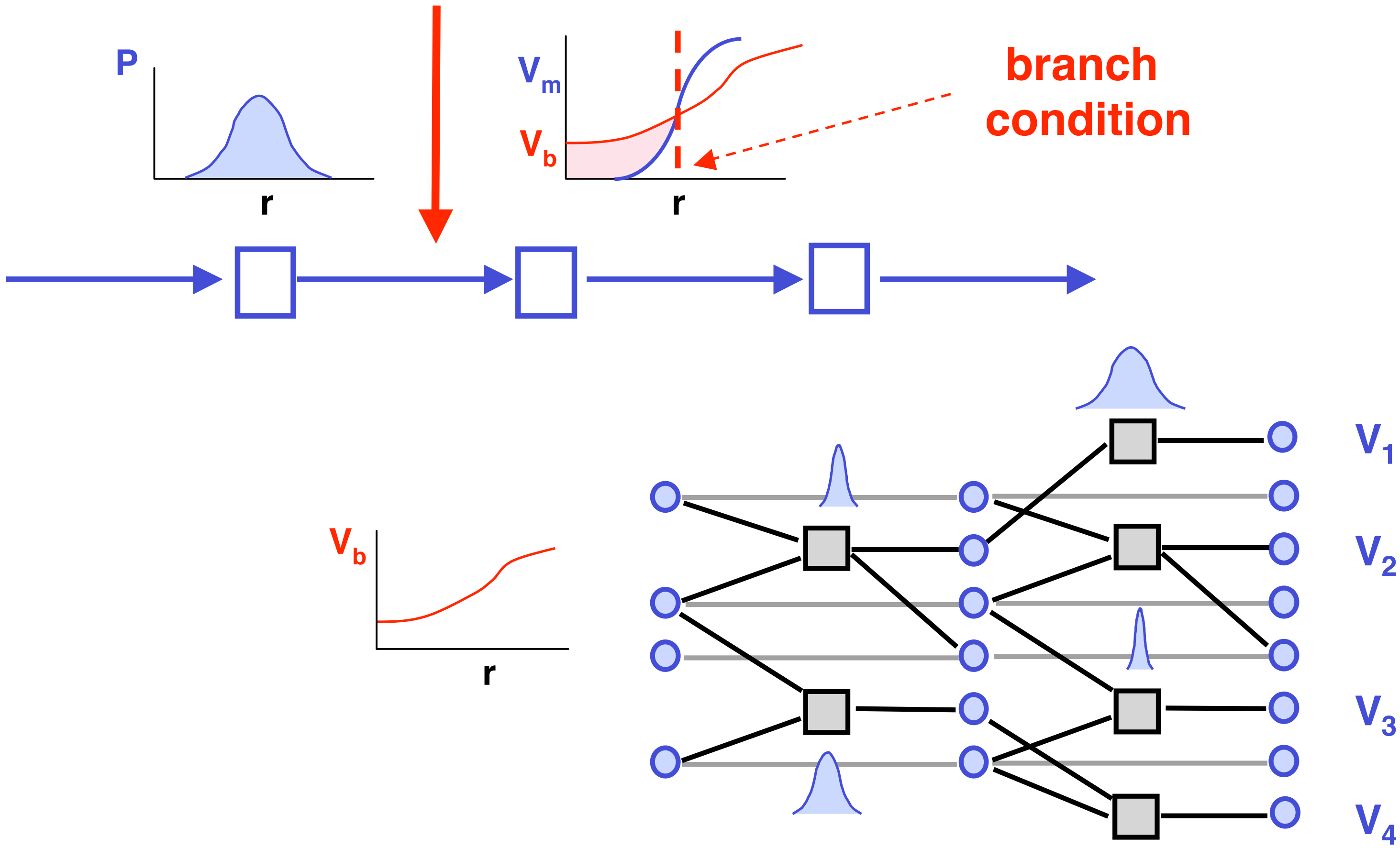


Estimating Branch Value



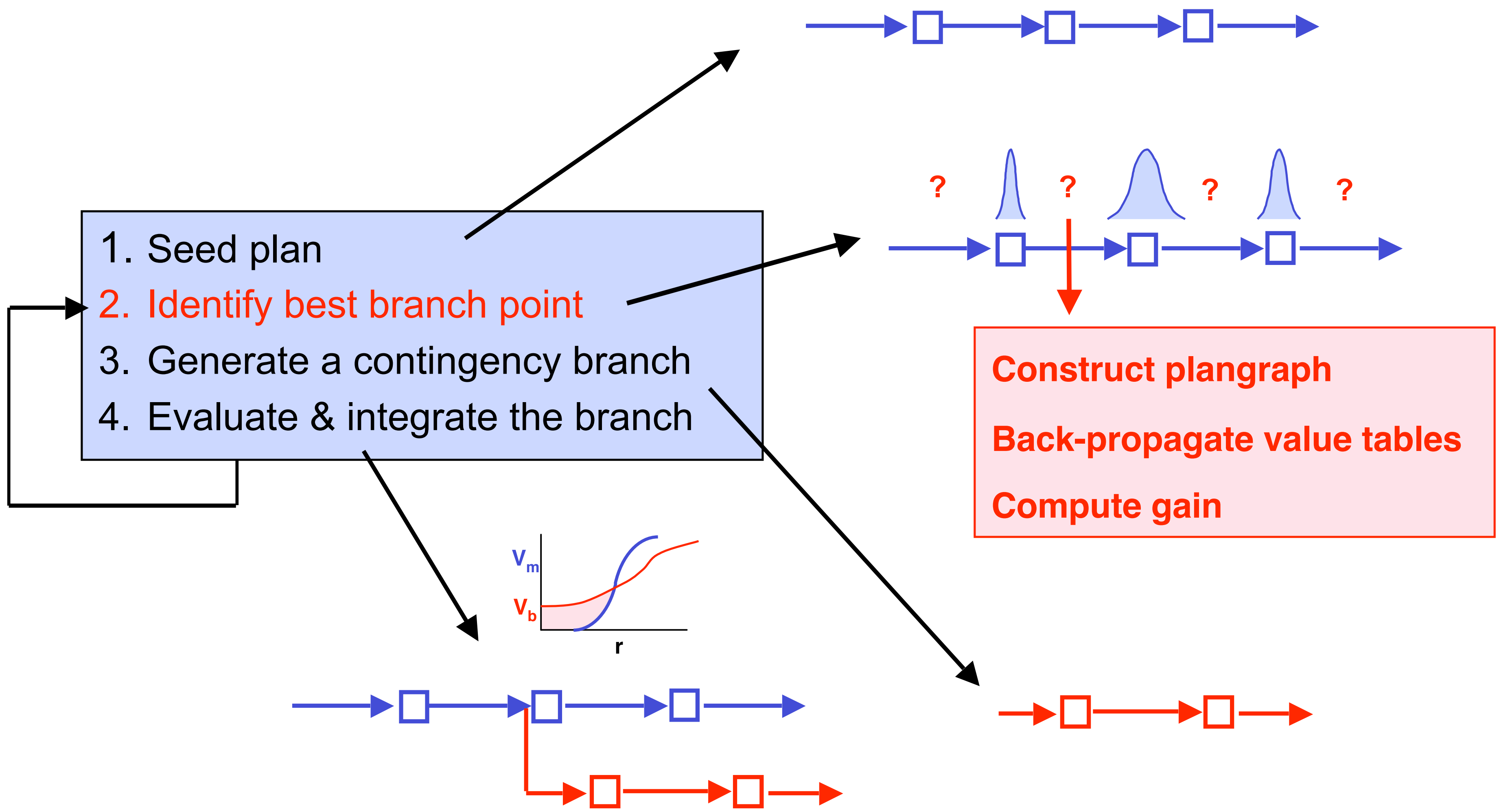
Expected Branch Gain

$$\text{Gain} = \int_0^{\infty} P(r)\ \max\{0, V_b(r) - V_m(r)\}\ dr$$



Identifying the Best Branch Point

1. Seed plan
2. Identify best branch point
3. Generate a contingency branch
4. Evaluate & integrate the branch



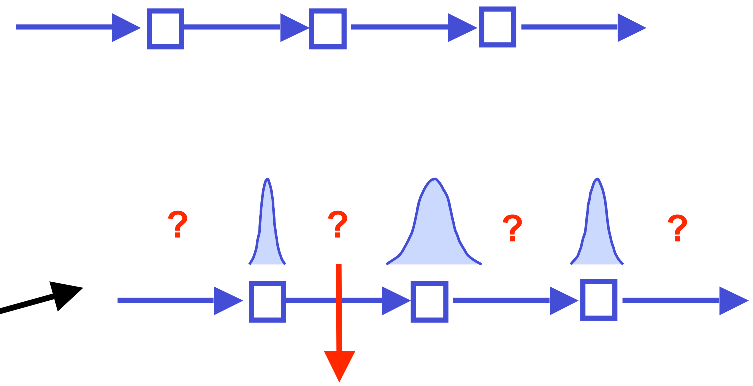
Construct plangraph

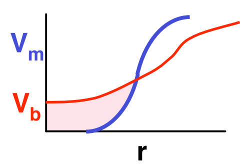
Back-propagate value tables

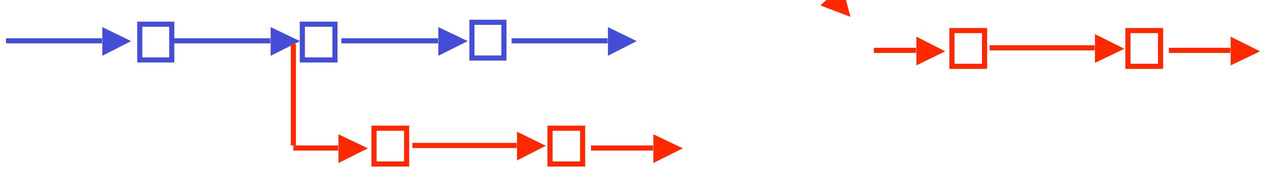
Compute gain

Generating the Branch

1. Seed plan
2. Identify best branch point
3. Generate a contingency branch
4. Evaluate & integrate the branch



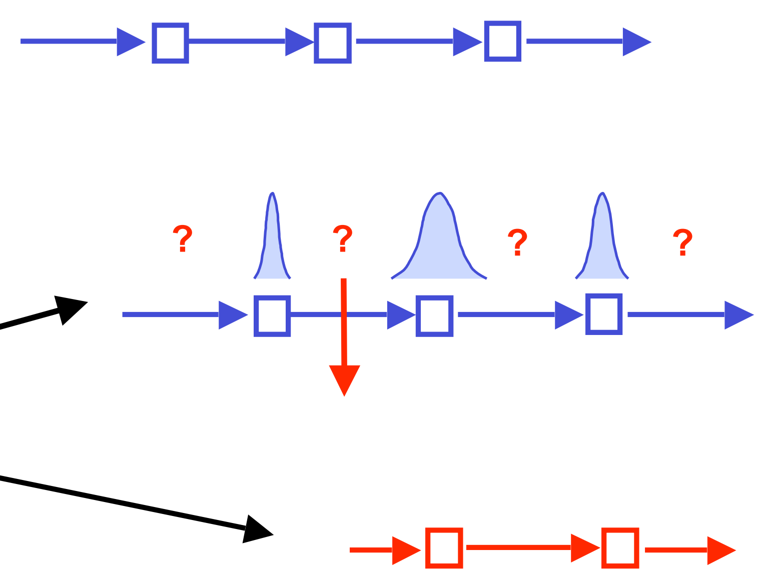


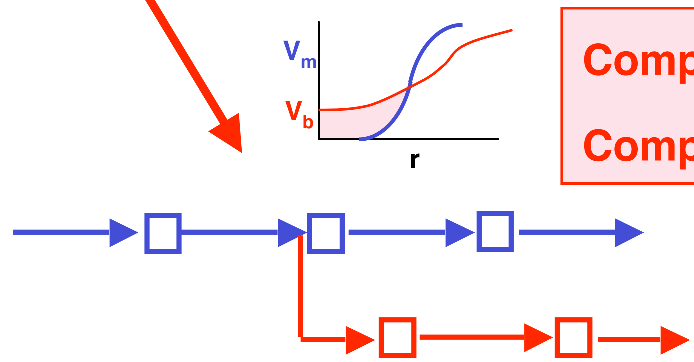


Plan for the branch/condition

Evaluating the Branch

1. Seed plan
2. Identify best branch point
3. Generate a contingency branch
4. Evaluate & integrate the branch





Compute value function

Compute actual gain

Outline

Incremental approaches

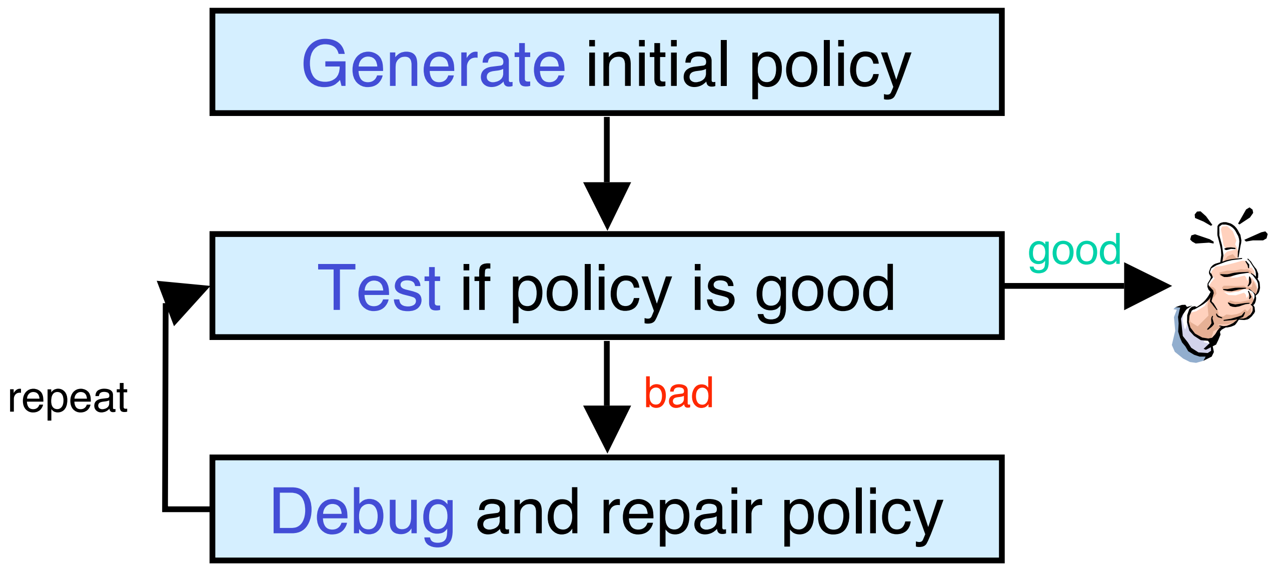
- JIC
- ICP
- Tempastic

When is contingency planning needed ?

Combining contingency planning & replanning

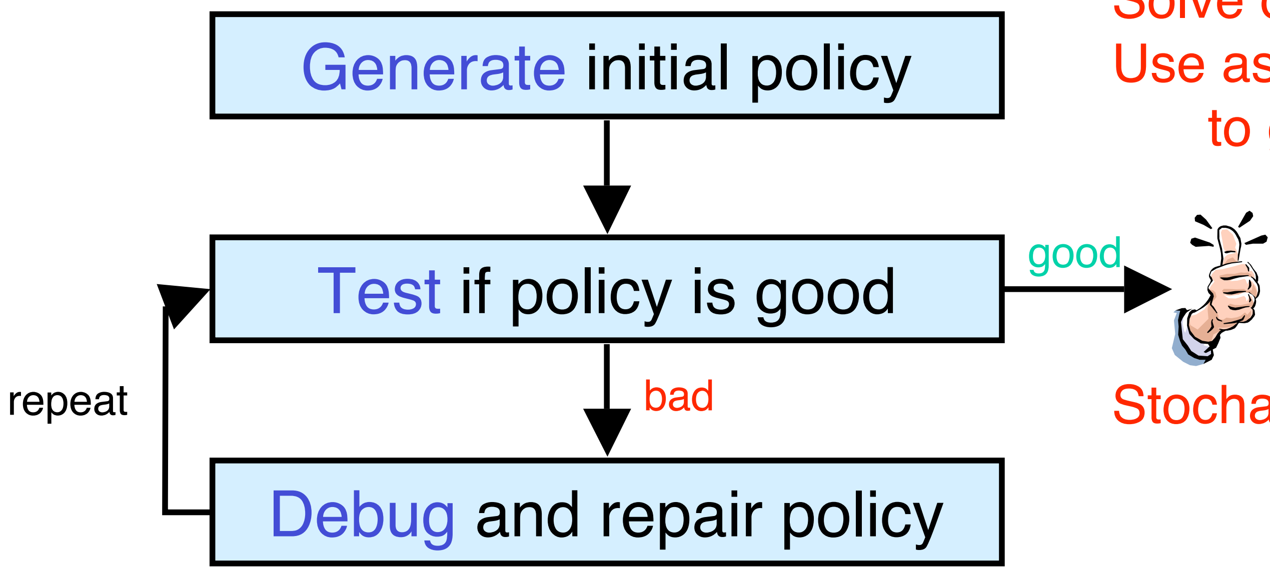
Applications

Tempastic



Ref: Younes & Simmons, ICAPS-04

Tempastic Details



Solve deterministic problem
Use as training data
to generate policy

Stochastic simulation

Rank bugs
Adapt deterministic problem
Solve deterministic problem
Use as training data
to improve policy

Policy Generation

Probabilistic planning problem

↓ Split discrete outcomes
Relax continuous outcomes

Deterministic planning problem

↓ Solve using VHPOP

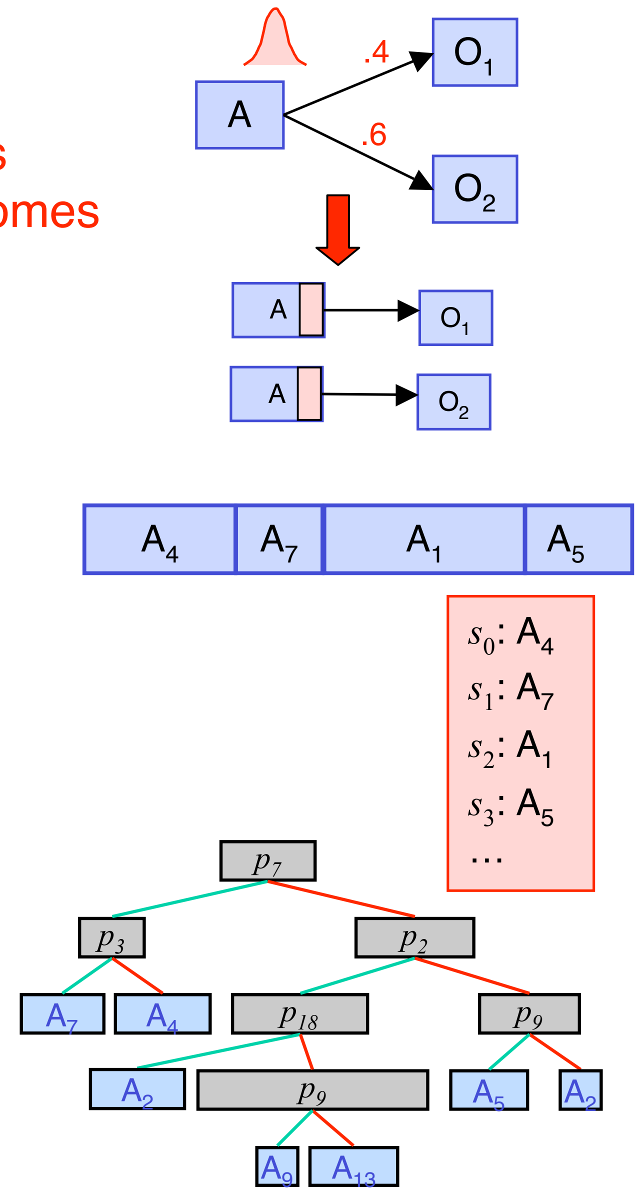
Temporal plan

↓ Generate training data
by simulating plan

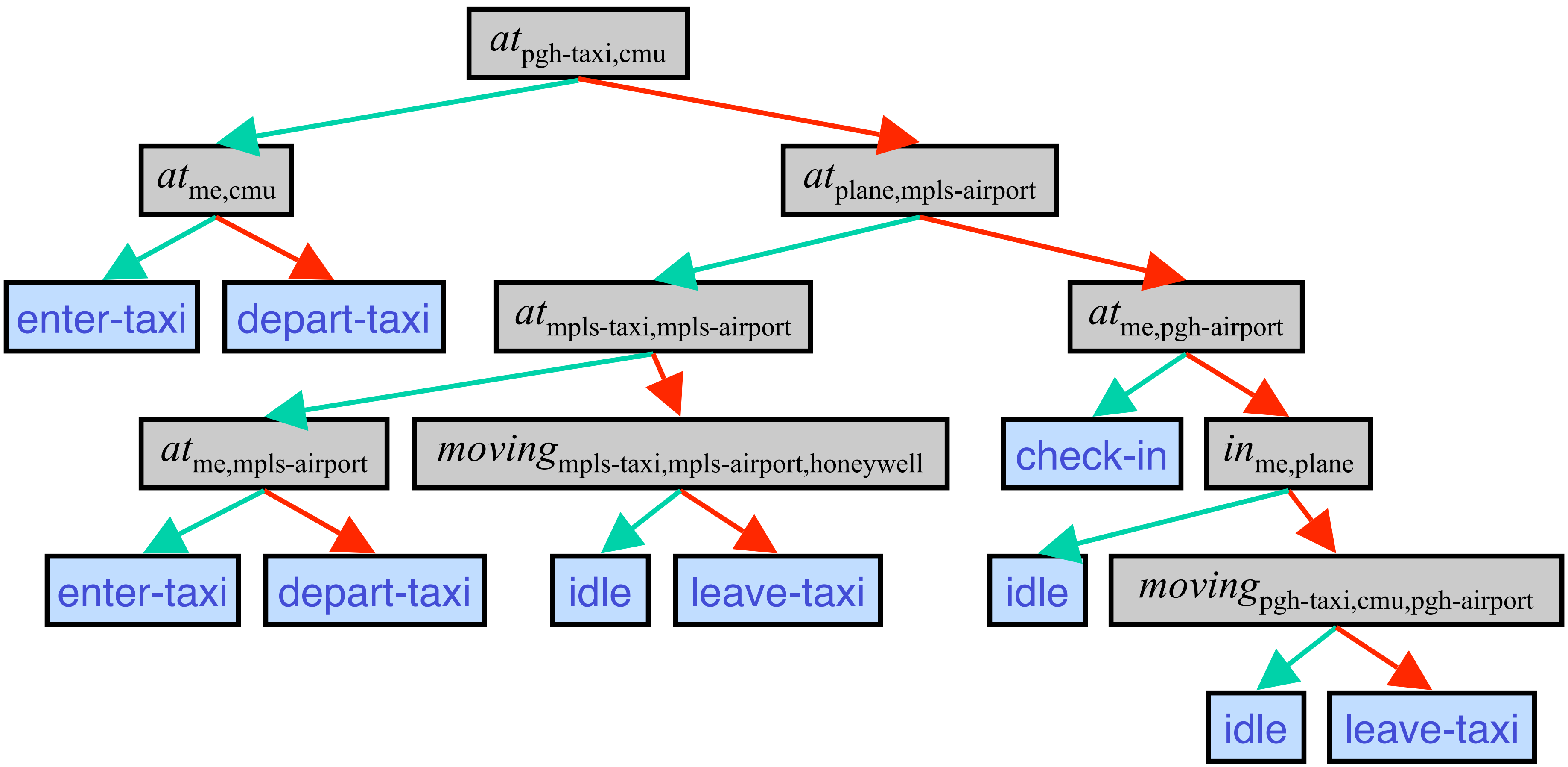
State-action pairs

↓ Decision tree learning

Policy (decision tree)

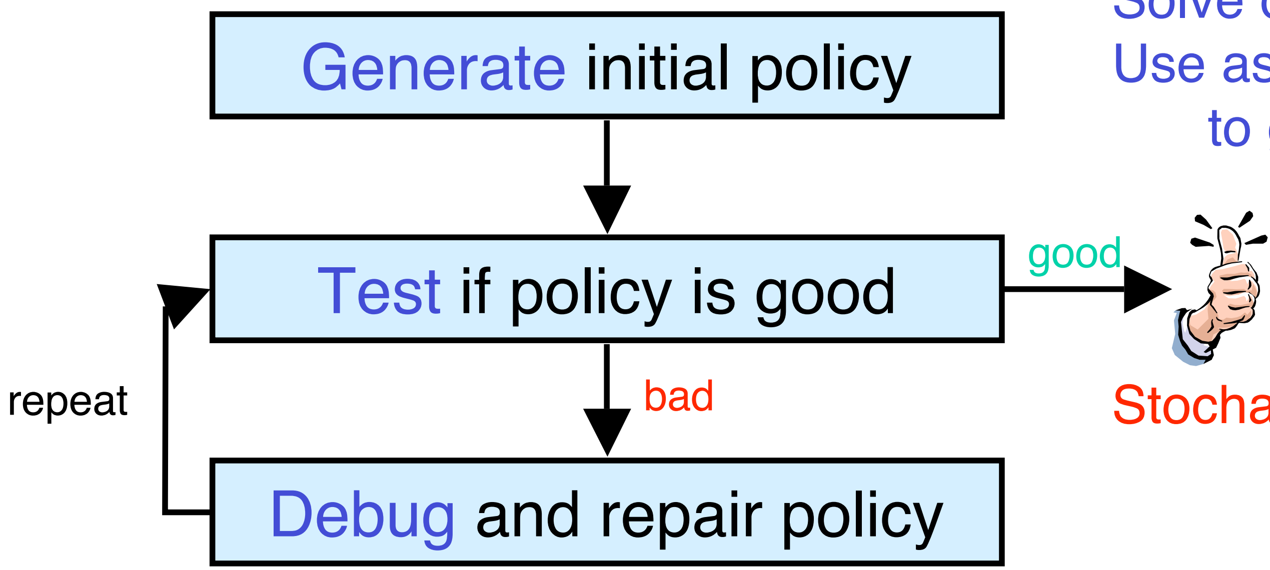


Policy Tree



$at_{pgh\text{-}taxi,cmu}$
$at_{me,cmu}$
$at_{plane,mpls\text{-}airport}$
enter-taxi
depart-taxi
$at_{mpls\text{-}taxi,mpls\text{-}airport}$
$at_{me,pgh\text{-}airport}$
$at_{me,mpls\text{-}airport}$
$moving_{mpls\text{-}taxi,mpls\text{-}airport,honeywell}$
check-in
$in_{me,plane}$
enter-taxi
depart-taxi
idle
leave-taxi
idle
$moving_{pgh\text{-}taxi,cmu,pgh\text{-}airport}$
idle
leave-taxi

Tempastic Details

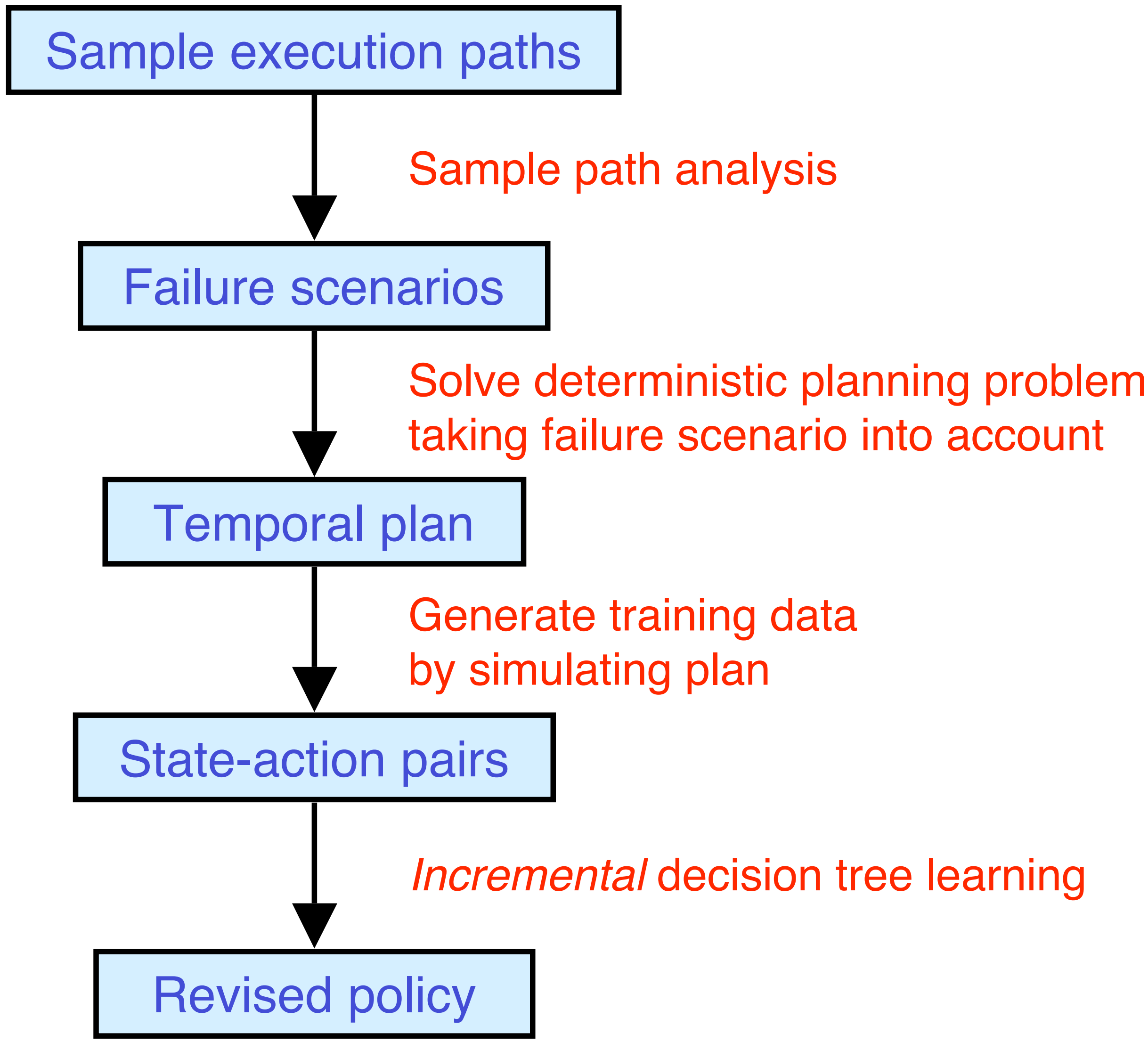


Solve deterministic problem
Use as training data
to generate policy

Stochastic simulation

Rank bugs
Adapt deterministic problem
Solve deterministic problem
Use as training data
to improve policy

Policy Debugging



Policy Debugging Details

Sample execution paths

Sample path analysis

Failure scenarios

Solve deterministic planning problem
taking failure scenario into account

Temporal plan

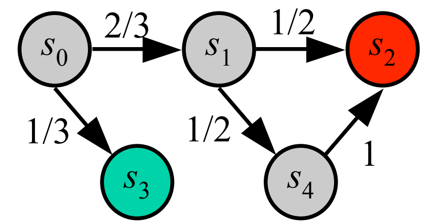
Generate training data
by simulating plan

State-action pairs

Incremental decision tree learning

Revised policy

Construct Markov chain:

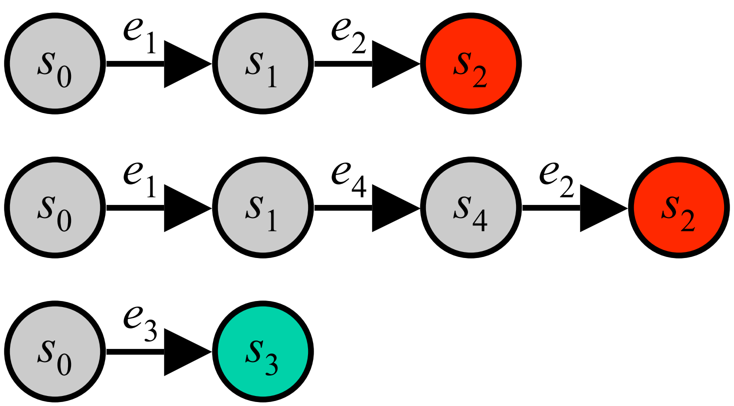


Bellman backups

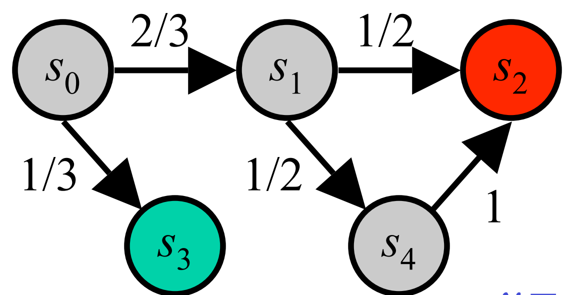
Incorporate most important failure
& force planner to work around it

Sample Path Analysis: Example

Sample paths:



Markov chain:



$\gamma = 0.9$

Event values:

$V(e_1) = 2 \cdot (V(s_1) - V(s_0)) = -1.284$

$V(e_2) = (V(s_2) - V(s_1)) + (V(s_2) - V(s_4)) = -0.245$

$V(e_3) = V(s_3) - V(s_0) = +1.213$

$V(e_4) = V(s_4) - V(s_1) = -0.045$

State values:

$V(s_0) = -0.213$

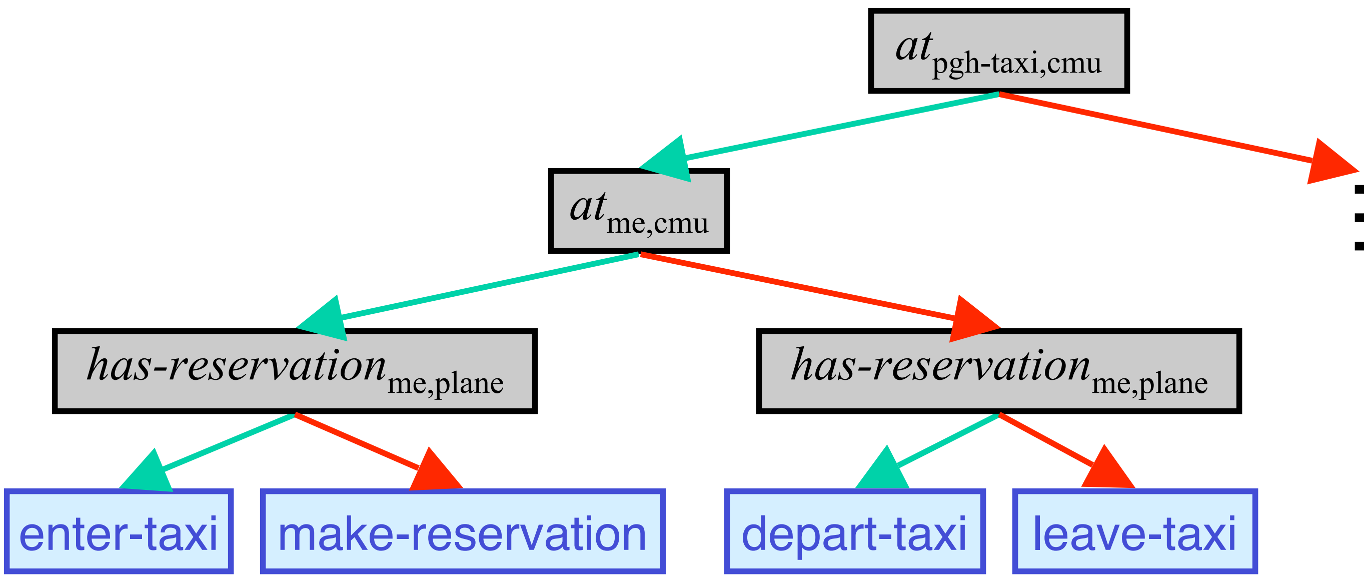
$V(s_1) = -0.855$

$V(s_2) = -1$

$V(s_3) = +1$

$V(s_4) = -0.9$

Revised Policy Tree



Tempastic Summary

Generate initial policy

Test if policy is good

good

bad

repeat

Debug and repair policy

Solve deterministic problem
Use as training data
to generate policy

Stochastic simulation

Rank bugs
Adapt deterministic problem
Solve deterministic problem
Use as training data
to improve policy

Advantages & Drawbacks

Advantages

Tractability

Anytime

Simple plans

Drawbacks

Sacrifice optimality

- seed plan
- repairs

Thrashing

Flaw Selection

- particularly for oversubscription

Outline

Incremental approaches

When is contingency planning really needed ?

Combining contingency planning & replanning

Applications

Alternative Approaches

Replanning

Improving robustness

- Conservatism
- Flexibility
- Conformance
- Conditionality

Alternative Approaches

Replanning

Improving robustness

- Conservatism
- Flexibility
- Conformance
- Conditionality

Not mutually exclusive

Which one when?

Requirements & Drawbacks

Approach	Requirements	Drawbacks
Replanning	Adequate time, computational power Time not critical resource No dead ends	Lost opportunity Non-optimal Failure
Improving robustness		
Conservatism	Resource usage	Lost opportunity
Flexibility	Limited uncertainty Sophisticated rep., planner, exec	Weak Computational
Conformant	Limited uncertainty Powerful actions	Weak Computational
Contingency	Model outcomes # of outcomes small	Computational

When?

Approach	When	ISS Examples
Replanning	Minor annoyances reversible outcomes low penalty Rich opportunities Highly stochastic	Misplaced supplies Loading, storage Job jar Obstacle avoidance
Improving robustness		
Conservatism	Critical resource	O_2, H_2O, food, power
Flexibility	Duration uncertainty Event time uncertainty	Daily tasks Communication
Conformant	Simple forcing actions	Computer reset
Contingency	<u>Only Critical situations</u>	Power inverter failure Pressure leak Fire

Point?

\start{soapbox}

Considered within larger context

- replanning

Different emphasis

- unrecoverable outcomes
  - (not just high probability/low value outcomes)

Impacts for Policy Search

Considered within larger context

replanning

Different emphasis

unrecoverable outcomes

(not just high probability/low value outcomes)

1. Don't care about having a complete policy
2. Policy must cover critical outcomes

\end{soapbox}

Outline

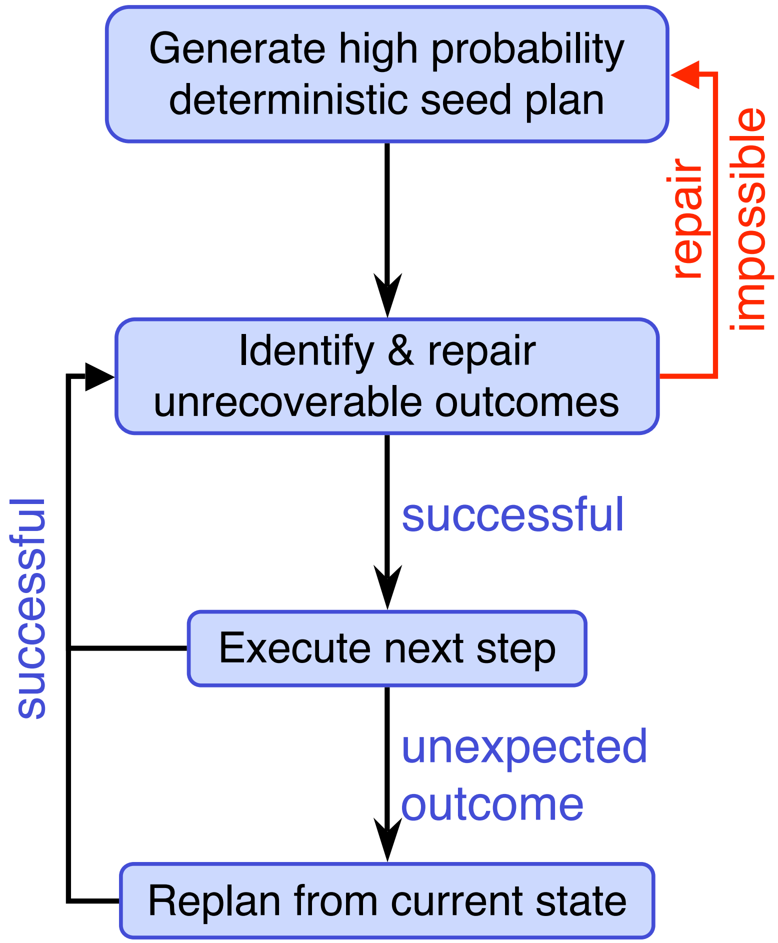
Incremental approaches

When is contingency planning really needed ?

Combining contingency planning & replanning

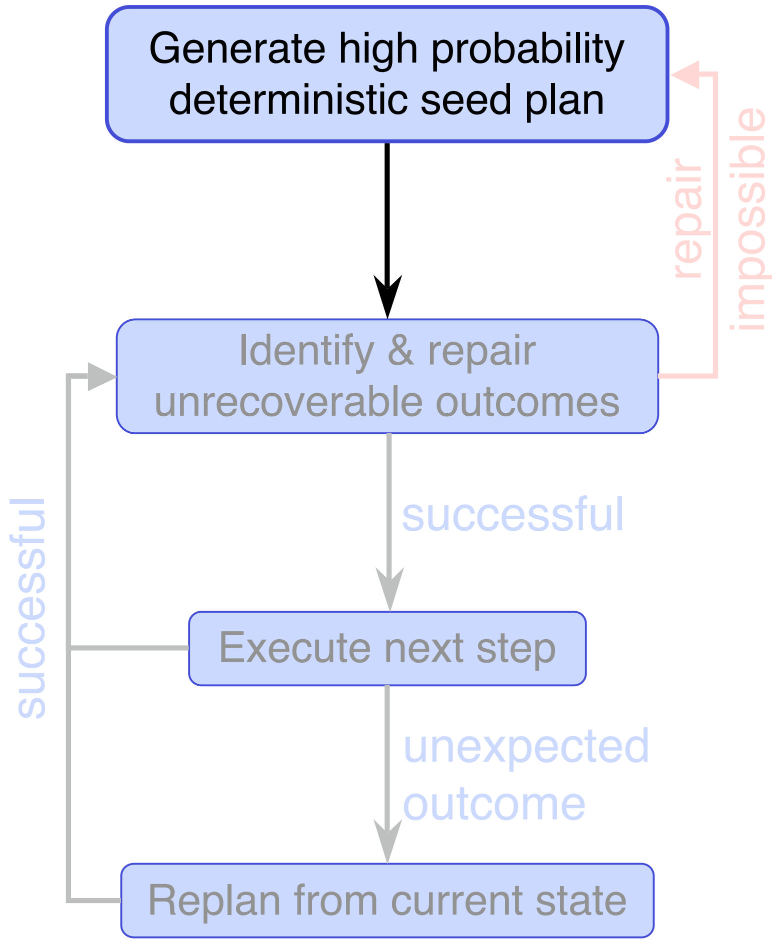
Applications

Precautionary Planning

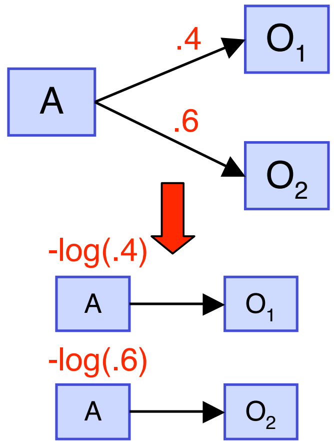


Ref: Foss, Onder & Smith, ICAPS-07 Wkshp

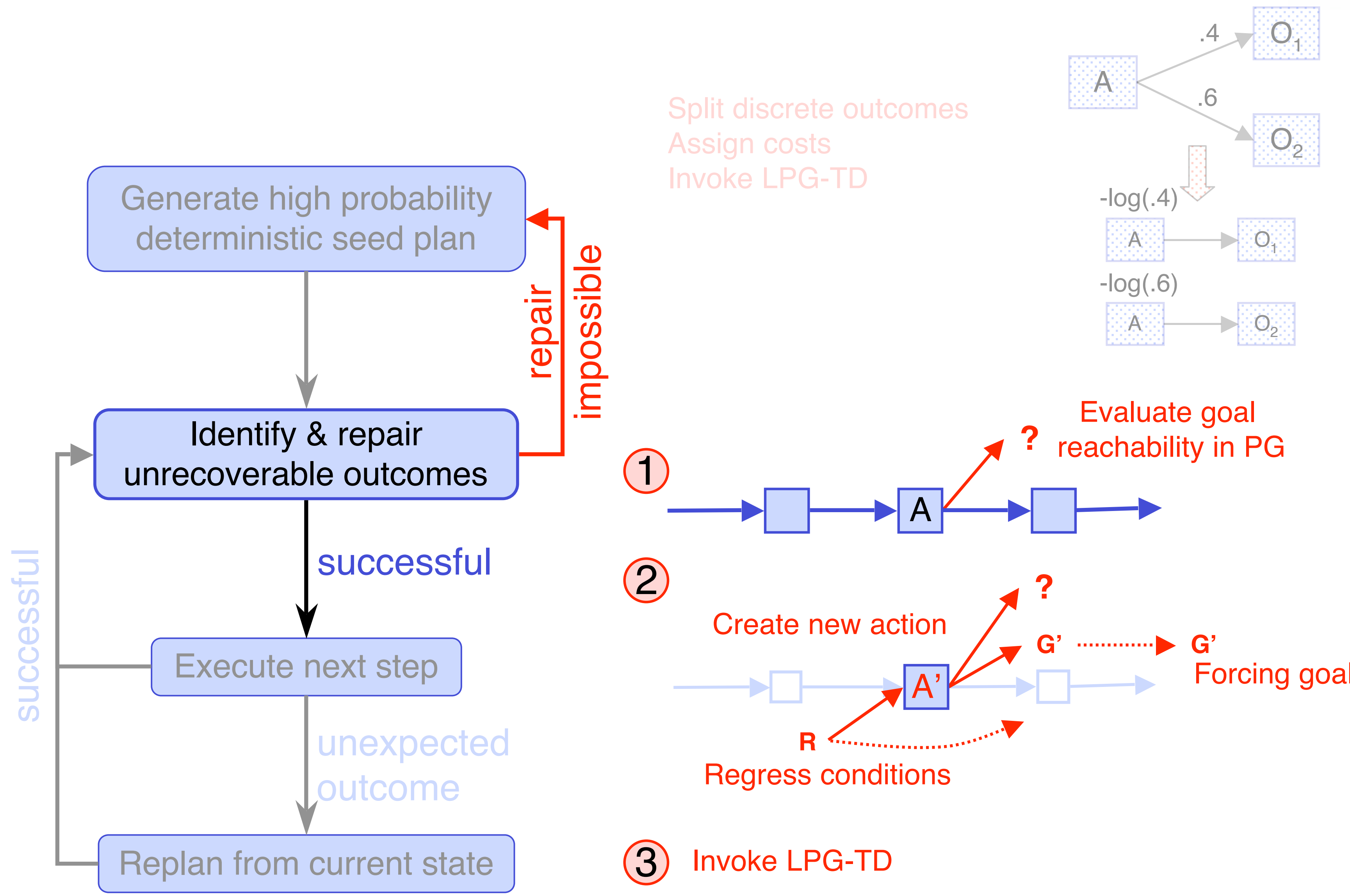
Seed Plan Generation



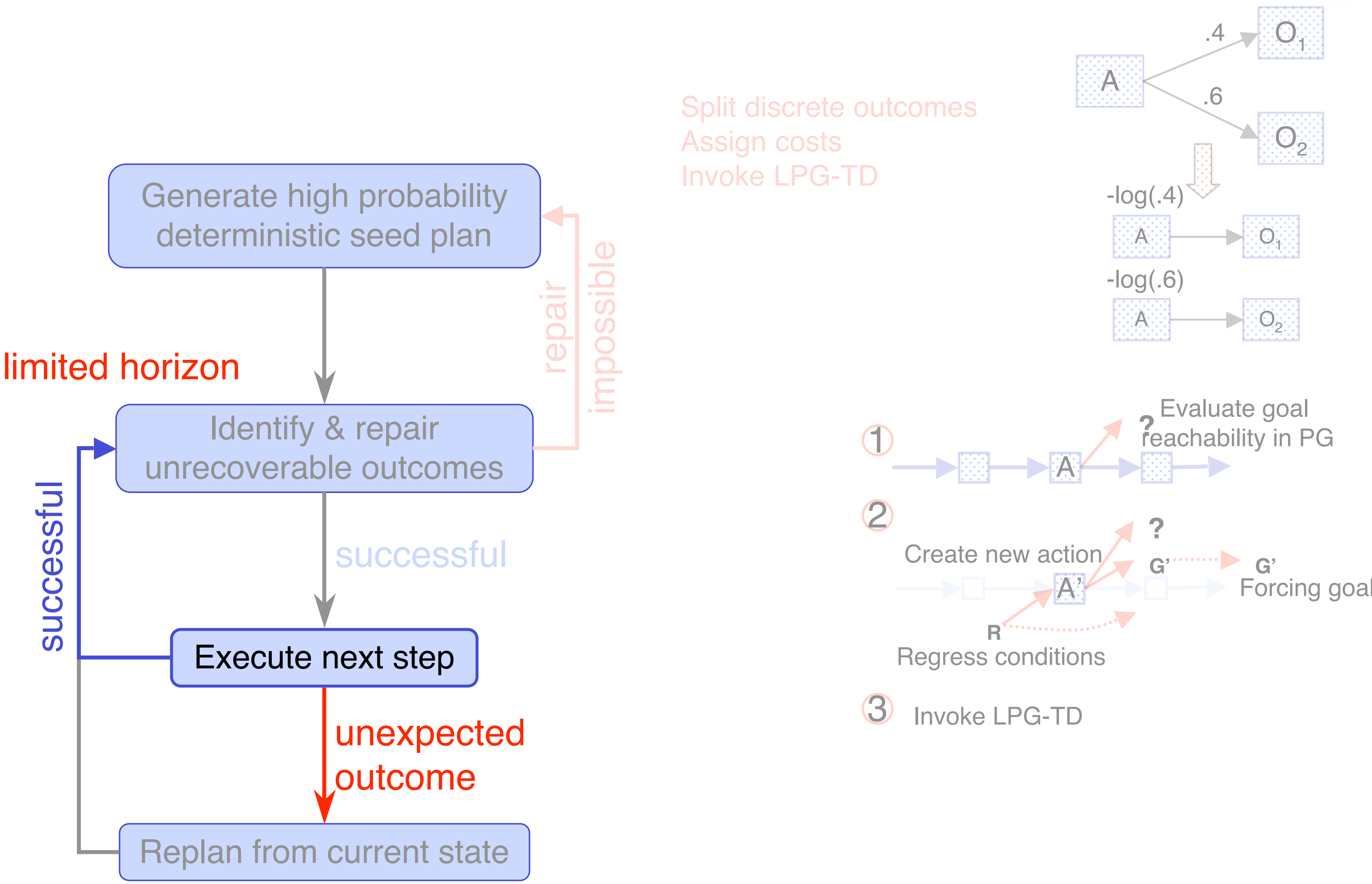
Split discrete outcomes
Expectations
Assign costs
Invoke LPG-TD



Unrecoverable Outcomes

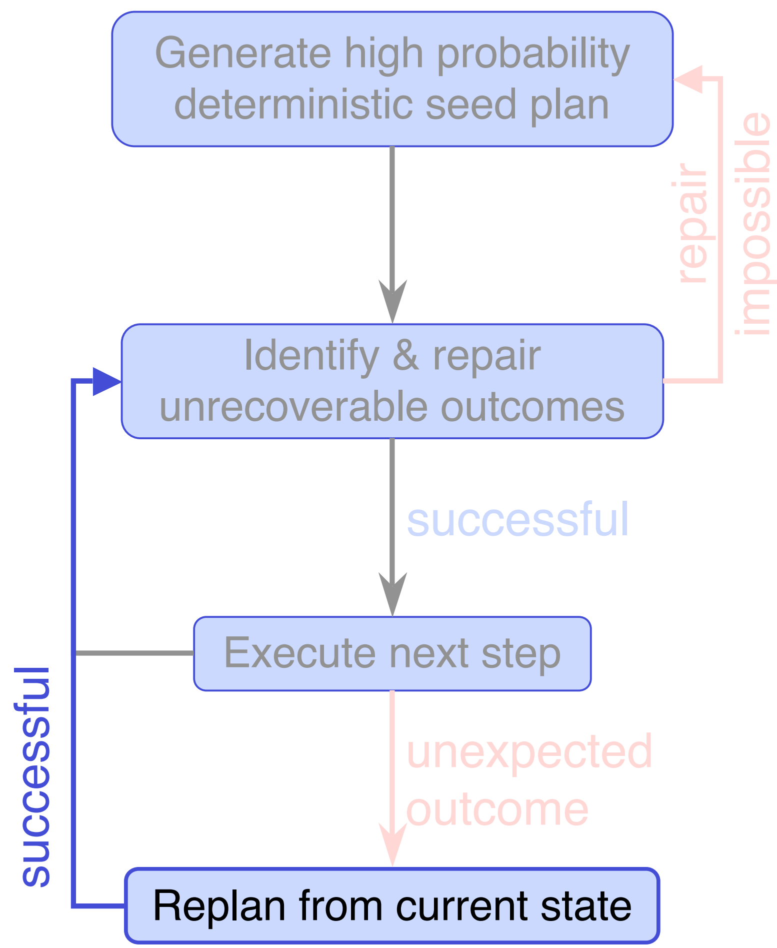


Execution

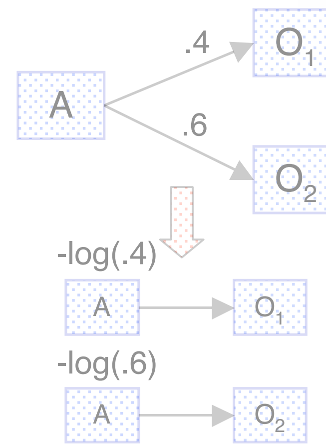


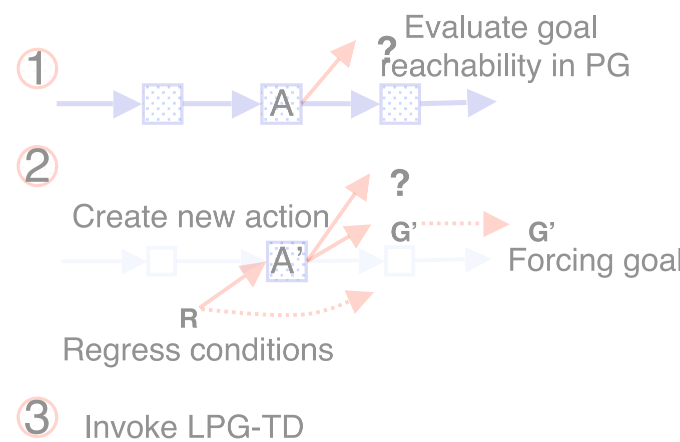
Split discrete outcomes
Assign costs
Invoke LPG-TD
A
.4
.6
O1
O2
-log(.4)
A
O1
-log(.6)
A
O2
Generate high probability deterministic seed plan
repair impossible
limited horizon
Identify & repair unrecoverable outcomes
successful
successful
Execute next step
unexpected outcome
Replan from current state
1
A
?
Evaluate goal reachability in PG
2
Create new action
A'
?
G'
G'
Forcing goal
R
Regress conditions
3
Invoke LPG-TD

Unplanned Outcomes



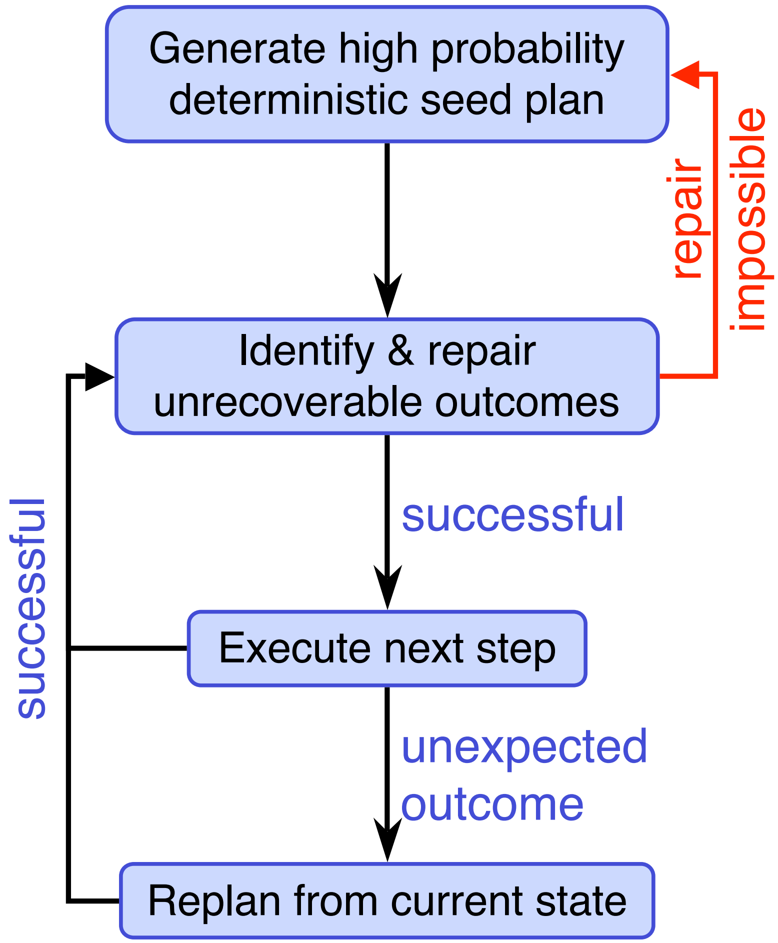
Split discrete outcomes
Assign costs
Invoke LPG-TD





Split discrete outcomes
Assign costs
Invoke LPG-TD

Main Points



ICP combined with replanning

Deterministic planner

Repair unrecoverable outcomes

Ref: Foss, Onder & Smith, ICAPS-07 Wkshp

Outline

Incremental approaches

When is contingency planning really needed ?

Combining contingency planning & replanning

Applications

- Military air campaign planning [Meuleau et al AAAI-98]
- Military operations planning [Aberdeen et al ICAPS-04]
- Rover planning [Pedersen et al IEEEaero-05] [Meuleau et al AAAI-04 Wkshp]

Military Air Campaign Planning

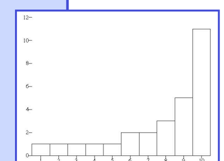
[Meuleau et al AAAI-98]

Customer: DARPA

Problem:

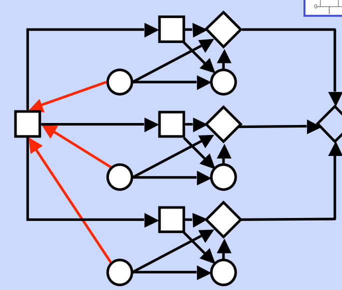
- military targets with time windows
- limited number of weapons (bombs) & aircraft
- strike outcomes uncertain, but observable
- objective – allocate aircraft & bombs to targets at each time step

Concurrency (1000)
Unit time actions
Discrete outcomes



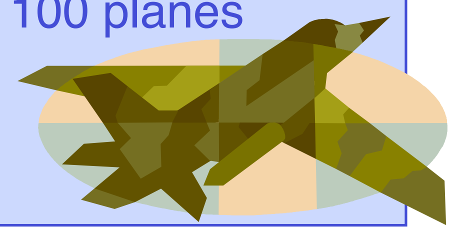
Approach

- Markov Task Decomposition (MTD)
  - offline: solve parameterized MDPs for each target
  - at each time step, allocate weapons across targets



Results

- Synthetic problems: 1000 targets, 10,000 weapons, 100 planes
- 35 minutes
- quality close to DP



Military Operations Planning

[Aberdeen et al ICAPS-04]

Customer: Australian Defence Science & Technology Organisation

Problem:

- set of military objectives (propositions)
- tasks (durative actions) make propositions true/false
- objective - achieve goals
  - minimize failure, makespan, resource cost

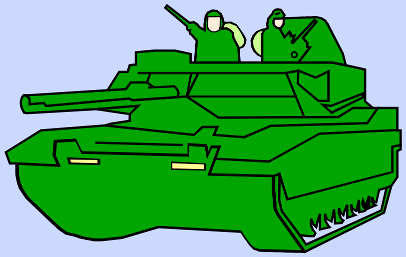
Concurrency (8)
Durative actions
Discrete outcomes

Approach

- LRTDP
- admissible heuristics – probability, makespan, resource usage
- pruning of states not recently visited (LRU)

Results

- synthetic problems (85) & military scenarios (2)
- biggest: 41 tasks, 51 facts, 19 resource types
  - 10 minutes



Rover Planning

[Pedersen et al IEEEaero-05]

Customer: NASA

Problem:

- set of science goals w/utilities, time constraints
- time & energy limitations
- duration & resource usage uncertain (driving)
- objective - maximize scientific reward

Approach

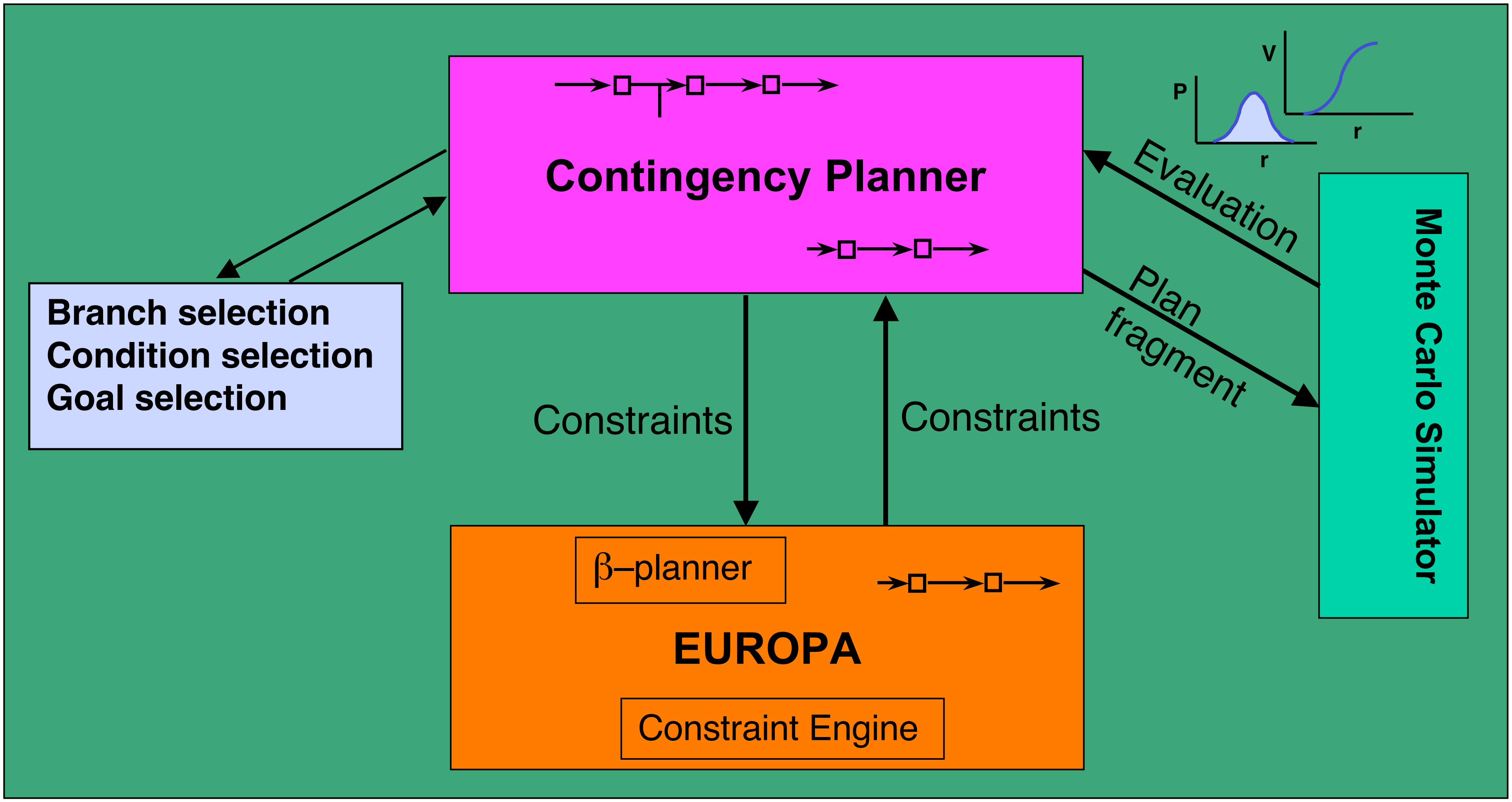
- ICP w/EUROPA planner
- heuristics
  - branch selection – utility drop
  - goal selection – orienteering

Results

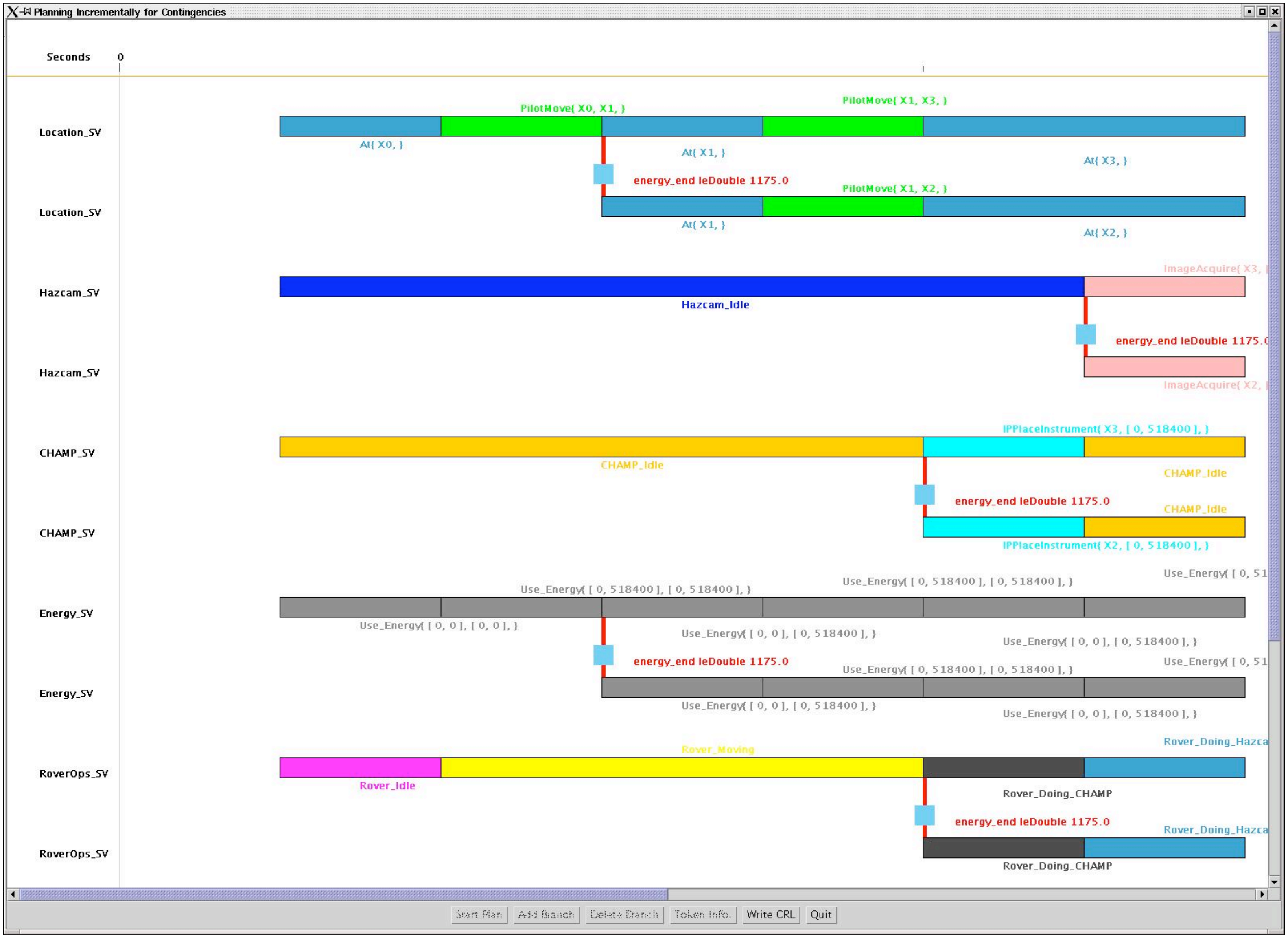
- simulator problems w/upto 20 objectives
- K9 rover - small problems (5 objectives)

Durative actions
Continuous outcomes
Oversubscription
Minor concurrency

Planner Architecture



Contingency Plan



Rover Planning

[Meuleau et al AAAI-04 Wkshp]

Customer: NASA

Oversubscription
Concurrency

Problem:

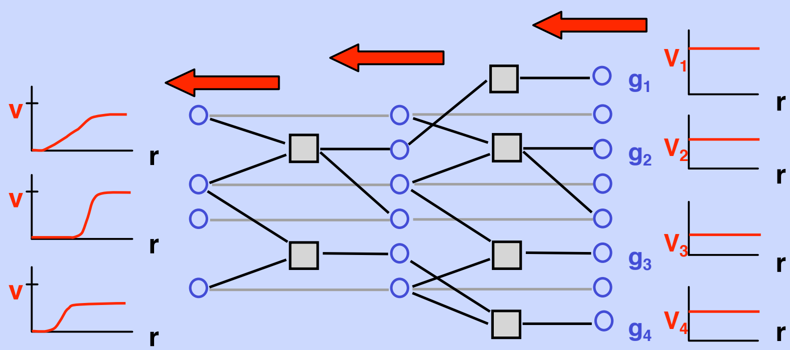
set of science goals w/utilities

objective - maximize scientific reward

Approach

Plangraph construction

DP regression of utility tables through PG



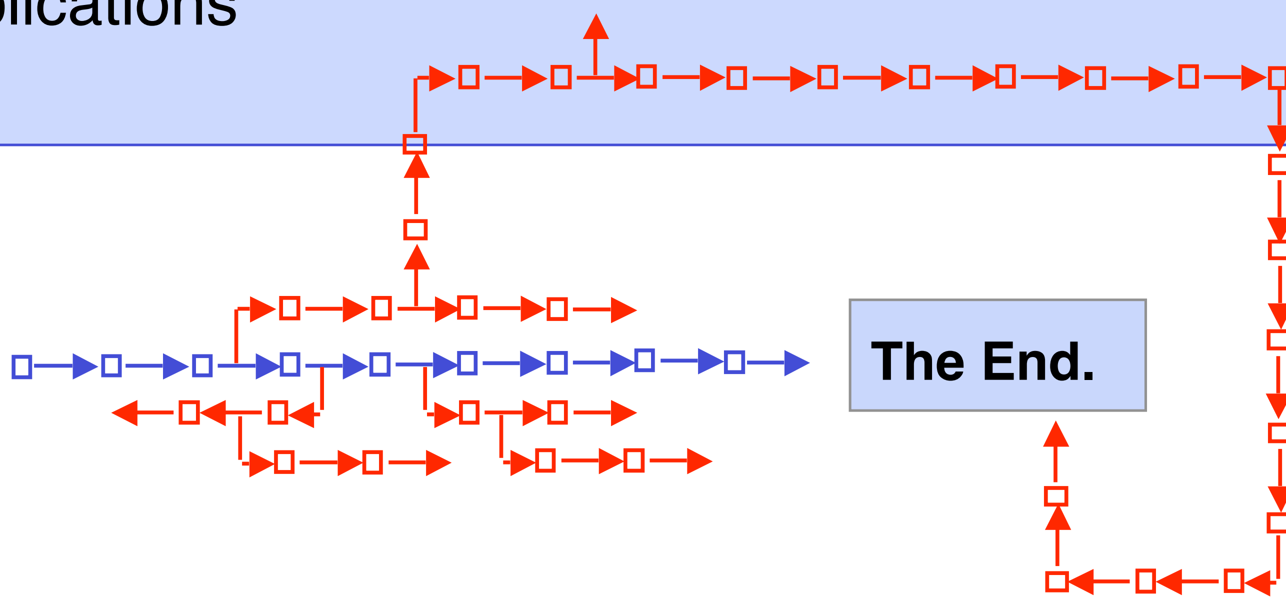
Results

synthetic problems w/upto 5 objectives, 75 paths

40s

Outline

- Incremental approaches
- When is contingency planning really needed ?
- Combining contingency planning & replanning
- Applications



References – Incremental Approaches

Dearden, R.; Meuleau, N.; Ramakrishnan, S.; Smith, D.; and Washington, R. Incremental contingency planning. *ICAPS-03 Wkshp on Planning under Uncertainty and Incomplete Information*.

Drummond, M.; Bresina, J.; and Swanson, K. Just In-Case scheduling. *AAAI-94*.

Foss, J., and Onder, N. A hill-climbing approach to planning with temporal uncertainty. *FLAIRS-06*.

Foss, J.; Onder, N.; and Smith, D. Preventing unrecoverable failures through precautionary planning. *ICAPS-07 Wkshp on Moving Planning and Scheduling Systems into the Real World*.

Long, D., and Fox, M. Singe-trajectory opportunistic planning under uncertainty. *2002 UK Planning and Scheduling SIG*.

Younes, H., and Simmons, R. Policy generation for continuous-time stochastic domains with concurrency. *ICAPS-04*.

References – Applications

Aberdeen, D.; Thiébaux, S.; and Zhang, L. Decision theoretic military operations planning. *ICAPS-04*.

Meuleau, N.; Dearden, R.; and Washington, R. Scaling up decision theoretic planning to planetary rover problems. *AAAI-04 Workshop on Learning and Planning in Markov Processes: Advances and Challenges*.

Meuleau, N.; Hauskrecht, M.; Kim, K.; Peshkin, L.; Kaelbling, L.; Dean, T.; and Boutilier, C. 1998. Solving very large weakly coupled Markov Decision Processes. *AAAI-98*.

Pedersen, L.; D.Smith; Dean, M.; Sargent, R.; Kunz, C.; Lees, D.; and Rajagopalan, S. Mission planning and target tracking for autonomous instrument placement. *2005 IEEE Aerospace Conf.*