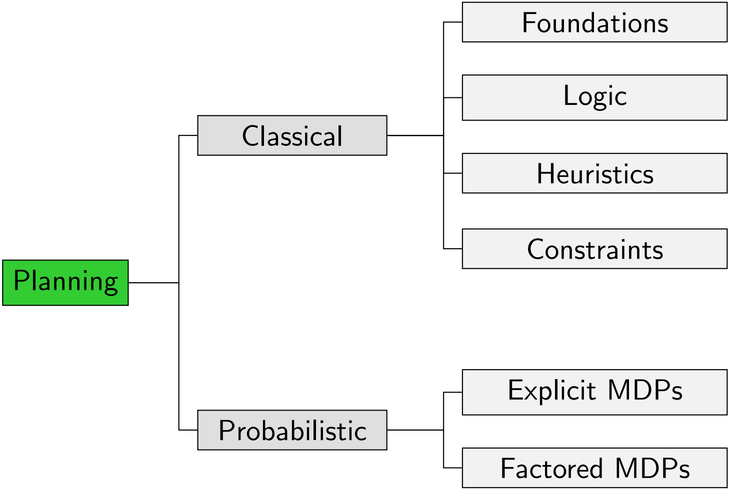
Planning and Optimization

F1. Markov Decision Processes

Malte Helmert and Gabriele Röger

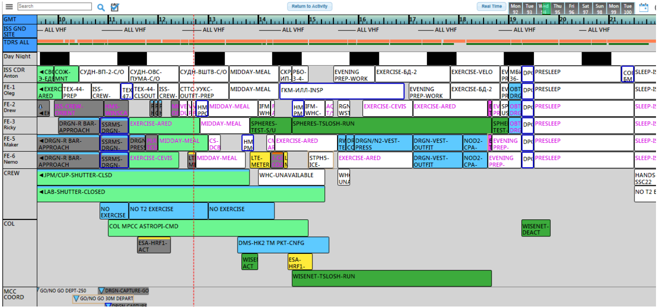
Universität Basel

Content of this Course



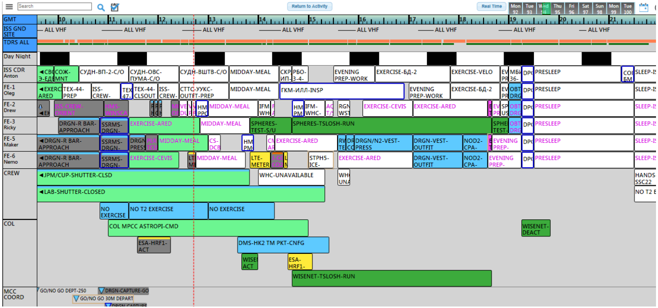
Motivation

Limitations of Classical Planning



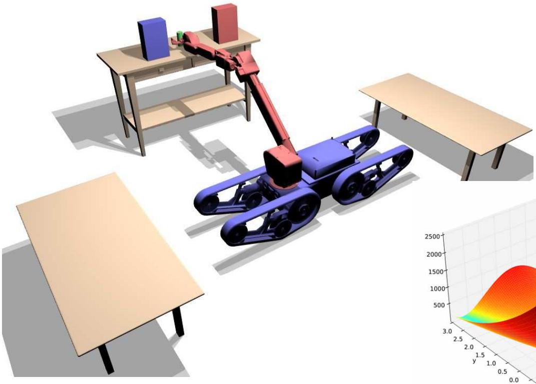
- timetable for astronauts on ISS

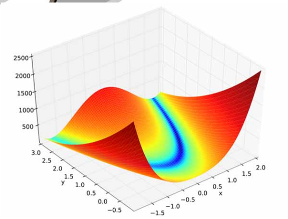
Generalization of Classical Planning: Temporal Planning



- timetable for astronauts on ISS
- concurrency required for some experiments
- optimize makespan

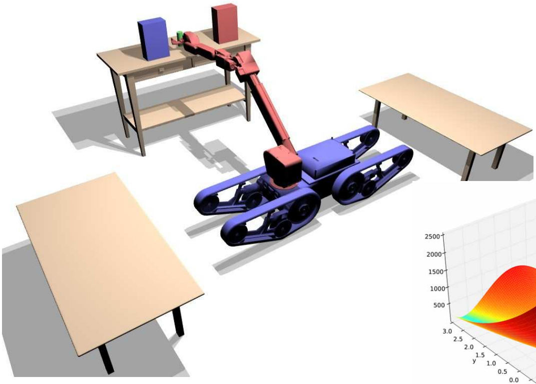
Limitations of Classical Planning

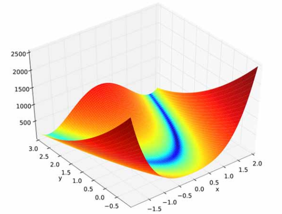




- kinematics of robotic arm

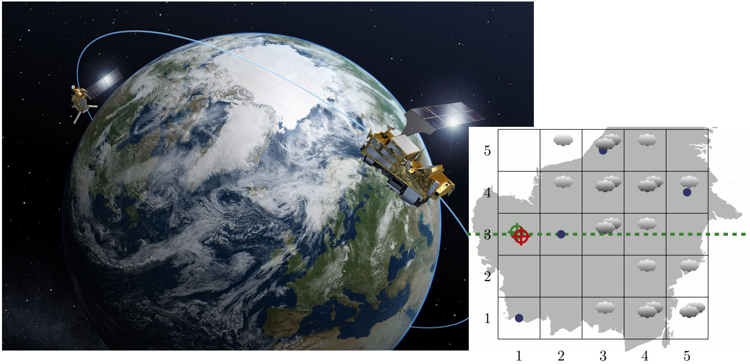
Generalization of Classical Planning: Numeric Planning





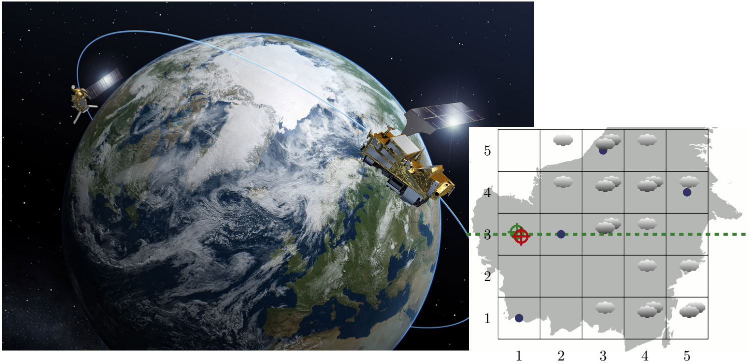
- kinematics of robotic arm
- state space is continuous
- preconditions and effects described by complex functions

Limitations of Classical Planning



- satellite takes images of patches on earth

Generalization of Classical Planning: MDPs



- satellite takes images of patches on earth
- weather forecast is uncertain
- find solution with lowest expected cost

Limitations of Classical Planning

- Chess

Generalization of Classical Planning: Multiplayer Games

- Chess
- there is an opponent with a contradictory objective

Limitations of Classical Planning

- Solitaire

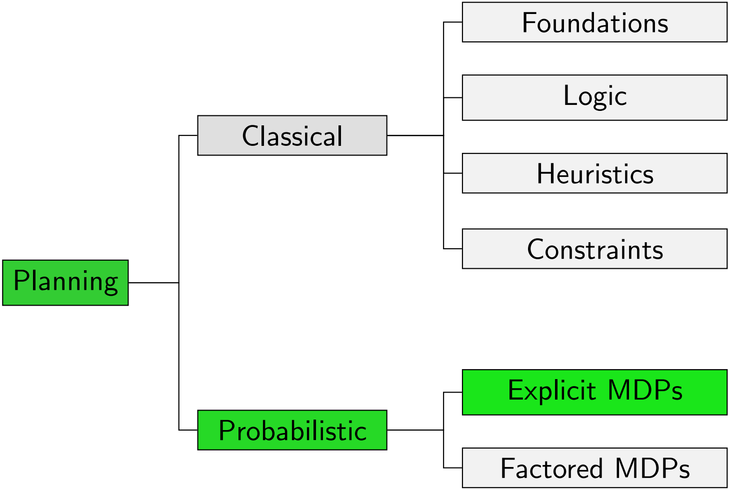
Generalization of Classical Planning: POMDPs

- Solitaire
- some state information cannot be observed
- must reason over belief for good behaviour

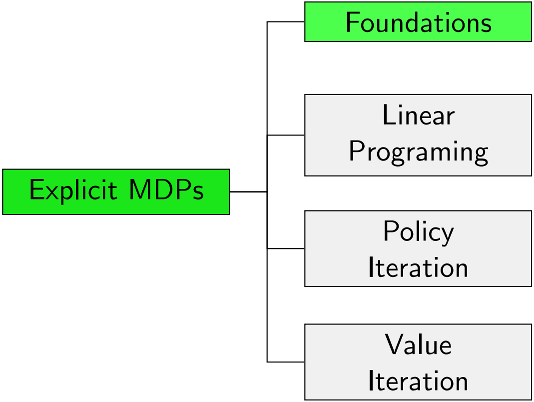
Limitations of Classical Planning

- many applications are combinations of these
- all of these are active research areas
- we focus on one of them:
  probabilistic planning with Markov decision processes
- MDPs are closely related to games (Why?)

Content of this Course



Content of this Course: Explicit MDPs



Markov Decision Process

Markov Decision Processes

- Markov decision processes (MDPs) studied since the 1950s
- Work up to 1980s mostly on theory and basic algorithms for small to medium sized MDPs ($\rightsquigarrow$ Parts F1–F4)
- Today, focus on large, factored MDPs ($\rightsquigarrow$ Part F5 and following)
- Fundamental datastructure for reinforcement learning (not covered in this course)
- and for probabilistic planning
- different variants exist

Reminder: Transition Systems

Definition (Transition System)

A transition system is a 6-tuple $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ where

- S is a finite set of states,
- L is a finite set of (transition) labels,
- $c : L \to \mathbb{R}_0^+$ is a label cost function,
- $T \subseteq S \times L \times S$ is the transition relation,
- $s_0 \in S$ is the initial state, and
- $S_\star \subseteq S$ is the set of goal states.

$\rightarrow$ goal states and deterministic transition function

Markov Decision Process

Definition (Markov Decision Process)

A (discounted reward) Markov decision process (MDP) is a 6-tuple $\mathcal{T} = \langle S, A, R, T, s_0, \gamma \rangle$, where

- S is a finite set of states,
- A is a finite set of actions,
- $R : S \times A \to \mathbb{R}$ is the reward function,
- $T : S \times A \times S \mapsto [0, 1]$ is the transition function,
- $s_0 \in S$ is the initial state, and
- $\gamma \in (0, 1)$ is the discount factor.

For all $s \in S$ and $a \in A$ with $T(s, a, s') > 0$ for some $s' \in S$, we require $\sum_{s' \in S} T(s, a, s') = 1$.

Reward instead of Goal States

- the agent does not try to reach a goal state but gets a (positive or negative) reward for each action application.
- infinite horizon: agent acts forever
- finite horizon: agent acts for a specified number of steps
- we only consider the variant with an infinite horizon
- immediate reward is worth more than later reward
  - as in economic investments
  - ensures that our algorithms will converge
- the value of a reward decays exponentially with γ
- now full value r, in next step γr, in two steps only $\gamma^2 r, \ldots$
- aim: maximize expected overall reward

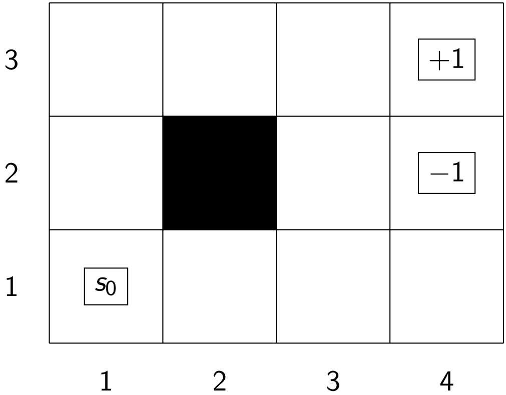
Markov Property

Why is this called a Markov decision process?

Russian mathematician
Andrey Markov (1856–1922)

Markov property: the probability distribution for the next state and the reward only depend on the current state (and the action) but not on previously visited states or earlier actions.

Example: Grid World



- moving *north* goes *east* with probability 0.4
- only applicable action in (4,2) and (4,3) is *collect*, which
  - sets position back to (1,1)
  - gives reward of +1 in (4,3)
  - gives reward of −1 in (4,2)

Solutions in MDPs

- classical planning
  - a solution is a sequence of operators
  - next state always clear
  - at the end we are in a goal state
- MDP
  - next state uncertain
  - we cannot know in advance what actions will be applicable in the encountered state
  - infinite horizon: act forever
  - $\rightarrow$ sequence of operators does not work
  - $\rightarrow$ policy: specify for each state the action to take
  - $\rightarrow$ at least for all states which we can potentially reach

Terminology (1)

- If $p := T(s, a, s') > 0$, we write $s \xrightarrow{p:a} s'$ (or $s \xrightarrow{p} s'$ if a is not relevant).
- If $T(s, a, s') = 1$, we also write $s \xrightarrow{a} s'$ or $s \rightarrow s'$.
- If $T(s, a, s') > 0$ for some s' we say that a is applicable in s.
- The set of applicable actions in s is $A(s)$. We assume that $A(s) \neq \emptyset$ for all $s \in S$.

Terminology (2)

- the successor set of s and a is $\mathrm{succ}(s, a) = \{s' \in S \mid T(s, a, s') > 0\}$.
- s' is a successor of s if $s' \in \mathrm{succ}(s, a)$ for some a.
- to indicate that s' is a successor of s and a that is sampled according to probability distribution T, we write $s' \sim \mathrm{succ}(s, a)$

Policy for MDPs

Definition (Policy for MDPs)

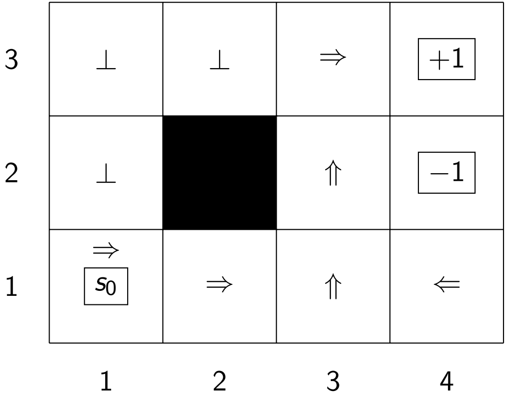
Let $\mathcal{T} = \langle S, A, R, T, s_0, \gamma \rangle$ be a (discounted-reward) MDP.
Let π be a mapping $\pi : S \to A \cup \{\bot\}$ such that $\pi(s) \in A(s) \cup \{\bot\}$ for all $s \in S$.

The set of reachable states $S_\pi(s)$ from s under π is defined recursively as the smallest set satisfying the rules

- $s \in S_\pi(s)$ and
- $\text{succ}(s', \pi(s')) \subseteq S_\pi(s)$ for all $s' \in S_\pi(s)$ where $\pi(s') \neq \bot$.

If $\pi(s') \neq \bot$ for all $s' \in S_\pi(s_0)$, then π is a policy for $\mathcal{T}$.

Example: Grid World



- moving *north* goes *east* with probability 0.4
- only applicable action in (4,2) and (4,3) is *collect*, which
  - sets position back to (1,1)
  - gives reward of +1 in (4,3)
  - gives reward of −1 in (4,2)

Stochastic Shortest Path Problem

I Want My Goal States Back!

- We also consider a variant of MDPs that are not discounted-reward MDPs.
- Stochastic Shortest Path Problems (SSPs) are closer to classical planning.
  - goal states
  - but still stochastic transition function
- We will use the same concepts for SSPs as for discounted-reward MDPs (e.g. policies)

Stochastic Shortest Path Problem

Definition (Stochastic Shortest Path Problem)

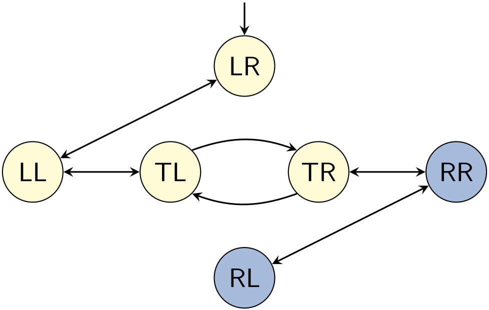
A stochastic shortest path problem (SSP) is a 6-tuple $\mathcal{T} = \langle S, A, c, T, s_0, S_\star \rangle$, where

- S is a finite set of states,
- A is a finite set of actions,
- $c : A \to \mathbb{R}_0^+$ is an action cost function,
- $T : S \times A \times S \mapsto [0, 1]$ is the transition function,
- $s_0 \in S$ is the initial state, and
- $S_\star \subseteq S$ is the set of goal states.

For all $s \in S$ and $a \in A$ with $T(s, a, s') > 0$ for some $s' \in S$, we require $\sum_{s' \in S} T(s, a, s') = 1$.

Note: An SSP is the probabilistic analogue of a transition system.

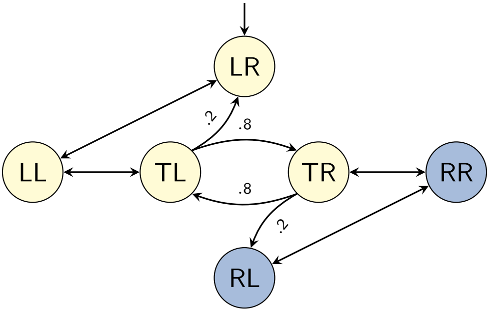
Transition System Example



Logistics problem with one package, one truck, two locations:

- location of package: domain $\{L, R, T\}$
- location of truck: domain $\{L, R\}$

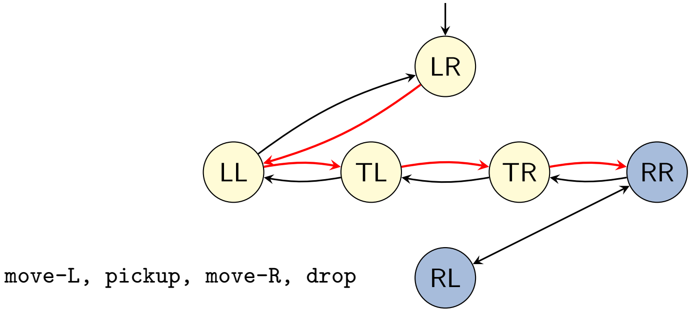
SSP Example



Logistics problem with one package, one truck, two locations:

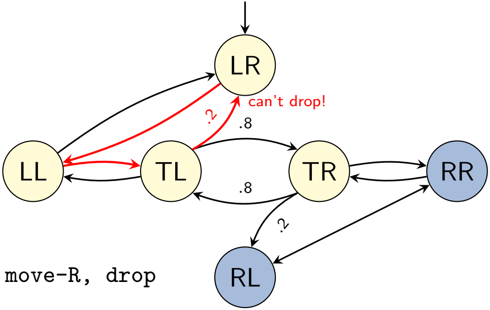
- location of package: $\{L, R, T\}$
- location of truck: $\{L, R\}$
- if truck moves with package, 20% chance of losing package

Solutions in Transition Systems



- in a deterministic transition system a solution is a plan, i.e., a sequence of operators that leads from s_0 to some $s_\star \in S_\star$
- an optimal solution is a cheapest possible plan
- a deterministic agent that executes a plan will reach the goal

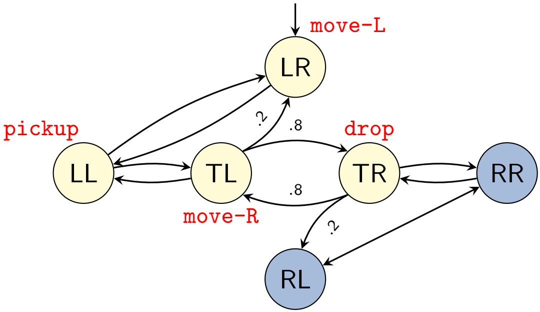
Solutions in SSPs



`move-L, pickup, move-R, drop`

- the same plan does not always work for the probabilistic agent (not reaching the goal or not being able to execute the plan)
- non-determinism can lead to a different outcome than anticipated in the plan
- need again a policy

Solutions in SSPs



Policy for SSPs

Definition (Policy for SSPs)

Let $\mathcal{T} = \langle S, A, c, T, s_0, S_\star \rangle$ be an SSP.
Let π be a mapping $\pi : S \to A \cup \{\perp\}$ such that $\pi(s) \in A(s) \cup \{\perp\}$ for all $s \in S$.

The set of reachable states $S_\pi(s)$ from s under π is defined recursively as the smallest set satisfying the rules

- $s \in S_\pi(s)$ and
- $\text{succ}(s', \pi(s')) \subseteq S_\pi(s)$ for all $s' \in S_\pi(s) \setminus S_\star$ where $\pi(s') \neq \perp$.

If $\pi(s') \neq \perp$ for all $s' \in S_\pi(s_0) \setminus S_\star$, then π is a policy for $\mathcal{T}$.
If the probability to eventually reach a goal is 1 for all $s' \in S_\pi(s_0)$ then π is a proper policy for $\mathcal{T}$.

Additional Requirements for SSPs

- We make two requirements for SSPs:
  - There is a proper policy.
  - Every improper policy incurs infinite cost from every reachable state from which it does not reach a goal with probability 1.
- We will only consider SSPs that satisfy these requirements.
- What does this mean in practise?
  - no unavoidable dead ends
  - no cost-free cyclic behaviour possible
- With these requirements every cost-minimizing policy is a proper policy.

Summary

Summary

- There are many planning scenarios beyond classical planning.
- For the rest of the course we consider probabilistic planning.
- (Discounted-reward) MDPs allow state-dependent rewards that are discounted over an infinite horizon
- SSPs are transition systems with a probabilistic transition relation.
- Solutions of SSPs and MDPs are policies.
- For MDPs we want to maximize the expected reward, for SSPs we want to minimize the expected cost.