Probabilistic Planning

Branislav Bošanský

PAH 2015/2016

Classical vs. Probabilistic Planning

- what have you learnt so far?
  - sequential decision making
  - deterministic effects of actions
  - static environment
  - perfect observation
  - perfect sensors

Classical vs. Probabilistic Planning

- the world is not perfect
  - actions take some time to execute
  - actions may fail or yield unexpected results
  - the environment may change due to other agents
  - the agent does not have knowledge about whole situation
  - other agents can have conflicting objectives
  - sensors are not precise

  - towards more realistic setting
  - planning with uncertainty

Classical vs. Probabilistic Planning

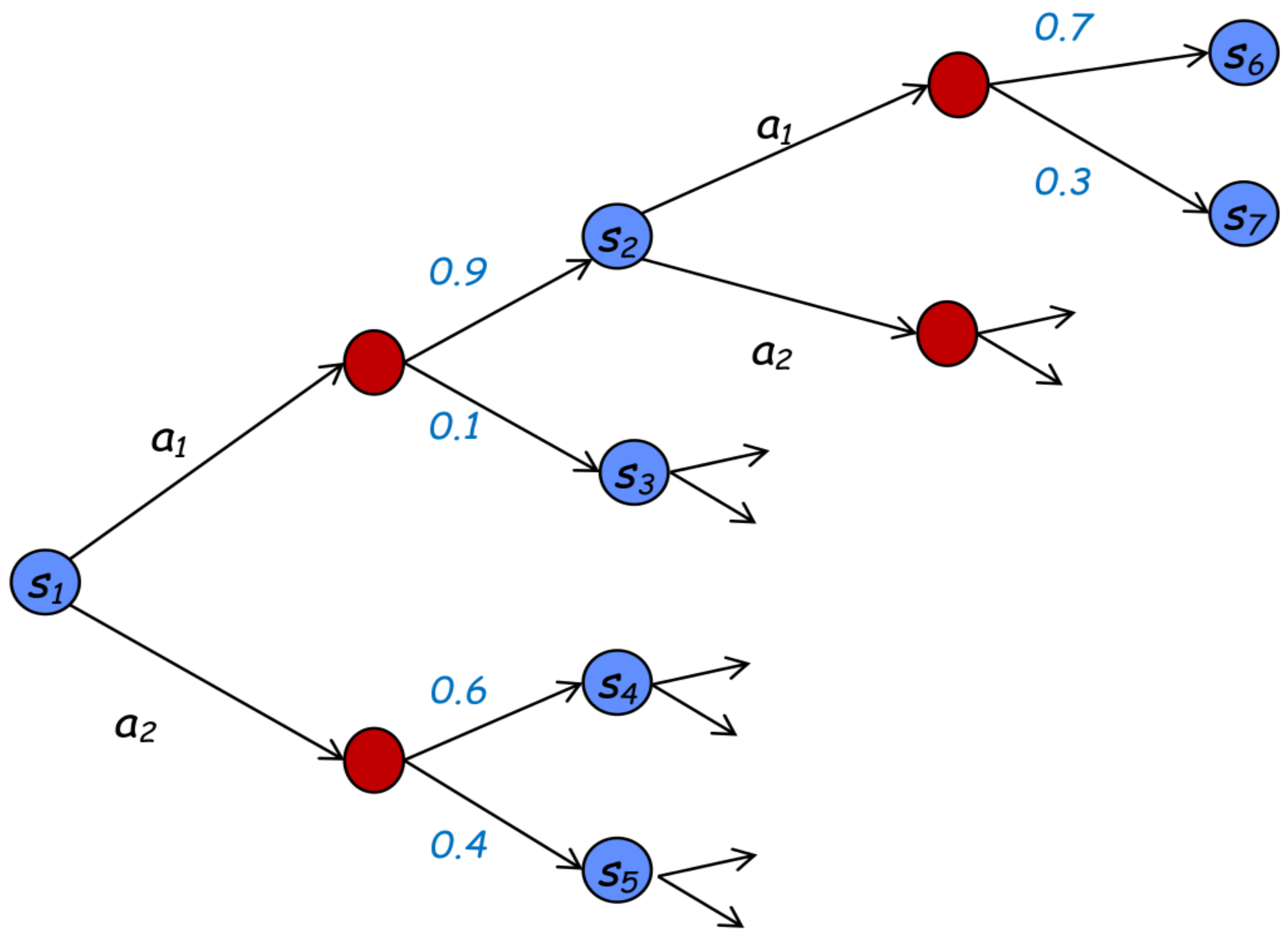
- Classical Planning: $\langle S, s_0, S_G, A, f, c \rangle$
  - states, initial state, goal state(s)
  - actions
  - transition function $f: S \times A \to S$
  - cost function

- Probabilistic Planning
  - probabilistic transition function $P: S \times A \times S \to [0,1]$

$$\sum_{s' \in S} P(s, a, s') = 1$$

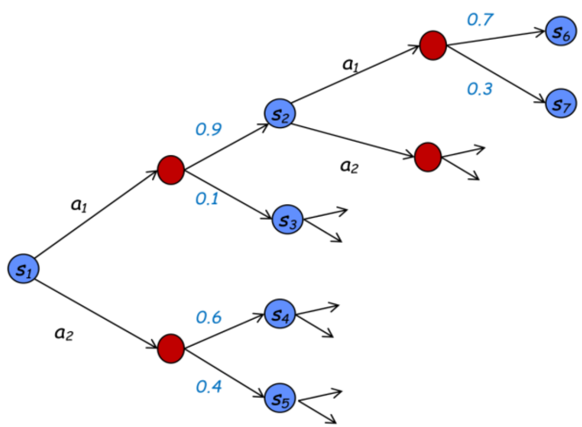
Q: why is this enough for modelling uncertainty in environment?

Probabilistic Planning - Visualization



Probabilistic Planning - Solution

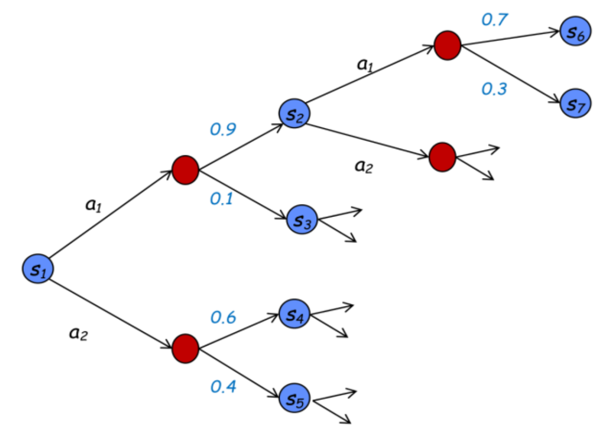
- what is the solution in classical planning?
  - sequence of (partially) ordered actions leading from initial state to the goal state



- this is not sufficient in the probabilistic case
  - what if the plan fails?
- we need **a contingency plan (policy)**
  - typically assumes *k* failures
  - if the number of failures is unbounded → policy

Probabilistic Planning - Solution

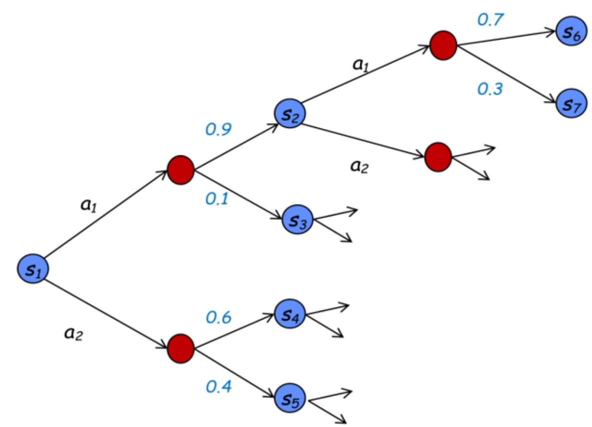
- in general we seek for a probabilistic history-dependent policy
  - $\pi: H \times A \rightarrow [0,1]$
  - where $h = s_1 a_1 s_2 a_2 \ldots s_t$
  - note that the policy may prescribe randomization over actions



- now we have a representation for plans (policy)
  - we need a method for plan evaluation

Probabilistic Planning - Evaluation

- costs are assigned to triplets (s, a, s')
- typically termed rewards (i.e., positive sense)
- executing a policy yields a sequence of rewards



- policy value – linear additive utility
  - $u(R_1, R_2, \ldots) = R_1 + \gamma R_2 + \gamma^2 R_3 + \cdots$
  - $u(\pi(s_0)) = E[u(R_1, \ldots)]$
- expected utility – what can happen?
  - optimal only for risk-neutral agent

Probabilistic Planning – Optimal Solution

- If the quality of every policy can be measured by its expected linear additive utility, there is a policy that is optimal at every time step.

(Stated in various forms by Bellman, Denardo, and others)

- we seek for π^* s.t. $u(\pi^*) \geq u(\pi)$ for all other policies π

- note: can be the case that the policy cannot be measured by expected linear additive utility?
  - yes (infinite state-space with non-discounted rewards, dead-ends, ...)

Probabilistic Planning – Algorithms

- this lecture
  - using classical planning to probabilistic planning
  - straightforward approach (FF-replan)
  - improved approach (Robust FF)

- next lectures
  - algorithms that directly use probability and uncertainty
    - formal definition MDP, strategy/policy iteration
    - current approaches for solving MDPs
  - uncertainty in observations
    - formal definition and current approaches for solving POMDPs

Probabilistic Planning – First Approach

- 2004 – first international probabilistic planning competition
- several participants, mainly based on MDP solvers
- winner?
  - FF-Replan
  - possibly the simplest algorithm you can think of …

FF-Replan

- outline of the algorithm

1. determinize the input domain (remove all probabilistic information from the problem)
2. synthesize a plan
3. execute the plan
4. should an unexpected state occur, replan

FF-Replan - Determinization

- what information can be discarded?
- two main heuristics
  - keep only one from all probabilistic outcomes of an action in a state (e.g., using the outcome with the highest probability)
  - keep all outcomes
    - generate a separate action for each possible outcome
- very simple, not sound, not optimal, but still good enough for simple domains
  - Q: In which cases you should adopt such techniques?
  - (outperformed also all participants in IPPC-06)

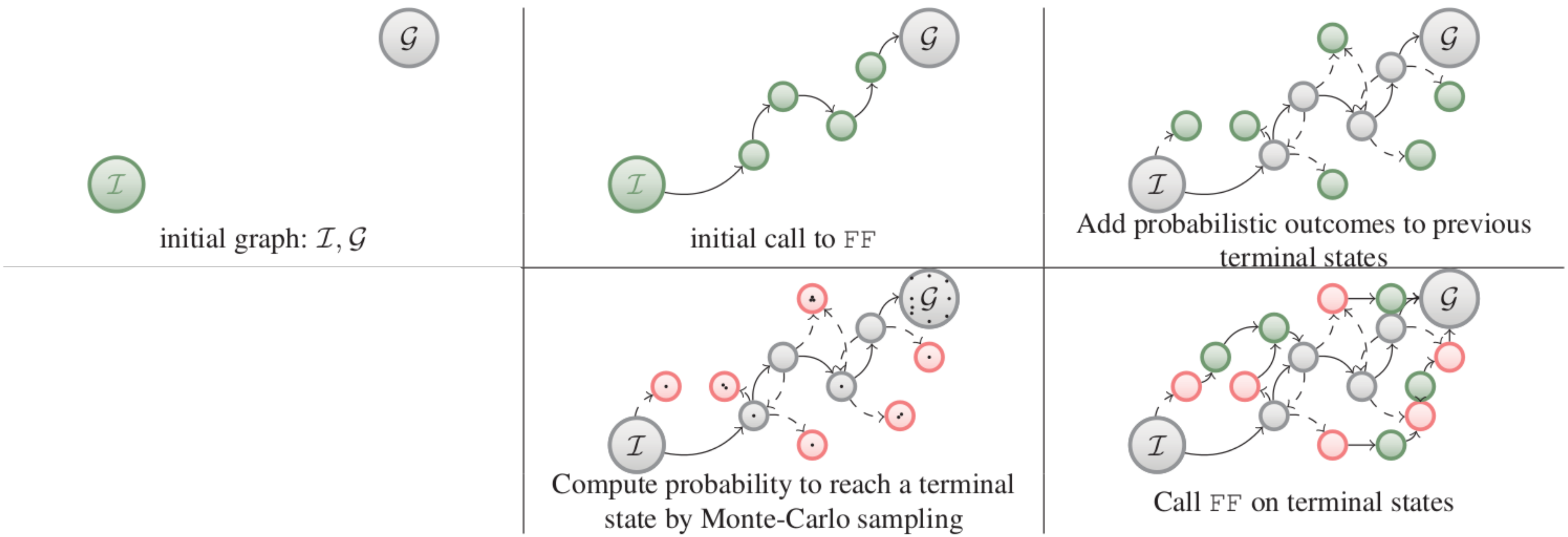
Probabilistic Planning (2)

- winner of IPPC 2008
  - Robust-FF
    - (Incremental Plan Aggregation for Generating Policies in MDPs, Konigsbuch, Kuter, Infantes 2010)
  - generalizes FF-Replan

1. determinize the problem
2. use classical planner to find partial plans
3. aggregate these plans into the partial policy
4. continue until the probability of replanning is below given threshold

Robust-FF

- outline of the algorithm



Robust-FF

- pseudocode of the algorithm

Algorithm 1: $\texttt{RFF}(M, s_0, G, \rho, N)$

$\mathcal{D} \leftarrow$ a deterministic relaxation of M
$T \leftarrow \{s_0\}$; $\pi \leftarrow \emptyset$; $\omega(s_0, \pi, s_0) \leftarrow 1$
repeat
    $T' \leftarrow \emptyset$ `// new terminal states`
    $X \leftarrow \emptyset$ `// new expanded states`
    for $s \in T$ *such that* $\omega(s_0, \pi, s) > \rho$ **do**
        pick $G_{\texttt{FF}} \subseteq G \cup S_\pi$
        $p \leftarrow \texttt{FF}(\mathcal{D}, s, G_{\texttt{FF}})$
        if $p \neq failure$ **then**
            $s' \leftarrow s$; let $p = \langle \hat{a}_1, \ldots, \hat{a}_k \rangle$
            for $1 \leqslant i \leqslant k$ **do**
                $X \leftarrow X \cup \{s'\}$
                $\pi(s') \leftarrow a_i$
                $T' \leftarrow T' \cup succ(s', a_i) \setminus (S_\pi \cup G)$
                $s' \leftarrow succ_{\mathcal{D}}(s', \hat{a}_i)$
        else $X \leftarrow X \cup \{s\}$
    $T \leftarrow (T \setminus X) \cup T'$
    $\{\omega(s_0, \pi, s) \mid s \in T)\} \leftarrow \texttt{Fail\_Prob}(s_0, \pi, T, N)$
    $\Omega(s_0, \pi) = \sum_{s \in T} \omega(s_0, \pi, s)$
    `// Next line is optional`
    Optimize the shortest stochastic path in S_π by considering all states in T as if they were unsolvable
until $\Omega(s_0, \pi) \leqslant \rho$ **or** $T = \emptyset$
if $\pi \neq \emptyset$ **then return** π
else return $failure$

Robust-FF

- number of options
  - selecting determinization (most probable, all outcomes)
  - selecting goals (only problem goals, random goals, best goals)
    - random/best goals – include also expanded states into G_{FF}; either k random, or k "best ones"
  - calculating probability of reaching terminal states (dynamic programming, Monte Carlo simulations)

- soundness vs. completeness of the algorithm?
  - only with selected methods (RFF_{AO})

- not (approximately) optimal in general

FF-Hindsight

- Approximate the value of a state
  - sample a set of determinized problems originating from a state
  - then solve these problems and combine their values

- Optimal value function
  - $$V^*(s,T) = \max_{\pi} \boldsymbol{E}[R(s,F,\pi)]$$
  - from state s, horizon T, policy π, random variable F, reward function R

- HOP value approximation
  - $$V^*(s,T) = \boldsymbol{E}[\max_{\pi} R(s,F,\pi)]$$

Robust-FF – Towards UCT

- incrementally builds the search space
- adds only such states and actions that lead to a goal
- the process of space-expansion does not guarantee optimality
- this was achieved by using theoretic results addressing the problem of exploration vs. exploitation
- In IPPC-12, the winner (and most of the other competitors) was based on UCT (Upper Confidence bounds applied on Trees)