IPC 2004
Probabilistic Planning Track

Michael L. Littman
Rutgers University

Håkan L. S. Younes
Carnegie Mellon University

Goal

- Provide shared benchmarks and evaluation metrics for the MDP and probabilistic planning communities

Domain Model Restrictions

- Discrete time
- Finite state space
- Full observability

Outline

- Input language
- Competition problems
- Plan representation
- Planner evaluation

Input Language

- PDDL 2.1 level 1 + probabilistic effects
- Rewards encoded using fluents
    - No numeric preconditions!

Stochastic Actions

- Variation of factored probabilistic STRIPS operators [Dearden & Boutilier 97]
- An action consists of a precondition ϕ and a consequence set $C = \{c_1, \ldots, c_n\}$
- Each c_i has a trigger condition ϕ_i and an effects list $E_i = \langle p_1^i, E_1^i; \ldots; p_k^i, E_k^i \rangle$
  - $\sum_j p_j = 1$ for each E_i

Stochastic Actions: Semantics

- An action is enabled in a state s if its precondition ϕ holds in s
- Executing a disabled action should be allowed, but does not change the state
    - Different from deterministic PDDL
    - Motivation: partial observability
    - Precondition becomes factored trigger condition

Stochastic Actions: Semantics (cont.)

- When applying an enabled action to s:
  - Select an effect set for each consequence with enabled trigger condition
  - The combined effects of the selected effect sets are applied atomically to s
  - Unique next state if consequences with mutually consistent trigger conditions have commutative effect sets

Syntax of Probabilistic Effects

```
<effect>      ::= <d-effect>
<effect>      ::= (and <effect>*)
<effect>      ::= (forall (<typed list(variable)>) <effect>)
<effect>      ::= (when <GD> <d-effect>)
<d-effect>    ::= (probabilistic <prob-eff>+)
<d-effect>    ::= <a-effect>
<prob-eff>    ::= <probability> <a-effect>
<a-effect>    ::= (and <p-effect>*)
<a-effect>    ::= <p-effect>
<p-effect>    ::= (not <atomic formula(term)>)
<p-effect>    ::= <atomic formula(term)>
<p-effect>    ::= (<assign-op> <f-head> <f-exp>)
<probability> ::= Any rational number in the interval [0, 1]
```

Correspondence to Components of Stochastic Actions

- Effects list:

  (probabilistic p_1^i E_1^i … p_k^i E_k^i)

- Consequence:

  (when ϕ (probabilistic p_1^i E_1^i … p_k^i E_k^i))

Stochastic Actions: Example

```
(:action stack
        :parameters (?x ?y)
        :precondition (and (holding ?x) (clear ?y))
        :effect (and (not (holding ?x)) (clear ?x) (handempty)
                    (probabilistic 0.95 (and (not (clear ?y)) (on ?x ?y))
                                   0.05 (ontable ?x))))
```

Rewards and Goals

- Rewards encoded using fluents
  - :effect (increase (reward) 100)
  - (:metric maximize (reward))
- Meaning of goals
  - (:goal ϕ) means maximize probability of achieving ϕ

Competition Problems

- Goal directed problems
  - Fuzzy blocks world and logistics domains
- Reward directed problems
  - Classical MDP type problems

Domain with Rewards

```
(define (domain tiger-domain)
    (:requirements :negative-preconditions :conditional-effects :probabilistic-effects :rewards)
    (:predicates (tiger-on-left) (hear-tiger-on-left))
    (:action  listen
              :effect (and  (when (tiger-on-left)
                                  (probabilistic 0.85 (hear-tiger-on-left)
                                                 0.15 (not (hear-tiger-on-left))))
                            (when (not (tiger-on-left))
                                  (probabilistic 0.85 (not (hear-tiger-on-left))
                                                 0.15 (hear-tiger-on-left)))))
    (:action  open-left-door
              :effect (and  (when (not (tiger-on-left)) (increase (reward) 100))
                            (when (tiger-on-left) (decrease (reward) 100))))
    (:action open-right-door
              :effect (and  (when (tiger-on-left) (increase (reward) 100))
                            (when (not (tiger-on-left)) (decrease (reward) 100)))))
```

MDP Planning Problem

```
(define (problem tiger-problem)
    (:domain tiger-domain)
    (:init (probabilistic 0.5 (tiger-on-left)))
    (:metric maximize (reward))
```

Plan Representation

- No explicit plan representation
  - Up to each individual planner
- Planner communicates with evaluator
  - Evaluator sends state updates to planner
  - Planner sends actions choices to evaluator

Distinction between planner and executor blurred

Planner Evaluation

- Sampling-based planner evaluation
  - Simulate execution for some time and accumulate reward
  - Take average accumulated reward over multiple sample executions
- Three speed categories (preliminary)
  - Real-time, intermediate, deliberative
  - Variation in time allowed per sample

Evaluation Issues

- How many samples?
- How much time per sample?
- Require replanning for each sample, or allot initial computation time?

Possible Subtracks

- Non-deterministic planning
  - Treat probabilistic effects as disjunctive effects
- Learning
  - Generalize from smaller problem instances

Resources

- On the web:

  http://www.cs.rutgers.edu/~mlittman/topics/ipc04-pt.html

- Mailing lists [mlittman@cs.rutgers.edu]:
  - probplan-panel (discussion the design of the competition)
  - probplan-announce (general announcements)