PDDL by Example

Manuela Veloso

Carnegie Mellon University
Computer Science Department

Planning, Execution, and Learning
Fall 2002

PDDL - Description Language

- A representation of Strips language
- A single language with all the extensions done to Strips
- Several levels, including more recently temporal constraints
- Level 1 is fine.

PDDL - Plain STRIPS Domain

```
(define (domain gripper-strips)
  (:predicates (room ?r)
               (ball ?b)
               (gripper ?g)
               (at-robby ?r)
               (at ?b ?r)
               (free ?g)
               (carry ?o ?g))
```

PDDL - Plain STRIPS Domain

```
(:action move
 :parameters (?from ?to)
 :precondition (and (room ?from) (room ?to) (at-robby ?from))
 :effect (and (at-robby ?to)
              (not (at-robby ?from))))
(:action pick
 :parameters (?obj ?room ?gripper)
 :precondition (and (ball ?obj) (room ?room) (gripper ?gripper)
                    (at ?obj ?room) (at-robby ?room) (free ?gripper)
 :effect (and (carry ?obj ?gripper) (not (at ?obj ?room))
              (not (free ?gripper))))
(:action drop
 :parameters (?obj ?room ?gripper)
 :precondition (and (ball ?obj) (room ?room) (gripper ?gripper)
                    (carry ?obj ?gripper) (at-robby ?room))
 :effect (and (at ?obj ?room) (free ?gripper)
              (not (carry ?obj ?gripper)))))
```

PDDL - Plain STRIPS Domain

```
(define (domain gripper-strips)
  (:predicates (room ?r)  (ball ?b) (gripper ?g) (at-robby ?r)
               (at ?b ?r) (free ?g) (carry ?o ?g))
  (:action move
   :parameters (?from ?to)
   :precondition (and  (room ?from) (room ?to) (at-robby ?from))
   :effect (and (at-robby ?to) (not (at-robby ?from))))
  (:action pick
   :parameters (?obj ?room ?gripper)
   :precondition (and (ball ?obj) (room ?room) (gripper ?gripper)
                      (at ?obj ?room) (at-robby ?room) (free ?gripper))
   :effect (and (carry ?obj ?gripper) (not (at ?obj ?room))
                (not (free ?gripper))))
  (:action drop
   :parameters (?obj ?room ?gripper)
   :precondition (and (ball ?obj) (room ?room) (gripper ?gripper)
                      (carry ?obj ?gripper) (at-robby ?room))
   :effect (and (at ?obj ?room) (free ?gripper)
                (not (carry ?obj ?gripper)))))
```

PDDL - Plain STRIPS Problem

```
(define (problem strips-gripper2)
  (:domain gripper-strips)
  (:objects rooma roomb ball1 ball2 left right)
  (:init (room rooma)
         (room roomb)
         (ball ball1)
         (ball ball2)
         (gripper left)
         (gripper right)
         (at-robby rooma)
         (free left)
         (free right)
         (at ball1 rooma)
         (at ball2 rooma))
  (:goal (at ball1 roomb)))
```

PDDL - STRIPS + Functions

```
(define (domain hanoi-domain)
  (:requirements :equality)
  (:predicates (disk ?x) (smaller ?x ?y) (on ?x ?y) (clear ?x))
  (:action move-disk
   :parameters (?disk ?below-disk ?new-below-disk)
   :precondition
     (and (disk ?disk)
          (smaller ?disk ?new-below-disk)
          (not (= ?new-below-disk ?below-disk))
          (not (= ?new-below-disk ?disk))
          (not (= ?below-disk ?disk))
          (on ?disk ?below-disk)
          (clear ?disk)
          (clear ?new-below-disk))
   :effect
     (and (clear ?below-disk)
          (on ?disk ?new-below-disk)
          (not (on ?disk ?below-disk))
          (not (clear ?new-below-disk)))))
```

PDDL - Typed Variables

```
(define (domain gripper-typed)
   (:requirements :typing)
   (:types room ball gripper)
   (:constants left right - gripper)
   (:predicates (at-robby ?r - room)
                (at ?b - ball ?r - room)
                (free ?g - gripper)
                (carry ?o - ball ?g - gripper))
   (:action move
      :parameters  (?from ?to - room)
      :precondition (at-robby ?from)
      :effect (and  (at-robby ?to)
                    (not (at-robby ?from))))
   (:action pick
      :parameters (?obj - ball ?room - room ?gripper - gripper)
      :precondition  (and  (at ?obj ?room) (at-robby ?room) (free ?gripper))
      :effect (and (carry ?obj ?gripper)
                   (not (at ?obj ?room))
                   (not (free ?gripper))))
```

PDDL - Type Hierarchy

```
(define (domain logistics-adl)
  (:requirements :adl :domain-axioms)
  (:types physobj - object
          obj vehicle - phssobj
          truck airplane - vehicle
          location city - object
          airport - location)
  (:predicates (at ?x - physobj ?l - location)
               (in ?x - obj ?t - vehicle)
               (in-city ?l - location ?c - city))
```

PDDL - Conditional Effects

```
(:action drive-truck
 :parameters (?truck - truck ?loc-from ?loc-to - location
              ?city - city)
 :precondition (and (at ?truck ?loc-from)
                    (in-city ?loc-from ?city)
                    (in-city ?loc-to ?city))
 :effect  (and (at ?truck ?loc-to)
          (not (at ?truck ?loc-from))
          (forall (?x - obj)
              (when (and (in ?x ?truck))
                    (and (not (at ?x ?loc-from))
                         (at ?x ?loc-to))))))
```