12 | STRIPS

Verónica Esther Arriola Ríos

En este proyecto implementarás una pequeña máquina de inferencias utilizando los conocimientos de la primera parte del curso.

Meta

El alumno experimentará de primera mano cómo programar los elementos esenciales de una máquina de inferencias para un sistema experto.

Objetivos

- Que el alumno se familiarice con el uso del lenguaje de descripción PDDL para expresar los pormenores de un problema de planeación.
- Familiarizarse con el uso del diseño orientado a objetos en el código, para facilitar la implementación de los algoritmos que resolverán el problema de planeación.
- Que el alumno implemente un algoritmo de búsqueda que trabaje sobre objetos en un dominio PDDL.
- Que el alumno valore la relevancia del algoritmo de unificación, en un caso sencillo, para la identificación de acciones aplicables a un estado del dominio.

Desarrollo

En este paquete se te entrega:

- Una copia de la gramática BNF del lenguaje para descripción de problemas de planeación PDDL. No utilizarás todas las opciones que se incluyen, así que sólo revísala para que te des una idea de lo que se puede hacer.

- Un script de Python con la definición de las clases que representarán a los objetos en el dominio, hechos, acciones posibles, estado inicial y meta de un problema de planeación. Lee con cuidado la documentación. Estas clases corresponden a los elementos de PDDL que vas a utilizar. Observa que las clases listan los atributos, pero esta representación es independiente de la notación con paréntesis de PDDL.

Escenario de prueba

El programa que vas a realizar deberá poder resolver un pequeño problema de planeación, definido según la notación de PDDL. El dominio siguiente se puede utilizar para probarlo.

```
;; Especificación en PDDL1 de un dominio DWR simplificado

(define (domain platform-worker-robot)
  (:requirements :strips :typing)
  (:types
    contenedor
    pila            ;tiene una base y una pila de contenedores.
    grúa            ;sostiene máximo un contenedor.
  )
  (:predicates
    (sostiene ?k - grúa ?c - contenedor) ;la grúa ?k sostiene al
       ↪ contenedor ?c
    (libre ?k - grúa)                    ;la grúa ?k está libre
    (en ?c - contenedor ?p - pila)       ;el contenedor ?c está en
       ↪ la pila ?p
    (hasta_arriba ?c - contenedor ?p - pila) ;el contenedor ?c se
       ↪ encuentra hasta arriba de la pila ?p
    (sobre ?k1 - contenedor ?k2 - contenedor);el contenedor ?k1 está
       ↪  sobre el contenedor ?k2
  )
  ;; toma un contenedor de una pila con la grúa
  (:action toma
     :parameters (?k - grúa ?c - contenedor ?p - pila)
     :vars (?otro - contenedor)           ;variables locales (extra
        ↪ a PDDL)
     :precondition (and (libre ?k) (en ?c ?p)
                        (hasta_arriba ?c ?p) (sobre ?c ?otro))
     :effect (and (sostiene ?k ?c) (hasta_arriba ?otro ?p)
                  (not (en ?c ?p)) (not (hasta_arriba ?c ?p))
                  (not (sobre ?c ?otro)) (not (libre ?k))))

  ;; pone al contenedor sostenido por la grúa en una pila
  (:action pon
     :parameters (?k - grúa ?c - contenedor ?p - pila)
     :vars (?otro - contenedor)           ;variables locales
     :precondition (and (sostiene ?k ?c) (hasta_arriba ?otro ?p))
     :effect (and (en ?c ?p) (hasta_arriba ?c ?p) (sobre ?c ?otro)
```

```
                    (not (hasta_arriba ?otro ?p)) (not (sostiene ?k ?c)
                       ↪ )
                    (libre ?k))))
)
```

A continuación se incluye un ejemplar de problema. Puedes diseñar otros tú y verificar que tu programa también los resuelva correctamente.

```
;; un problema sencillo del dominio DWR simplificado
(define (problem dwrpb1)
  (:domain platform-worker-robot)

  (:objects
   k1 k2 - grúa
   p1 q1 p2 q2 - pila
   ca cb cc cd ce cf pallet - contenedor)

  (:init
   (en ca p1)
   (en cb p1)
   (en cc p1)

   (en cd q1)
   (en ce q1)
   (en cf q1)

   (sobre ca pallet)
   (sobre cb ca)
   (sobre cc cb)

   (sobre cd pallet)
   (sobre ce cd)
   (sobre cf ce)

   (hasta_arriba cc p1)
   (hasta_arriba cf q1)
   (hasta_arriba pallet p2)
   (hasta_arriba pallet q2)

   (libre k1)
   (libre k2))

   ;; el problema consiste en mover todos contenedores
   ;; ca y cc a la pila p2, los demás a q2
   (:goal
     (and (en ca p2)
          (en cb q2)
          (en cc p2)
          (en cd q2)
          (en ce q2)
          (en cf q2))
    )
```

```
45 )
```

Tareas a realizar

Las tareas que debes realizar son las siguientes:

Parte I

1. Dibuja el estado inicial. Puedes incluirlo como comentario en tu código como arte ASCII o adjuntar un archivo de imagen (no debe ser un archivo pesado).
2. Crea a mano los objetos correspondientes al dominio y problema anteriores. Imprímelos y observa que todo esté correcto.
3. Observa que el estado del mundo se representa como una lista de predictados aterrizados. Agrega el código necesario (clases o funciones) para que, dado un estado determine si una acción es aplicable o no y con qué sustitución (qué objeto se asigna a qué variable).
4. Agrega el código para determinar si un estado satisface las condiciones indicadas en el campo *meta*.
5. Incluye código con pruebas que muestren que tus implementaciones son correctas.

Parte II

6. Ahora agrega el código necesario para realizar una búsqueda en amplitud de la secuencia de acciones que se debe seguir para alcanzar la meta. Haz que imprima lo que está haciendo de modo que puedas ver si se comporta como deseas.
7. Utiliza tu programa para buscar una solución al problema anterior. ¿Es suficiente información? ¿Cómo se comporta tu programa?