Práctica 2. Planificación

Objetivos:

- Conocer el formato PDDL. Conocer la funcionalidad de un planificador.
- Ejemplo ZenoTravel (dominio + problema). Evaluar, conocer y modificar.
- Diseñar un nuevo dominio + problema en PDDL y aplicarlo.

Poliformat:

- Artículos sobre lenguaje PDDL 2.1, 1.2
- Ejemplo de Dominio y Problemas: ZenoTravel
- Boletín Practica. Incluye Enunciado problema a resolver.
- Presentación Práctica
- Planificador (lpg para ejecución en Windows + cygwin1.dll).

Estructura de PDDL: Dominio

Se proporcionan varios ejemplos de dominios

(define (domain <name>)

(:requirements <:req 1>… <:req n>) ; requisitos necesarios para entender el dominio

(:types <subtype1>… <subtype n> – <type1> <typen>) ; Tipos a usar. Espacios entre –

(:constants <cons1> … <cons n>) ; definición de constantes (si se van a usar)

; Info sobre el estado del problema

(:predicates <p1> <p2>… <p n>) ; definición de predicados (info proposicional)

(:functions <f1> <f2>… <fn>) ; definición de funciones (info numérica)

; ---------- Acciones ----------

(:durative-action 1 *; acción con duración*

:parameters (?par1 – <subtype1> ?par2 – <subtype2> ...)

:duration <value>

:condition (*[and]* <condition$_1$>… <condition$_n$>) ; *"at start", "over all", "at end"*

:effect (*[and]* <effect$_1$>… <effect$_n$>) ; *"at start", "at end"*

)

……

(**:durative-action** n) ; o **:action** en el caso de acciones sin duración

)

EJEMPLO

Objetos: personas, aviones y ciudades. Las personas pueden embarcar/desembarcar en/de los aviones, que vuelan entre distintas ciudades.
Existen dos formas de volar, una rápida y una lenta con distintos consumos de combustible.

(define (domain zeno-travel)

(:requirements :durative-actions :typing :fluents)

(:types aircraft person city - object) ***; tres tipos de objetos: avión, persona y ciudad***

(:predicates (at ?x - (either person aircraft) ?c - city) ***; en qué ciudad está una persona o avión***
--- ***; Otros predicados…..***

(:functions (fuel ?a - aircraft) ***; función numérica: nivel de combustible de un avión***
(distance ?c1 - city ?c2 - city) ***; función numérica: distancia entre 2 ciudades***
(boarding-time) ***; función numérica: tiempo que se tarda en embarcar.***
--- ***; Otras funciones……***

; A continuación vienen las acciones

(:durative-action board ***; acción de embarcar***

:parameters (?p - person ?a – aircraft ?c - city) ***; hay 3 parámetros: persona, avión y ciudad***

:duration (= ?duration (boarding-time)) ***; la duración viene dada por la función "boarding-time"***

:condition (and (at start (at ?p ?c)) ***; dos condiciones: al principio de la acción ("at start")***
(over all (at ?a ?c))) ; ***la persona tiene que estar en la ciudad; y durante toda la ejecución***
; Condicion ("over all"): el avión debe permanecer en la ciudad

:effect ***; se generan dos efectos***
(and (at start (not (at ?p ?c))) ***; al principio de la acción ("at start") la persona deja de estar en la ciudad;***
(at end (in ?p ?a)))) ; ***al final de la acción ("at end") la persona pasa a estar dentro del avión.***

Estructura de PDDL: PROBLEMA

Se proporcionan varios ejemplos de problemas sobre cada dominio

(define (problem <name>)

(:domain <name >) *; nombre del dominio al que pertenece este problema*

(:objects <obj$_1$> - <type$_1$> … <obj$_n$> - <type$_n$>) *; **objetos y sus tipos.*** Espacios entre –

(:init *; **estado inicial***

(<predicate$_1$>) … (<predicate$_i$>) *; parte proposicional. Sin and.*

(= <function$_1$> <value$_1$>) … (= <function$_n$> <value$_n$>)) *; parte numérica (TODAS)*

(:goal *; **objetivos***

(and ((<predicate$_1$>) … (<predicate$_i$>) *; objetivos proposicionales, con and*

(<operator$_1$> <function$_1$> <value$_1$>) …

(<operator$_j$> <function$_j$> <value$_j$>))) *; objetivos numéricos*

(:metric minimize|maximize <expression>) *; opcional, métrica a min/maximizar*

*; que representa la **calidad** del plan*

)

Zenotravel: Dominio y Problemas en Poliformat

```
(define (problem ZTRAVEL-1-2)     ;nombre del problema
(:domain  zeno-travel          ; nombre del dominio – debe corresponderse con el definido en el dominio
(:objects                plane1 – aircraft        ; objetos existentes en el problema
                         person1 - person
                                  …..)
(:init          ; estado inicial .  Todos los predicados que se cumplen y  valor de todas las funciones.
                (at plane1 city0)                ; el avión plane1 en city0 – información proposicional
                (= (slow-speed plane1) 198)      ; la velocidad lenta de plane1 – información numérica
                (= (distance city0 city0) 0)     ; distancias entre pares de ciudades…
                (= (distance city0 city1) 678)
                ……
                (= (total-fuel-used) 0)          ; valor inicial del combustible acumulado
                (= (boarding-time) 0.3)          ; duración necesaria para embarcar
                (= (debarking-time) 0.6)         ; duración necesaria para desembarcar
)

(:goal     (and                                  ; objetivos a conseguir
                (at plane1 city1)                ; plane1 tiene que acabar en city1 – objetivo proposicional
                (at person1 city0)               ; person1 tiene que acabar en city0
                (at person2 city2)               ; person2 tiene que acabar en city2
                (< (total-fuel-used) 300)        ; ejemplo de objetivo numérico
                ))

(:metric     minimize (+ (* 4 (total-time))  (* 0.005 (total-fuel-used))))    ; calidad (métrica) a optimizar
```

PLANIFICADOR *(ejecución desde terminal Windows (necesario cygwin1.dll)*

lpg-td (http://zeus.ing.unibs.it/lpg/)

- Búsqueda local heurística que utiliza grafos de planificación como base de estimaciones.
- ¡No determinista!

Ejecución: **lpg-td –o dominio.pddl –f problema.pddl –n 1** *(*soluciones*)*

Solución: **TIEMPO: (ACCION PARAMETROS) [DURACIÓN] [COSTE]**

0.0003: (**POP-UNITARYPIPE** S13 B1 A1 A3 B5 LCO OCA1 TA1-1-OCA1 TA3-1-LCO)
[**D**: 2.0000; **C**: 0.1000]
……..: …… *;acciones no necesariamente ordenadas en tiempo*

***NOTA**: Alternativamente a la opción "–n", también se puede utilizar la opción –speed o –quality.*

***–speed**: trata de encontrar una solución tan rápidamente como pueda.*

***–quality**: trata de encontrar una solución de buena calidad (aunque sin garantía de optimalidad).*

lpg-td-1.0 –o dominio.pddl –f problema.pddl –n 1
lpg-td-1.0 –o dominio.pddl –f problema.pddl –speed
lpg-td-1.0 –o dominio.pddl –f problema.pddl –quality

Plan Obtenido:

TIEMPO: (ACCION PARAMETROS) [DURACIÓN] [COSTE]

Duracion: *Duración de la acción.*

Coste: *Coste que supone la ejecución de la acción en la métrica definida para el problema.*

```
Solution number: 1
Total time:      0.03
Search time:     0.03
Actions:         21
Execution cost:  2.10
Duration:        130.182
Plan quality:    130.182
     Plan file:       plan_problema.pddl_1.SOL
```

Ejecución lpd-td (rover)

```
Parsing domain file:  domain 'ROVER' defined ... done.
Parsing problem file:  problem 'ROVERPROB1234' 



Modality: Incremental Planner

Number of actions            :      64
Number of conditional actions :       0
Number of facts              :      36


Analyzing Planning Problem:
        Temporal Planning Problem: YES
        Numeric Planning Problem: YES
        Problem with Timed Initial Litearals: N
        Problem with Derived Predicates: NO

Evaluation function weights:
     Action duration 1.00; Action cost 0.00


Computing mutex... done

Preprocessing total time: 0.00 seconds

Searching ('.' = every 50 search steps):
.. solution found:

Plan computed:
   Time: (ACTION) [action Duration; action Cost]
 0.0000: (SAMPLE_ROCK ROVER0 ROVER0STORE WAYPOINT3) [D:8.0000; C:0.1000]
 8.0000: (NAVIGATE ROVER0 WAYPOINT3 WAYPOINT1) [D:5.0000; C:0.1000]
 8.0000: (DROP ROVER0 ROVER0STORE) [D:1.0000; C:0.1000]
 13.0000: (NAVIGATE ROVER0 WAYPOINT1 WAYPOINT2) [D:5.0000; C:0.1000]
 18.0000: (SAMPLE_SOIL ROVER0 ROVER0STORE WAYPOINT2) [D:10.0000; C:0.1000]
 28.0000: (CALIBRATE ROVER0 CAMERA0 OBJECTIVE1 WAYPOINT2) [D:5.0000; C:0.1000]
 33.0000: (NAVIGATE ROVER0 WAYPOINT2 WAYPOINT1) [D:5.0000; C:0.1000]
 38.0000: (NAVIGATE ROVER0 WAYPOINT1 WAYPOINT3) [D:5.0000; C:0.1000]
 43.0000: (NAVIGATE ROVER0 WAYPOINT3 WAYPOINT0) [D:5.0000; C:0.1000]
 48.0000: (RECHARGE ROVER0 WAYPOINT0) [D:7.2727; C:0.1000]
 55.2727: (NAVIGATE ROVER0 WAYPOINT0 WAYPOINT3) [D:5.0000; C:0.1000]
 60.2727: (NAVIGATE ROVER0 WAYPOINT3 WAYPOINT1) [D:5.0000; C:0.1000]
 65.2727: (NAVIGATE ROVER0 WAYPOINT1 WAYPOINT3) [D:5.0000; C:0.1000]
 70.2727: (NAVIGATE ROVER0 WAYPOINT3 WAYPOINT0) [D:5.0000; C:0.1000]
 75.2727: (RECHARGE ROVER0 WAYPOINT0) [D:2.9091; C:0.1000]
 78.1818: (TAKE_IMAGE ROVER0 WAYPOINT0 OBJECTIVE1 CAMERA0 HIGH_RES) [D:7.0000; C
:0.1000]
 85.1818: (NAVIGATE ROVER0 WAYPOINT0 WAYPOINT3) [D:5.0000; C:0.1000]
 90.1818: (COMMUNICATE_IMAGE_DATA ROVER0 GENERAL OBJECTIVE1 HIGH_RES WAYPOINT3 W
AYPOINT0) [D:15.0000; C:0.1000]
 105.1818: (NAVIGATE ROVER0 WAYPOINT3 WAYPOINT1) [D:5.0000; C:0.1000]
 110.1818: (COMMUNICATE_SOIL_DATA ROVER0 GENERAL WAYPOINT2 WAYPOINT1 WAYPOINT0)
[D:10.0000; C:0.1000]
 120.1818: (COMMUNICATE_ROCK_DATA ROVER0 GENERAL WAYPOINT3 WAYPOINT1 WAYPOINT0)
[D:10.0000; C:0.1000]
```

```
Solution number: 1
Total time:      0.03
Search time:     0.03
Actions:         21
Execution cost:  2.10
Duration:        130.182
Plan quality:    130.182
     Plan file:       plan_problema.pddl_1.SOL
```

Plan quality (calidad del plan según la métrica definida en el problema (por defecto, duración del plan)

Evaluación

Práctica a realizar:

Planificación de transporte multimodal de mercancías

Evaluación:

- Realizad el ejercicio propuesto (se necesitará para el día de la evaluación, en el que se plantearán ampliaciones y/o modificaciones)

Calendario:

Sem	LABORATORIO	Evaluación
5-X	Planificación	
19-X	Planificación	
26-X		P2: Planificación

Aplicación y evaluación de Planificación (15%) P2

Notas (FAQ's)

- Inicializar todas las funciones **(fluents)** y estados de los objetos.
- Expresar bien las **condiciones** al inicio (START), al final (END) y OVERALL de las acciones durativas
- Expresar bien los **efectos** al inicio (START), al final (END) de las acciones durativas
- El planificador LPG no soporta ciertas precondiciones numéricas como "over all" (error: "PRECONDITION TYPE NOT HANDLED YET"). La alternativa es tratar esa precondición mediante predicados.
- En planificador LPG buscad solo unas pocas soluciones (por temas de ejecución). Preferiblemente, solo 1 **(-n 1).** Además, si se indican más podría NO PARAR!

  Se puede usar también la opción –speed o –quality
- LPG-td no es determinista
- Asegurarse de que el planificador, dominio y problema están en misma carpeta

Probad LPG y analizad los planes solución (y métricas) obtenidos