Tema 3: Planificación Clásica
Ejercicios

Sistemas Inteligentes - 3º

2018/2019

1. Transporte aereo de carga
   Tenemos dos cargas (C1 y C2) estan en 2 aeropuertos (SFO, JFK) y dos aviones (A1 y A2) para transportar dichas cargas, cada uno en un aeropuerto. El objetivo es que C1 acabe en JFK y C2 en SFO.

   Las acciones posibles en este caso son: (carga(c, av, aerop),descarga(c, av, aerop),volar(a, orig, dest)

   - Describir el estado inicial
   - Describir el objetivo:
   - Describir las acciones de cargar, descargar y volar.
   - Ejecutar el algoritmo STRIPS y dar varias soluciones.

Problema de Modelización en STRIPS: Gestión de equipajes en un aeropuerto

Se desea construir un sistema automático para gestionar el transporte de equipajes de una terminal de aeropuerto cuya planta puede verse en la Figura 1.

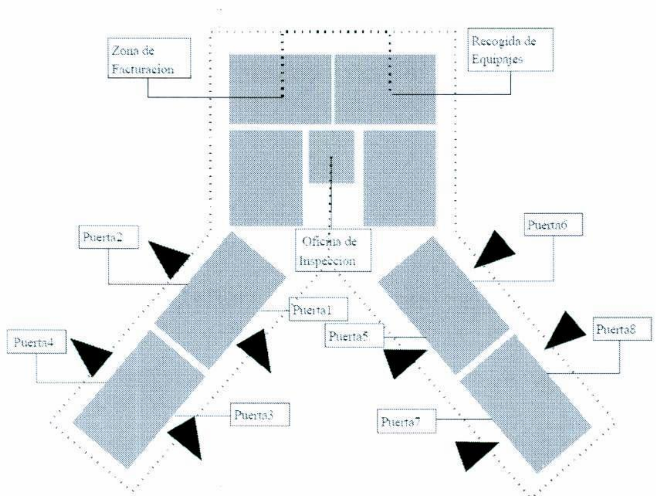


Figura 1 Planta de la terminal del aeropuerto

Las características del sistema son las siguientes:

- El transporte se realiza mediante convoyes. Estos están formados por dos elementos diferentes: la máquina y los vagones. La máquina es el elemento tractor y no lleva equipaje. Los vagones van enganchados a la máquina para poder moverse y se puede encadenar cualquier número de ellos. Cada vagón tiene una capacidad para 2 equipajes. Un vagón se puede encadenar y desencadenar en cualquier momento. Solo se puede cargar y descargar equipaje de vagones previamente encadenados a una máquina.
- Los convoyes solo pueden moverse a localizaciones conectadas por la línea de puntos. Para pasar de la puerta 2 a la oficina de inspección es necesario pasar por la puerta 4, por la puerta 3 y finalmente por la puerta 1.
- Tanto en la facturación de equipajes como en la llegada de un vuelo, los agentes de seguridad marcan ciertos equipajes como sospechosos. Estos equipajes obligatoriamente tienen que ser investigados en la oficina de inspección antes de ser llevados a su destino final. Una vez inspeccionados dejan de ser sospechosos y pasan a entregarse al destino final.

Se pide:

- Representar el dominio en STRIPS.
- Representar el siguiente problema: se han facturado 2 equipajes que tienen que ir a la puerta 4. Uno de ellos es sospechoso. Además, han llegado 2 equipajes a la puerta 7 y de nuevo uno de ellos es sospechoso. Los vehículos de transporte están distribuidos de la forma siguiente: tres vagones sueltos y descargados en la puerta 1, dos vagones sueltos y descargados en la puerta 5 y dos máquinas en la zona de recogida de equipajes.

INTELIGENCIA ARTIFICIAL   Plan 96

2º Parcial 2006

Ejercicio nº 2 (5 puntos) (60 minutos)

NO SE RESPONDERA NINGUNA PREGUNTA SOBRE EL EJERCICIO, LAS INTERPRETACIONES QUE DESE REALIZAR INDIQUELAS EN SU RESPUESTA.

Realizar un sistema basado en STRIPS para resolver el siguiente problema:

Un robot debe desplazarse desde un punto (x) a otro (y) y volver al lugar de partida (x), para ello debe pasar por un lugar intermedio (z), equidistante por igual de los dos puntos x e y. Para desplazarse debe hacer uso de carburante.

El robot dispone de un depósito que le posibilita el desplazamiento.

La capacidad del depósito posibilita recorrer la distancia existente entre el punto x e y.

El robot puede transportar, además del contenido de su depósito, un barril de carburante, la capacidad de este barril es la mitad de la que cabe en el depósito del robot.

En el estado inicial solo existen barriles llenos (en número ilimitado) en el punto de origen (x).

En el punto de origen el robot puede repostar hasta completar la capacidad de su depósito, sin ninguna limitación.

1. Descripción del estado inicial y final. **(0,5 puntos)**
2. Describir los operadores y predicados. **(1.5 puntos)**
3. Desarrollar el plan que de solución al problema. **(3 puntos)**

INTELIGENCIA ARTIFICIAL Plan 96

Junio 2010 Final

Ejercicio nº 2 (3 puntos) (30 minutos)

El Mars Exploration Rover necesita planificar un viaje de ida al Monte Olimpo, una montaña (M) situada en Marte, a 50 Km en línea recta del campo base (B) donde está situado inicialmente el Rover. Como puede verse en la figura, entre el campo base y la montaña existen 4 localizaciones intermedias que distan de B (y de M) 10, 20, 30 y 40 Km respectivamente.

El Rover puede viajar exactamente 10 Km cuando el depósito de combustible está completamente lleno. Adicionalmente, este puede transportar un máximo de 2 bidones de combustible en la bodega. La capacidad del depósito de combustible es exactamente la de un bidón. Inicialmente, el depósito está vacío y la bodega está descargada. Además, hay un bidón de combustible en B, tres en L_1, uno en L_2 y uno en L_4. El Rover puede realizar las siguientes acciones: (1) ir desde una localización a otra, que solamente podrá realizarse si el combustible existente en el depósito es suficiente para recorrer la distancia que separa ambas localizaciones; (2) recoger un bidón ubicado en una localización y almacenarlo en la bodega; (3) descargar en una localización un bidón almacenado en la bodega, (4) llenar el depósito de combustible del Rover en caso de que esté vacío. El repostaje puede realizarse únicamente en B, M, o en cualquiera de las localizaciones intermedias. Además, el Rover no puede utilizar un bidón almacenado en la bodega para repostar, ya que el Rover necesita que el bidón esté en tierra para poder manipularlo. Se asumirá que una vez se haya realizado el repostaje el bidón vacío será destruido automáticamente por el Rover, y que la única acción que consume combustible es la acción de desplazamiento.

Se pide:

(2.5 puntos) Representar el <u>dominio</u> de planificación descrito en el enunciado mediante STRIPS. Indicar brevemente el significado de cada predicado y operador mediante una frase en lenguaje natural.

(0.5 puntos) Plantear el <u>problema</u> de planificación especificado en el enunciado identificando claramente cada uno de sus componentes.

4.7.4 Supóngase un mundo en el que existen: R, robots de dos tipos, grandes y pequeños; O, objetos de dos tipos, pesados y ligeros; y H, habitaciones conectadas o no entre sí. En este mundo, los objetos pesados sólo pueden ser levantados por los robots grandes. Además, los robots no pueden llevar mas de un objeto. Se pide definir en STRIPS los operadores y predicados necesarios para resolver problemas del tipo de transportar objetos entre habitaciones.

4.7.5 Supóngase una fábrica en la que hay varios tipos de robots y varios tipos de objetos. Los robots pueden ser grandes y pequeños. Los robots pequeños pueden ser curvos o cuadrados. Los objetos pueden ser pesados y ligeros. Los objetos ligeros pueden ser cubos o cilindros. Las restricciones del problema son:

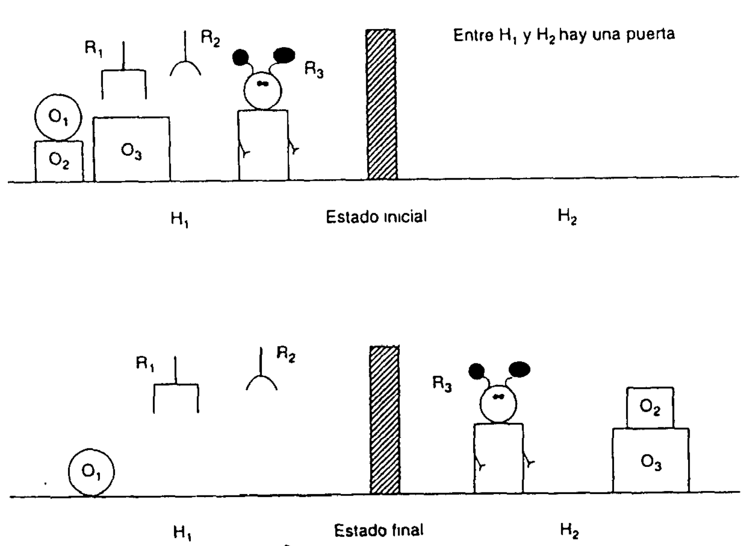
- Los objetos pesados sólo pueden ser cogidos por los robots grandes.
- Los cubos no pueden ser cogidos por los robots curvos, mientras que los cilindros no pueden ser cogidos por los robots cuadrados.
- Los robots pequeños no pueden moverse entre habitaciones, sólo pueden sujetar un objeto cada vez y pueden dar el objeto que lleven a un robot grande.
- Los robots grandes pueden llevar un objeto pesado y otro ligero, como máximo.
- Los objetos no pueden apilarse encima de otros objetos.

Se pide:

a) Describir el dominio de esta fábrica, con lodos sus componentes, para que sirva como entrada a STRIPS.
b) Suponiendo que se quita la última restricción, y se añade que los cilindros no pueden tener ningún objeto encima, ¿cuál sería la descripción del dominio?.
c) Dado el estado inicial y final, descritos en la figura 4.42, relativos al dominio descrito en el ejercicio (b), describir una(s) posible(s) soluciones) al problema, de acuerdo a STRIPS. En la figura, R3 es un robot grande, Rl y R2 son pequeños, 01 y 02 son objetos ligeros, y 03 es pesado.

FIGURA 4.42

Ejemplo de problema para el ejercicio 4.7.4



Cuestiones de autocomprobación

Hacer ejercicios de autocomprobación del libro Inteligencia Artificial (2), apartados 4.7.4, 4.7.5 y 4.7.6.

1. Supóngase un dominio de una fábrica de piezas para maquinaria pesada. En ella existen cuatro tipos de máquinas:

- Marcadoras: marcan un punto en una pieza donde se deben realizar las siguientes tareas. Necesita tener sujeta la pieza.
- Taladradoras helicoidales: hacen un agujero de forma helicoidal en una pieza que no sea de hierro. Necesitan tener sujeta la pieza y marcada la posición del agujero.
- Taladradoras cilíndricas: hacen un agujero de forma cilíndrica en una pieza. Necesitan tener sujeta la pieza y marcada la posición del agujero. Pueden ser grandes y pequeñas. Las grandes no pueden actuar sobre piezas pequeñas.
- Limpiadoras: todas las operaciones anteriores dejan las piezas sucias. Antes de realizar cualquier acción, las piezas deben estar limpias.

No hay piezas pequeñas de madera. Se pide describir este dominio para que puedan ser resueltos los problemas mediante STRIPS.

2. Supóngase un mundo de los bloques, con las siguientes características adicionales:

- Cada bloque tiene un color en cada cara, que puede ser blanco, negro, azul o rojo.
- Hay dos brazos de robot.
- Cada brazo de robot puede levantar un bloque por una cara C si la cara C está hacia arriba, libre, y el otro brazo está libre o tiene cogido un bloque por una cara C', tal que C y C' tienen el mismo color.
- Un bloque sólo puede estar encima de otro si las dos caras en contacto tienen el mismo color.
- Cualquiera de los dos brazos de robot que esté libre puede empujar por una cara a un bloque que esté encima de la mesa y libre, de forma que el bloque girará y la cara empujada pasará a estar hacia arriba.
- Cualquiera de los dos brazos de robot que esté libre puede pintar un bloque de cualquiera de los cuatro colores.

Se pide:

1. Modelizar el dominio completamente para que pueda ser utilizado por STRIPS. No se permite la utilización de ORs en la parte izquierda de los operadores ni ninguna formalización equivalente.
2. Representar el siguiente problema en la formalización escogida:
3. Estado inicial:
4. Un bloque A con una cara blanca hacia arriba, una cara negra (la opuesta a la cara blanca) y cuatro caras rojas. Está encima de la mesa.
5. Un bloque B con una cara roja hacia arriba, una cara blanca (la opuesta a la cara roja) y cuatro caras azules. Está encima del bloque A.
6. Un bloque C con una cara azul hacia arriba, una cara roja (la opuesta a la cara azul) y cuatro caras negras. Está encima del bloque B.
7. Estado final: el bloque A encima del bloque C.
8. Mostrar los cinco primeros pasos de la resolución del problema mediante STRIPS, mostrando el árbol generado, con todas las alternativas.

3\. Se pide establecer cuáles serían unos posibles valores de criticalidad del dominio del problema 1 de acuerdo a AB STRIPS.

4\. Se pide desarrollar el orden parcial que NOAH generaría para el problema de la **figura 1.**

Soluciones de autocomprobación

1. Supóngase un dominio de una fábrica de piezas para maquinaria pesada. En ella existen cuatro tipos de máquinas:

- Marcadoras: marcan un punto en una pieza donde se deben realizar las siguientes tareas. Necesita tener sujeta la pieza.
- Taladradoras helicoidales: hacen un agujero de forma helicoidal en una pieza que no sea de hierro. Necesitan tener sujeta la pieza y marcada la posición del agujero.
- Taladradoras cilíndricas: hacen un agujero de forma cilíndrica en una pieza. Necesitan tener sujeta la pieza y marcada la posición del agujero. Pueden ser grandes y pequeñas. Las grandes no pueden actuar sobre piezas pequeñas.
- Limpiadoras: todas las operaciones anteriores dejan las piezas sucias. Antes de realizar cualquier acción, las piezas deben estar limpias.

No hay piezas pequeñas de madera. Se pide describir este dominio para que puedan ser resueltos los problemas mediante STRIPS.

Se van a definir los siguientes predicados con sus significados:

- pieza (p): p es una pieza
- madera (p): p es una pieza de madera
- hierro (p): p es una pieza de hierro
- grande (p): p es una pieza grande
- pequeña (p): p es una pieza pequeña
- marcadora (m): m es una marcadora
- taladradora (m): m es una taladradora
- helicoidal (m): m es una taladradora helicoidal
- cilíndrica (m): m es una taladradora cilíndrica
- cilíndrica-grande (m): m es una taladradora cilíndrica grande
- cilíndrica-pequeña (m): m es una taladradora cilíndrica pequeña
- limpiadora (m): m es una limpiadora
- sujeta (p, m): p está sujeta en la máquina m
- libre-pieza (p): p está libre
- libre-máquina (p): p está libre
- limpia (p): p está limpia
- sucia (p): p está sucia
- punto (p, x, y): p tiene un punto (x, y)
- marcada (p, x, y): p está marcada en el punto (x, y)
- hueco (p, x, y): p tiene un hueco en el punto (x, y)
- hueco-helicoidal (p, x, y): p tiene un hueco helicoidal en el punto (x, y)
- hueco-cilíndrico-grande (p, x, y): p tiene un hueco cilindrico grande en el punto (x, y).
- hueco-cilíndrico-pequeño (p, x, y): p tiene un hueco cilindrico pequeño en el punto (x,y)

Los operadores serían, suponiendo que una vez que se actúa sobre una pieza, la máquina correspondiente la suelta.

sujetar (p, m)

precondiciones: pieza (p) ∧ libre-pieza (p) ∧ máquina (m) ∧ libre-máquina (m)
añadidos: sujeta (p, m)
borrados: libre-pieza (p) ∧ libre-máquina (m)

limpiar (p, m)

precondiciones: pieza (p) ∧ limpiadora (m) ∧ sujeta (p, m) ∧ sucia (p)
añadidos: limpia (p) ∧ libre-pieza (p) ∧ libre-máquina (m)
borrados: sucia (p) ∧ sujeta (p, m)

marcar (p, m, x, y)

precondiciones: pieza (p) ∧ marcadora (m) ∧ sujeta (p, m) ∧ punto (p, x, y) ∧ ~hueco(p,x,y)
añadidos: marcada (p, x, y) ∧ libre-pieza (p) ∧ libre-máquina (m) ∧ sucia (p)
borrados: sujeta (p, m) ∧ limpia (p)

taladrar-helicoidal (p, m, x, y)

precondiciones: madera (p) ∧ helicoidal (m) ∧ sujeta (p, m) ∧ punto (p, x, y) ∧ marcada (p, x, y)
añadidos: hueco-helicoidal (p, x, y) ∧ libre-pieza (p) ∧ libre-máquina (m) ∧ sucia (p) ∧ hueco (p, x, y)
borrados: sujeta (p, m) ∧ limpia (p) ∧ marcada (p, x, y)

taladrar-cilíndrico-grande (p, m, x, y)

precondiciones: grande (p) ∧ cilíndrica-grande (m) ∧ sujeta (p, m) ∧ punto (p, x, y) ∧ marcada (p, x, y)
añadidos: hueco-cilíndrico-grande (p, x, y) ∧ libre-pieza (p) ∧ libre-máquina (m) ∧ sucia (p) ∧ hueco (p, x, y)
borrados: sujeta (p, m) ∧ limpia (p) ∧ marcada (p, x, y)

taladrar-cilíndrico-pequeño (p, m, x, y)

precondiciones: pieza (p) ∧ cilíndrica-pequeña (m) ∧ sujeta (p, m) ∧ punto (p, x, y) ∧ marcada (p, x, y)
añadidos: hueco-cilíndrico-pequeño (p, x, y) ∧ libre-pieza (p) ∧ libre-máquina (m) ∧ sucia (p) ∧ hueco (p, x, y)
borrados: sujeta (p, m) ∧ limpia (p) ∧ marcada (p, x, y)

2. Supóngase un mundo de los bloques, con las siguientes características adicionales:

- Cada bloque tiene un color en cada cara, que puede ser blanco, negro, azul o rojo.
- Hay dos brazos de robot.
- Cada brazo de robot puede levantar un bloque por una cara C si la cara C está hacia arriba, libre, y el otro brazo está libre o tiene cogido un bloque por una cara C', tal que C y C' tienen el mismo color.
- Un bloque sólo puede estar encima de otro si las dos caras en contacto tienen el mismo color.

- Cualquiera de los dos brazos de robot que esté libre puede empujar por una cara a un bloque que esté encima de la mesa y libre, de forma que el bloque girará y la cara empujada pasará a estar hacia arriba.
- Cualquiera de los dos brazos de robot que esté libre puede pintar un bloque de cualquiera de los cuatro colores.

Se pide:

1. Modelizar el dominio completamente para que pueda ser utilizado por STRIPS. No se permite la utilización de ORs en la parte izquierda de los operadores ni ninguna formalización equivalente.
2. Representar el siguiente problema en la formalización escogida:
    - Estado inicial:
        - Un bloque A con una cara blanca hacia arriba, una cara negra (la opuesta a la cara blanca) y cuatro caras rojas. Está encima de la mesa.
        - Un bloque B con una cara roja hacia arriba, una cara blanca (la opuesta a la cara roja) y cuatro caras azules. Está encima del bloque A.
        - Un bloque C con una cara azul hacia arriba, una cara roja (la opuesta a la cara azul) y cuatro caras negras. Está encima del bloque B.
    - Estado final: el bloque A encima del bloque C.

3. Mostrar los cinco primeros pasos de la resolución del problema mediante STRIPS, mostrando el árbol generado, con todas las alternativas.

1. Lo que viene a continuación es, como siempre, una posible solución de las muchas válidas que habría. Los operadores necesarios serían los que aparecen seguidamente, mientras que los predicados son los que aparecen en los operadores y ninguno más.

Quitar-Otro-Libre (R, B, B')

preconditiones: encima (B, B') $\wedge$ libre (B) $\wedge$ brazo-libre (R) $\wedge$ brazo-libre (R') $\wedge$ R $\neq$ R'
añadidos: sujeto (R, B) $\wedge$ libre (B')
borrados: brazo-libre (R) $\wedge$ libre (B) $\wedge$ encima (B, B')

Quitar-Otro-Ocupado (R, B, B')

precondiciones: encima (B, B') $\wedge$ libre (B) $\wedge$ brazo-libre (R) $\wedge$ sujeto (R', B'') $\wedge$ cara-superior (B, Cara, C) $\wedge$ cara-superior (B'' , Cara', C)
añadidos: sujeto (R, B) $\wedge$ libre (B')
borrados: brazo-libre (R) $\wedge$ libre (B) $\wedge$ encima (B, B')

Levantar- Otro-Libre (R, B)

precondiciones: en-mesa (B) $\wedge$ libre (B) $\wedge$ brazo-libre (R) $\wedge$ brazo-libre (R') $\wedge$ R $\neq$ R'
añadidos: sujeto (R, B)
borrados: brazo-libre (R) $\wedge$ libre (B) $\wedge$ en-mesa (B)

Levantar-Otro-Ocupado (R, B)

precondiciones: en-mesa (B) ∧ libre (B) ∧ brazo-libre (R) ∧ sujeto (R', B') ∧ cara-superior (B, Cara, C) ∧ cara-superior (B', Cara', C)

añadidos: sujeto (R, B)

borrados: brazo-libre (R) ∧ libre (B) ∧ en-mesa (B)

Poner (R, B', B)

precondiciones: libre (B) ∧ sujeto (R, B') ∧ cara-superior (B, Cara, C) ∧ cara-inferior (B', Cara', C)

añadidos: brazo-libre (R) λ libre (B') λ encima (B', B)

borrados: sujeto (R, B') λ libre (B)

Dejar (R, B)

precondiciones: sujeto (R, B)

añadidos: brazo-libre (R) ∧ libre (B) ∧ en-mesa (B)

borrados: sujeto (R, B)

Empujar (R, B)

precondiciones: libre (B) ∧ en-mesa (B) ∧ brazo-libre (R) ∧ cara-superior (B, Cara, C) ∧ cara-inferior (B, Cara', C') ∧ cara-lateral (B, Cara'', C'') ∧ cara-lateral (B, Cara'' ', C''') ∧ cara-opuesta (B, Cara'', Cara'' ')

añadidos: cara-superior (B, Cara'', C'') ∧ cara-inferior (B, Cara''', C''') ∧ cara-lateral (B, Cara, C) ∧ cara-lateral (B, Cara', C')

borrados: cara-superior (B, Cara, C) ∧ cara-inferior (B, Cara', C') ∧ cara-lateral (B, Cara'', C'') ∧ cara-lateral (B, Cara''',C''')

Pintar (R, B, C)

precondiciones: color (C) ∧ brazo-libre (R) ∧ cara-superior (B, Cara, C')

añadidos: cara-superior (B, Cara, C)

borrados: cara-superior (B, Cara, C')

También se podría haber hecho lo mismo para todos los lados, sin más que añadir los predicados correspondientes en las precondiciones, lista de añadidos y lista de borrados.

2. Estado inicial (El):

• Referente al bloque A:

cara-superior (A, CaraA, Blanca) λ cara-inferior (A, CaralA, Negra) λ cara-lateral (A, Cara2A, Roja) λ cara-lateral (A, Cara3A, Roja) λ cara-lateral (A, Cara4A, Roja) λ cara-lateral (A, Cara5A, Roja) λ cara-opuesta (A, CaraA, CaralA) λ cara-opuesta (A, CaralA, CaraA) λ cara-opuesta (A, Cara2A, Cara3A) λ cara-opuesta (A, Cara3A, Cara2A) λ cara-opuesta (A, Cara4A, Cara5A) λ cara-opuesta (A, Cara5A, Cara4A) λ en-mesa(A) λ

• Referente al bloque B:

λ cara-superior (B, CaraB, Roja) λ cara-inferior (B, CaralB, Blanca) λ cara-lateral (B, Cara2B, Azul) λ cara-lateral (B, Cara3B, Azul) λ cara-lateral (B, Cara4B, Azul) λ cara-lateral (B, Cara5B, Azul) λ cara-opuesta (B, CaraB, CaralB) λ cara-opuesta (B, Ca-ralB, CaraB) λ cara-opuesta (B, Cara2B, Cara3B) λ cara-opuesta (B, cara3B, Cara2B) λ cara-opuesta (B, Cara4B, Cara5B) λ cara-opuesta (B, Cara5B, Cara4B) λ encima (B, A) λ

• Referente al bloque C:

λ cara-superior (C, CaraC, Azul) λ cara-inferior (C, CaralC, Roja) λ λ cara-lateral (C, Cara2C, Negra) λ cara-lateral (C, Cara3C, Negra) λ λ cara-lateral (C, Cara4C, Negra) λ cara-lateral (C, Cara5C, Negra) λ λ cara-opuesta (C, CaraC, CaralC) ∧ cara-opuesta (C, CaralC, Ca-raC) λ cara-opuesta (C, Cara2C, Cara3C) λ cara-opuesta (C, Cara3C, Cara 2C) λ cara-opuesta (C, Cara4C, Cara5C) λ cara-opuesta (C, Cara5C, Cara4C) λ encima (C, B) λ libre (C) λ

• Predicados generales:

color (Blanca) λ color (Negra) λ color (Roja) λ color (Azul) λ brazo-libre (Rl) λ brazo-libre (R2)

Meta: encima (A, C)

3. Los cinco primeros nodos del árbol serían:

El primer nodo tendría:

encima (A, C)	E1

Los descendientes serían todas aquellas combinaciones de valores para las variables libres del único operador que añade encima (x, y), Poner (R, B, B'), donde la sustitución de variables sería [(B x C) (B' x A)]. Como quedan libres las variables R, Cara, Cara', y C, habría tantos descendientes como posibles combinaciones de valores para esas variables haya. Supongamos que elige una de las correctas, que sería:

Poner (R1, A, C) encima (A, C)	E1

• El único descendiente sería aquel que pondría las precondiciones del operador en la pila de metas:

Libre (C) ∧ sujeto (R1, A) ∧ cara-superior (C, Cara2C, Negra) ∧ ∧ cara-inferior (A, Cara1A, Negra) Poner (R1, A, C) encima (A, C)	E1

Este nodo tendría tantos descendientes como combinaciones de las metas de la cima de la pila hubiera. Vamos a poner un solo nodo descendiente.

Libre (C) Sujeto (R1, A) cara-superior (C, Cara2C, Negra) cara-inferior (A, Cara1A, Negra) libre (C) ∧ sujeto (R1, A) ∧ cara-superior (C, Cara2C, Negra) ∧ ∧ cara-inferior (A, Cara1A, Negra) Poner (R1, A, C) encima (A, C)	E1

• Por último, los descendientes de este nodo serían todos los operadores instanciados que añaden libre (C). Los operadores serían:
Quitar-Otro-Libre, Quitar-Otro-Ocupado, Poner, y Dejar. Uno de esos descendientes podría ser:

Dejar (R1, C) libre (C) sujeto (R1, A) cara-superior (C, Cara2C, Negra) cara-inferior (A, Cara1A, Negra) libre (C) ∧ sujeto (R1, A) ∧ cara-superior (C, Cara2C, Negra) ∧ ∧ cara-inferior (A, Cara1A, Negra) Poner (R1, A, C) encima (A, C)	E1

3. Se pide establecer cuáles serian unos posibles valores de criticalidad del dominio del problema 1 de acuerdo a ABSTRIPS.

Una forma de asignar los valores de criticalidad consiste en dar un alto valor de criticalidad a los predicados estáticos, aquellos que no pueden ser añadidos o borrados por ningún operador, y dar un valor bajo a los predicados no estáticos. Así, los predicados estáticos serían: pieza (p), madera (p), hierro (p), grande (p),

Departamento de Cs. e Ingeniería de la Computación
Universidad Nacional del Sur

Conceptos de Inteligencia Artificial

Trabajo Práctico N° 8

Representación de Acciones y Cambio - Planificación

Segundo Cuatrimestre de 2018

Ejercicios

1. Conceptos Básicos

    a) ¿Cómo se define un problema de planificación?

    b) ¿En qué consiste un dominio de planificación?

    c) ¿Qué diferencia hay entre un operador y una acción?

    d) ¿Qué es una solución a un problema de planificación?

2. `STRIPS`

    a) ¿En qué consiste el lenguaje de representación de `STRIPS`?

    b) ¿Cómo se representan los estados, las metas y las acciones?

3. Mundo de Bloques

    Considere el dominio de planificación *Mundo de Bloques* visto en clase, que utiliza las relaciones $libre(X)$, $mesa(X)$ y $sobre(X, Y)$ para representar estados; y las acciones:

    - $apilar(A, B)$: apila un bloque A sobre un bloque B. Esta acción puede llevarse a cabo sólo si ambos bloques están libres y A está sobre la mesa.
    - $desapilar(A, B)$: desapila sobre la mesa el bloque A que está sobre el bloque B. Esta acción sólo puede realizarse si el bloque A está sobre el bloque B y el bloque A está libre.

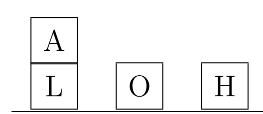


Figura 1: Estado inicial para el Mundo de Bloques.

    a) Haciendo uso de las relaciones descriptas anteriormente, enumere los hechos que representan el estado inicial ilustrado en la Figura 1.

    b) Defina los operadores *apilar* y *desapilar* de acuerdo a la representación de acciones utilizada por `STRIPS`.

c) Siguiendo la representación de STRIPS, describa el mundo resultante luego de llevar a cabo la siguiente secuencia de acciones (proyección) a partir del estado inicial ilustrado en la Figura 1:

$$desapilar(\texttt{A,L}), apilar(\texttt{L,O}), apilar(\texttt{A,L})$$

Por cada acción realizada, indique claramente cuáles son los estados intermedios que se van obteniendo.

4. ¿Qué es un sistema de planificación? ¿Qué se entiende por forward planning y backward planning?

5. Describir el algoritmo de planificación denominado "STRIPS Planner" en [PMG98].

6. Considere las relaciones y acciones del dominio de planificación del Ejercicio 3 *Mundo de Bloques*.

   a) Considere el estado inicial que se muestra en la Figura 1 y suponga que se quiere alcanzar la meta $\{sobre(H, O), sobre(A, H)\}$. Realizar una traza del algoritmo de STRIPS Planner, indicando ***claramente*** la evolución del algoritmo. Esto es, qué submeta se selecciona en cada paso, qué acción se elige para obtener dicha submeta y, en consecuencia, cuáles son las metas restantes.

      Verifique la solución encontrada utilizando el planificador STRIPS (*strips_planner.pl*) provisto por la cátedra.

   b) ¿Qué sucede si en la ejecución del STRIPS Planner considerada en el ejercicio anterior el algoritmo selecciona para resolver en primera instancia la meta *sobre(A,H)* y luego *sobre(H,O)*? ¿Cómo es posible solucionarlo?

7. Planificador de Orden Parcial

   a) ¿Qué es un plan parcial (plan parcialmente ordenado)?

   b) Describa el funcionamiento de un planificador de orden parcial como el presentado en [PMG98].

   c) ¿Qué se obtiene como resultado de la ejecución de un planificador de orden parcial?

   d) ¿Qué representa una *solución* en el contexto de planificación de orden parcial?

8. Considere la siguiente definición de acciones:

acción	pre	add	del
dormir	{relajado}	{descansado}	{}
bañarse	{toalla_seca}	{limpio}	{toalla_seca}
comprar_ropa	{descansado, limpio}	{ropa}	{}
limpiar_pileta	{descansado}	{pileta_limpia}	{limpio}

el estado inicial E_I = {relajado, toalla_seca} y la meta M = {ropa, pileta_limpia}.

a) Halle 2 planes parciales que permitan lograr M a partir de E_I, empleando el algoritmo de Planificación de Orden Parcial descripto en el Ejercicio 7. Muestre claramente cada paso de la ejecución del algoritmo, indicando la solución correspondiente a cada uno de los planes parciales encontrados.

b) Para alguna de las soluciones halladas en el inciso anterior, obtenga 2 planes (totales) asociados.

c) Vuelva a realizar el Ejercicio 8*a*) considerando las metas que conforman M en distinto orden. ¿Cambian las soluciones encontradas?

Referencias

[PMG98] Poole, D., Mackworth, A., and Goebel, R. *Computational Intelligence: A Logical Approach.* Oxford University Press, 1998.

[RN02] Russel, S., and Norvig, P. *Artificial Intelligence: A Modern Approach*, 2nd ed. Prentice Hall, 2002.