Práctica 3

Representación de dominios y resolución de problemas con técnicas de planificación

---

Técnicas de los Sistemas Inteligentes - GII
Grupo 2
Curso 19-20

Universidad de Granada
Cumbreras Torrente, Paula
49087324-B

Índice de contenidos

Introducción

El objetivo de esta práctica es aprender a diseñar e implementar dominios de planificación clásicos. Para ello se empleará el planificador Metric-FF, un planificador que permite ejecutar planes definidos en el lenguaje PDDL. Para esta práctica, basada en el videojuego StarCraft1 de Blizzard Entertainment, se pide desarrollar 6 ficheros PDDL en los cuales, de manera ascendente, se irá completando con varias funcionalidades el dominio de planificación desarrollado.

Ejercicios

Ejercicio 1

En este ejercicio se desarrolla una primera versión del dominio. Se han añadido los tipos unidad, edificio, recurso y localización. Además se han definido los siguientes constantes:

- VCE de tipo unidad.
- CentroDeMando y Barracones de tipo edificio
- Minerales y Gas de tipo recurso

Además de los predicados solicitados para la práctica (unidadEn, edificioEn, existeCamino, asignarMaterial, extrayendo) se han añadido 3 predicados esTipo (uno para cada tipo de objeto, usados para relacionar objetos y constantes), un predicado necesitaRecurso (indica qué material necesita un determinado tipo de edificio para ser construído) y otro trabajando (será cierto cuando una unidad VCE esté trabajando, es decir, haya sido asignada a un nodo). Se ha añadido también una función cantidadRecursos que cuantificará la cantidad de material que se ha recolectado de un tipo en concreto. En esta versión sólo se incluyen tres acciones:

- Navegar: si existe camino entre localización destino y la localización en la que se encuentra la unidad, ésta dejará de estar en la actual para aparecer en la localización deseada.
- Asignar: solo se podrá asignar una unidad a un nodo de recurso si esa unidad es VCE, se encuentra libre y está en la misma localización que el nodo. El efecto de esta acción en el mundo es que la unidad comienza a extraer del nodo (deja de estar libre) y además se incrementa la cantidad de recursos del material del nodo en una unidad. Esto sería ineficiente para problemas grandes, pues al asignar un VCE a un nodo sólo se obtendría una unidad del recurso. Sin embargo, dado que en los primeros ejercicios no hay coste de construcción ni se pide la creación de muchos objetos, se ha definido de este modo.

- Construir: se construirá si la unidad es VCE, se encuentra en la localización en la que se desea construir el edificio y está libre, si la localización no ha sido ocupada por otro edificio y si se poseen suficientes recursos (al menos uno). Como resultado aparecerá el edificio y se reducirá la cantidad de recursos del material necesario para el edificio en una unidad.

En la inicialización del problema se ha creado un grid de 5x5 conectadas entre sí haciendo uso las herramientas del editor online sugerido para la práctica. Se definen tres unidades de tipo VCE, dos edificios (un barracón y un centro de mando) y cinco nodos de recursos (dos de gas y tres de mineral). Se sitúan en localizaciones aleatorias y se establecen las necesidades de materiales para la construcción de cada tipo de edificio. Por último se inicializa la cantidad de recursos de gas y mineral a 0. El objetivo será construir un barracón.

```
ff: found legal plan as follows
step    0: NAVEGAR U1 L5_3 L5_2
        1: NAVEGAR U2 L1_4 L1_5
        2: NAVEGAR U1 L5_2 L5_1
        3: ASIGNAR U2 MIN3 MINERALES L1_5
        4: CONSTRUIR U1 BARRACON BARRACONES L5_1 MINERALES

time spent:    0.00 seconds instantiating 495 easy, 0 hard action templates
               0.00 seconds reachability analysis, yielding 246 facts and 417 actions
               0.00 seconds creating final representation with 194 relevant facts, 2 relevant fluents
               0.00 seconds computing LNF
               0.00 seconds building connectivity graph
               0.00 seconds searching, evaluating 6 states, to a max depth of 1
               0.00 seconds total time
```

Ejercicio 2

Para el ejercicio 2 se pide modificar el dominio de forma que para extraer recursos de un nodo de gas, antes ha de ser construído en él un extractor. Para ello se ha definido otro objeto constante Extractor de tipo edificio. Se ha añadido un predicado extractorConstruido el cual indicará si se ha construido un extractor en una localización dada. Las acciones han sufrido las siguientes modificaciones:

- Asignar: se añade como precondición que, si se trata de un nodo de gas, se haya construido un extractor en dicha localización(*).
- Construir: si el edificio a construir es un extractor se comprobará que hay un nodo de gas en la localización en la que se va a construir el edificio (para ello se ha usado el cuantificador lógico exists de :adl). Se añade como efecto la información de que el extractor ha sido construido, en caso de que así haya sido, haciendo uso del condicional when.

En la inicialización del problema se ha incluido un edificio de tipo extractor y se ha añadido la información sobre su construcción. Dado que el objetivo no ha variado, y no influyen los cambios en la creación de un barracón, los pasos encontrados son los mismos que en el ejercicio 1, por lo que solo se añaden los tiempos

```
time spent:    0.01 seconds instantiating 240 easy, 168 hard action templates
               0.00 seconds reachability analysis, yielding 275 facts and 408 actions
               0.00 seconds creating final representation with 200 relevant facts, 2 relevant fluents
               0.00 seconds computing LNF
               0.00 seconds building connectivity graph
               0.00 seconds searching, evaluating 5 states, to a max depth of 1
               0.01 seconds total time
```

(*) Todas las condiciones en las que no se especifique el condicional usado se han llevado a cabo mediante anidado de expresiones lógicas.

Ejercicio 3

Se pide la modificación del dominio de forma que un edificio pueda requerir el uso de más de un tipo de material. Debido a esto, solamente ha sufrido cambios la acción Construir: a la hora de comprobar que tenemos suficientes recursos, en lugar de pasar el tipo de recurso (Gas o Minerales) como parámetro, se usa el cuantificador lógico forall e la expresión lógica imply para tener en cuenta todos los tipos de recurso que necesita el edificio:

```
(forall (?tipoN - recurso)
   (imply (necesitaRecurso ?tipoE ?tipoN) (>= (cantidadRecurso ?tipoN) 1))
)
```

Del mismo modo, al reducir la cantidad de recurso se vuelve a usar forall, ésta vez con un condicional when, de tal forma que si se ha utilizado el material para la construcción, se reducirá. Al usar forall se permite usar la misma acción para edificios que necesiten varios tipos de recursos y para los que únicamente necesiten uno.

En la inicialización se han añadido la nueva información para la construcción de los edificios involucrados. Al igual que en el ejercicio anterior, se sigue manteniendo el objetivo, al que siguen sin afectar los cambios en los pasos para la resolución. Se muestra a continuación el tiempo tardado.

```
time spent:    0.02 seconds instantiating 320 easy, 224 hard action templates
               0.00 seconds reachability analysis, yielding 307 facts and 544 actions
               0.00 seconds creating final representation with 232 relevant facts, 2 relevant fluents
               0.00 seconds computing LNF
               0.00 seconds building connectivity graph
               0.00 seconds searching, evaluating 6 states, to a max depth of 1
               0.02 seconds total time
```

Ejercicio 4

Para este ejercicio se pide añadir dos nuevos objetos constantes de tipo unidad: Segador y Marine. A su vez, se han añadido los predicados recursoUnidad (indica qué tipo de material o materiales necesita para reclutar un tipo concreto de unidad) y generaUnidad (almacena información sobre el tipo de edificio que puede generar a un tipo de unidad). Se ha añadido una nueva acción:

- Reclutar: una nueva unidad podrá ser reclutada en un edificio si éste edificio es del tipo que genera al tipo de unidad a reclutar, si no ha sido reclutada anteriormente y si para todos los recursos necesarios para su reclutamiento se tiene una cantidad igual o superior a 1. Como efecto, aparecerá dicha unidad en la misma localización que el

edificio y todas las cantidades de los recursos que hayan sido empleadas para el reclutamiento disminuirán en una unidad.

La acción Asignar ha sido modificada, aumentando el número de recursos que se obtenían a 5 por cada asignación, dado que era imposible resolver el problema obteniendo solo 1.

En la inicialización se definen tres nuevas unidades (dos marines y un segador). Se eliminan dos VCE, dejando únicamente sobre el tablero 1 VCE y un centro de mando. El objetivo será que haya un marine en una localización y un marine y un segador en otra.

```
ff: found legal plan as follows
step    0: NAVEGAR U1 VCE L5_3 L4_3
        1: NAVEGAR U1 VCE L4_3 L3_3
        2: NAVEGAR U1 VCE L3_3 L3_2
        3: NAVEGAR U1 VCE L3_2 L3_1
        4: ASIGNAR U1 MIN1 MINERALES L3_1
        5: RECLUTAR U7 VCE CENTRO CENTRODEMANDO L4_2
        6: NAVEGAR U7 VCE L4_2 L3_2
        7: RECLUTAR U2 VCE CENTRO CENTRODEMANDO L4_2
        8: NAVEGAR U2 VCE L4_2 L5_2
        9: NAVEGAR U2 VCE L5_2 L5_3
       10: NAVEGAR U2 VCE L5_3 L5_4
       11: NAVEGAR U7 VCE L3_2 L2_2
       12: NAVEGAR U7 VCE L2_2 L1_2
       13: CONSTRUIR U2 BARRACON BARRACONES L5_4
       14: NAVEGAR U2 VCE L5_4 L4_4
       15: CONSTRUIR U2 EXTRACTOR EXTRACTOR L4_4
       16: ASIGNAR U2 GAS1 GAS L4_4
       17: CONSTRUIR U7 BARRACON BARRACONES L1_2
       18: NAVEGAR U7 VCE L1_2 L1_3
       19: NAVEGAR U7 VCE L1_3 L1_4
       20: NAVEGAR U7 VCE L1_4 L1_5
       21: ASIGNAR U7 MIN3 MINERALES L1_5
       22: RECLUTAR M1 MARINE BARRACON BARRACONES L1_2
       23: RECLUTAR S1 SEGADOR BARRACON BARRACONES L1_2
       24: RECLUTAR M2 MARINE BARRACON BARRACONES L5_4

time spent:    0.03 seconds instantiating 800 easy, 642 hard action templates
               0.00 seconds reachability analysis, yielding 808 facts and 1442 actions
               0.00 seconds creating final representation with 653 relevant facts, 2 relevant fluents
               0.00 seconds computing LNF
               0.00 seconds building connectivity graph
               0.02 seconds searching, evaluating 54 states, to a max depth of 6
               0.05 seconds total time
```

Ejercicio 5

Se pide incorporar al dominio un nuevo tipo investigacion, así como dos nuevos objetos constantes: ImpulsorSegador de dicho tipo y BahiaDeIngenieria de tipo edificio. Se han añadido además los predicados esTipoInvestigacion, investigacionFinalizada (indicará si se ha realizado una investigación concreta) y recursoInvestigar (almacena información sobre el tipo de recurso o recursos que necesita un tipo de investigación concreta para investigarse). Se ha añadido la siguiente acción:

- Investigar: para poder realizar una investigación ha de haberse construido una bahía de ingeniería, se han de poseer los recursos necesarios de los tipos que precise la

investigación y ésta no ha podido ser investigada anteriormente. Como resultado, se guardará la información sobre la investigación y se reducirán en una unidad la cantidad de los tipos de materiales empleados en una unidad.

Se ha modificado la acción Reclutar de forma que, si estamos tratando de reclutar a un Segador, se comprobará previamente si se ha investigado ImpulsoSegador; para ello se ha usado el cuantificador lógico exists.

En la inicialización del problema solo se ha definido una nueva unidad imp_seg y la información correspondiente a los materiales necesarios para la investigación ImpulsoSegador. Aunque se mantiene el objetivo del ejercicio 4, en esta ocasión los pasos para alcanzarlo sí se ven afectados por los cambios (será necesario realizar una investigación para poder reclutar a un segador)

```
ff: found legal plan as follows
step    0: NAVEGAR U1 VCE L5_3 L4_3
        1: NAVEGAR U1 VCE L4_3 L3_3
        2: NAVEGAR U1 VCE L3_3 L3_2
        3: NAVEGAR U1 VCE L3_2 L3_1
        4: ASIGNAR U1 MIN1 MINERALES L3_1
        5: RECLUTAR U7 VCE CENTRO CENTRODEMANDO L4_2
        6: NAVEGAR U7 VCE L4_2 L3_2
        7: RECLUTAR U2 VCE CENTRO CENTRODEMANDO L4_2
        8: NAVEGAR U2 VCE L4_2 L5_2
        9: NAVEGAR U2 VCE L5_2 L5_3
       10: NAVEGAR U2 VCE L5_3 L5_4
       11: NAVEGAR U7 VCE L3_2 L2_2
       12: NAVEGAR U7 VCE L2_2 L1_2
       13: CONSTRUIR U2 BARRACON BARRACONES L5_4
       14: NAVEGAR U2 VCE L5_4 L4_4
       15: CONSTRUIR U2 EXTRACTOR EXTRACTOR L4_4
       16: ASIGNAR U2 GAS1 GAS L4_4
       17: CONSTRUIR U7 BARRACON BARRACONES L1_2
       18: NAVEGAR U7 VCE L1_2 L1_3
       19: NAVEGAR U7 VCE L1_3 L1_4
       20: NAVEGAR U7 VCE L1_4 L1_5
       21: ASIGNAR U7 MIN3 MINERALES L1_5
       22: RECLUTAR U3 VCE CENTRO CENTRODEMANDO L4_2
       23: CONSTRUIR U3 BAHIA BAHIADEINGENIERIA L4_2
       24: INVESTIGAR IMP_SEG IMP_SEG BAHIA L4_2
       25: RECLUTAR M1 MARINE BARRACON BARRACONES L1_2
       26: RECLUTAR S1 SEGADOR BARRACON BARRACONES L1_2
       27: RECLUTAR M2 MARINE BARRACON BARRACONES L5_4

time spent:    0.09 seconds instantiating 800 easy, 917 hard action templates
               0.00 seconds reachability analysis, yielding 887 facts and 1717 actions
               0.00 seconds creating final representation with 707 relevant facts, 2 relevant fluents
               0.01 seconds computing LNF
               0.00 seconds building connectivity graph
               0.02 seconds searching, evaluating 84 states, to a max depth of 6
               0.12 seconds total time
```

Ejercicio 6

Se pide trabajar ahora con información numérica, aunque en mi caso ya había tratado con ella en el primer ejercicio, por lo que la adaptación del dominio a las nuevas características no ha supuesto un cambio tan drástico.

- Se ha añadido un nuevo objeto constante del tipo edifcio: Deposito.
- Se ha añadido un predicado almacenLleno que funcionará como un booleano indicando si se ha alcanzado ya la capacidad máxima de almacenamiento para algún tipo de material. Haciendo que el predicado sea por tipo en lugar de general, aunque la capacidad máxima sea la misma, permite que se pueda seguir extrayendo de un tipo de recurso aunque el otro haya llegado a su capacidad máxima.
- Se han añadido 6 nuevas funciones: maximoAlmacen (capacidad máxima de almacenamiento), capacidadRecoleccion (cantidad de recursos que puede recoger una unidad cada vez), costeConstruccion, costeReclutamiento, costeInvestigacion (cantidad de recursos de un tipo concreto para la construcción de un edificio / reclutamiento de una unidad / investigación respectivamente) y numTrabajadores (cantidad de trabajadores asignados a un nodo de recurso)
- Se ha modificado la acción Asignar de forma que no aumente la cantidad de recursos al asignar a una unidad a un nodo, pero sí aumenta la cantidad de trabajadores asignados a dicho nodo en una unidad.
- Las acciones Construir, Reclutar e Investigar también han sido modificadas: en lugar de comprobar que se posee de una cantidad mayor o igual a 1 de cada tipo de recurso necesario, se comprueba que sea mayor o igual al coste asignado para dicho recurso y objeto a construir/reclutar/investigar, respectivamente. Del mismo modo, como efecto se restará de las cantidades de recursos el coste empleado en lugar de una unidad. Además, en el efecto de la acción Construir, si se ha construido un depósito se aumentará la capacidad máxima del almacén en 100 unidades.
  Se han añadido dos nuevas acciones
- Desasignar: una unidad podrá desasignarse de un nodo de recursos si estaba previamente extrayendo de él. Como consecuencia, el VCE dejará de extraer, pasará a estar libre y se reducirá la cantidad de trabajadores de dicho nodo en una unidad.
- Recolectar: se podrá recolectar de un recurso si el almacén no está lleno y si la cantidad que se posee de dicho recurso no supera el máximo del almacén menos el número de trabajadores por la capacidad de recolección (como es mostrado al final del párrafo). Con esta última condición nos aseguramos de que en ningún momento se excederá la capacidad máxima, habiendo que reducir el número de trabajadores para recolectar en caso de que hubiera demasiados. Como consecuencia, la cantidad del recurso deseado incrementará en número de trabajadores por su capacidad de recolección. Se añade también una condición when, de forma que si la cantidad ahora supera el máximo del almacén menos el número de trabajadores por la capacidad de recolección, se declarará lleno el almacén.

```
(<= (cantidadRecurso ?tipo) (- (maximoAlmacen)(* (numTrabajadores ?tipo) (capacidadRecoleccion))))
```

En la inicialización del problema se definen nuevos edificios del tipo Deposito y se añade la información necesaria para esta práctica, es decir, la información númerica: costes de construcción, reclutamiento e investigación, capacidad máxima del almacén y capacidad de recolección. Se inicializan a 0 el número de trabajadores para cada recurso.

Me han surgido problemas relacionados con el tiempo empleado en ejecutarse, tales que hacen inviable la ejecución del mismo con los parámetros sugeridos en el guión de la práctica. Haciendo varias pruebas, he obtenido lo siguiente:

- Disminuir todos los costes de construcción, reclutamiento e investigación: al realizar esto he sido capaz de ejecutar el problema con una capacidad de recolección de 10 unidades por trabajador, sin embargo he tenido que reducir tanto dichos valores para lograrlo que difiere demasiado del ejercicio propuesto, por lo que queda rechazado.
- Aumentar la capacidad de recolección: con una capacidad de recolección de 25 y los valores de coste iniciales obtiene un tiempo de 26.13 segundos, el cual sigue pareciéndome excesivo. Con una capacidad mayor, por ejemplo, de 40, se obtiene un tiempo de 4.04 segundos.
- Comenzar con los almacenes llenos: Como en muchos juegos del estilo de StarCraft, al inicio de una partida se comienza con los almacenes al máximo, pues la recolección desde 0 con poca mano de obra puede resultar tediosa. Sin embargo, aun haciendo esto, con una capacidad de recolección de 10 sigue sin ser factible. Aumentando la capacidad a 25, con esta nueva característica, el tiempo se reduce a 2.26 segundos. Esta idea quedó descartada pues en ningún momento se afirma que pueda realizarse esta acción, aunque la he añadido en la memoria para la comparación.
- Colocar más VCE sobre el tablero o eliminar su coste de reclutamiento: Añadiendo 4 VCE más al tablero, el tiempo empleado con una capacidad de recolección de 10 pasa a ser 4.37 segundos. Eliminando su coste y con una capacidad de recolección de 25, apenas tarda 0.84 segundos. En cualquier juego esto iría en contra de las reglas y sería considerado “hacer trampas”, por lo que también queda descartada.

Ninguna de esas ideas me convencía por separado así que mezclé varias de ellas, obteniendo la siguiente inicialización: El almacén de minerales tendrá 50 unidades, el de gas estará vacío y la capacidad de recolección será de 25 unidades de recurso por cada VCE asignado. De esta forma el primer paso del plan es reclutar a un nuevo VCE, reduciendo considerablemente el tiempo de ejecución. El plan consta de 59 pasos por lo que no ha sido incluído en la memoria.

```
time spent:    0.23 seconds instantiating 840 easy, 1351 hard action templates
               0.00 seconds reachability analysis, yielding 1047 facts and 2023 actions
               0.00 seconds creating final representation with 813 relevant facts, 18 relevant fluents
               0.00 seconds computing LNF
               0.00 seconds building connectivity graph
               5.12 seconds searching, evaluating 5181 states, to a max depth of 13
               5.35 seconds total time
```

Bibliografía

- http://editor.planning.domains *"PDDL Editor"* (*)
- https://github.com
    - *"primaryobjects/strips: AI Automated Planning with STRIPS and PDDL in Node.js"*
    - *"pold87/myPDDL-Atom: PDDL syntax highlighting for Atom"* (*)
    - *"negative preconditions example·Issue #8·hfoffani/pddl-lib*
- https://elvex.ugr.es - *"P9PDDL.pptx"*
- https://www.ida.liu.se
- https://www.cs.upc.edu - *"FastForward.pdf"*
- http://www.cs.toronto.edu - *"PDDL by Example"*
- https://planning.wiki
    - *"PDDL Requirements-Planning.wiki-The AI Planning & PDDL Wiki"*
    - *"SMTPlan+-Planning.wiki-The AI Planning & PDDL Wiki"*
    - *"Planners by tag-Planning.wiki-The AI Planning & PDDL Wiki"*

(*) Sugerido por los profesores