Memoria Práctica 2

Álvaro Beltrán

May 22, 2020

Grupo 3, Jose Ángel Segura Muros

1 Introducción

En esta práctica se pide al estudiante resolver la siguiente relación de problemas usando PDDL para la solución de un problema de planificación sobre el juego Starcraft.

2 Ejercicios

2.1 Ejercicio 1

Para el ejercicio 1 he definido 4 tipos que heredan de objeto (Edificios Recursos movible Localizaciones) y otro que hereda de movible (Unidades). Luego he definido los tipos de estos objetos como constantes : TipoUnidades - Unidades, TipoRecursos - Recursos, TipoEdificios - Edificios, VCE - TipoUnidades, CentroDeMando Barracones - TipoEdificios, Minerales Gas - TipoRecursos.

En cuanto a los predicados, he definido doce. Tres de ellos para saber de que tipo es el objeto, cinco asociados a los predicados que pide el enunciado, otro para saber si una unidad está asignada, otro para saber si la localización está libre, otro para indicar que se ha recogido un recurso y un último para saber si una instancia de edificio se ha construido ya.

En este ejercicio se nos pide hacer tres acciones: Navegar, Asignar y construir:

- **Navegar**. Necesitamos tres parámetros: una unidad y dos localizaciones. como precondición es necesario que la unidad se encuentre en una de las localizaciones y que haya un camino entre las dos localizaciones. El resultado de esta acción es que ahora la unidad está en la otra localización y que ya no se encuentra en la antigua.
- **Asignar**. Necesitamos tres parámetros: una unidad, un recurso y una localización. La unidad no debe estar asignada y debe estar en la localización; además el recurso debe estar asignado a esa localización. El resultado es que la unidad ya está asignada, ese recurso se ha recogido y la unidad está extrayendo ese recurso.
- **Construir**. Necesitamos cuatro parámetros: una unidad, un recurso, una localización y un edificio. La unidad no debe estar asignada y debe estar en la localización; además el recurso debe estar recogido y el edificio debe necesitar ese recurso para ser construido, por último el edificio no ha debido ser construido, además de que la localización debe estar libre para que se pueda construir. El resultado es que el edificio está en esa localización, que ese edificio ya se ha construido y que esa localización ya no está libre.

el mapa que uso es el siguiente, donde Gx son los gases, Mx son los minerales, CM el centro de mando y B el lugar donde voy a colocar los Barracones:

VCEs	X	X	X	B
X	M2	X	G1	X
X	X	X	X	X
X	M3	X	X	CM
G2	M1	X	X	X

En el Dominio he definido los respectivos objetos que aparecen en el mapa y he puesto como verdaderos los predicados pertinentes. He creado un grid con el editor online y por último he definido con los predicados de que tipo es el objeto creado.

El objetivo es construir un edificio Barracones en la localización indicada, y al ejecutar lo hace sin problemas.

2.2 Ejercicio 2

Para el ejercicio 2 he añadido un tipo en Edificios denominado Extractor y un predicado que me indica si hay un extractor en una posición. He hecho cambios en estas dos acciones:

- **Asignar**. He cambiado la precondición con una disyunción para que se entre si es mineral igual que antes, pero si es gas en el nodo debe haber construido un extractor.
- **Construir**. En construir en los efectos si el edificio es un extractor debemos indicar con el predicado anteriormente definido que hay un extractor en esa posición.

En el dominio he añadido un objeto extractor el cual necesita de mineral para ser construido y lo demás todo igual que el ejercicio anterior. He usado el mismo mapa que en el ej1.

Al igual que en el ejercicio 1 tenemos el mismo objetivo, el cual realiza creando antes un extractor en un nodo de Gas para extraer Gas y así luego poder construir Barracones.

2.3 Ejercicio 3

Para el ejercicio 3 he tenido que cambiar los predicados necesitaRecurso (he cambiado para que en vez de aceptar el objeto acepte el tipo) y RecursoRecogido (he cambiado el objeto por el tipo). También he añadido un nuevo predicado por si el edificio necesita dos recursos (necesitaDosRecursos).

También he tenido que cambiar la acción construir de forma que acepte como precondición que necesite un tipo de recurso o dos. Para ello compruebo que recurso necesita y si lo he recogido. De esta forma el recurso no está como parámetro.

En el dominio he añadido que el centro de mando necesita minerales y gas.

el mapa que uso es el siguiente, el objetivo es construir el centro de mando (CM) para probar que funciona y un Barracón por el enunciado:

VCEs	X	X	X	CM
X	M2	X	G1	X
X	X	X	X	X
X	M3	X	X	B
G2	M1	X	X	X

Llega a la solución sin problemas construyendo lo necesario.

2.4 Ejercicio 4

En este ejercicio nos piden hacer una nueva acción llamada reclutar cuyos parámetros dos la unidad, el tipo de unidad, el edificio y tipo donde se construyen y la localización. Las precondiciones son que la unidad no esté reclutada y que se haya recogido el recurso necesario para reclutar esa unidad. Como efecto, tenemos que la unidad ha sido reclutada, no está asignada y he tomado que aparece en la posición del edificio donde se recluta. He supuesto que no se necesita de una unidad para reclutar.

A parte, tenía que crear dos tipos nuevos de unidades y para implementar la acción me he ayudado de 4 predicados: dos para saber que recursos necesita cada unidad (unidadNecesitaRecurso unidadNecesitaDosRecursos), otro para saber si la unidad está reclutada (unidadReclutada) y otro para saber donde se recluta cada tipo de unidad (unidadReclutadaEn).

Por último en el dominio he añadido un segador y dos marines, y he inicializado con este mapa para probar:

VCE	X	X	X	CM
X	M1	X	G1	X
X	X	X	X	X
X	M2	X	X	X
G2	M3	X	X	X

El objetivo es reclutar a un segador y dos marines, un segador y un marine en una posición y el otro marine en otra posición distinta. El programa encuentra una solución correcta y coloca las unidades en las localizaciones indicadas en goal.

2.5 Ejercicio 5

Para este ejercicio he añadido un tipo de edificio nuevo llamado BahiaDeIngenieria, he creado un tipo nuevo llamado Investigaciones y un tipo de investigaciones llamado Im-

pulsorSegador. Para saber que investigación se ha completado he añadido un predicado llamado InvestigacionCompletada a la que le paso un tipo de investigación.

Por último, he implementado la acción investigar cuyos parámetros son una localización, un edificio y un tipo de investigación. Las precondiciones son: que el edificio sea BahiaDeIngenieria, que el edificio esté en esa posición (con esto nos aseguramos que está construido), que tengamos minerales y gas (necesarios para investigar según el foro de prácticas) y que no se haya investigado ese tipo. El efecto es que ese tipo de investigación se ha completado.

En el dominio he tenido que añadir un objeto investigación de tipo ImpulsorSegador y un edificio BahiaDeIngenieria que necesita minerales y gas según el foro de prácticas.

El objetivo es el mismo que en el anterior ejercicio y podemos ver como investiga ImpulsorSegador antes de reclutarlo como debe.

2.6 Ejercicio 6

Para este ejercicio he tenido que usar las directrices functions de :fluent para poder hacer operaciones aritméticas. Para ello he definido 4 functions: uno llamado consume para saber cuanto de un tipo de recurso consume una tarea, otro llamado hayAlmacenado para saber cuanto hay almacenado de un tipo de recurso, otro asignados nodo para llevar la cuenta de cuantos trabajadores hay asignados a un nodo y un último maximoRecAlmacenado para saber la capacidad de almacenaje que tenemos.

He hecho cambios en todas las acciones:

- **Asignar**. He añadido un efecto que es aumentar en uno los trabajadores asignados a ese nodo.
- **Construir**. Tengo que comprobar si hay suficientes recursos almacenados. En efectos añado que si es un edificio tengo que aumentar el máximo de capacidad disponible, también tengo que restar los recursos usados.
- **Reclutar**. Tengo que comprobar si hay suficientes recursos almacenados y tengo que restar los recursos usados.
- **Investigar**. Tengo que comprobar si hay suficientes recursos almacenados y tengo que restar los recursos usados.

He hecho añadido dos nuevas acciones:

- **Desasignar**. Lo que hacemos es comprobar que la unidad estaba asignada al recurso indicado, para luego dejarla sin asignar y disminuir en uno los trabajadores asignados al nodo.

- **Recolectar**. Comprobamos si hay trabajadores asignados al nodo del que queremos recolectar y si al recolectar cabe comprobando con el máximo de capacidad actual. Como efecto obtenemos lo recolectado añadiéndolo al almacén.

He tenido que tomar la decisión de aumentar la cantidad de recolectado a 25 porque ni con 10 ni con 20 consigue encontrar solución. Esto creo que se debe a que 25 es divisor de todos los números asignados a consume, de esta forma el planificador encuentra más fácil soluciones exactas de tantos recolectados para realizar tal acción.
Por último en el dominio he añadido dos depósitos (uno de ellos construido para poder empezar), las restricciones que aparecen en la tabla, he igualado el máximo de los almacenes a 100, he igualado los asignados y los recursos almacenados a 0; y he inicializado con este mapa para probar:

VCE	X	X	X	CM
X	M1	X	G1	X
X	X	X	X	D1
X	M2	X	X	X
G2	M3	X	X	X

En este ejercicio volvemos a tener el mismo objetivo que en el anterior y si analizamos cada una de las acciones realiza correctamente el plan, aunque en este caso al ser más complejo tarda un poco más.

3 Conclusión

Una vez acabada la práctica y mirándola con retrospectiva veo que es una práctica muy útil, ya que nunca hemos programado en un lenguaje como este y nos hace pensar de otra manera.