

Técnicas de los Sistemas Inteligentes

Enunciado de la Práctica 3

Introducción

El objetivo de la tercera práctica de la asignatura de *Técnicas de los Sistemas Inteligentes* consiste en diseñar e implementar un dominio de planificación clásico basado en el juego StarCraft¹. Para el desarrollo de la práctica los estudiantes deberán confeccionar un conjunto de ficheros PDDL y ejecutarlos en el planificador Metric-FF para obtener un plan válido. Este documento describe el mundo del juego, así como las tareas a realizar para superar la práctica, e información sobre la entrega y su evaluación.

Descripción del mundo

El mundo sobre el que se va a desarrollar la práctica está inspirado en juegos de estrategia militar y gestión de recursos, tales como StarCraft o Age of Empires. En StarCraft, un grupo de vehículos de construcción espacial (VCEs) deben recolectar una serie de recursos para construir nuevas estructuras, entre otras posibles acciones. Para la realización de esta práctica usaremos una versión simplificada del juego StarCraft 1 de Blizzard Entertainment (Figura 1).



Figura 1: Ejemplo de entorno en el que se desarrolla la acción en StarCraft.

¹ Videojuego de estrategia en tiempo real de ciencia ficción militar desarrollado por Blizzard Entertainment, y lanzado en 1998. Se trata de uno de los juegos para ordenador más vendidos de la historia, con más de 11 millones de copias vendidas (a fecha de Febrero de 2009). Más información en [https://en.wikipedia.org/wiki/StarCraft_\(video_game\)](https://en.wikipedia.org/wiki/StarCraft_(video_game))

En nuestro mundo tenemos una pequeña base Terran², en la que podemos construir edificios y reclutar unidades. Para poder realizar estas acciones necesitaremos extraer recursos del entorno. Los recursos que podremos extraer serán de dos tipos:

- Minerales



- Gas Vespeno

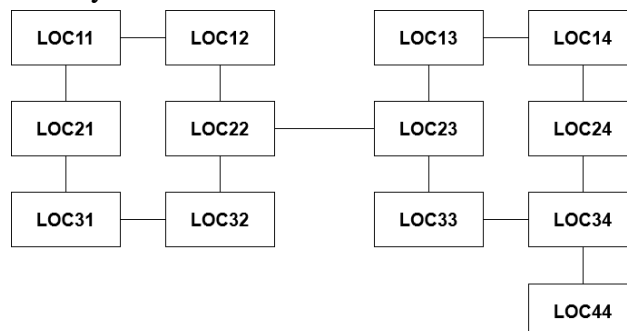


Ambos recursos deben ser recolectados por un tipo de unidad: el VCE. Los VCE son los trabajadores encargados de recolectar recursos y construir edificios. Los nuevos edificios que se construyan desbloquean nuevas capacidades en la base, ya sea habilitando el reclutamiento de nuevos tipos de unidades, o permitiendo el desarrollo de investigaciones que a su vez expandirán aún más dichas capacidades. Para poder extraer recursos, los VCE deben asignarse a un “nodo de recursos” (el emplazamiento donde habrá o bien minerales o bien gas vespeno) y se encargarán de, periódicamente, obtener/extraer el recurso. Un VCE asignado a un recurso no podrá moverse por el mapa ni construir edificios hasta que sea desasignado del nodo³. A un nodo de recursos se pueden asignar tantos VCEs libres como se desee (se entiende por VCE libre aquellos no asignados a otros nodo de recursos).



Figura 2: Ejemplo de VCE para recolección de recursos

Para representar el mapa usaremos el siguiente grafo. Cada nodo del grafo representará una “localización”, en la que puede haber edificios y unidades, y las aristas entre localizaciones representan un camino entre ambas. No hay límite en cuanto al número de unidades que pueden encontrarse en una localización determinada, pero solo puede haber un edificio construido en cada localización. Cada nodo de recursos se encuentra en alguna localización. En una localización solo puede haber un nodo de recursos, aunque puede construirse un edificio en una localización que contenga un nodo de recurso. Es decir, una misma localización puede contener al mismo tiempo un nodo de recursos, un edificio y número ilimitado de unidades.



² Una de las tres razas del videojuego StarCraft, junto con los Protoss y los Zerg. Los Terran son humanos exiliados de la Tierra y altamente evolucionados, con una gran capacidad de adaptación a cualquier situación.

³ Por simplicidad en esta práctica, un VCE asignado a un nodo de recurso ya no podrá hacer nada más el resto de la ejecución puesto que no se implementará ninguna acción para desasignarlo.



Descripción de la tarea a realizar

Los estudiantes deberán definir un dominio de planificación escrito en PDDL que represente el mundo presentado en la sección anterior y, a medida que se realicen los ejercicios propuestos, ir añadiéndole nuevas funcionalidades.

Ejercicio 1

Diseñar y programar una versión inicial del dominio de planificación propuesto. Los dominios diseñados deben cumplir con los siguientes requisitos:

- Representar en el dominio los **tipos: Unidades, Edificios, y Localizaciones**. Usar **constantes** para representar el tipo de unidad **VCE**, los tipos de edificio **CentroDeMando** y **Barracones**, y los tipos de recurso **Minerales** y **Gas**.
- Definir los **predicados** necesarios para:
 - Determinar si un edificio o unidad está en una localización concreta.
 - Representar que existe un camino entre dos localizaciones.
 - Determinar si un edificio está construido.
 - Asignar un nodo de un recurso concreto a una localización concreta.
 - Indicar si un VCE está extrayendo un recurso.

Nota: En caso de que el estudiante considere que necesita más tipos, constantes o predicados para realizar correctamente el ejercicio, tiene libertad total para crearlos.

- El dominio debe contener **únicamente** las siguientes dos **acciones** definidas:
 - **Navegar:** Mueve una unidad entre dos localizaciones.
 - Parámetros: Unidad, Localización origen, Localización destino
 - **Asignar:** Asigna un VCE a un nodo de recurso. Además, en este ejercicio será suficiente asignar un único VCE a un nodo de recursos de un tipo (minerales o gas vespeno) para tener ilimitados recursos de ese tipo.
 - Parámetros: Unidad, Localización del recurso, Tipo de recurso

Nota: Es **imprescindible respetar tanto el nombre de las acciones como el orden de los parámetros** para que el plan generado de vuestro dominio pueda ser comparado con la solución de la práctica.

- El **problema** debe definir el mapa localizaciones expuesto anteriormente. En la localización LOC11 debe encontrarse el edificio CentroDeMando1 (de tipo CentroDeMando) y la unidad VCE1 (de tipo VCE). En el mapa existen dos recursos de mineral en las localizaciones LOC22 y LOC32.
- El **objetivo** (:goal) de este ejercicio es generar recursos de tipo Mineral.

Ejercicio 2

Partiendo del ejercicio anterior, crear/modificar las siguientes **acciones**:

- **Construir:** Ordena a un VCE libre que construya un edificio en una localización. En este ejercicio, cada edificio sólo requerirá un único tipo de recurso para ser construido. Adicionalmente y por simplicidad, en este ejercicio se permite que existan varios edificios en la misma localización.
 - Parámetros: Unidad, Edificio, Localización, Recurso



- **Asignar:** Para poder obtener Gas Vespeno (es decir, asignar un VCE a un nodo de gas vespeno), debe existir un edificio Extractor construido previamente sobre dicho nodo de recurso. No hay cambios para obtener recursos de mineral.

Adicionalmente se debe definir el **predicado** para:

- Definir qué recurso necesita cada edificio para ser construido.

Este ejercicio también requiere la definición del tipo de edificio **Extractor**.

En el **problema**, existen nodos de mineral en las localizaciones LOC22 y LOC32, y un nodo de gas vespeno en la localización LOC44. Estas localizaciones para los recursos se mantendrán constantes para toda la práctica. La construcción de los extractores requiere minerales. Este ejercicio requiere dos unidades VCE (VCE1 y VCE2, localizadas inicialmente en el CentroDeMando1, LOC11): una se encargará de extraer el mineral y la otra se ocupará de la construcción del extractor y posteriormente de extraer el gas vespeno.

El **objetivo** de este ejercicio es generar recursos de tipo Gas Vespeno.

Nota1: Aunque un edificio no haya sido construido (por ejemplo Extractor1 de tipo Extractor), éste debe aparecer en la lista de objetos del problema, para representarlo y poderlo referenciar posteriormente por la acción Construir.

Nota2: cuando una unidad está extrayendo un recurso en una parte del mapa, otra unidad puede construir un edificio (que requiera dicho recurso) en otra parte del mapa. Es decir, no es necesario que esa unidad u otra se desplace por el mapa “portando” el recurso allí donde es necesario. Se asume que las otras unidades ya tienen acceso a dichos recursos para operar con ellos una vez dicho recurso está siendo extraído.

Ejercicio 3

Partiendo del ejercicio anterior, modificar la acción **Construir** para que tenga en cuenta que un edificio puede requerir más de un tipo de recurso. Esta acción debe inferir por sí misma si se tiene el tipo de recursos necesarios para poder ejecutarse. Además, debe evitar que se construya más de un edificio en la misma localización.

- Parámetros: Unidad, Edificio, Localización

Para los siguientes ejercicios es muy importante definir esta restricción de forma que solo se tenga que instanciar una única vez por tipo de edificio. Por ejemplo, para incluir información sobre los recursos necesarios para los Barracones solo debe escribirse una única vez

(necesita Barracones Minerales)

independientemente del número de barracones que existan en el problema.

El **problema** debe definir que la construcción de Barracones necesita la obtención previa tanto de minerales como de gas vespeno. Se definirán tres unidades VCE en la localización LOC11. Los tres nodos de recursos (dos de mineral y uno de gas vespeno) se localizarán en las mismas ubicaciones que el ejercicio anterior, así como el Centro de Mando.

El **objetivo** es la construcción de unos Barracones (llamados Barracones1) en la localización LOC33.

Ejercicio 4

Partiendo del ejercicio anterior, crear una acción **Reclutar** para crear nuevas unidades. En este problema se incluyen dos nuevos tipos de unidades: el Marine y el Soldado. Al igual que los edificios, la creación de cada unidad requiere distintos recursos. La creación de VCEs y Marines



requiere minerales, mientras que los Soldados requieren mineral y gas vespeno. Cada unidad se genera en un edificio concreto: los VCEs se reclutan en el Centro de Mando, mientras que Marines y Soldados se reclutan en los Barracones. Los parámetros de esta acción son:

- Parámetros: Edificio, Unidad, Localización

El fichero de **problema** contendrá inicialmente un único VCE1 (al igual que en los casos anteriores, localizado inicialmente en LOC11), aunque se deben definir otras dos unidades VCE2 y VCE3 (que podrán ser reclutadas en CentroDeMando1 durante la ejecución) para que el problema se pueda resolver. Al igual que con anterioridad, los nodos de minerales estarán en las localizaciones LOC22 y LOC32, mientras que el nodo de gas vespeno estará en la localización LOC44. El Centro de Mando estará ubicado en LOC11.

El **objetivo** de este problema es disponer de un marine (Marine1) en la localización LOC31, otro marine (Marine2) en la localización LOC24, y un soldado (Soldado1) en la localización LOC12. Nótese que para ello hace falta construir un extractor (Extractor1) con el que obtener gas vespeno, con el que posteriormente se construirá unos barracones (Barracones1) donde se reclutarán estas unidades militares. En este ejercicio, los barracones (Barracones1) se deben construir en la localización LOC32.

Ejercicio 5

Partiendo del ejercicio anterior, incluir en el dominio todos los elementos necesarios para usar una nueva acción **Investigar**, cuyos **parámetros** son *Edificio* e *Investigación*, y que permitirá realizar nuevas investigaciones para la base. En este ejercicio, todas las investigaciones se desbloquean en un nuevo edificio “**Bahía de Ingeniería**”, cuya construcción por parte de un VCE requiere de mineral y gas vespeno. Al igual que los edificios y unidades, las investigaciones requieren de varios recursos. De cara a crear una investigación “**Investigar Soldado Universal**”, se requiere tanto minerales como gas vespeno. Modifíquese la acción **Reclutar** para que no se puedan crear Soldados hasta que no se haya finalizado dicha investigación.

El **objetivo** de este ejercicio es construir unos barracones en la localización LOC14, construir una Bahía de Ingeniería en la localización LOC12, y que el Marine1, el Marine2 y el Soldado1 estén en la localización LOC14.

Ejercicio 6

MetricFF no garantiza encontrar el plan óptimo (aunque intenta aproximar lo más posible). Como consecuencia, en ocasiones devolverá un plan válido, pero no óptimo.

En este ejercicio se pide implementar un método que garantice a MetricFF que el plan devuelto sea óptimo. Para ello se trabajará a partir de la versión del ejercicio anterior, con los mismos objetivos y características/definición del problema. En concreto, se debe implementar una función que represente el coste del plan calculado (en el Ejercicio 5, el coste de cada acción es de una unidad), la cual debe ser inicializada y modificada a medida que se incrementa el plan, y sobre la cual se debe añadir una restricción adicional en el objetivo: que el valor de esta función (en el estado objetivo) sea estrictamente menor que un valor dado.

Para encontrar el plan óptimo, este valor dado se establecerá inicialmente con el número de acciones encontradas en el plan calculado en el Ejercicio 5. Posiblemente, la ejecución de este problema devolverá un plan con un menor número de acciones que el encontrado en el Ejercicio 5. De forma iterativa, se irá modificando este valor dado con el número de acciones del último plan encontrado, hasta que MetricFF devuelva que no existe solución (MetricFF imprime “*best first search space empty! problem proven unsolvable*”), momento en el que se sabe que el plan óptimo es el encontrado en la iteración anterior. Si bien este proceso podría ser automatizado, para la



realización de este ejercicio es suficiente con realizar las modificaciones del valor dado de forma manual.

La solución entregada debe devolver el plan óptimo; es decir, si se decrementara en una unidad el valor dado, MetricFF debería devolver que no existe solución.

Ejercicio 7

Partir del **Ejercicio 4** y modificar todos los elementos necesarios del dominio para que este sea capaz de usar información numérica. En concreto, se pide lo siguiente:

1. Incluir una nueva acción **Recolectar** para extraer recursos de un nodo y almacenarlos. Cada vez que se llame a esta acción se actualizará el número de minerales o gas vespeno almacenado. En concreto, la acción se ejecuta sobre un nodo de recurso, y como efecto se incrementará la cantidad de este recurso en 10 unidades por cada VCE asignado a dicho nodo. Además, se debe establecer un límite a la cantidad de recursos almacenados; es decir, esta acción no se podrá llevar a cabo si la cantidad recolectada más la almacenada previamente exceden el límite de recursos almacenable. Este límite se establece en 60 unidades, tanto para minerales como para gas vespeno. Inicialmente, dichos depósitos están vacíos.
 - Parámetros: Recurso, Localización
2. Modificar las acciones de **Construir** y **Reclutar** para consumir un cierto número de recursos. Estos recursos deben estar almacenados y ser suficientes para poder ejecutar la acción. Los costes de cada elemento (edificio o unidad) son los siguientes:

Elemento	Minerales	Gas Vespeno
Barracones	30	10
Extractor	10	0
VCE	5	0
Marine	10	15
Soldado	30	30

Las características del **problema** (disposición del mapa, localización de recursos, edificios y unidades) así como el **objetivo** del ejercicio son las siguientes: se debe construir un barracón (Barracon1) en LOC32, y se deben reclutar dos marines (Marine1 y Marine2) y un soldado (Soldado1), que serán destinados a las localizaciones LOC31, LOC24 y LOC12, respectivamente. Al igual que en todos los ejercicios anteriores, el único VCE disponible desde el principio (VCE1) se encuentra en LOC11 (misma localización del CentroDeMando1).

En este ejercicio no es obligatorio implementar las funciones y restricciones necesarias para que MetricFF devuelva el plan óptimo, como se especificaba en el ejercicio anterior, pero se valorará positivamente que el plan obtenido sea el más cercano al óptimo.

Ejercicio 8

Partir del ejercicio anterior (ejercicio 7) e incluir una nueva función para controlar el tiempo que tarda cada **acción** del plan:

1. La acción **Asignar** no tiene coste.



2. La acción **Recolectar** tiene un coste fijo de 5 unidades de tiempo.
3. La acción **Navegar** tiene un coste dependiente de la distancia entre dos localizaciones y la velocidad a la que se desplazan las unidades. La distancia entre cualesquiera dos localizaciones conectadas es de 20 unidades de distancia, mientras que las velocidades de las unidades son:
 - a. VCE: 1 unidad de distancia / unidad de tiempo
 - b. Marine: 5 unidades de distancia / unidad de tiempo
 - c. Soldado: 10 unidades de distancia / unidad de tiempo
4. Los tiempos de creación de cada elemento (edificio para la acción **Construir**, o unidad para la acción **Reclutar**) son los siguientes:

Elemento	Tiempo
Barracones	50
Extractor	20
VCE	10
Marine	20
Soldado	30

Incluir una métrica para encontrar un plan que minimice el tiempo del plan. Recuerde que, aunque se defina una métrica distinta a la longitud del plan (*plan length*), MetricFF aproxima el óptimo pero no garantiza que la solución encontrada lo sea. De nuevo, en este ejercicio no es estrictamente necesario implementar la estrategia de optimización del Ejercicio 6, pero igualmente se valorará positivamente que el plan obtenido sea lo más cercano posible al plan óptimo.

Las características del problema (disposición del mapa, localización de recursos, edificios y unidades) y el objetivo, al igual que en ejercicio anterior, son los siguientes: se debe construir un barracón (Barracon1) en LOC32, y se deben reclutar dos marines (Marine1 y Marine2) y un soldado (Soldado1), que serán destinados a las localizaciones LOC31, LOC24 y LOC12, respectivamente. Al igual que en todos los ejercicios anteriores, el único VCE disponible desde el principio (VCE1) se encuentra en LOC11 (misma localización del CentroDeMando1).

Nota: Se proporciona a los estudiantes el fichero `getCostFromPlanEj8.py`, que les permitirá calcular el coste total de su plan con facilidad y, de ese modo, poder analizar y estudiar sus soluciones con mayor comodidad y rapidez. Al comienzo de dicho fichero se proporciona información sobre su ejecución. Para poder utilizarlo se deben emplear exactamente los nombres de las acciones definidos hasta el momento: RECOLECTAR, RECLUTAR, CONSTRUIREDIFICIO, ASIGNAR, y NAVEGAR. También se deben emplear los nombres adecuados para las unidades (VCE, MARINE, SOLDADO) y los tipos de edificio (EXTRACTOR, BARRACON).



Ejecución de los dominios de planificación

Para el desarrollo de esta práctica usaremos el planificador MetricFF en su primera versión, que se puede descargar en:

<https://fai.cs.uni-saarland.de/hoffmann/ff/Metric-FF.tgz>

Nota: No se debe usar la versión 2.0 o 2.1 de este planificador, puesto que implementa un mayor número de técnicas de búsqueda y heurística que pueden dificultar la resolución de esta práctica.

Una vez compilado, se ejecutará con el siguiente comando:

```
$ ./ff -o <dominio.pddl> -f <problema.pddl> -O -g 1 -h 1
```

La opción “-O” le indica al planificador que se desea optimizar el plan encontrado (si bien no existen garantías de que el plan encontrado sea óptimo), mientras que las opciones “-g 1” y “-h 1” nos sirven para establecer la función de coste y la función heurística usada por A* para encontrar el plan. Usando estos valores se facilita que MetricFF se acerque a la solución óptima.

Evaluación y Entrega

Los estudiantes deben entregar en la actividad creada en PRADO, **antes de las 23:59 del 08 de junio de 2022**, una carpeta comprimida en formato ZIP con el siguiente contenido:

- **Un fichero de dominio y otro fichero de problema para cada uno de los ocho ejercicios planteados. Dichos ficheros deben respetar los nombres “dominioX.pddl” y “problemaX.pddl”, donde “X” corresponde al número del ejercicio.** Es decir, por cada ejercicio debe haber un fichero de dominio y otro de problemas.
- **Una breve memoria (3 páginas como máximo)** donde se responda razonadamente a las preguntas presentadas en la siguiente sección.

Todos los ficheros deben estar adecuadamente **comentados, explicando y justificando la solución presentada, así como las variables, tipos y predicados usados para codificar cada problema, y cualquier otro aspecto relativo a las decisiones tomadas durante el proceso de implementación. Un código cuyos comentarios sean, a criterio del profesor, extremadamente deficientes podrá invalidar total o parcialmente aquellos ejercicios a los que afecte.**

Nota: Los problemas con **errores sintácticos** en PDDL automáticamente califican con una puntuación de **0 (cero) puntos**.

La puntuación para la entrega será la siguiente:

- Ejercicio 1: 1 punto
- Ejercicio 2: 1 punto
- Ejercicio 3: 1 punto
- Ejercicio 4: 1 punto
- Ejercicio 5: 1 punto
- Ejercicio 6: 1 punto



- Ejercicio 7: 1 punto
- Ejercicio 8: 1 punto
- Memoria: 2 puntos.

Memoria

La memoria debe contener la respuesta a las siguientes preguntas:

1. Tabla de resultados (1 punto). Se pide entregar una tabla de resultados donde se especifique el número de acciones del plan encontrado (junto con su coste en el caso del Ejercicio 8) y el tiempo invertido por MetricFF para encontrar dicho plan para cada ejercicio. Valore los resultados obtenidos.
2. Pregunta #1 (0.5 puntos). En las distintas llamadas a MetricFF necesarias para resolver el Ejercicio 6, ¿MetricFF tarda aproximadamente el mismo tiempo en todas ellas? ¿A qué cree que se debe este fenómeno? Razone su respuesta.
3. Pregunta #2 (0.5 puntos). En base a los tiempos de ejecución obtenidos, ¿cree que el dominio de planificación planteado en esta práctica es de dificultad moderada/media/alta? ¿Cuáles son las limitaciones de la planificación automática en otros dominios?