



UNIVERSIDAD DE GRANADA

TÉCNICAS DE LOS SISTEMAS INTELIGENTES
GRADO EN INGENIERÍA INFORMÁTICA

PRÁCTICA 3

PLANIFICACIÓN HTN

Autor

Vladislav Nikolov Vasilev

Rama

Computación y Sistemas Inteligentes



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

CURSO 2018-2019

Índice

1. Ejercicio 1	2
2. Ejercicio 2	3
3. Ejercicio 3	4
4. Ejercicio 4	5
4.1. Problema 1	6
4.2. Problema 2	6
4.3. Problema 3	7
4.4. Problema 4	7

1. Ejercicio 1

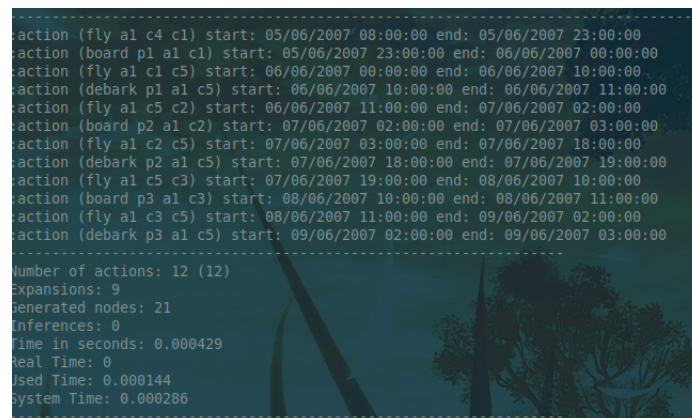
En este ejercicio se ha modificado la tarea de transportar a personas añadiendo un nuevo método en el cuál si el avión y la persona están en ciudades distintas, lo primero que se hace es trasladar el avión a la ciudad de la persona y luego transportarla, tal y como se hacia en el Caso2. A continuación se puede ver la tarea con la nueva funcionalidad añadida:

Listing 1: Modificación de la tarea de transportar personas.

```
(:task transport-person
  :parameters (?p - person ?c - city))

(:method WaitCatchFlight
  :precondition (and (at ?p - person ?c1 - city)
                    (at ?a - aircraft ?c2 - city)
                  )
  :tasks (
    (mover-avion ?a ?c2 ?c1)
    (board ?p ?a ?c1)
    (mover-avion ?a ?c1 ?c)
    (debark ?p ?a ?c)
  )
)
```

El resultado se puede ver a continuación:



```
.....
action (fly a1 c4 c1) start: 05/06/2007 08:00:00 end: 05/06/2007 23:00:00
action (board p1 a1 c1) start: 05/06/2007 23:00:00 end: 06/06/2007 00:00:00
action (fly a1 c1 c5) start: 06/06/2007 00:00:00 end: 06/06/2007 10:00:00
action (debark p1 a1 c5) start: 06/06/2007 10:00:00 end: 06/06/2007 11:00:00
action (fly a1 c5 c2) start: 06/06/2007 11:00:00 end: 07/06/2007 02:00:00
action (board p2 a1 c2) start: 07/06/2007 02:00:00 end: 07/06/2007 03:00:00
action (fly a1 c2 c5) start: 07/06/2007 03:00:00 end: 07/06/2007 18:00:00
action (debark p2 a1 c5) start: 07/06/2007 18:00:00 end: 07/06/2007 19:00:00
action (fly a1 c5 c3) start: 07/06/2007 19:00:00 end: 08/06/2007 10:00:00
action (board p3 a1 c3) start: 08/06/2007 10:00:00 end: 08/06/2007 11:00:00
action (fly a1 c3 c5) start: 08/06/2007 11:00:00 end: 09/06/2007 02:00:00
action (debark p3 a1 c5) start: 09/06/2007 02:00:00 end: 09/06/2007 03:00:00
.....
Number of actions: 12 (12)
Expansions: 9
Generated nodes: 21
Inferences: 0
Time in seconds: 0.000429
Real Time: 0
Used Time: 0.000144
System Time: 0.000286
.....
```

Figura 1: Plan obtenido en el ejercicio 1.

2. Ejercicio 2

En este ejercicio se ha modificado la tarea de mover el avión para que considere también el repostaje en caso de tener insuficiente fuel. Si no tiene suficiente, lo único que tiene que hacer primero es repostar y luego volar. A continuación se puede ver la implementación:

Listing 2: Modificación de la tarea de mover avión.

```
(:task mover-avion
:parameters (?a - aircraft ?c1 - city ?c2 -city)

(:method RefuelFlyAircraft
:precondition (not (hay-fuel ?a ?c1 ?c2))
:tasks(
  (refuel ?a ?c1)
  (fly ?a ?c1 ?c2)
)
)
)
```

También se ha añadido un predicado derivado para comprobar si se tiene suficiente combustible, el cuál es utilizado en la tarea del movimiento (es el predicado **hay-fuel**, el cuál está asociado a un avión y mira si hay combustible para ir de *c1* a *c2*. El plan que se obtiene del problema se puede ver a continuación:

```
.....
action (fly a1 c4 c1) start: 05/06/2007 08:00:00 end: 05/06/2007 23:00:00
action (board p1 a1 c1) start: 05/06/2007 23:00:00 end: 06/06/2007 00:00:00
action (refuel a1 c1) start: 06/06/2007 00:00:00 end: 16/06/2007 10:00:00
action (fly a1 c1 c5) start: 16/06/2007 10:00:00 end: 16/06/2007 20:00:00
action (debark p1 a1 c5) start: 16/06/2007 20:00:00 end: 16/06/2007 21:00:00
action (fly a1 c5 c2) start: 16/06/2007 21:00:00 end: 17/06/2007 12:00:00
action (board p2 a1 c2) start: 17/06/2007 12:00:00 end: 17/06/2007 13:00:00
action (refuel a1 c2) start: 17/06/2007 13:00:00 end: 27/06/2007 23:00:00
action (fly a1 c2 c5) start: 27/06/2007 23:00:00 end: 28/06/2007 14:00:00
action (debark p2 a1 c5) start: 28/06/2007 14:00:00 end: 28/06/2007 15:00:00
action (refuel a1 c5) start: 28/06/2007 15:00:00 end: 04/07/2007 21:00:00
action (fly a1 c5 c3) start: 04/07/2007 21:00:00 end: 05/07/2007 12:00:00
action (board p3 a1 c3) start: 05/07/2007 12:00:00 end: 05/07/2007 13:00:00
action (refuel a1 c3) start: 05/07/2007 13:00:00 end: 11/07/2007 19:00:00
action (fly a1 c3 c5) start: 11/07/2007 19:00:00 end: 12/07/2007 10:00:00
action (debark p3 a1 c5) start: 12/07/2007 10:00:00 end: 12/07/2007 11:00:00
.....
Number of actions: 16 (16)
Expansions: 9
Generated nodes: 25
Inferences: 0
Time in seconds: 0.000963
Real Time: 0
Used Time: 0.000963
System Time: 0
.....
```

Figura 2: Plan obtenido en el ejercicio 2.

3. Ejercicio 3

En este ejercicio se ha modificado la tarea de mover el avión para considerar distintos tipos de desplazamientos, priorizando siempre que se realice lo más rápido posible. Esto se hace al especificar el orden en el que se aplican los métodos. Primero se van a comprobar aquellos en los que el avión puede realizar un “zoom”, y si no puede, se mirará si puede realizar un vuelo normal. También se han añadido 2 nuevos predicados derivados para saber si un avión tiene suficiente combustible para hacer un “zoom” o un vuelo normal, y se ha modificado el predicado derivado del apartado anterior para comprobar si tiene combustible para los dos tipos de vuelos (cada uno consume una cantidad diferente).

El primer método que se comprueba es si el avión puede realizar un “zoom” sin repostar, es decir, si tiene suficiente fuel para ir al destino, y en caso de poder hacerlo, solo se desplazará el avión mediante un “zoom”. El otro método comprueba si puede desplazarse con un “zoom” pero sin tener suficiente combustible, en cuyo caso repostará primero y luego se desplazará mediante un “zoom”. Otro método será si puede realizar un vuelo lento y tiene suficiente combustible para hacerlo, en cuyo caso volará al destino. Y el último método comprueba si puede realizar un vuelo lento pero no tiene suficiente combustible, en cuyo caso repostará primero y luego volará.

Debido a que la implementación ocupa demasiadas líneas, la explicación ofrecida anteriormente se considera suficiente y se recomienda mirar el dominio del ejercicio para ver exactamente como está implementada. Vamos a ver, no obstante, cuáles son los resultados al resolver el problema:

```
:action (refuel a1 c4) start: 05/06/2007 08:00:00 end: 09/06/2007 12:00:00
:action (zoom a1 c4 c1) start: 09/06/2007 12:00:00 end: 09/06/2007 20:00:00
:action (board p1 a1 c1) start: 09/06/2007 20:00:00 end: 09/06/2007 21:00:00
:action (refuel a1 c1) start: 09/06/2007 21:00:00 end: 22/06/2007 09:00:00
:action (zoom a1 c1 c5) start: 22/06/2007 09:00:00 end: 22/06/2007 14:00:00
:action (debark p1 a1 c5) start: 22/06/2007 14:00:00 end: 22/06/2007 15:00:00
:action (refuel a1 c5) start: 22/06/2007 15:00:00 end: 30/06/2007 23:00:00
:action (zoom a1 c5 c2) start: 30/06/2007 23:00:00 end: 01/07/2007 07:00:00
:action (board p2 a1 c2) start: 01/07/2007 07:00:00 end: 01/07/2007 08:00:00
:action (refuel a1 c2) start: 01/07/2007 08:00:00 end: 13/07/2007 20:00:00
:action (zoom a1 c2 c5) start: 13/07/2007 20:00:00 end: 14/07/2007 04:00:00
:action (debark p2 a1 c5) start: 14/07/2007 04:00:00 end: 14/07/2007 05:00:00
:action (refuel a1 c5) start: 14/07/2007 05:00:00 end: 26/07/2007 17:00:00
:action (fly a1 c5 c3) start: 26/07/2007 17:00:00 end: 27/07/2007 08:00:00
:action (board p3 a1 c3) start: 27/07/2007 08:00:00 end: 27/07/2007 09:00:00
:action (refuel a1 c3) start: 27/07/2007 09:00:00 end: 02/08/2007 15:00:00
:action (fly a1 c3 c5) start: 02/08/2007 15:00:00 end: 03/08/2007 06:00:00
:action (debark p3 a1 c5) start: 03/08/2007 06:00:00 end: 03/08/2007 07:00:00
-----
Number of actions: 18 (18)
Expansions: 10
Generated nodes: 31
Inferences: 0
Time in seconds: 0.002198
Real Time: 0
Used Time: 0.001649
System Time: 0.000549
-----
```

Figura 3: Plan obtenido en el ejercicio 3.

4. Ejercicio 4

En este ejercicio se han modificado tanto el dominio como las primitivas para poder permitir embarcar múltiples pasajeros, imponer límites de tiempo para los aviones y cambiar la forma en la que se representa el límite de combustible para hacer que cada avión tenga el suyo. Con tal de no superar el número máximo de páginas, se van a describir de palabra las modificaciones realizadas, sin entrar en demasiado detalle en la implementación:

- Se han modificado las primitivas **board** y **debark**. Ahora la acción de embarcar comprueba que no se haya llegado al límite dado de pasajeros para ese avión, y en caso de que no suceda, se embarca el pasajero y se cuenta un nuevo pasajero embarcado. En la acción de desembarcar, se comprueba que no se haya vaciado completamente el avión, y en caso de que no sea así, se desembarca el pasajero.
- Se han creado dos nuevas tareas para embarcar y desembarcar a múltiples pasajeros de forma recursiva, comprobando siempre los destinos de las personas para saber si deben o no subir o bajar de ese avión.
- Se ha añadido un predicado para representar el destino de una persona determinada, tal y como se ha especificado en el enunciado.
- Se han añadido una serie de nuevas funciones y predicados derivados. Entre las funciones se incluyen, por ejemplo, el número de pasajeros que lleva un determinado avión en ese momento, la cantidad máxima de pasajeros que puede llevar un avión, el límite de tiempo de un avión (un avión no puede realizar un viaje de duración superior al que se le ha especificado en el límite de tiempo; si por ejemplo se le ha especificado un límite de tiempo de 20 horas, no puede realizar un vuelo de duración superior) y se ha modificado el límite de combustible para que cada avión tenga el suyo propio. En cuanto a los predicados derivados, se han añadido dos nuevos predicados que sirven para comprobar si la acción de volar lento o rápido (mediante un “*zoom*”) se pasan del tiempo límite. Estos predicados derivados serán utilizados en las precondiciones de los métodos de vuelo para, además de comprobar si se tiene combustible suficiente, comprobar que no se supere el límite de tiempo establecido. Los tiempos se calculan utilizando las distancias entre las ciudades y las velocidades.

Para especificar las tareas objetivo se tiene que especificar a dónde se quiere transportar cada persona, tal y como se hacía en los anteriores ejercicios. Por tanto, en las tareas objetivo aparecerán una serie de **transport-person**, donde la ciudad

a la que se quiere transportar la persona tiene que ser la misma que la ciudad destino, ya que en caso contrario se tendrán problemas.

Respecto a la estrategia que se sigue en este ejercicio en cuanto a tiempos de viaje y gasto de combustible, se sigue un enfoque igual que en el ejercicio anterior. Es decir, se prioriza el tiempo de viaje (por tanto, **zoom** tendrá una mayor prioridad que **fly**), aumentando por tanto el gasto de combustible. Esto se debe a que minimizar los dos objetivos a la vez es imposible, ya que volar en menos tiempo implica gastar más combustible, y gastar menos combustible implica viajar más lento. Por tanto, se ha elegido seguir un enfoque que priorice los tiempos de viajes, dando por tanto prioridad a la hora de expandir el árbol de decisiones a la acción **zoom** y después a **fly**. No obstante, se ha tenido en cuenta cuáles son las capacidades de combustible de los aviones y los tiempos máximos que estos pueden viajar. Como cada uno tiene sus propias características, el planificador se encargará de escoger la más adecuada. En cuanto a las políticas de embarque se ha intentado que se embarquen todos los pasajeros posibles cuyo destino no es la ciudad actual en la que se encuentran. Sin embargo, el planificador se encargará de decidir quiénes deben embarcar, lo cuál puede resultar en que casi siempre se dejen pasajeros en el aeropuerto ya que no comparten destino, creando por tanto planes subóptimos.

A continuación se van a explicar brevemente los problemas planteados. Los planes no van a ser incluidos en la memoria, debido a que son más largos. Si se quieren consultar, se han adjuntado algunos archivos que los contienen.

4.1. Problema 1

Se ha planteado un problema con 12 personas y todos los aeropuertos españoles con las mismas distancias que aparecen en el enunciado. Además, se ha planteado utilizar 6 aviones con velocidades, consumos, capacidades y número máximo de pasajeros distintos. Todos ellos tienen la misma restricción de tiempo de 300 horas, a las cuáles no van a llegar, siendo por tanto esta restricción inexistente. El planificador consiguió encontrar un plan, el cuál se puede ver en el archivo *Solucion_problema1.txt*, donde se puede ver que alterna entre vuelos rápidos y lentos con cada avión procurando no superar la cantidad máxima de combustible establecido.

4.2. Problema 2

Se ha realizado una modificación del problema anterior, cambiando el destino de una persona y reduciendo el número de aviones a 4. En este caso, el planificador

de nuevo consiguió encontrar un plan, alternando los vuelos lentos y las rápidos con tal de no superar la cantidad máxima de combustible. El resultado se puede ver en *Solucion_problema2.txt*.

4.3. Problema 3

Se ha modificado el problema 1 cambiando las horas máximas de vuelo de cada avión, reduciéndolas, algunas en mayor medida y otras en menor. El plan obtenido es diferente al que se había obtenido inicialmente, y se puede ver en el archivo *Solucion_problema3.txt*.

4.4. Problema 4

Se ha modificado el problema 1 añadiendo personas hasta un total de 20, modificando los destinos de aquellas personas que ya estaban en el problema 1. Se han añadido más aviones hasta un total de 9, poniéndoles unas capacidades lo suficientemente grandes para poder hacer algunos recorridos y modificando el número de horas máximo por trayecto, asignando de forma manual un número de horas razonable, ya que hay trayectos que llevarían más horas que otros. También se han modificado algunos consumos y velocidades, debido a que en caso contrario, no se hubiese encontrado una solución satisfactoria al problema. Todas estas decisiones (destinos, consumos, velocidades, etc.) han sido tomadas a base de probar si el planificador era capaz de encontrar algún plan para una configuración de predicados y funciones. Debido a que en muchos casos el planificador ha estado hasta un total de 40 minutos sin encontrar ningún plan, finalmente tras mucho probar se ha escogido la configuración de valores que se ve reflejada en el archivo del problema. Esto también ha influido en que la mayoría de personas tienen como destino ciudades cercanas a ellos o la misma ciudad en la que están, ya que cambiando ligeramente la descripción del problema se podía llegar a una en la que no había solución.

Con los valores especificados en la descripción del problema, el planificador pudo encontrar una solución, aunque le llevó algo de tiempo, más que en los casos anteriores. Esta solución se puede ver en el archivo *Solucion_problema4.txt*.