Alberto García Doménech
Pablo Daurell Marina

Práctica 2B (Ascensores)

Inteligencia artificial

Representación de estados

La representación elegida consta de dos tuplas, una para las personas y otra para los ascensores:

estado_ini = ((personas), (ascensores))
estado_goal = (personas)

El índice de la tupla indica la persona/ascensor y el valor la planta en la que se encuentra.
El último ascensor de la lista siempre es un ascensor rápido (se mueve entre plantas pares).

Las restricciones de nuestra representación son que cada ascensor solo puede llevar a una persona a la vez, siempre hay exactamente un ascensor rápido y los ascensores se mueven de forma secuencial.
Con esta representación podemos añadir tantas personas y ascensores normales como queramos.

Representación de acciones

Hemos definido tres acciones distintas: Llamar ascensor, Subir planta y Bajar planta.

- Llamar ascensor mueve dicho ascensor desde su planta actual a una planta donde se encuentre una persona.
- Subir planta y Bajar planta mueven a un ascensor y una persona (que se encuentran en la misma planta) una planta arriba o abajo respectivamente.

Hemos definido tres acciones más para el ascensor rápido, igual que las anteriores pero moviéndose sólo entre plantas pares: Llamar Rápido, Subir Rápido y Bajar Rápido.

Función de coste

La función de coste que hemos definido atribuye un coste de 1 a las acciones de Subir y Bajar (ya que suben una planta por acción) y un coste de el número de plantas que se ha movido el ascensor para la acción de Llamar.
En el caso de las acciones del ascensor rápido, el coste de todas ellas se reduce a la mitad (como el ascensor rápido solo se mueve por las plantas pares, el coste mínimo de subir y bajar sería 1 porque sube las plantas de dos en dos, nunca tendrá un coste de 0,5 ya que nunca se mueve sólo una planta).

Heurísticas

Antes de añadir el ascensor rápido definimos dos heurísticas para los ascensores normales:

- **descolocados**: Cuenta el número de personas que están en una planta distinta a la del destino.

- **distancias**: Calcula la distancia de cada persona a su planta destino.

• La heurística **descolocados** es admisible ya que si tenemos N personas mal colocadas, el valor de la heurística será N, mientras que el coste será mayor o igual que N ya que para colocar a esas personas deberemos realizar al menos N acciones y todas las acciones cuestan al menos 1.
(P.ej. Si tenemos a una persona mal colocada y sólo necesita moverse una planta, nuestro coste real sería 1 y nuestra heurística nos daría un coste de 1 también, si la persona necesitase moverse más de una planta o tuviese que llamar a un ascensor el coste real aumentaría y nuestro coste heurístico seguiría siendo menor)

Si añadiésemos un ascensor rápido (cuyo coste fuese la mitad de uno normal) esta heurística seguiría siendo admisible ya que si hay N personas descolocadas el coste heurístico será N y como mínimo las personas se tendrán que desplazar una vez, lo cual supone un coste real de N*1, y nuestras acciones tienen como coste mínimo 1 (a un ascensor normal le cuesta 1 subir o bajar y a uno rápido, al moverse sólo de dos en dos, también).

Esta heurística será consistente tanto con ascensor rápido como sin él. El coste para pasar de un estado a otro será 1 (subir/bajar) o más (llamar). Si en un estado tenemos N personas descolocadas en el siguiente estado tendremos N o N-1 descolocados, y N-N o N-(N-1) siempre será menor o igual que 1, por tanto la heurística es consistente.

• La heurística **distancias** también es admisible por el mismo motivo que la anterior.
El valor máximo de esta heurística se daría en el caso de que todas las personas estuvieran M plantas alejadas de su destino (siendo M el número total de plantas), lo que daría un coste heurístico igual a N*M (siendo N el número total de personas). Dado que todas las acciones valen al menos 1, el coste total va a ser siempre mayor o igual que el coste de la heurística.
(P.ej. Si tenemos a una persona en la planta 0 y quiere subir a la 12, eso supondría un coste real de 12 y un coste heurístico de 12 también, pero si además tiene que llamar a un ascensor el coste real aumentaría mientras que el coste heurístico no).

También es una consistente porque el mayor cambio que podemos realizar de un estado a otro es modificar la distancia de una persona a su destino en 1 (subiendo o bajando) lo que supondrá que la diferencia de los costes heurísticos de un estado y del siguiente siempre será de 1 como mucho y como nuestras acciones valen 1 o más podemos concluir que es una heurística consistente.

Si añadiésemos un ascensor rápido podría darse el caso de tener que subir dos plantas (ambas pares) lo cual supondría un coste heurístico de 2 pero un coste real de 1 (usando el ascensor rápido), por lo que esta heurística dejaría de ser admisible
Por este mismo caso la heurística también dejaría de ser consistente ((2 - 0)> 1).

• Entre estas dos heurísticas, *distancias* sería mejor debido a que se acerca más al valor del coste real, N < N*M (distancias es más informada que descolocados).

Al añadir un ascensor rápido, hemos creado una nueva heurística:

- **distancias2**: Calcula la distancia de cada persona a su planta destino, teniendo en cuenta que el ascensor rápido reduce los costes a la mitad. Teniendo en cuenta la paridad de las plantas podemos definir una heurística más cercana al coste real sin llegar a superarlo (usando la división entera: //).

• Esta nueva heurística (**distancias2**) es admisible tanto con ascensores normales como con un ascensor rápido ya que el coste de mover un ascensor entre dos plantas pares es D/2 (siendo D la diferencia entre plantas) como mínimo (usando el ascensor rápido) mientras que el coste real de mover un ascensor entre una planta par e impar es como mínimo (D+1//2), por lo que el coste heurístico nunca será mayor al coste real.

Además vemos que esta heurística es consistente ya que la máxima diferencia de valor heurístico entre estados adyacentes es de 1: si nos movemos entre plantas pares la diferencia entre ellas siempre va a ser de dos, por lo que 2//2 = 1 y si en cambio vamos de una par a una impar o viceversa la diferencia de plantas siempre va a ser de uno, por lo que (1//2)+1 = 1.
El coste real de la transición entre dos estados es como mínimo 1, en el caso de subir o bajar(ya sea con un ascensor rápido o no) ya que mover una planta con un ascensor normal cuesta 1 y mover dos con uno rápido cuesta también 1. El coste podría ser mayor si hubiese que llamar a un ascensor a una planta.
Por lo tanto siempre se cumple que la diferencia entre heurísticas es menor o igual al coste real.

Como se ha mencionado antes, la heurística *descolocados* seguiría siendo admisible en el caso de añadir el ascensor rápido, sin embargo *distancias2* sería más informada. Por ejemplo, si tenemos una persona en la planta 0 que quiere subir a la 12 tendríamos un coste real de 12, con *descolocados* el coste sería 1 y con *distancias2* el coste sería 6 que se acerca mucho más al coste real.

Distintas instancias del problema

Como es de esperar, cuantas más personas y más ascensores haya en el problema, más larga será la solución, ya que aumentará el número de acciones posibles. También influirá lo alejadas que estén las personas de su destino, cuanto más lejos, más larga será la solución. Sin embargo, lo alejados que estén los ascensores no influye en la solución (ya que acercarlos solo supone una acción), pero si estos están en la misma planta que las personas la solución se simplificará un poco (ya que no hay que llamar a los ascensores).
El estado de los ascensores no afectará significativamente a la longitud de la solución aunque si influirá en el proceso de resolución del problema, ya que, salvo que todas las personas estén en la misma planta que todos los ascensores, surgirán muchas posibilidades en las que alguien pueda llamar a un ascensor. Por otro lado, si ningún ascensor coincide con ninguna persona las posibilidades serán menores ya que no habrá acciones de subir, ni de bajar.
Otro detalle a tener en cuenta son las personas que se encuentren en plantas pares y cuyo destino también lo sea, por lo que será más probable que usen el ascensor rápido y lleguen antes a su destino, acortando la solución.

Modificaciones del problema

Con la representación elegida:

• Cambiar plantas de origen y destino: simplemente modificamos los valores de la tupla de estado_ini y el valor de la tupla estado_goal con las nuevas plantas.

• Tener en cuenta el tiempo de subir y bajar pasajeros para cambiar de ascensor: dado que todos los ascensores llegan a todas las plantas, no hay personas que cambien de ascensor al cambiar de bloque.

• Incluir nuevos pasajeros: añadimos un nuevo elemento a la tupla estado_ini y a la tupla estado_goal que represente la planta inicial donde se encuentra el nuevo pasajero y la planta a la que quiere llegar.

• Cambiar capacidad de los ascensores: debido a la implementación escogida la capacidad de todos los ascensores es de 1, por lo que no podríamos modificarla fácilmente.

• Añadir nuevos ascensores: podemos añadir todos los ascensores normales que queramos, tan solo hace falta añadirlos a la segunda tupla de estado_inicial. Debido a la representación del ascensor rápido que hemos elegido no es posible añadir fácilmente otro tipo de ascensores rápidos, ya que utilizamos la última posición de la tupla para indicar el ascensor rápido que se mueve entre plantas pares (por lo que además siempre tiene que haber un ascensor rápido).

Análisis de resultados

[Las pruebas de ejecución se encuentran en el jupyter notebook de la práctica]

Hemos usado los algoritmos de búsqueda en estrella (astar_search) y de primero el mejor (best_fisrt_graph_search). Sólo hemos tenido en cuenta las heurísticas descolocados y distancias2, debido a que el resto son inadmisibles para nuestra representación actual (con un ascensor rápido).

- Dos personas en plantas pares y dos ascensores: ((0,0),(0,0)) -> (4,10)
    - A* (descolocados): 97,9 ms
    - A* (distancias 2): 17,7 ms
    - Primero el mejor (descolocados): 8,74 ms
    - Primero el mejor (distancias2): 482 µs
- Dos personas en plantas impares y dos ascensores: ((1,3),(0,0)) -> (7,5)
    - A* (descolocados): 187 ms
    - A* (distancias2): 162ms
    - Primero el mejor (descolocados): 7,36 ms
    - Primero el mejor (distancias2): 3,16 ms
- Dos personas en plantas pares e impares y dos ascensores: ((2,4),(0,0)) -> (7,1)
    - A* (descolocados): 208 ms
    - A* (distancias2): 119 ms
    - Primero el mejor (descolocados): 4,52 ms
    - Primero el mejor (distancias2): 3,9 ms

- Tres personas y dos ascensores: ((2,4,1),(0,0)) -> (10,1,8)
    - A* (descolocados): 1,56 min
    - A* (distancias2): 5,03 s
    - Primero el mejor (descolocados): 151 ms
    - Primero el mejor (distancias2): 5,99 ms
- Dos personas y tres ascensores: ((2,4),(0,1,6)) -> (10,1)
    - A* (descolocados): 13 s
    - A* (distancias2): 716 ms
    - Primero el mejor (descolocados): 1,58 s
    - Primero el mejor (distancias2): 2,99 ms
- Cinco personas y dos ascensores: ((2,4,1,8,1),(0,0)) -> (3,11,12,1,9)
  (No usamos A en este caso porque tarda demasiado)*
    - Primero el mejor (descolocados): 7,08 s
    - Primero el mejor (distancias2): 2,41 s

De todas estas pruebas vemos que al añadir más ascensores aumenta el tiempo de ejecución y al añadir más personas aumenta aún más.
Si las personas están y van a plantas pares, el tiempo se reduce sobretodo cuando usamos la heurística *distancias2.*
A* es más eficiente con la heurística *distancias2* ya que es más informada que *descolocados*. Además encuentra siempre una solución óptima, sin embargo esto hace que cuantos más elementos haya en el estado inicial más tarde en ejecutarse.
Por otro lado Primero el mejor también es más eficiente con *distancias2* que con *descolocados.* No encuentra siempre una solución óptima pero es más eficiente (no gasta tiempo buscando la solución más óptima)
Si buscamos la solución más óptima de la manera más eficiente lo mejor es usar A* con la heurística *distancias2*.
Sin embargo, si buscamos la máxima eficiencia independientemente de que la solución sea óptima o no, lo mejor es usar Primero el mejor con la heurística *distancias2*.

Alberto García Doménech
Pablo Daurell Marina