Práctica 2B: Problema de los Ascensores

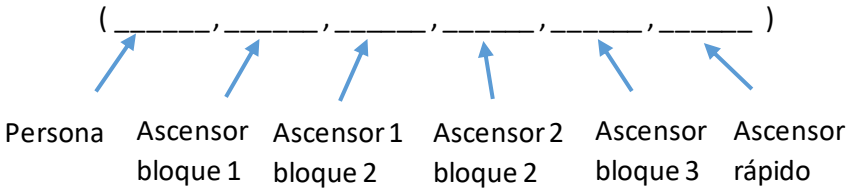
Resoluciones con AIMA

Ignacio de la Cruz Crespo
Sergio José Gómez Cortés
Aroa Ayuso Muñoz

1. Resolución del problema reducido.

Para tomar contacto con la práctica primero decidimos resolver un problema simplificado en el que había 1 persona y 5 ascensores.

Para ello definimos el estado como una tupla de 6 elementos en la que, el primero elemento era la persona, el segundo el ascensor del bloque 1, el tercero, el primer ascensor del bloque 2, el cuarto, el segundo ascensor del bloque dos, el quinto, el ascensor del bloque tres y el sexto, el ascensor rápido.



Las acciones del problema las definimos por cada caso posible que se podía plantear, así quedó que las posibles acciones eran "Bajar X", "Subir X", "Bajar X con P", "Subir X con P", siendo P la persona y "X" el ascensor que se quiere mover, nos quedaron 20 acciones. Como en el problema original, estas acciones cumplen las restricciones que el ascensor del bloque 1 sólo puede ir de la planta 0 a la 4, los del bloque 2 de la 4 a la 8, el del boque 3 de la planta 8 a la 12 y el rápido de la 0 a la 12 parando únicamente en las pares. Como el goal está formado por la persona y los ascensores, para evitar que la persona se esté moviendo, se comprueba que la persona esté en la planta que indica el goal para que no se añada la acción de mover ascensor con P, de esta forma evitamos que la persona se nos descoloque mientras los ascensores llegan a las plantas que indica el goal.

Para generar el estado resultante de aplicar la acción, en el método result comprueba la acción e incrementa o decrementa en 1 el valor de la persona y del ascensor que tenga esa acción, en el caso de los lentos e incrementa o decrementa en 2 en el caso del rápido.

Se diseñaron dos heurísticas, la primera tiene en cuenta únicamente la distancia a la que está la persona de su objetivo. Esta heurística es admisible ya que siempre se va a dar que la heurística es menor que el coste real, esto es porque para pasar de un estado a otro el coste real siempre es 1, por lo que para llegar a un objetivo es el número de movimientos que tienes que hacer, mientras que para la heurística sólo cuentas los movimientos que tienes que hacer para colocar a la personas, si esta ya está colocada, el valor es 0 frente al número de movimientos que sigues contando para colocar los ascensores, que es el coste real. Esta heurística también es consistente debido a que, para cualquier par de nodos, el número de movimientos que los separa va a ser siempre mayor o igual a la diferencia de la heurística, ya que por cada movimiento se diferencia como mucho en 1 la heurística de un estado al del siguiente hasta llegar al objetivo, así la suma de las diferencias que sería el camino completo, como mucho es el valor de movimientos del camino, que es el coste real.

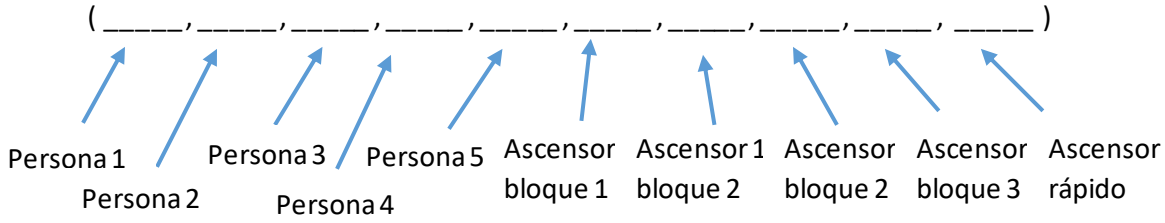
Como, tras ejecutar el problema con esta heurística, los resultados eran peores que en alguna búsqueda ciega, decidimos cambiarla y añadir el resto de los elementos a la operación. Esta heurística también es lineal y consiste en sumar las distancias entre el estado actual y el objetivo de todos los componentes de la tupla, es decir, de la persona y los ascensores. Esta heurística no es admisible esto es porque para el caso de estado actual (1,1,4,4,8,12) en el que la persona

y el ascensor están en la misma planta, y estado objetivo (2,2,4,4,8,12), el valor de la heurística es 2 pero el coste real es 1, ya que con la acción subir ascensor 1 con persona se llega.

2. Resolución del problema casi completo

Una vez hicimos que la primera parte funcionara decidimos ampliar el problema, para esta resolución escogimos 5 personas, 5 ascensores, uno que estuviera en el bloque 1, dos en el bloque 2, uno en el bloque 3 y el rápido que pasa por los pares, para este problema no tuvimos en cuenta que hubiera un máximo de personas por ascensor.

La definición de estados que hemos escogido es una tupla de 10 elementos, los 5 primeros corresponden a las personas y los 5 últimos a los ascensores, por orden, el del primer bloque, los del segundo bloque, el del tercero y el rápido.



Las acciones que se pueden realizar son: “Subir ALB1”, Subir el ascensor del bloque 1, “Bajar ALB1”, bajar el ascensor del bloque 1, “Subir AL1B2”, subir el ascensor 1 del bloque 2, “Bajar AL1B2”, bajar el ascensor 1 del bloque 2, “Subir AL2B2”, subir el ascensor 2 del bloque 2, “Bajar AL2B2”, bajar el ascensor del bloque 2, “Subir ALB3”, subir el ascensor del bloque 3, “Bajar ALB3”, bajar el ascensor del bloque 3, “Subir AR”, subir el ascensor rápido, “Bajar AR”, bajar el ascensor rápido.

Para generar estas acciones a partir de un estado, en el método actions cogemos las posiciones de los ascensores y comprobamos si están en la posición más baja en la que se permite, en caso de que no, se añade a la lista de acciones que pueden bajar, si tampoco están en la planta más alta en la que se les permite, se añade que pueden subir:

- Si el ascensor del bloque 1 “ALB1” no está en la 0 entonces puede bajar, si no está en la 4, puede subir.
- Si los ascensores del bloque 2 “AL1B2” y “AL2B2”, no están en la 4, pueden bajar, si no están en la 8, pueden subir.
- Si el ascensor del bloque 3 “ALB3” no está en la 8 puede bajar, si no está en la 12, puede subir.
- Si el ascensor rápido no está en la 0 puede bajar y si no está en la 12 puede subir.

En el método result, para modificar el estado lo que hacemos es:

Por cada acción recorremos las personas y si está en la misma planta que el ascensor que estamos moviendo, y si esa persona quiere subir, en el caso que la acción sea subir ascensor x, o bajar, en el caso de que la acción sea bajar ascensor x, esa persona se mueve con el ascensor, para ello comparamos la posición en el estado actual con la posición en el goal. Si la persona está en la planta a la que quiere ir, no se moverá.

Aunque los ascensores formen parte del estado, como lo que queremos es que las personas lleguen a la planta que quieren independientemente de dónde se coloque el ascensor, hemos definido un goal test en el que comprobamos si el estado es goal fijándonos solo en las personas.

Búsquedas ciegas

Para probar estas búsquedas hemos usado como estado inicial (1, 3, 0, 7, 0, 0, 4, 4, 8, 6) como goal para las personas es: (2, 4, 1, 10, 1). Los resultados obtenidos han sido:

Búsqueda	Tiempo	Longitud camino	Nodos expandidos
Primero en profundidad con control de repetidos	1,44 seg	539	540
Primero en profundidad sin control de repetidos	No terminó	--	--
Primero en anchura sin control de repetidos	--	--	--
Primero en anchura con control de repetidos	6,27s	10	5302

Hemos escogido estos estados, inicial y goal, ya que es sencilla y la solución se encuentra a una profundidad de 10 nodo, esto nos permite que estas búsquedas puedan encontrar la solución en un tiempo no muy elevado para poder hacer mejor las comparaciones y el análisis.

Tanto para anchura como profundidad sin control de repetidos, tree search, la búsqueda no termina, después de un tiempo elevado ejecutando nos devuelve un memory error, genera demasiados nodos, en el caso de primero en profundidad y en el primero en anchura hace que el sistema operativo reinicie el ordenador. Haciendo un print de los estados que abre hemos podido ver como entra en bucles en los que sube y baja los mismos ascensores constantemente.

Para las búsquedas con control de repetidos, graph search, ambos encuentran una solución. Para la búsqueda en profundidad tarda menos ya que abre menos nodos, pero a cambio la solución que encuentra tiene una longitud mucho mayor, ya que es la hoja de la rama más a la izquierda, esta solución tiene una longitud de 539. Para la búsqueda en anchura el tiempo es mayor que el anterior dado que abre 5302 nodos, pero a cambio encuentra el camino más corto, que es de longitud 10, esto es porque todos los estados tienen al menos 5 acciones posibles por lo que en cada nivel el número de estados que se crean escala muy rápido haciendo que se tengan que abrir una cantidad de nodos muy elevada aun estando a una distancia no muy grande de la raíz.

Búsquedas informadas

Para intentar mejorar los resultados anteriores vamos a implementar una heurística que ayude en la búsqueda A*. Hemos creado tres heurísticas:

Para las dos primeras heurísticas se calculan creando una matriz en la que las columnas son las personas y las filas los ascensores, para cada celda (i, j) se calcula cuantas plantas tiene que recorrer el ascensor i para llegar a la persona j más las plantas que recorrería después hasta el objetivo de esa persona. Si el objetivo está fuera del rango del ascensor se suma igualmente, pero si el ascensor no puede llegar hasta esa persona se tiene como nulo y no se tiene en cuenta.

Basándonos en esa matriz la primera heurística coge el resultado menor de esta y es el valor para ese nodo. Es admisible porque el valor que coge es como mucho el coste real de colocar a una única persona. Como solo se tiene en cuenta de colocar a la persona con menor coste y además en caso de que el objetivo esté fuera de rango, no se cuentan los movimientos que tiene que hacer el ascensor que sí llega, hasta la persona, por esto este valor va a ser siempre menor que el coste real.

Para la segunda heurística nos quedamos con el recorrido mínimo de cada ascensor, es decir, el valor mínimo de cada fila, que después vamos sumando y es lo que devolvemos. No es admisible ya que existen varios ejemplos en los que no se cumple que para un nodo el valor de la heurística sea menor o igual al coste real, un ejemplo es el siguiente:

Para el nodo: (1,4,4,1,1,1,6,8,9,0) con objetivo en (1,6,8,1,1), la heurística vale 13 (0+4+6+3) mientras el coste real es 3, subiendo el ascensor rápido a la 4, coger a las dos personas, subir a la 6, se baja una y sube a la 8, se baja la otra.

La matriz que se calcula para la heurística es la siguiente:

	Orig: 1 Des:1	Orig: 4 Des:6	Orig: 4 Des:8	Orig: 1 Des:1	Orig: 1 Des:1
Ascensor 1	0	5	7	0	0
Ascensor 2	-	4	6	-	-
Ascensor 3	-	6	8	-	-
Ascensor 4	-	-	-	-	-
Ascensor 5	-	3	4	-	-

La tercera es lineal, se basa en recorrer el estado y el goal y hacer la suma acumulada de la distancia entre la posición actual y su destino final, sólo tiene en cuenta a las personas. Esta heurística no es admisible ya que el coste de un nodo al siguiente es 1 y el valor de la heurística cuanta 1 por cada planta que tiene entre medias de su objetivo, para cada persona, por lo que para el ejemplo nodo actual (1, 1, 1, 2, 3, 1, 4, 4, 8, 10) y queremos ir al nodo (2, 2, 2, 2, 3 …) el valor de la heurística es 3 mientras que el coste real es 1, porque con un único movimiento "Subir ascensor 1" se llega al nodo objetivo. Como no se cumple que para todo nodo el valor de la heurística es menos o igual al coste real, esta heurística no es admisible.

Heurística	Tiempo	Longitud camino	Nodos explorados
Heurística 1	39 min 21 seg	10	56426
Heurística 2	49 min 28 seg	12	56426
Heurística 3	1,94 seg	10	634

Tras ejecutar la búsqueda podemos decir que, la tercera es la más eficiente ya que, aunque la longitud de las soluciones es muy similar dos de ellas encuentran la óptima y la segunda con una longitud de dos nodos más, la que consigue encontrar una solución en menor tiempo es la tercera, con una diferencia muy elevada, esta lo resuelve en apenas dos segundos, un tiempo muy bueno dado la complejidad del problema, y las otras dos en 40 y 50 min. Con esto y comparándolo con las búsquedas ciegas podemos decir que las dos primeras heurísticas en vez de informar lo que hacen es desinformar haciendo que explore nodos que en otros casos no exploraría por estar más alejados de la solución.

Para las búsquedas con heurísticas, en este caso A*, con una heurística lineal, la tercera implementada, el tiempo de ejecución es corto, de 2 seg, da la solución de mejor longitud y analiza menos nodos. Se puede concluir que es la más eficiente.

Cabe destacar que, aunque en este problema en particular los tiempos de las búsquedas ciegas con control de repetido es muy similar al de A* con la tercera heurística, en problemas más complejos en los que la solución no se encuentra tan cerca de la raíz, los tiempos de las búsquedas ciegas es del orden de minutos mientras que el de A* es de segundo.

3. Cuestiones de la memoria

Dada la representación que hemos escogido sí se pueden cambiar las plantas de origen y destino de cualquiera de los pasajeros, esto es porque a la hora de crear el problema se le introducen dos tuplas una de longitud 10, que va a ser el estado inicial de los pasajeros, las 5 primeras posiciones, y de los ascensores, las 5 posiciones siguientes y otra tupa de 5 elementos, que va a llevar las posiciones a las que quieren ir cada persona, es decir, el goal. Estas tuplas pueden tener los valores que se quieran, siguiendo las restricciones del problema.

No se tiene en cuenta el tiempo que tarda un pasaje en cambiar de ascensor, quedarse en la planta o meterse en el ascensor. Lo que se hace es mover al pasajero a la panta a la que le corresponde según la acción y se da por hecho que se baja en cuanto llega, en cada movimiento, si puede y quiere, se monta en un ascensor se desplaza y se baja.

Para incluir nuevos pasajeros habría que modificar algunos valores usados en bucles, pero escalar el código para que se pudieran introducir más personas sería algo sencillo, la representación del problema seguiría siendo el mismo.

No le hemos puesto capacidad límite a los ascensores.

Al igual que para los pasajeros, añadir nuevos ascensores supondría cambios pequeños como añadir nuevas acciones y sus restricciones en el result que no supondrían cambios muy grandes en el volumen y estructura del código, la representación seguiría siendo igual.