Departamento de Informática
Universidad Carlos III de Madrid

Ingeniería en Informática

Inteligencia Artificial

Noviembre 2007

Hoja de Ejercicios 3: Funciones heurísticas

Comentarios generales sobre los ejercicios

- Asumiendo que se conocen los contenidos teóricos, el tiempo estimado para realizar los ejercicios es de **2 horas**
- Describir las soluciones a los ejercicios de una manera lo más formal posible

Ejercicio 1

En la orilla de un río hay 3 misioneros y 3 caníbales y todos ellos pretenden cruzar al otro lado. La barca que se utiliza para cruzarlo sólo tiene capacidad para dos personas, con lo que alguien ha de estar volviendo siempre a la orilla inicial mientras quede gente sin cruzar. Además, si en alguna ocasión y en cualquiera de las orillas se encuentran un número mayor de caníbales que de misioneros, los primeros se comerán a los segundos.

1. ¿Cómo representarías los estados?
2. ¿Cuáles serían los operadores?
3. ¿Qué heurísticas existen para este problema?¿Son admisibles?

Ejercicio 2

"Las torres de Hanoi" es un juego matemático ideado en el siglo XVIII. Este juego consiste en pasar 64 discos de diámetro decreciente, de un poste a otro poste, utilizando un tercer poste auxiliar para los pasos intermedios, tal y como muestra la Figura 1.

Figura 1: Juego de las Torres de Hanoi para 8 discos.

Cada vez sólo se puede mover un disco, los discos siempre deben estar en algún poste y no se puede colocar un disco sobre otro de menor tamaño.

1. ¿Cómo representarías los estados?
2. ¿Cuáles serían los operadores?
3. ¿Qué heurísticas existen para este problema?¿Son admisibles?

Ejercicio 3

En algunas aplicaciones reales, el espacio de problemas se suele formular con un único estado inicial, s, y un conjunto arbitrariamente grande de estados finales o meta $\Gamma : \{t_1, t_2, \ldots, t_n\}$. Considerando los espacios de problemas formulados de esta manera, se pide:

- Dada una función heurística admisible $h(n, m)$ que estima el esfuerzo para llegar hasta un único estado m desde otro n, ¿es posible obtener una nueva función de evaluación heurística, h_Γ, que resuelva el problema anterior y sea admisible?

Ejercicio 4

La red de Metro de Madrid esta compuesta por 12 líneas que se entrecruzan. De esta forma, es frecuente que para ir de una estación a otra existan diferentes alternativas, tal y como se muestra en la Figura 2. Asumiendo que representamos esta red como un árbol de búsqueda responde a las siguientes preguntas:

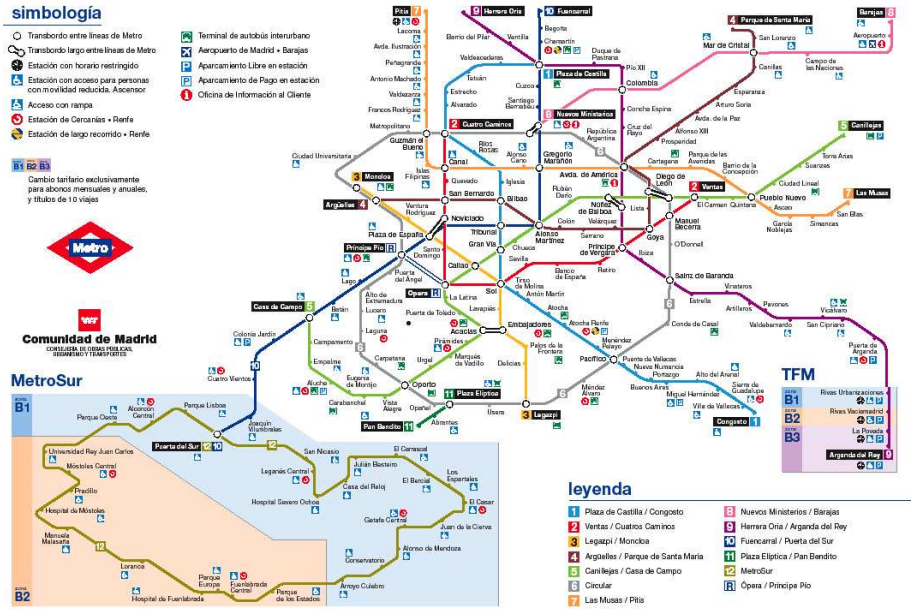

Figura 2: Mapa del Metro de Madrid

1. ¿Cómo representarías un estado en este dominio? ¿Cómo representarías los operadores que permiten pasar de un estado a otro?
2. ¿Cuál es el factor de ramificación del árbol de búsqueda?
3. Obtener una heurística.
4. Describe las heurísticas que utilizamos las personas para determinar el camino más corto entre 2 estaciones.
5. ¿El valor de estas heurísticas sería el mismo si utilizáramos el mapa real de Madrid en lugar de un esquema de la red de Metro?

Ejercicio 5

Un acertijo consiste en dados 4 números y un resultado determinar las operaciones de suma o resta que hay que realizar sobre los números para obtener ese resultado. Por ejemplo:

- Números: 1,4,3,2
- Resultado: 0
- Solución: 4 - 3 -2 + 1

Suponiendo que resolvemos el acertijo como un problema de búsqueda, responde las siguientes cuestiones:

1. Propón una representación de los estados y explica como se generarían los estados sucesores.
2. ¿Cuál sería el tamaño del espacio de estados. ¿Y si el acertijo en lugar de ser con 4 números es con 5? Propón una fórmula general. ¿Cuántos nodos generaría el algoritmo de amplitud si buscara tosas las posibles soluciones?
3. ¿Qué tipo de algoritmo de búsqueda no informada sería mejor utilizar? ¿Por qué?
4. Define heurísticas para este problema. ¿qué otros mecanismos podríamos incluir para hacer la búsqueda más eficiente?

Solución Ejercicio 1

Estados

Se podría indicar la posición de la barca, junto con el número de misioneros y caníbales que hay en cada lado.

Se podría pensar que, dado que el número de personas en el extremo final puede calcularse a partir de los que hay en el inicial, basta con indicar la posición de la barca y los misioneros y caníbales que quedan en el extremo inicial. Sin embargo, la primera aproximación permite describir más fácilmente las precondiciones y efectos de los operadores, por lo que mantenemos la representación inicial. Además, el espacio de estados tiene el mismo tamaño e idéntica semántica con ambas representaciones.

Formalmente, por tanto, un estado es una terna (M_i, C_i, B, M_f, C_f) en la que:

- $B \in [i, f]$ indica la posición de la barca, por lo que toma el valor i si está en el extremo inicial, o f si está en el final
- $M_i, C_i, M_f, C_f \in [0, \ldots, 3]$ indican el número de misioneros y caníbales que quedan en el extremo inicial y final del río, respectivamente

De esta manera, el estado inicial se representa como $(3, 3, i, 0, 0)$ y el final como $(0, 0, f, 3, 3)$

Operadores

El conjunto de operadores es el mostrado en al Tabla 1. Se dispone de 5 operadores. Cada uno de ellos consiste en transportar personas de la orilla x a la orilla y: $Mover1C(x, y)$ transporta 1 caníbal desde la orilla x hasta la orilla y; $Mover2M(x, y)$ tranporta 2 misioneros desde x hasta y; y así sucesivamente. Las precondiciones de cada operador se dividen en tres grupos de requisitos. El primer grupo (primera columna de precondiciones) hace referencia a que para transportar a una persona desde x, esa persona debe estar en x. Así, el operador $Mover1M1C(x, y)$ requiere que en x haya al menos un caníbal y un misionero. El segundo grupo de condiciones (segunda columna) hace referencia a la posción de la barca: para todos los operadores, la barca debe estar en la posición de origen. El tercer grupo de condiciones evita que los caníbales se coman a los misioneros, evitando hacer movimientos que dejen a más caníbales que misioneros en cualquier orilla.

Operador	Precondiciones			Efectos
Mover1C(x,y)	$C_x \geq 1$	$B = x$	$M_y \geq C_y + 1 \vee M_y = 0$	$B = y$, $C_x = C_x - 1, C_y = C_y + 1$
Mover2C(x,y)	$C_x \geq 2$	$B = x$	$M_y \geq C_y + 2 \vee M_y = 0$	$B = y$, $C_x = C_x - 2$, $C_y = C_y + 2$
Mover1M(x,y)	$M_x \geq 1$	$B = x$	$(M_x \geq C_x + 1 \vee M_x = 1) \wedge M_y \geq C_y - 1$	$B = y$, $M_x = M_x - 1$, $M_y = M_y + 1$
Mover2M(x,y)	$M_x \geq 2$	$B = x$	$(M_x \geq C_x + 2 \vee M_x = 2) \wedge M_y \geq C_y - 2$	$B = y$, $M_x = M_x - 2$, $M_y = M_y + 2$
Mover1M1C(x,y)	$M_x \geq 1$,$C_x \geq 1$	$B = x$	$M_y \geq C_y$	$B = y$, $M_x = M_x - 1$, $M_y = M_y + 1, C_x = C_x - 1, C_y = C_y + 1$

Cuadro 1: Operadores del problema de los caníbales y los misioneros

En la tabla también se incluyen los efectos del operador, que suele incluir el cambio en la posición de la barca, así como el cambio en el número de caníbales y misioneros en cada orilla.

Heurísticas

En este caso, vamos a obtener las heurísticas por relajación del problema original. Para dicha relajación, partimos de las precondiciones expuestas en la Tabla 1. Estas precondiciones se pueden relajar fácilmente teniendo en cuenta los tres tipos de precondiciones definidas anteriormente:

1. Eliminar primer grupo de precondiciones. Si eliminamos ese primer grupo de condiciones, se obtiene un problema relajado que no parece ser mucho más fácil de resolver que el problema original, por lo que no tiene mucho sentido.
2. Eliminar segundo grupo de precondiciones. Al eliminar el segundo grupo de condiciones, el problema resultante es más fácil que el original, puesto que se puede asumir que hay infinitas barcas tanto en un lado como en otro. Este problema tiene una solución muy sencilla, que es asumir que siempre viajan un caníbal y un misionero juntos, con el operador $Mover1M1C(i, f)$. Por tanto, la heurística resultante de este problema relajado es:

$$h_1(n) = \frac{C_i + M_i}{2} \tag{1}$$

asumiendo que en el estado n se cumplen los requisitos definidos por el grupo de precondiciones 3.

3. Al eliminar el tercer grupo de condiciones, obtenemos un problema relajado en el que los caníbales nunca se comen a los misioneros. Entonces, en cada viaje de ida y vuelta, podemos transportar a una persona (dado que la otra tendrá que volver para llevar la barca). Por tanto, la heurística resultante es:

$$h_2(n) = \begin{array}{ll} 2 \text{ x } \max(0,\ C_i + M_i - 2) + 1, & \text{Si B=i} \\ 2 \text{ x } (C_i + M_i) & \text{, Si B=f} \end{array} \tag{2}$$

Una cuarta posibilidad sería eliminar el grupo de condiciones 2 y 3 a la vez. Sin embargo, como hemos visto anteriormente, este problema es equivalente a eliminar sólo el grupo de condiciones 2.

Las dos heurísticas son admisibles, puesto que son resultado de relajar el problema original. Para decidir qué heurística elegir, h_1 o h_2, estudiamos cuál es la más informada, puesto que será la que, siendo admisible, tendrá un valor más cercano a h^*. Se observa fácilmente que la más informada es h_2, puesto que $h_2(n) \geq h_1(n)$, sea cual sea el valor de C_i y de M_i para el estado n. Por tanto, elegimos $h_2(n)$

Solución Ejercicio 2

Estados

Un estado puede representarse como una lista con los discos que hay en cada uno de los postes. En caso de tres postes, tenemos una lista con tres sublistas. Cada disco viene representado con un identificador que indica además su diámetro.

- Estado: (P_1, P_2, P_3), donde $P_i (i = 1, 2, 3)$ es una lista de valores entre 1 y 64
- Estado inicial: ((1, 2, 3, 4, 5, ..., 62, 63, 64) ()())
- Estado final: (()()(1, 2, 3, 4, 5, ..., 62, 63, 64))

Existe un único operador, $Mover(P_i, P_j)$ que mueve un disco del poste P_i al poste P_j, por lo que $i, j \in [1, 2, 3], i \neq j$.

Las precondiciones del operador son:

1. $P_i \neq []$
2. $(P_j = []) \vee$ (para $P_i = [x|L_i], P_j = [y|L_j], x<y$)

Donde L_i y L_j contienen la lista de discos que hay en P_i y P_j respectivamente, pero eliminando el primer disco. Los efectos del operador $Mover(P_i, P_j)$, donde $P_i = [x|L_i], P_j = [y|L_j]$, son:

- $P_i = [L_i]$
- $P_j = [x|y|L_j]$

Heuristica

Para generar una heurística, obtenemos un problema relajado a partir del original, simplemente eliminando alguna precondición del operador $Mover$. En este caso, no tiene sentido eliminar la precondición 1, puesto que eso añadiría más discos a nuestro problema, así que eliminamos sólo la precondición 2. Al eliminar esta segunda precondición, resolver el problema es muy sencillo. Por ejemplo, si se supone la situación inicial, en la que todos los discos están en el primer poste, sólo hay que ir moviendo todos los discos uno a uno al poste intermedio, y de ahí nuevamente uno a uno al definitivo. Esto produce dos movimientos por cada disco que no esté colocado correctamente en el poste adecuado, excepto para el último (que puede ir directamente al poste final sin pasar por el intermedio). Para otros estados distinto al inicial, esta forma de resolver el problema es, al menos, una cota

inferior, asegurando que la heurística es admisible para todos los estados del problema origina. Esto nos genera la siguiente heurística:

$$h_1(n) = 2 \times \|P_1\| + 2 \times \|P_2\| - 1 \rightarrow h(n) = 2 \times (\|P_1\| + \|P_2\|) - 1 \tag{3}$$

donde el estado n es $(P_1 P_2 P_i)$, $\|P_1\|$ es el número de elementos en el poste P_1, y $\|P_2\|$ es el número de elementos en el poste P_2. Para ser exactos, a la heurística anterior habría que restarle 1 por cada poste en el que haya discos (es decir, si los dos postes tienen discos, se le resta 2, y si sólo uno de ellos tiene discos, se le resta 1). Si no se tiene en cuenta este detalle, la heurística se puede simplificar a $h_2(n) = (\|P_1\| + \|P_2\|)$.

Sin embargo, esta heurística sólo es válida si asumimos que los discos colocados en el poste final están, en verdad, bien colocados (desde el de mayor tamaño, al de menor, con diferencias de 1 en el diámetro de cada disco colocado). En caso contrario, la heurística debería penalizar también los discos colocados en el tercer poste, haciendo que:

$$h_3(n) = 2 \times (\|P_1\| + \|P_2\| + \|P_3\|) - 1 \tag{4}$$

Aun así, el valor de esta heurística todavía no corresponde con el valor real del coste óptimo del problema relajado. Sin embargo, todas las heurísticas descritas están menos informadas que la del problema relajado, y subestiman el coste. Por tanto, son también admisibles, aunque menos informadas.

Otra posible relajación de este problema es la resultante de eliminar la restricción de que sólo hay tres postes. Es decir, podemos asumir que hay tantos postes intermedios como sean necesarios. Con esta relajación obtenemos la misma heurística definida anteriormente.

Solución Ejercicio 3

Dada una función heurística $h(n, m)$ que estime el esfuerzo para llegar desde un estado n hasta otro cualquiera m, es posible estimar, asimismo, el esfuerzo **mínimo** para llegar desde un estado n hasta otro cualquiera t_i, de una colección de estados $\Gamma : \{t_1, t_2, \ldots, t_n\}$ como sigue:

$$h_\Gamma(n, \Gamma) = \min_{i=1,k} \{h(n, t_i)\}, \quad t_i \in \Gamma$$

Obviamente, si la función heurística $h(n, m)$ es admisible y, por lo tanto, no sobreestima nunca el esfuerzo para llegar hasta el estado m, la función heurística h_Γ será asimismo admisible, puesto que toma el mínimo de las distancias hasta uno cualquiera de los nodos en el conjunto Γ.

Solución Ejercicio 4

Estados y Operadores

Un estado viene indicado por una tupla $< estacion, linea >$ que indica la estación en la que nos encontramos. Dado que el coste de continuar un viaje en el mismo vagón es el mismo que el de hacer un trasbordo, no es necesario indicar la línea de metro en la que nos encontramos, puesto que toda la red se puede considerar como un grafo no dirigido.

Existen dos operadores. El primero, $viajar(i, j, l)$, permite trasladar al viajante desde la estación i hasta la j a través de la línea l. Por tanto, las precondiciones del operador son:

1. $estacion = i$
2. $conectado(i, j, l)$, donde $conectado(i, j, l)$ indica si en el grafo del metro existe una línea de metro, l que conecta las estaciones i y j directamente, y sin ninguna otra estación intermedia.

El efecto de ejecutar el operador $viajar(i, j, l)$ es que $estacion = j$, mientras que linea permenece invariante.

El segundo operador, $trasbordo(i, l_m, l_n)$ permite al viajante cambiar de la línea l_m a la línea l_n en la estación i. Las precondiciones de este operador son:

1. $estacion = i$
2. $parada(l_m, i) \wedge parada(l_n, i)$, donde el predicado $pasa(l, i)$ indica que la línea l tiene una parada en la estación i

Factor de Ramificación

Dado un estado, el factor de ramificación viene dado por el número de trasbordos que puedo realizar, más dos (uno por cada sentido de la marcha). A este valor hay que eliminar 1 si la estación en la que se encuentra el viajante es una estación principio o final de línea.

Heurísticas

Cualquier relajación del problema original nos lleva a problemas que se resuelven en un único paso. Sin embargo, si hacemos que el coste de cada paso no sea uniforme, sino que venga dado por la distancia euclídea a la meta, entonces podemos usar la distancia euclídea como heurística.

Heurística de Personas

- Distancia visual del mapa: Según a donde queramos ir vamos eligiendo las opciones que visualmente se vayan acercando a nuestra parada de destino. El problema se resueve visualmente, y en caso de duda, se cuenta el número de estaciones.
- Número de trasbordos: Además suele influir en la selección de opciones la ruta que presente un menor número de trasbordos, dado que el coste de un trasbordo suele ser muy alto con respecto al de ir de una estación a otra por la misma línea.

Uso del mapa real de Madrid

El mapa de metro suele ser un mapa esquemático, en el que las distancias en las longitudes de cada tramo es sólo aproximado. En un mapa de Madrid, las distancias son reales, pudiéndose utilizar la distancia euclídea como heurística. Por ejemplo, si tienes en el mapa esquemático un cuadrado y vas a ir a vértices opuestos, tu valor heurístico "a ojo" te dirá que ambos lados te salen igual, hasta que lo miras en el mapa real de la red y de das cuenta de que las distancias son muy distintas. Esto es un ejemplo de cómo el tipo de representación puede variar las soluciones que podemos obtener, y de cómo una definición inexacta de los costes de los operadores puede llevarnos a soluciones subóptimas.

Solución Ejercicio 5

Un estado viene representado por una tupla $< s_1, s_2, s_3, s_4 >$, en la que $s_i \in [o, +, -]$ representa el signo del número en la posición i. El valor o indica que aun no se ha asignado ningún signo.

El estado inicial es la tupla $< o, o, o, o >$. El estado final viene representado implíciatemente, de forma que debe cumplier:

- $s_i \neq o$
- $\sum_{i=1} 4 s_i n_i$, donde $s_i n_i$ es el número original n_i con el signo s_i.

Tamaño del Espacio de Estados

El tamaño del espacio de estados es 3^n, donde n es la cantidad de números que nos dan en el problema. Ahora bien, el número de nodos que generaría el algoritmo de amplitud depende de los operadores que se definan.

Una posible opción es que en cada nivel i se asigne un signo al número i-ésimo. En el ejemplo, en el primer nivel asignaríamos un signo al número 1, en el segundo nivel al número 4, en el tercero al número 3 y en el último al 2. Por tanto, tenemos un factor de ramificación de 2, con una profunidad de 4, con lo que obtenemos un número de nodos de $2^4 = 16$ nodos. Este valor es muy inferior aal de 3^4, puesto que hay estados a los que, con esta representación de los operadores, nunca se llegaría.

Mejor algoritmo de búsqueda no informada

Profundidad, ya que todas las soluciones tienen la misma profundidad en el árbol.

Heurísticas

Tal y como está definido el problema, los operadores no tienen restricciones, con lo que no es posible obtener un problema relajado que nos proporcione una heurística.

Sin embargo, se pueden introducir mecanismos que hagan la búsqueda más eficiente mediante la poda de ramas que, por conocimiento del dominio, se puede saber que no llevarán a la solución.

La primera de ellas es la paridad. Si en un nodo del árbol tenemos un número par, el objetivo es impar, y todos los números que tenemos que sumar y restar son pares, entonces no tiene sentido seguir esa rama. Se pueden generar varias reglas de poda siguiendo este sistema.

Otro mecanismo es la distancia a meta. Por ejemplo, si el valor actual de un estado dista del valor objetivo más que la suma de los valores restantes por asignar, también se puede hacer la poda de esa rama.