TEMA 3. BUSQUEDA INFORMADA O HEURISTICA

En los métodos de búsqueda no informada del tema anterior, la búsqueda se realizaba explorando sistemáticamente los nodos del grafo, a veces teniendo en cuenta lo ya gastado en la exploración de la parte del camino recorrido (coste uniforme), pero sin utilizar ninguna información disponible (que en ocasiones puede existir) sobre preferencia de unas vías sobre otras o sobre proximidad de las metas en las partes de los caminos aún por recorrer.

Cuando se dispone de alguna clase de información en este segundo sentido se habla de ***heurísticas*** y los métodos correspondientes son los ***métodos heurísticos*.**

En ellos, básicamente lo que se hace es utilizar dicha información para elegir el desarrollo de unos estados frente a otros, tomando para desarrollar de la lista de abiertos antes aquellos que aparenten estar más cerca de la meta.

EJEMPLOS

En el problema del 8-*puzzle*: el número de piezas por colocar o la suma de las distancias de cada pieza descolocada a su posición definitiva son indicadores de la "distancia" de cada estado hasta la solución.

En problemas de laberintos o de mapas: distancias (aérea o en cuadrícula) desde cada estado a la meta.

FUNCIONES HEURÍSTICAS. BÚSQUEDA HEURÍSTICA

La información sobre proximidad a la meta consiste en una función numérica ***h*** del espacio de estados a $[0, \infty [= R^{+} U \{0\}$, que estime la "distancia" de cada estado al objetivo más próximo, valiendo **0** en las metas.

Esta estimación es información incierta (de no serlo, desaparecería el problema(heurística perfecta, h*, su valor es desconocido en problemas reales)), y puede conducir por sí sola a engaños. Al usarla se sacrifica certeza para obtener economía, "podando" el árbol de búsqueda para centrarse en las ramas o direcciones más prometedoras.

Combinando adecuadamente la información heurística (sobre el futuro, incierta) con la del gasto realizado (sobre el pasado, conocida), se puede establecer métodos seguros y económicos, que combinan útilmente las ventajas respectivas de ambas informaciones

MÉTODOS VORACES

Búsqueda en *Escalada (o irrevocable, ingl. Hill Climbing)*

Esta búsqueda, rudimentaria, sin memoria, consiste en seguir siempre sólo el camino hacia el que, según lo apuntado por la función heurística, parezca mejor sucesor, mientras éste mejore al estado en que se esté:

1. Empezar con *ACTUAL* = lista formada por el estado inicial
2. Hasta que *ACTUAL* = meta o no haya cambios en *ACTUAL* , hacer:
   a) Presentar la lista de *ACTUAL* si su estado final es una meta y parar
   b) Tomar los sucesores de *ACTUAL* y usar *h* para puntuar cada uno de ellos.
   c) Si uno de los sucesores tiene mejor (o sea, menor valor de *h*) puntuación que *ACTUAL*, hacer *ACTUAL* igual a la lista anterior aumentada en el sucesor.
3. Presentar FALLO y terminar

Búsqueda *Primero el mejor*

Como la anterior, usa la heurística para elegir siempre al nodo sucesor aparentemente mejor situado, pero usa memoria (manteniendo abiertas las posibilidades alternativas mediante una lista de ABIERTOS) y así varios caminos posibles, no sólo uno sin alternativas.

Funciona de modo análogo al de la búsqueda (desinformada) optimal o de coste uniforme gestionando la lista ABIERTOS como una cola de prioridad, pero usando sólo la función heurística *h* para ordenarla y priorizar, examinando antes los nodos que tengan valor heurístico más bajo (que , según la información aportada por *h* estén “más cerca” del objetivo). Esta búsqueda es similar a la búsqueda en profundidad en el modo que prefiere seguir un camino hacia el objetivo, pero cuando llegue a un camino sin salida, volverá hacia atrás.

Por supuesto, la obtención de resultado y la calidad del mismo, dependerá de lo fidedigna que sea la función heurística.

1. Empezar con *ABIERTOS* = lista formada por el estado inicial
2. Mientras *ABIERTOS* no esté vacio hacer:

   2.1 Quitar de *ABIERTOS* la lista con el mejor nodo terminal y ponerla en *ACTUAL*

   2.2 Si el nodo terminal de *ACTUAL* es meta, presentar su lista y terminar.

   2.3 En otro caso, tomar los sucesores de dicho nodo, usar *h* para puntuar cada uno de ellos e incorporar las listas resultantes de ampliar *ACTUAL* con dichos nodos a la lista *ABIERTOS* en su orden.

3. Presentar FALLO y terminar.

Si se dispone de una buena función heurística, la búsqueda *primero el mejor* puede lograr en la práctica sustanciales descensos en el coste computacional de su ejecución, si bien no queda garantizado que el camino obtenido sea óptimo. Hay que tener además en cuenta el coste suplementario que la evaluación de *h* en cada nodo suponga. Además de que esta búsqueda sufre los mismos defectos que la búsqueda en profundidad, no se óptima ni completa

El ALGORITMO A^*

La idea en la que se basa la búsqueda **A*** consiste en intentar combinar las ventajas de la búsqueda desinformada de coste uniforme (completitud y optimalidad) con las de la búsqueda heurística primero el mejor (eficiencia, por la poda del árbol de búsqueda).

Ello se consigue priorizando en la lista de abiertos aquellos caminos en que menor resulte una **combinación** de los valores del **gasto hecho** en la parte del camino recorrido (suma de los costes de sus aristas etapa) con la **distancia** restante hasta la meta según la información dada por *h* en el trozo de camino por recorrer:

Para hacerlo, en el paso 2.3 del algoritmo anterior se substituye la función ***h*** por la ***f*** dada por

$$f(nodo)=g(nodo)+h(nodo)$$

lo que ordena a los nodos de *ABIERTOS* según la media aritmética de los valores de ***g*** y de ***h*** o por alguna otra combinación adecuada que agregue ambos valores. Siendo g(nodo) el coste del camino que va desde estado inicial hasta nodo y h(nodo) el coste estimado del camino más barato desde n al estado objetivo.

Si la heurística es adecuada (admisible), y el grafo no es "patológico", con este método se consigue un algoritmo completo, óptimo y con un coste computacional menor, equivalente en la práctica una disminución del factor de ramificación.

Como grafo no patológico se entiende aquel grafo en el que ningún nodo tiene infinitos sucesores (lo que no significa que el grafo sea finito) y en el que el coste de todas las aristas del grafo sea mayores que una cierta cantidad ε positiva y real.

PROPIEDADES DE A^* Y DE LAS HEURÍSTICAS

Si se denotan con h^* a la heurística ideal, que daría la información perfecta sobre el coste óptimo, por el mejor camino, desde cada nodo hasta la meta más cercana, y con $f^*(n)=g(n)+h^*(n)$ al coste total de la mejor solución que pase por n, dada una posible función heurística ***h***, puede ocurrir:

a) Que ***h(n)=0*** para todo ***n***, es decir, que la heurística no dé información. En este caso, A^* se convierte en la búsqueda optimal.

b) Que $h(n)=h^*(n)$ para cada ***n***, entonces no habría problema, pues en cada paso se sabría qué dirección hay que seguir para llegar a la meta más cercana del inicio por el mejor camino.

c) Que $h(n) > h^*(n)$ para algún ***n.*** Entonces A^* puede resultar no óptimo.

d) Que $h(n) \leq h^*(n)$ para cada ***n.*** Entonces A^* termina y resulta ser completo, óptimo y con coste computacional tanto más reducido cuanto mayor sea ***h*** (es decir, cuanto más parecida sea ***h*** a h^*), oscilando entre la búsqueda – a ciegas- de coste uniforme del caso (a) y la ausencia se búsqueda del caso (b).

Las heurísticas consideradas en (d) se llaman ***admisibles*.**

Se dice que una heurística h(n) es optimista cuando se cumple que para todo n, 0<=h(h)<=h*(n), es decir que dicha h(n) subestima el coste real de llegar desde cada estado n a la meta más cercana

Una heurística ***h*** es ***más informativa*** que otra ***h´*** si cumplen $0 \leq h'(n) \leq h(n) \leq h(n)^*$ para cada ***n***. Cuanto más informativa sea una heurística, más económica resultará la búsqueda mediante ella con A^*, es decir, menos nodos habrá que examinar para encontrar la solución.

Una heurística es ***consistente*** cuando para cada par de nodos adyacentes ***n*** y ***n´*** con el segundo sucesor del primero cumple que $h(n)-h(n') \leq C(n,n')$ **.** Es ***monótona*** si se cumple que siempre que ***n´*** sea un sucesor de ***n*** se tendrá que $f(n') \geq f(n)$. Cuando una heurística es consistente, entonces también es monótona y se cumple que la primera vez que un nodo sea escogido de la lista *ABIERTOS* para ser examinado, se habrá llegado desde el inicial hasta él por el camino más corto posible(camino óptimo) (es decir, que en la aplicación de ***A**** no habrá que "rectificar" ningún camino)

Nota: las pruebas de estas afirmaciones, véase Nilsson 9.2.1-9.2.4 o Poole-Mackworth 3

Búsqueda heurística con memoria limitada

Estos mecanismos van orientados a la evitación del crecimiento exponencial producido por los procedimientos de búsqueda heurística. Uno de ellos es el algoritmo IDA.

Algoritmo IDA (Korf)

Funciona de forma muy parecida al descenso iterativo convencional, basado en la ejecución de sucesivas búsquedas en profundidad. Este algoritmo se basa en una serie de iteraciones buscando con f(n)=g(n)+h(n) con límitse de profundidad que viene dado por h(inicial) y sus sucesivas mejoras.