Diseño de Algoritmos - Algoritmos II

Nazareno Aguirre
Departamento de Computación
Facultad de Ciencias Exactas, Físico-Químicas y Naturales
Universidad Nacional de Río Cuarto

Clase 9: Búsqueda

Estrategias de Diseño de Algoritmos

Ya hemos visto una variedad de estrategias de diseño de algoritmos, que nos brindan herramientas para atacar el problema de construir soluciones algorítmicas para problemas.

Veremos ahora una familia de técnicas de diseño de algoritmos, que denominaremos Técnicas de Búsqueda. Dado que muchos problemas pueden formularse en términos de búsqueda (como veremos más adelante), estas técnicas tienen un alcance importante.

Problemas de Búsqueda

Un gran número de problemas puede ser resuelto aplicando técnicas de búsqueda.

Para poder aplicar esta técnica, es esencial poder describir el problema como una búsqueda de configuraciones exitosas a partir de ciertas configuraciones iniciales, mediante la aplicación de reglas predefinidas de reconfiguración o avance.

En general, un problema P corresponde a un problema de búsqueda cuando se cuenta con:

- una descripción precisa de qué constituye el estado o configuración
- una descripción del estado de partida (estado inicial)
- una descripción de aquellas configuraciones que se consideran exitosas (estados finales)
- una descripción de cómo se puede avanzar, en un movimiento, desde un estado hacia otros estados sucesores

El problema P es entonces encontrar, a partir del estado de partida, alguna configuración exitosa, usando las reglas de avance predefinidas.

Ejemplo: El Problema de las Jarras de Agua

Consideremos el siguiente problema:

> Tenemos dos jarras, con capacidades de 4 y 3 litros respectivamente. Las jarras no tienen líneas de medición, y queremos que la primera quede con exactamente 2 litros de agua.
>
> Contamos con una fuente de agua ilimitada, y podemos llenar alguna de las jarras, o volcar el contenido de una de las jarras en la otra.

estados posibles: la combinación de los posibles contenidos de las jarras A (de 0 a 4 litros) y B (de 0 a 3 litros)

estado inicial: las dos jarras vacías

estados finales exitosos: la jarra A con 2 litros, y la jarra B con cualquier contenido

reglas de avance: (i) vaciar jarra A, (ii) vaciar jarra B, (iii) llenar jarra A, (iv) llenar jarra B, (v) volcar el contenido de A en B (hasta que A se vacíe o B esté llena), (vi) volcar el contenido de B en A (hasta que B se vacíe o A esté llena)

Árboles de Búsqueda

Usando las descripciones de los elementos anteriores podemos construir un árbol de búsqueda, de la siguiente forma:

- los estados posibles son los nodos del árbol
- el estado inicial es la raíz del árbol
- Un estado e_h es hijo de otro estado e_p en el árbol ssi existe un movimiento (atómico) que lleva de la configuración e_p a la configuración e_h

El problema se reduce entonces a encontrar un camino que nos lleve del estado inicial a alguno de los estados exitosos.

Exploración del Espacio de Estados

Podemos intentar construir un camino desde la raíz del árbol de búsqueda hasta alguno de los nodos exitosos (es decir, una solución al problema) mediante la visita a los nodos del árbol.

Ya conocemos, de otras asignaturas, algunas estrategias simples de visita de nodos en árboles:

- Primero a lo ancho (breadth-first search): visita de los nodos por niveles hasta encontrar un estado exitoso
- Primero en profundidad (depth-first search): visita de los nodos en profundidad, considerando los hijos de un nodo antes que sus hermanos, hasta encontrar un estado exitoso

Ventajas y Desventajas de las Estrategias de Búsqueda Simples

Primero a lo ancho:

- Ventajas: Si existe un estado exitoso de profundidad finita, entonces será encontrado. Encuentra soluciones de profundidad mínima.
- Desventajas: Es necesario almacenar un conjunto de nodos exponencial con respecto al nivel que se está explorando.

Primero en profundidad:

- Ventajas: Sólo es necesario almacenar un conjunto de nodos linealmente proporcional a la profundidad que se está explorando.
- Desventajas: Las soluciones encontradas no son necesariamente de profundidad mínima. Aún habiendo estados exitosos de profundidad finita, puede no encontrarlos (puede perderse en ramas infinitas sin soluciones).

Una Combinación de Estrategias Simples de Búsqueda

R. Korf publicó en 1985 una estrategia que puede verse como una combinación inteligente de las estrategias simples primero en profundidad y primero a lo ancho. Esta estrategia se llama profundización iterativa (iterative deepening) y consiste en realizar búsquedas depth-first sucesivas, cada una con un límite de profundidad incrementado en uno con respecto al anterior. Es decir:

1. realizar depth-first search hasta profundidad 1
2. si no se encontró una solución, realizar depth-first search hasta profundidad 2
3. si no se encontró una solución, realizar depth-first search hasta profundidad 3

...

Ventajas y Desventajas de Iterative Deepening

Desventajas

- Se vuelve a visitar múltiples veces un gran número de nodos

Ventajas

- Si existe un estado exitoso de profundidad finita, entonces será encontrado
- Encuentra soluciones de profundidad mínima
- Sólo es necesario almacenar un conjunto de nodos linealmente proporcional a la profundidad que se está explorando
- Las múltiples visitas a nodos no afecta la tasa de crecimiento asociada al tiempo de ejecución (similar a depth-first search)

Otros Ejemplos

Otros ejemplos de problemas que pueden formularse como problemas de búsqueda son los siguientes:

- búsqueda de salidas en laberintos
- el problema de las 8 baldosas (8-tile problem, o 8-puzzle)
- el problema de las 8 reinas
- Acertijos del estilo del de Indiana Jones, o cruce del río con la gallina, el zorro, el maíz y el granjero
- juegos (e.g., Sudoku)