UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

Métodos de Búsqueda

Alan Reyes-Figueroa

Inteligencia Artificial

(Aula 05b) 22.enero.2024

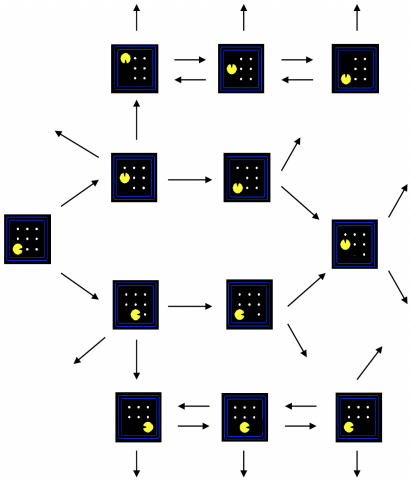
Espacios de Estados y Árboles de Búsqueda

Un **grafo del espacio de estados** (G, V, E) es una representación matemática del problema de búsqueda.

- Nodos: configuraciones del espacio de estados.
- Aristas: sucesores, o resultados de una acción.
- el conjunto objetivo es un subconjunto $F \subseteq V$.

En el espacio de estados (y en su grafo), cada configuración ocurre una sola vez.

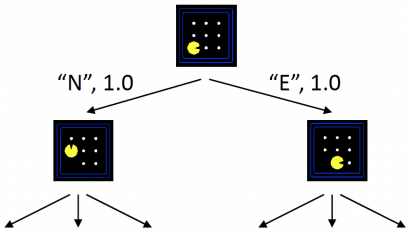
Para un problema real, difícilmente podemos reconstruir el grafo completo en la memoria.



Espacios de Estados y Árboles de Búsqueda

Un **árbol de búsqueda** $T = (G', V', E')$ es una representación matemática del plan de acción o movimiento a través del grafo de estados.

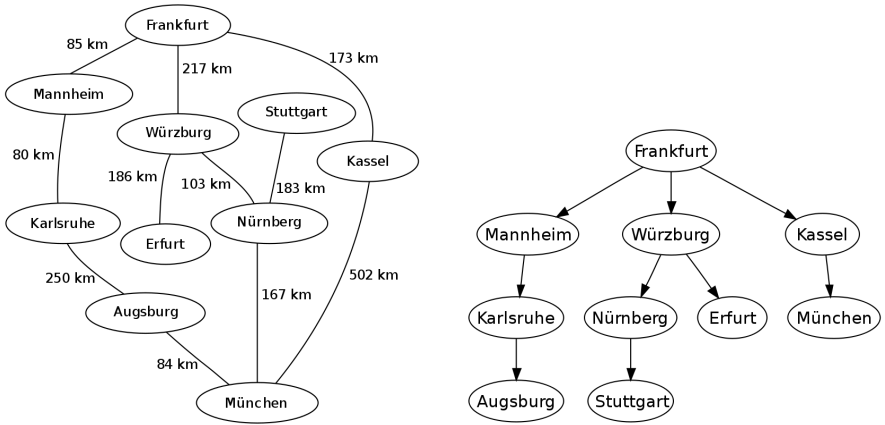
- Planes y resultados.
- El estado inicial S es el nodo raíz.
- Los descendientes en el árbol corresponden a los sucesores o movidas en la función de acción.
- Los nodos representan estados, pero corresponden a planes para alcanzar dichos estados.



Los estados pueden repetirse, los planes no.

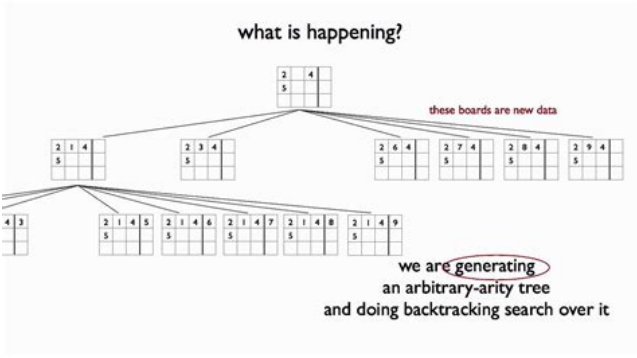
Para un problema real, difícilmente podemos reconstruir todo el árbol de búsqueda.

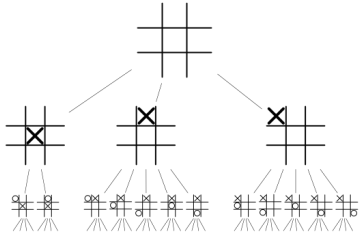
Espacios de Estados y Árboles de Búsqueda



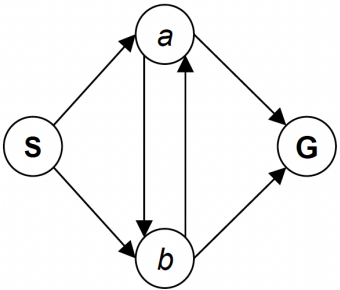
(a) Grafo del espacio de estados, (b) Árbol de búsqueda.

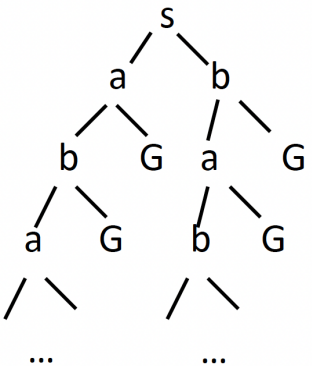
Espacios de Estados y Árboles de Búsqueda





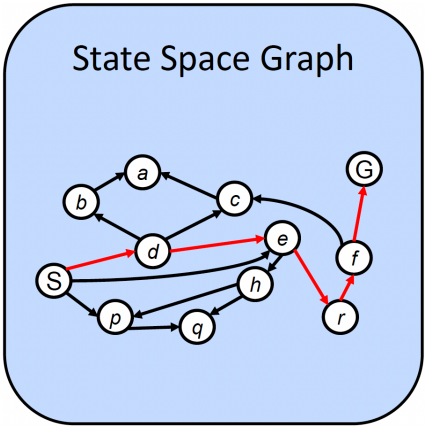
Espacios de Estados y Árboles de Búsqueda

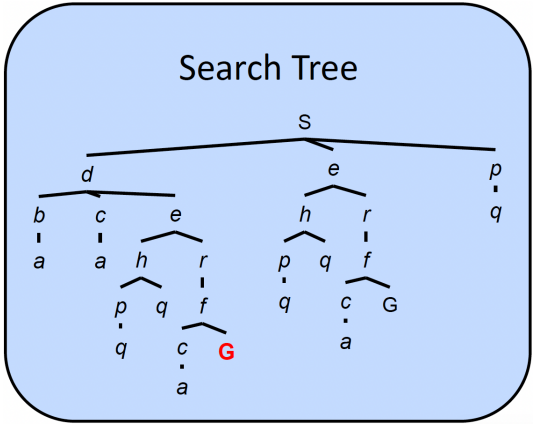




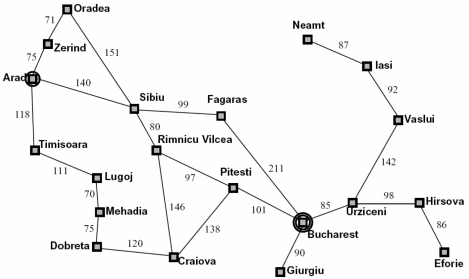
Un ejemplo donde los nodos en el árbol de búsqueda pueden aparecer varias veces. Hay mucha estructura que se puede repetir en un árbol de búsqueda.
Sin embargo cada plan de búsqueda sólo ocurren una vez.

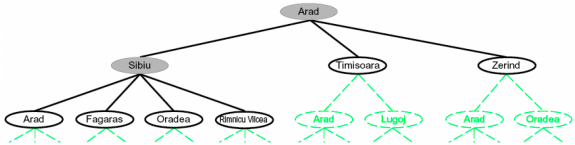
Espacios de Estados y Árboles de Búsqueda





Métodos de Búsqueda





Vamos a aplicar estrategias o algoritmos de búsqueda sobre el árbol *T*:

- Expandir los potenciales planes de acción.
- Mantener una **franja** de planes parciales bajo consideración.
- Tratar de expandir la menor cantidad de planes posible.

Métodos de Búsqueda

Algoritmos general de búsqueda:

```
function TREE-SEARCH( problem, strategy) returns a solution, or failure
    initialize the search tree using the initial state of problem
    loop do
        if there are no candidates for expansion then return failure
        choose a leaf node for expansion according to strategy
        if the node contains a goal state then return the corresponding solution
        else expand the node and add the resulting nodes to the search tree
    end
```

Es importante tener en cuenta:

- Conjunto de expansión (estados activos)
- Franja de expansión
- Estrategia exploratoria

Métodos de Búsqueda

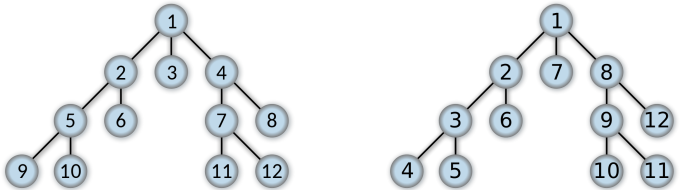
Existen diferentes estrategias de búsqueda en grafos. Mencionamos las más importantes:

- BFS (*breadth-first search*) búsqueda en anchura,
- DFS (*depth-first search*) búsqueda en profundidad,
- (*iterative deepening*) profundidad iterada,
- *Best-first search*,
- *Greedy search*,
- UCS (*uniformed cost search*) o búsqueda por costo,
- *Backtracking*,
- Heurísticas.

Métodos de Búsqueda

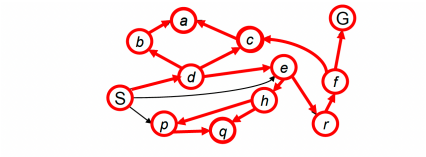
DFS: La **búsqueda en profundidad** (DFS) comienza en el nodo raíz (seleccionando algún nodo arbitrario como nodo raíz en el caso de un grafo) y explora en la medida de lo posible a lo largo de cada rama antes de retroceder.

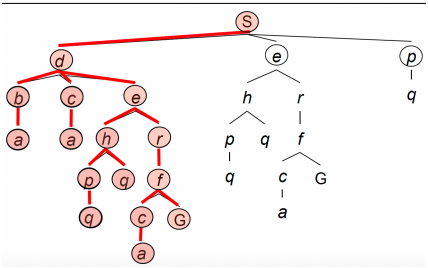
BFS: La **búsqueda en amplitud** (BFS) comienza en la raíz del árbol y explora todos los nodos en la profundidad actual antes de pasar a los nodos del siguiente nivel de profundidad. Se necesita memoria adicional, generalmente una cola, para realizar un seguimiento de los nodos secundarios que se encontraron pero que aún no se exploraron.



Estrategias de búsqueda: (a) BFS, (b) DFS.

Búsqueda en Profundidad (DFS)



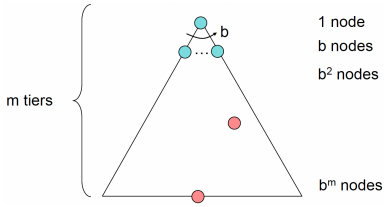


- Estrategia: expandir el nodo más profundo primero.
- Implementación: a través de un stack LIFO (*last input first output*).

Búsqueda en Profundidad (DFS)

Propiedades de la búsqueda DFS:

- ¿Es completo?: ¿garantiza hallar una solución si existe?
- ¿Es optimal?: ¿garantiza encontrar la mejor solución posible?
- ¿Complejidad temporal?
- ¿Complejidad espacial?



Esquema:

- $b =$ *branching factor*.
- $m =$ profundidad máxima.
- Soluciones a diferentes profundidades.

Número de nodos $= 1 + b + b^2 + \ldots + b^m = O(b^m)$.

Búsqueda en Profundidad (DFS)

¿Cuáles nodos expande el DFS?

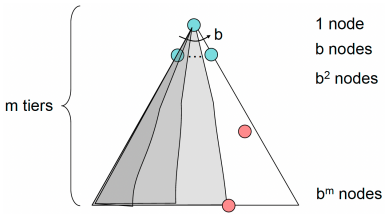
- Prefijos a la izquierda en T.
- Puede procesar todo el árbol.
- Si m es finito, toma $O(b^m)$ tiempo.
- En términos de espacio, requiere $O(bm)$.

¿Es completo?

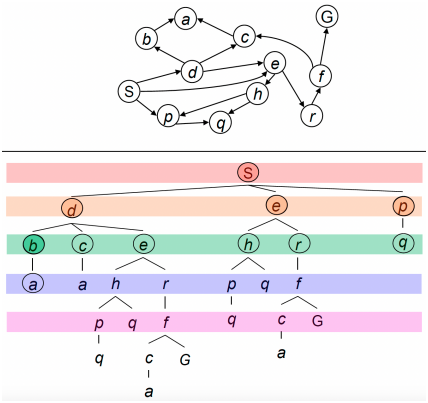
- En el caso finito sí.
- Si $m = \infty$, es completo siempre que se puedan evitar ciclos.

¿Es optimal?

- No, encuentra la primera solución objetivo a la izquierda (*leftmost*).



Búsqueda en Anchura (BFS)



- Estrategia: expandir el nodo menos profundo primero.
- Implementación: a través de un queue FIFO (*first input first output*).

Búsqueda en Anchura (BFS)

¿Cuáles nodos expande el BFS?

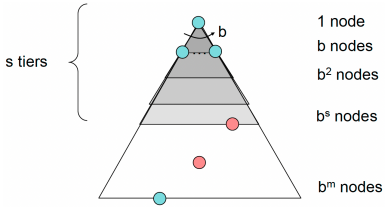
- Procesa todos los nodos antes de la solución menos profunda (suponga *d* la profundidad de esta solución).
- Si *m* es finito, toma $O(b^d)$ tiempo.
- Aproximando toda la última capa, requiere $O(b^d)$.

¿Es completo?

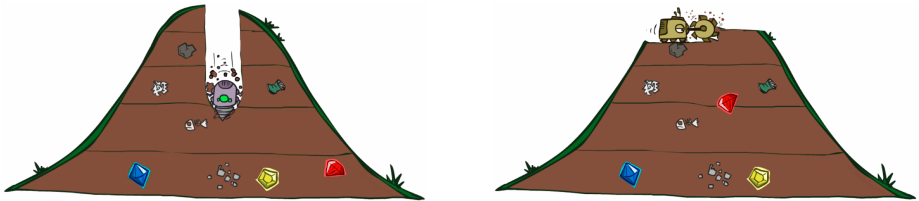
- *d* es finito si hay solución. En este caso sí.

¿Es optimal?

- Sí, siempre que los costos sean todos 1 o uniformes (veremos eso más adelante).



DFS vs. BFS



Diferencia esquemática entre las estrategias de búsqueda (a) DFS, (b) BFS.

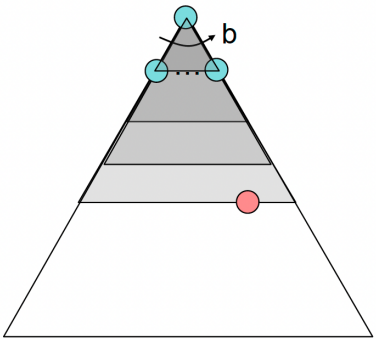
Iterando en Profundidad

Idea: obtener las ventajas espaciales del DFS con las ventajas temporales de BFS.

- Útil para soluciones poco profundas:
- Correr un DFS con profundidad máxima 1. Si no hay solución
- Correr un DFS con profundidad máxima 2. Si no hay solución
- Correr un DFS con profundidad máxima 3, ...

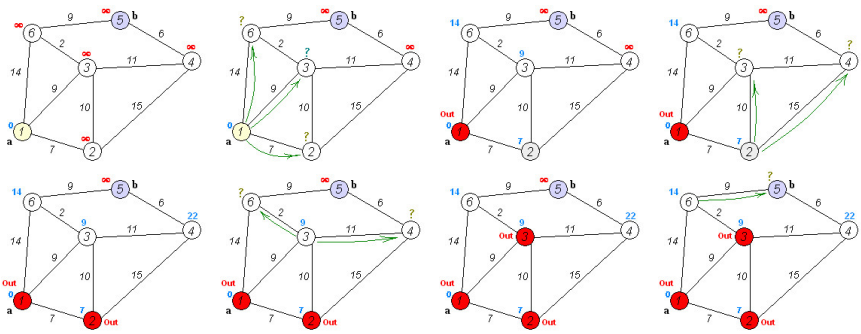
¿No es esto muy redundante?

- De hecho sí lo es.
- Pero el método en general es eficiente para soluciones que aparecen a poca profundidad.



Ejemplos

Por ejemplo, el algoritmo de Dijkstra para hallar la ruta más corta a partir de un nodo origen, sigue una estrategia tipo BFS:



Primeras etapas durante el algoritmo de Dijkstra.

Esquemas de Búsqueda

Para una comparación de las diferentes estrategias, por ejemplo ver:
`https://visualgo.net/en/sssp`