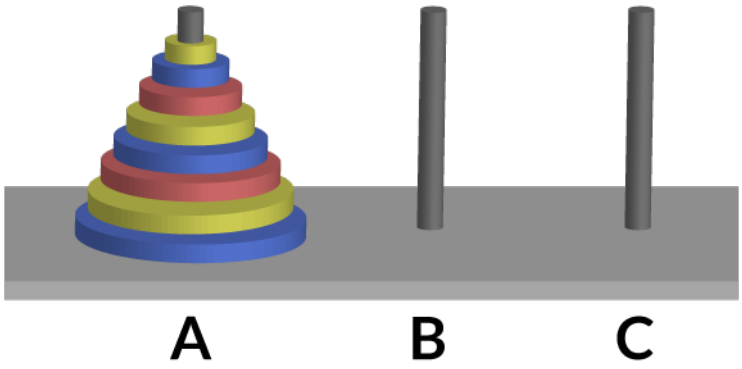
Técnicas de Búsqueda Ciega
Sistemas Inteligentes

Diego Rodríguez Riera

February 18, 2018

1 Torres de Hanoi



1.1 Reglas

1. Solo puedes mover el disco superior de cada columna.
2. Solo puedes mover el disco superior de cada columna.
3. Ningún disco puede estar encima de un disco más pequeño.

1.2 objetivo

mover la torre entera a otra columna

1.3 Resolución

1. se efectua el primer movimiento según el numero de anillos
   - si n es impar, el primer movimiento será de A a C
   - si n es par, el primer movimiento será de A a B
2. Nungún anillo par se deberá poner directamente encima de otro anillo par
3. Nungún anillo impar se deberá poner directamente encima de otro anillo impar
4. Si hay más de un movimiento posibles, (siendo las opciones una columna vacía y otra no lo está) pon el disco en la columna que no esta vacía
5. Nunca muevas un disco el cual se ha movido en la rotación anterior
6. Repetir desde el paso 2 hasta completar

2 Arbol

2.1 Enunciado

Consideremos un árbol finito de profundidad d con un factor de ramificación b (un solo nodo raíz, con profundidad 0, y b sucesores por cada nodo, etc Supongamos que el nodo objetivo menos profundo se encuentra a una profundidad g $\leq$ d

1. ¿Cuál es el número mínimo de nodos generados por la búsqueda primero en profundidad con una profundidad límite d?. ¿Y el máximo?.

2. ¿Cuál es el número mínimo de nodos generados por la búsqueda primero en anchura?. ¿Y el máximo?.

3. ¿Cuál es el número mínimo de nodos generados por el descenso iterativo?. ¿Y el máximo?. (Considérese que comenzamos con una profundidad límite inicial de 1 y la vamos incrementando una unidad cada iteración).

2.2 Resolución

1. El numero de nodos mínimo generados por la función primero por profundidad será:
$$(d) \tag{1}$$
ya que, si el nodo que estamos buscando esta contenido en la primera rama que buscamos, solo hará falta llegar hasta el de una forma lineal. Por otra parte, tenemos que el máximo será el numero totales de nodos menos la profundidad que nos quita d del total.
$$[n^b - (n - d)] \tag{2}$$
Debido a que en el peor de los casos, pasaremos por todas las ramas erróneas y en la ultima, encontraremos a el nodo deseado en el nivel de profundidad d

2. El numero mínimo de nodos generados utilizando primero en anchura será:
$$[(d - 1^b) + 1] \tag{3}$$
Ya que habrá que explorar todos los nodos de los niveles anteriores más el nodo buscado.
En el caso contrario, el máximo seria que este sea el ultimo nodo del nivel d, quedando que el numero de nodos generados por la búsqueda seria:
$$(d^b) \tag{4}$$

3. El numero minino de nodos generados por descenso iterativo sera:
$$(\sum_{i=0}^{(d-1)} b^i) + 1 \tag{5}$$

Siendo la totalidad de los niveles menores que d (d-1) más el nodo buscado del nivel d.
El maximo seria exactamente igual, pero sumándole todos los del nivel de profundidad d, quedando la siguiente expresión:

$$\sum_{i=0}^{d} b^i \tag{6}$$

Siendo el nodo buscado el ultimo en ser accedido en el nivel d

3 Sol y Sombra

1. Solo puedes mover un elemento a la vez.
2. Los elementos pueden saltar, pero solo a otro elemento contiguo en su dirección de movimiento.
3. Todos lo elementos solo se pueden mover hacia la dirección contraria del tablero, es decir, los de la derecha solo se pueden mover hacia la izquierda.

4 Objetivo

- cambiar de posición ambos elementos (soles con sombras)

5 Resolución

EL conjunto de pasos que da solución son dos, depende del primer movimiento. exploramos una de las ramas.

iteración	p1	p2	p3	p4	p5	movimiento
iteración 0	o	o		x	x	posición inicial
iteración 1	o	o	x		x	se mueve x
iteración 2	o		x	o	x	se mueve o
iteración 3		o	x	o	x	se mueve o
iteración 4	x	o		o	x	se mueve x
iteración 5	x	o	x	o		se mueve x
iteración 6	x	o	x		o	se mueve o
iteración 7	x		x	o	o	se mueve o
iteración 8	x	x		o	o	se mueve x, posición final

5.1 Algoritmo

5.1.1 Directivas

1. El primer movimiento es libre.
2. Se mueven dos fichas (una antes y la otra después, en dos turnos) del símbolo contrario del símbolo de la ficha movida en el turno inicial, solo hay un movimiento legal por cada turno.

3. Repetir paso dos hasta completar.

Este algoritmo es valido n=[1,2] donde n ∈ "numero de fichas de cada signo".

5.1.2 Código del algoritmo

Para ver el código que resuelve este problema ir a:
https://github.com/riera90