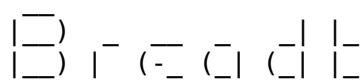
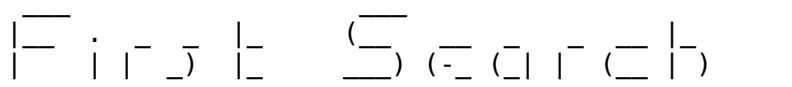
"Análisis de Algoritmos"
José Helo Guzmán





Contenido

▬▬▬ 1. Introducción.

Otra forma de encontrar un camino en un árbol es mediante Breadth First Search, anchura primero. Analicemos y veamos las diferencias entre este método y el método anterior.

▬▬▬ 2. Definición de un grafo.

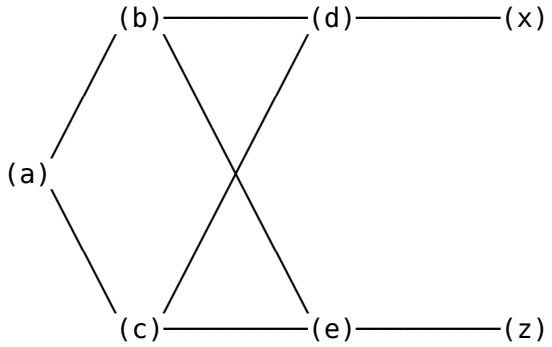
Un grafo se define como un conjunto de nodos “V” (“vertex”) y un conjunto de arcos “E” (“Edges”). Por ejemplo:

```
V = [a, b, c, d, e, x, z]

E = [ [a,b], [a,c],
      [b,a], [b,d], [b,e],
      [c,a], [c,d], [c,e],
      [d,b], [d,c], [d,x],
      [e,b], [e,c], [e,x],
      [x,d],
      [z,e]
    ]
```

Otra forma más sencilla de representar el grafo consiste en enumerar el conjunto d
de nodos y para cada nodo, indicar con quien se conecta:

```
V = [a, b, c, d, e, x, z]

E = [ [a, [b,c]],
      [b, [a,d,e]],
      [c, [a,d,e]],
      [d, [b,c,x]],
      [e, [b,c,z]],
      [x, [d]],
      [z, [e]]
    ];
```

La anterior descripción sirve para representar este grafo.



▬▬▬ 3. Breadth First Search.

El proceso de anchura primero va desarrollando cada nivel del árbol antes de bajar
al siguiente nivel.

Veamos como se logra encontrar la primera solución:

```
[ [a] ]


[ [a,b],
  [a,c]
]


[ [a,c],
  [a,b,d],
  [a,b,e]
]


[ [a,b,d],
  [a,b,e],
```

```
  [a,c,d],
  [a,c,e]
]


[ [a,b,e],
  [a,c,d],
  [a,c,e],
  [a,b,d,c],
  [a,b,d,x]
]



[ [a,c,d],
  [a,c,e],
  [a,b,d,c],
  [a,b,d,x],
  [a,b,e,c],
  [a,b,e,z]
]


[ [a,c,e],
  [a,b,d,c],
  [a,b,d,x],
  [a,b,e,c],
  [a,b,e,z],
  [a,c,d,b],
  [a,c,d,x]
]


[ [a,b,d,c],
  [a,b,d,x],
  [a,b,e,c],
  [a,b,e,z],
  [a,c,d,b],
  [a,c,d,x],
  [a,c,e,b],
  [a,c,e,z]
]

[ [a,b,d,x],
  [a,b,e,c],
  [a,b,e,z],
  [a,c,d,b],
  [a,c,d,x],
  [a,c,e,b],
  [a,c,e,z],
  [a,b,d,c,e]
]


[ [a,b,e,c],
  [a,b,e,z],
  [a,c,d,b],
  [a,c,d,x],
  [a,c,e,b],
  [a,c,e,z],
  [a,b,d,c,e]
]


[ [a,b,e,z],
  [a,c,d,b],
  [a,c,d,x],
```

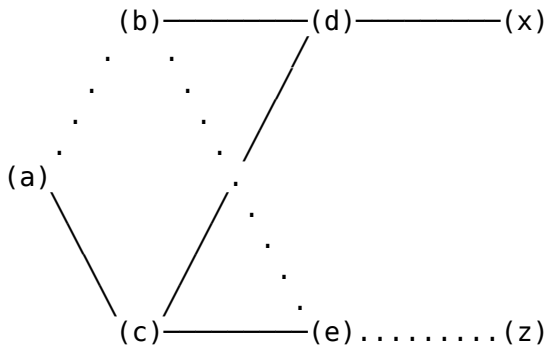
```
  [a,c,e,b],
  [a,c,e,z],
  [a,b,d,c,e],
  [a,b,e,c,d]
]
```

Aquí encontramos una solución en el primer lugar de la lista:

[a,b,e,z]

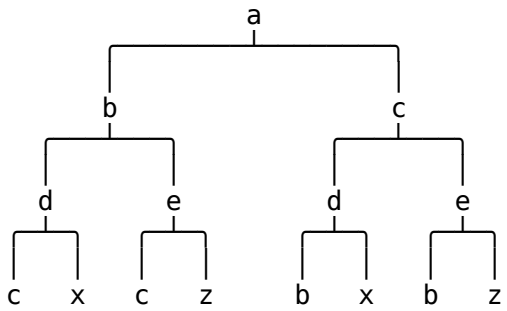
●● Ruta en el Grafo.

Aquí podemos ver la ruta que se encontró. Es la ruta más corta.



●● Árbol de Búsqueda.

Podemos ver en el árbol como se encuentra la primera solución:



▬▬▬ 4. Todas las Rutas con Breath First Search.

Si seguimos buscando las rutas encontraremos este proceso.
Ya conocemos el camino a la primera ruta:

>>> Primera Solución:

[[a]]

┆

[[a,b,e,z], [a,c,d,b], [a,c,d,x],

  [a,c,e,b], [a,c,e,z],
  [a,b,d,c,e],
  [a,b,e,c,d]]

>>> Para la segunda solución:

[[a,c,d,b], [a,c,d,x], [a,c,e,b], [a,c,e,z], [a,b,d,c,e], [a,b,e,c,d]]

[[a,c,d,x], [a,c,e,b], [a,c,e,z], [a,b,d,c,e], [a,b,e,c,d],[a,c,d,b,e]]

[[a,c,e,b], [a,c,e,z], [a,b,d,c,e], [a,b,e,c,d], [a,c,d,b,e]]

[[a,c,e,z], [a,b,d,c,e], [a,b,e,c,d], [a,c,d,b,e], [a,c,e,b,d]]

>>> Para la tercera solución:

[[a,b,d,c,e], [a,b,e,c,d], [a,c,d,b,e], [a,c,e,b,d]]

[[a,b,e,c,d], [a,c,d,b,e], [a,c,e,b,d], [a,b,d,c,e,z]]

[[a,c,d,b,e], [a,c,e,b,d], [a,b,d,c,e,z], [a,b,e,c,d,x]]

[[a,c,e,b,d], [a,b,d,c,e,z], [a,b,e,c,d,x], [a,c,d,b,e,z]]

[[a,b,d,c,e,z], [a,b,e,c,d,x], [a,c,d,b,e,z], [a,c,e,b,d,x]]

>>> Para la cuarta solución

[[a,b,e,c,d,x], [a,c,d,b,e,z], [a,c,e,b,d,x]]

[[a,c,d,b,e,z], [a,c,e,b,d,x]]

>>> Finaliza el proceso:

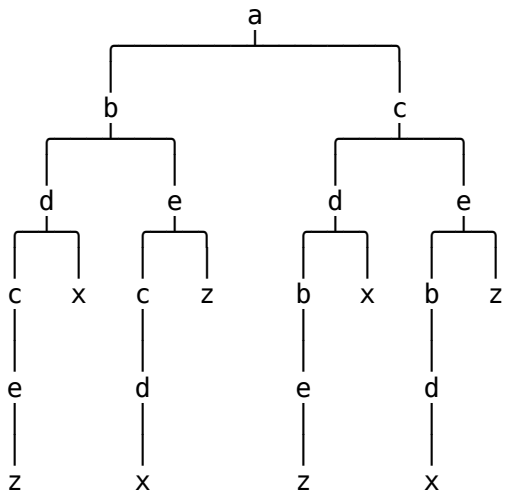
[[a,c,e,b,d,x]]

[]

Por lo tanto se obtienen las siguientes rutas:

```
[  [a,b,e,z],
   [a,c,e,z],
   [a,b,d,c,e,z],
   [a,c,d,b,e,z]
]
```

Al igual que con Deep First Search, se recorre todo el árbol, pero en una forma diferente:



▬▬▬ 5. Análisis del Tiempo de Big Oh.

Si suponemos que tenemos un grafo completamente conectado, tendremos el mismo tiempo de complejidad que para el DFS.

Esto es debido a que únicamente cambia la forma en la que encontramos la ruta que vamos a desarrollar, pero el resto del algoritmo es el mismo.

▬▬▬ 6. Resumen.

• Tando Deep First Search como Breadth First Search nos indican si hay un camino en un grafo.

• Ambos algoritmos se puede utilizar para encontrar todos los posibles caminos de un problema.

• Es relativamente fácil convertir un algoritmo en otro para estos problemas de búsqueda.

▬▬▬ 7. Ejercicios.

●● Ejercicio 1.

Los mismos ejercicios que el capítulo anterior, pero deben hacerlos con BFS.