Laboratory practice No. 3: Backtracking

María Sofía Uribe
Universidad Eafit
Medellín, Colombia
msuribec@eafit.edu.co

Isabel Cristina Graciano
Universidad Eafit
Medellín, Colombia
icgracianv@eafit.edu.co

3) Practice for final project defense presentation

3.1 Dijkstra: This is an algorithm for finding the shortest path from an unique vertex (initial node) to another one (final node). It is one of the most famous searching algorithms because its complexity is:
O(|E| + |V| log |V|).

Bellman Ford: It is an algorithm that allows to find the shortest path among a source vertex to all the other in a weighted graph, which opposite to Dijkstra's algorithm is slower because its complexity is *O(|V||E|)* but it is capable to work with negative weights edges.
This algorithm initializes all the distances in the graph in infinity and as the program is running it replaces the minimum cost by better results until they finish the process.

Floyd - Warshall: The Floyd-Warshall algorithm as Bellman-Floyd cand calculate the shortest path from one source node to all the others even if the edges have negative weights, but in this case it can not handle with negative cycles.
When we execute it once it will find the lengths of shortest paths between all pairs of vertices and because of its functioniting it has a complexity of *O(|V|^3)*

Johnson: This algorithm finds every shortest path between all pairs of vertices in a directed graph, and just like Floyd-Warshall it allows to have negative numbers in the edges but no negative weight cycles.
This consists in starting from a source vertex using the Bellman-Ford algorithm, let's call it 'b' and find for each vertex the minimum weight from b to all the others; if in this process it finds a negative cycle the it is finished but if it does not the edges of the original graph are reweighted based on the last process, so, for example, if we have the length *w=(u,v)* the new length will be *w(u,v) + h(u) - h(v);* And finally, b is removed and we proceed to use Dijkstra algorithm to find the shortest path from each node to every other vertex.

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

3.2 The number of paths with edges ($1 \leq k \leq P-1$) between two distinct vertices in the complete graph Kp is:

$(P-2)(P-3)...(P-k) = \frac{(P-2)!}{(P-k-1)!}$

So the total number is:

$(P-2)!(\frac{1}{(P-2)!} + \frac{1}{(P-3)!} + ... + \frac{1}{1!} + 1)$

And the value in the brackets is close to Ɛ (euler number), so we can conclude the next formula:

$(P-2)!Ɛ$

3.3

Value of N	Execution time (Brute Force)	Execution time (Backtracking finding one solution)
4	1 ms	0 ms
5	5 ms	0 ms
6	56084 ms (approx 1 minute)	1 ms
...	....	....
32	more than 5 minutes	48904 ms (approx 49 seconds)
N	O(n!)	O(n^n)

3.4 We use BFS when we need to find the shortest path between two nodes in which the first one will be the source node and the other one will be the destination; also, this algorithm allows to look up for all the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level.

On the other hand, we apply DFS when we want to know all the possibilities and verify which one is the best, and in order to do this the algorithm counts all the possible ways and at the end it makes a comparison among all the final values.

3.5 The algorithm uses an array to keep track of the distances from the source vertex to all other vertices, we establish that the distance from the source vertex to itself will be zero and the distance to every other vertex will be infinity.Then we perform dfs from the source vertex and therefore visit every child node.When we visit a node we check if the distance from the source can be optimized and update if possible, then we traverse from the child node using dfs .

3.6 The complexity of this algorithm is O(V + E)

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

3.7 V is the number of vertices in the graph and E is the number of edges

3.8 The algorithm is similar to the one described in 3.5 , we use dfs to traverse the graph and keep track of the distances with an array , additionally we use a list to keep track of the path visited so far. When a child node is visited and the distance can be optimized we add it to the shortest distance path , once we have visited one possibility of a path we try a different path to see if it's a better one, we do this step by backtracking (taking elements out of the path).

4) Practice for midterms

4.1

4.1.1 `n-a,a,b,c`

4.1.2 `res,solucionar(n-b,a,b,c) + 1`

4.1.3 `res,solucionar(n-c,a,b,c) + 1`

4.2

4.2.1 `pos == graph.length`

4.2.2 `v, graph, path, pos`

4.2.3 `graph, path, pos + 1`

4.3

4.3.1 DFS

NOTICE: Because the arrow between 1 and 2 does not have a direction we assume 2→ 1 and 1→ 2

0→ 3→7→ 4→ 2 → 1→ 5 → 6
1→ 0→ 3→ 7→ 2→ 4→ 6→ 5
2→ 1→ 0→ 3→ 7→ 4→ 2→ 6→ 5
3→ 7
4→ 2→ 1→ 0→ 3→ 7→ 5→ 6
5
6→ 2→ 1→ 0→ 3→ 7→ 4→ 5→ 6
7

4.3.2 BFS

NOTICE: Because the arrow between 1 and 2 does not have a direction we assume 2→ 1 and 1→ 2

0→ 3 → 4 → 7→ 2→ 1→ 6 → 5
1→ 0 → 2→ 5→ 3→ 4 → 6 → 7
2→ 1→ 4→ 6→ 0→ 5→ 3→7
3→ 7
4→ 2→ 1→ 6→ 0 → 5 → 3→ 7
5
6→ 2→ 1→ 4→ 0→ 5→ 3→ 7

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

7

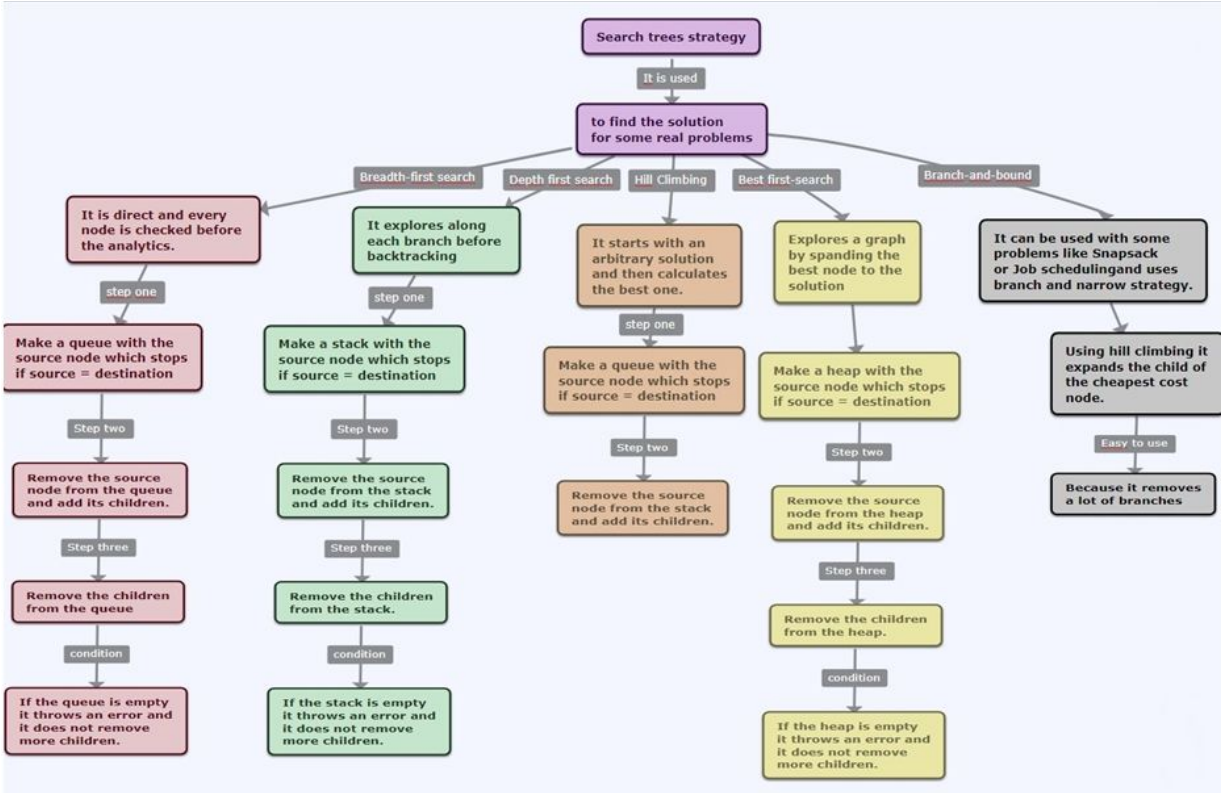
4.5
- *4.5.1 1*
- *4.5.2 ni,nj*
- *4.5.3 O(2^n)*

4.6
- *4.6.1 c)*
- *4.6.2 a)*
- *4.6.3 O(2^n)*

4.7
- *4.7.1 r==N*
- *4.7.2 i*
- *4.7.3 r+1*

5) Recommended reading (optional)

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473



6) Team work and gradual progress (optional) Discussing

6.1 Meeting minutes

TEAM MEMBER	DATE	DONE	DOING	TO DO
ISABEL	14/03/2019	Discussing task distribution		Writing laboratory
SOFIA	14/03/2019	Discussing task distribution		Writing laboratory
ISABEL	15/03/2019	Answer exercise 3.1, 3.2, 3.4		Optional Reading (concept diagram)
SOFIA	15/03/2019	Answer exercise 3.3, 3.5, 3.6, 3.7, 3.8		Code comments and practice for midterms
ISABEL	16/03/2019	Concept map and optional reading		Practice for midterms
SOFIA	15/03/2019	Practice for midterms		
ISABEL	16/03/2019	Practice for midterms		

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473