ESTRUCTURA DE DATOS 2
Código ST0247

Laboratory practice No. 3: BackTracking

Juan Sebastián Pérez salazar
Universidad Eafit
Medellín, Colombia
jsperezs@eafit.edu.co

Yhoan A. Guzmán García
Universidad Eafit
Medellín, Colombia
yaguzmang@eafit.edu.co

3) Practice for final project defense presentation

3.1 There are more effective computational techniques than backtracking or brute force, some of these techniques are Dijkstra, Prim, or greedy algorithms. They are techniques that reduce the execution time and are more favorable to apply them.
3.2 The number of possible paths is n! if we are talking about simple path, of one vertex to other.
3.3

N value	Backtracking execution time (seconds)	Brute force execution time (seconds)
4	0	0
5	0	0
6	0	0
7	0	0
8	0,002	0,015
9	0,002	0,4
10	0,003	0,5
11	0,003	3
12	0,09	24
13	0	185
14	0,12	1531
15	0,1	> 50min
16	0,5	> 50min
17	0,3	> 50min
18	2	> 50min
19	0,1	> 50min
20	15	> 50min

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

21	0,7	> 50min
22	151	> 50min
23	2,4	> 50min
24	41	> 50min
25	5	> 50min
26	44	> 50min
27	61	> 50min
28	410	> 50min
29	224	> 50min
30	> 50min	> 50min
31	> 50min	> 50min
32	> 50min	> 50min

3.4 - It is better to use BFS when you want to find the shortest path from a certain source node to a certain destination. (Or more generally, the smallest number of steps to reach the end state from a given initial state.)
- When it is needed to exhaust all possibilities and check which one is the best/count the number of all possible ways, DFS should be used.
- When it is just needed to check connectedness between two nodes on a given graph, both BFD and DFS can be used. (Or more generally, whether you could reach a given state to another.)

3.5 In point 2.1 a backtracking method was used together with the DFS method, this helped to find all the possible paths with certain restrictions, in this way we arrived at the most optimal solution, both in a complete graph, as in a simple graph . The algorithm start in vertex 0, and from that vertex makes a deep search recursively until vertex n-1. The algorithm will storing the value of the path, and this will only be updated if there is one less than him. In the end the algorithm returns the cost of the path and the path.

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

3.7 V represent the number of vertices and E represent the number of edges

3.8 In point 1.1 a backtracking method was used, this helped to find all the possible paths with certain restrictions as in point 2.1, in this way the most optimal solution was reached, both in a complete graph, as in a simple graph. The algorithm receives a graph, an initial vertex and a final vertex, and from that initial vertex makes a deep search recursively. The algorithm will store the value of the road, and this will only be updated if there is one less than it, in turn after each complete iteration, the algorithm cleans the visited table to avoid inconveniences when looking for another path. In the end the algorithm returns the cost of the path.

4) Practice for midterms

4.1 1) int res = solucionar(n - a, a, b, c) + 1

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

2) (res, solucionar(n - b, a, b, c) + 1)
3) (res, solucionar(n - c, a, b, c) + 1)

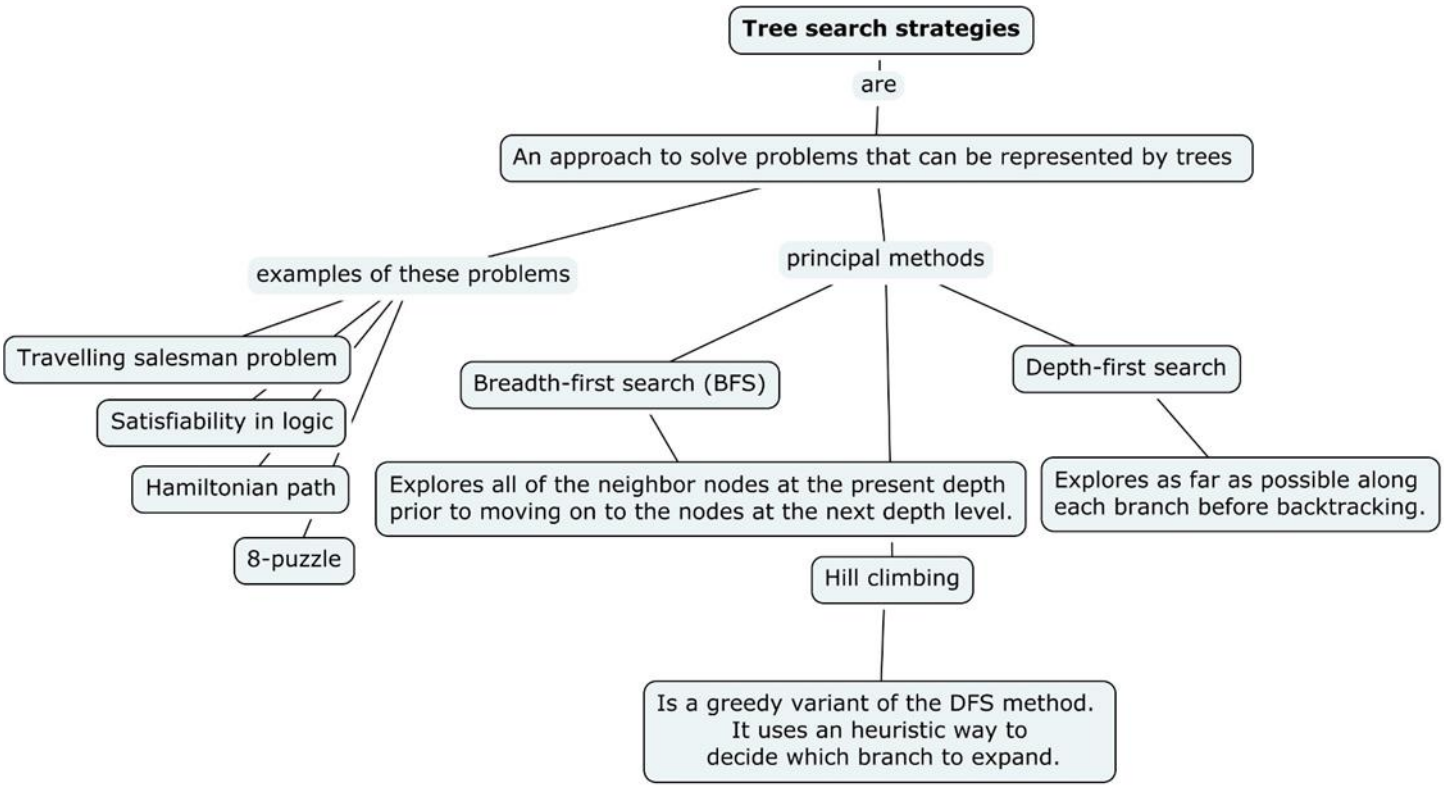
4.2 1) pos == path.length
2) sePuede(v, graph, path, pos)
3) cicloHamilAux(graph, path, pos+1)

4.5 1) 1+ lcs(i - 1, j - 1, s1, s2);
2) Math.max(ni, nj)
3) T(n) = 2T(n-1) + C

4.6 1) c. 0, 1, 4, 3, 2
2) a. 0, 1, 2, 3, 4

4.7 1) r == N
2) a[r] = i
3) sol(a, 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)

6.1 Meeting minutes

Member	Date	Done	Doing	To do
Sebastián	16/03/2019	**Think about point 1.1, 1.5, 2.1**		Implement a solution to points 1.1, 1.5, 2.1
Sebastián	16/03/2019	**I implemented a solution to point 1.1, 1.5**	Doing a solution to point 2.1	Test for points, 1.1, 1.5
Sebastián	16/03/2019	**Solution to point 2.1**	Test for points 1.1, 1.5, 2.1	Make the laboratory report
Sebastián	16/03/2019	**tests**	Laboratory report	Practice for midterms
Sebastián	17/03/2019	**parcial laboratory report and practice for mindterms**		Points 1.2, 1.3
Sebastián	17/03/2019	**Points 1.2 and 1.3**		The other part of the laboratory report
Yhoan	17/03/2019	**The other part of the laboratory report**		recommended reading
Yhoan	17/03/2019	**recommended reading**		upload the laboratory

6.2 History of changes of the code

History changes of code		
Version	**Code**	**Status**
1.0	1.1	
1.0	1.2	
2.0	1.2	
1.0	1.3	
1.0	1.5	
1.0	2.1	
2.0	2.1	

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