Laboratory practice No. 3: Backtracking

Santiago Moreno Rave
Universidad Eafit
Medellín, Colombia
smorenor@eafit.edu.co

Valeria Suárez Mejía
Universidad Eafit
Medellín, Colombia
vsuarezm@eafit.edu.co

3) Practice for final project defense presentation

3.1 we have algorithms that do their same function to find the shortest path between two points with variations in speed and performance are the Dijkstra algorithm, the Bellman Ford algorithm, the Floyd-Warshall algorithm, etc. And a computational technique we can find SSSP (shortest route from a single source).

3.2

n (n-1) / 2 n is the number of vertices.

3.3

n	Execution time (Brute Force)	Execution time (Backtracking)
	More than 5 minutes	65220 ms

3.4

In the case of DFS, we use it in problems that need an in-depth tour, for example when we want to find the shortest path between two nodes so we can quickly implement this algorithm and for BSF in those problems that need a "wide" path. that all the successors of each of the nodes are evaluated at the same time.

3.5 2.1: O(n^2)

3.6 The data structure used was backtracking, since it allows us to find the shortest path from an initial vertex to a destination vertex, with the variation that it is not necessary to go through all the vertices, the objective is to reach the destination in the cost minimum. The unguided graph is interpreted and the recursive call is started to know which of the arcs is the minimum, then all the minimum arcs are added and the path of vertices is shown with its cost.

3.7 boolean []visit is the arrangement that tells me if the vertices are visited
Digraph g is the graph that they give me already properly interpreted
int o is the origin vertex
int d is the destination vertex
int [] cost is the minimum cost of the arc
int minimumCost is the minimumCost of each vertex's brach
int roadCost is the cost of the each road

3.8 The data structure used for saving information of the point 1.1 is a hashmap thas saves vertexID as a key and his information and list of edges as the value of the hashmap. For the algorithm we use two queues. One of them is the main queue and the other one is auxiliar. So the auxiliar is going to be fill by pairs with vertex and their distance, and when the recusion finds the destination vertex, compares his total distance with the main queue, and if the distance of the auxiliar is smallest than the main, we replace it.

4. Practice for midterms

4.1.

4.1.1 *solucionar(n-a,a,b,c)+1;*

4.1.2 res = Math.max(res,*solucionar(n-b,a,b,c)*)+1;

4.1.2 res = Math.max(res,*solucionar(n-c,a,b,c)*)+1;

4.2.

4.2.1 if (pos == path[0])

4.2.2 if (sePuede(v,graph [] [], path[] ,pos))

4.2.3 if (cicloHamilAux(graph[][] ,path[],pos))

4.3.

4.3.1 since 0: 0,3,7,4,2,1,5,6
since 1: 1,0,3,7,4,2,6,5
since 2: 2,1,0,3,7,4,5,6
since 3: 3,7
since 4: 4,2,1,0,3,7,5,6
since 5: 5
since 6: 6,2,1,0,3,7,4,5

since 7: 7

4.3.2 since 0: 0,3,4,7,2,1,6,5
since 1: 1,0,2,5,3,4,6,7
since 2: 2,1,4,6,0,5,3,7
since 3: 3,7
since 4: 4,2,1,6,0,5,3,7
since 5:5
since 6: 6,2,1,4,0,5,3,7
since 7: 7

4.5.

4.5.1 return 1+ lcs(i - 1, j - 1, s1, s2);

4.5.2 return Math.max(ni,nj);

4.5.3 O(2^n)

4.6.

4.6.1 c)

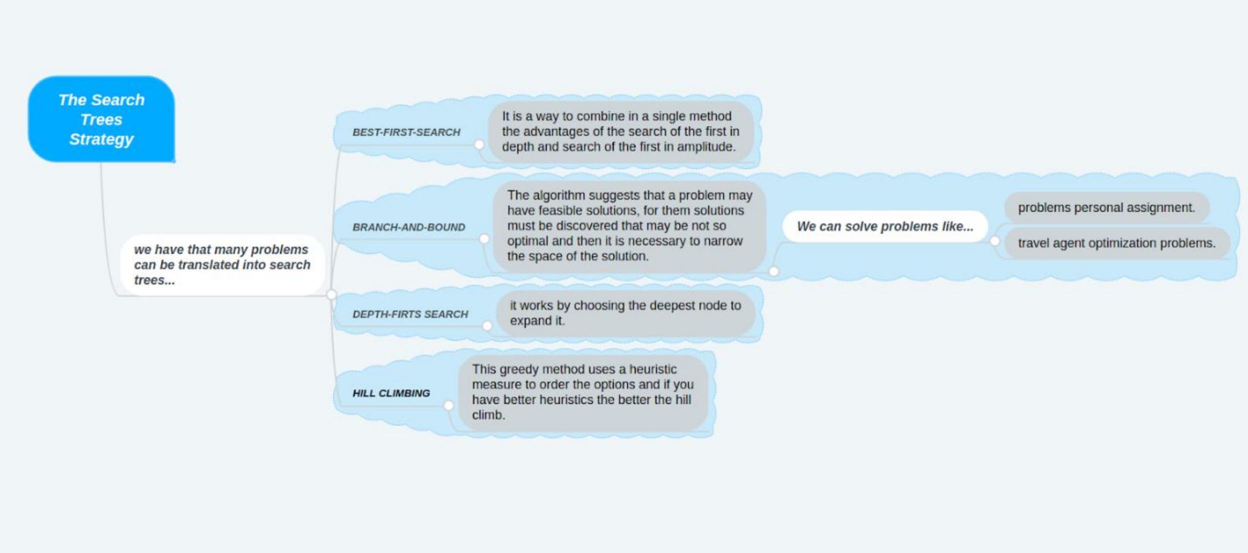
4.6.2 a)

4.7.1 if (r==N)

4.7.2 a[r]= i;

4.7.3 if(place(a,r)) sol(a, r+1);

5) Recommended reading (optional)



The Search Trees Strategy
we have that many problems can be translated into search trees...
BEST-FIRST-SEARCH
It is a way to combine in a single method the advantages of the search of the first in depth and search of the first in amplitude.
BRANCH-AND-BOUND
The algorithm suggests that a problem may have feasible solutions, for them solutions must be discovered that may be not so optimal and then it is necessary to narrow the space of the solution.
We can solve problems like...
problems personal assignment.
travel agent optimization problems.
DEPTH-FIRTS SEARCH
it works by choosing the deepest node to expand it.
HILL CLIMBING
This greedy method uses a heuristic measure to order the options and if you have better heuristics the better the hill climb.

6) Team work and gradual progress (optional)

1. Meeting minutes

work hours/day	***13/03/2019***	***16/03/2019***	***17/03/2019***
2hr	***working valeria and santiago***		
4hr		***working valeria and santiago***	
7hr			***working valeria and santiago***

6.2 History of changes of the code
These modifications are made in a code shared for us.

modifier / day	***13/03/2019***	***16/03/2019***	***17/03/2019***
Santiago	***1:00pm mofidication time***	***2:30pm modification time***	***3:00pm modification time*** ***7:30pm modification time***
Valeria	***3:00pm modification time***	***3:30pm modification time*** ***4:30pm modification time***	***8:30pm modification time***

6.3 History of changes of the report

modifier / day	***13/03/2019***	***16/03/2019***	***17/03/2019***
Santiago	***3:30pm modification time***	***6:20pm modification time***	***1:30pm modification time*** ***3:30pm modification time***
Valeria	***1:00pm modification time***	***12:00pm modification time***	***9:30pm modification time***

		2:40pm modification time ***4:00pm modification time*** ***6:00pm modification time***	

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