Breadth First Search

Exercício. Seu amigo(a) está resolvendo uma instância de um jogo chamado "Sudoku M", onde temos que preencher n números x1, x2, ... xn, obedecendo umas certas regras (ele(a) não entrou em detalhe).

O problema é que essa instância pode ter sido gerada errada e não ter solução.

Ao analisar a instância, seu amigo(a) encontrou um conjunto C de condições do tipo "**xa = xb**" e "**xi ≠ xj**". Ele(a) quer saber:

Existe alguma forma de atribuir números x1,x2,...,xn de forma a satisfazer as restrições do conjunto C ?

Modele essa questão como problema em grafo, e dê um algoritmo para resolve-lo. Analise brevemente sua complexidade.

Exercises

Exercise 1:

(1 pts) Dado um grafo e uma de suas arestas (u, v), encontre o menor **ciclo** no grafo contendo essa aresta, onde o tamanho do ciclo é o número de arestas que possui. Seu algoritmo deve ter complexidade $O(n + m)$

Exercise 2: Can we use **BFS** to detect cycles in **undirected** graphs?
How?

Careful: BFS does not directly detects cycles in directed graphs!

Exercises

Exercise 3: Suppose your graph is an undirected tree. If run BFS starting from the root of the tree, in which order are the nodes explored? What about in DFS?

Exercise 4: Using the BFS/DFS tree, show that every connected undirected graph has a node that can be removed keeping the graph still connected

[show example]

Exercise 5: Suppose your undirected graph has a value x(v) for each node. Modify DFS to compute

z(v)=sum of values of all descendants of v in the DFS tree,

for all nodes. The algorithm should still run in $O(n + m)$

Applications of topological order

Problem 1: Given a list of courses a student needs to take and the prerequisites between then, give an algorithm that finds the minimum number of semesters needed for the student to finish all the courses

[give concrete example on the board]

Possible solution: Compute for each node u the first semester f(u) that we can do that course:

1) Do topological sort of the graph
2) Scan nodes in the order of the topological sorting
3) At node u, compute

$$f(u) = 1 + \min_{v \text{ antineighbor of } u} \{f(v)\}$$

Time complexity is O(n + m) (construct reverse graph to obtain anti-neighbors)

Exercises: Shortest Paths

Exercise 1: Run Dijkstra's algorithm on the following graph [desenho no quadro], starting from node **s**

Exercise 2: Can we run Dijkstra's algorithm on undirected graphs? How?

Exercise 3: Explain why the algorithm does not need to update unvisited non-neighbors of u

Exercise 4: Show that Dijkstra's algorithm may not return the correct distance if there are negative lengths (construct a graph)

Exercise 5: Consider a different problem: You are given a directed graph and costs on the **nodes**. You want to find the shortest cost path from s to t, where the cost of a path is the sum of the costs of the nodes in the path

Find an algorithm to solve this problem. (Hint: run Dijkstra on a modified graph)