# MATH.APP.270 Algorithms for graphs

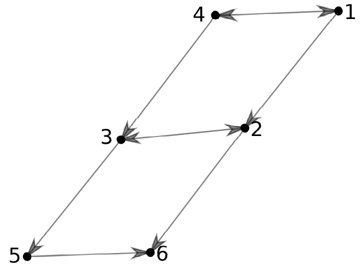
## Programming assignment 1 (Student must submit to get points) 2023

In this assignment you will need to make modifications to Algorithm 1 in the course notes (breadth-first search algorithm). Given a digraph $G = (V, E)$ and a starting node $s$, Algorithm 1 currently computes two things:

1. The distance $d(s, x)$ from $s$ to $x$ for all vertices $x \in V$.
2. For each vertex $x \in V$ for which $d(s, x) \neq \infty$, a parent $p[x]$. Using the parents we can construct a BFS-tree.

We consider here one example. Suppose our starting digraph $G_1$ is the following.

$G_1 =$

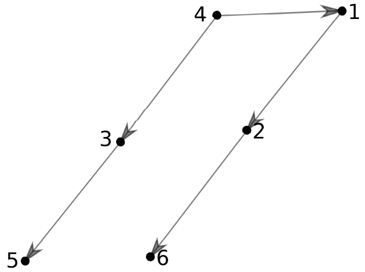


When the starting node $s = 4$, then one possible output from Algorithm 1 is the following:

| $x$ | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| $d[x]$ | 1 | 2 | 1 | 0 | 2 | 3 |
| $p[x]$ | 4 | 1 | 4 | $nil$ | 3 | 2 |

This output corresponds to the following BFS-tree:

$T_1 =$

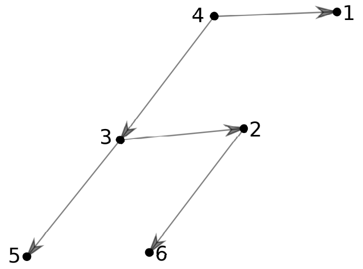


In this assignment your new algorithm will take 4 inputs:

- a digraph $G = (V, E)$
- a starting vertex $s$
- a final vertex $u$
- a subset of vertices $B \subset V$

The algorithm you are to write should find a shortest directed path from $s$ to $u$ such that the shortest directed path contains as many of the vertices in $B$ as possible. For example, suppose in addition to graph $G_1$ above, the inputs are $s = 4$, $u = 6$ and $B = \{2, 3\}$. For these inputs, the path in the BFS-tree $T_1$ from $s = 4$ to $u = 6$ is $p_1 = \langle 4, 1, 2, 6 \rangle$. Path $p_1$ contains 1 of the vertices in $B$, since $|p_1 \cap B| = |\{2\}| = 1$. The question is, 'Does there exist another shortest path $p$ from $s = 4$ to $u = 6$ that contains more than 1 of the vertices that belong to $B$?' For this example, the answer is yes. Another possible BFS-tree from graph $G_1$ is the following:

$T_2 =$



Now the path from $s = 4$ to $u = 6$ is $p_2 = \langle 4, 3, 2, 6 \rangle$ and $p_2$ contains 2 of the vertices from $B$.

Your algorithm should return a single non-negative integer $a$. This integer is the largest number of vertices from $B$ that can be included in any shortest directed path from $s$ to $u$. Since $d(s, u)$ is a fixed value, $0 \leq a \leq d(s, u) + 1$, always. The algorithm need not return the actual path from $s$ to $u$.

It is highly recommended that you write your algorithm as a Python function. You may use some other language, but if you do you must also meet the following requirements:

- Your code should accept as an input a digraph $G$ that is stored in the form specified on the course Moodle page.
- Your code must also accept a set of vertices $B$, a starting vertex $s$ and a final vertex $u$.
- You must provide clear and detailed instructions on how to compile your code and how to run your code. You cannot assume that a user has any particular knowledge. For example, you should not assume that user knows anything about Java or how to use Netbeans.