



Q1 Manually execute breadth-first search on the undirected graph in Figure 6.1, starting from vertex s . Then, use it as an example to illustrate the following properties:

- (a) The results of breadth-first search may depend on the order in which the neighbours of a given vertex are visited.
- (b) With different orders of visiting the neighbours, although the BFS tree may be different, the distance from starting vertex s to each vertex will be the same.

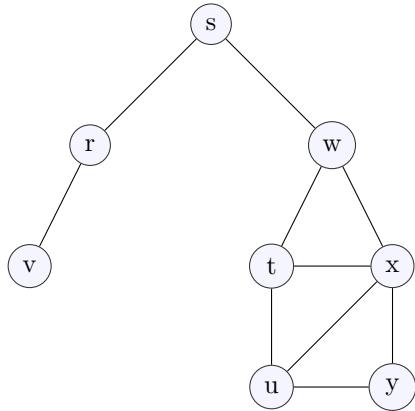


Figure 6.1: Graph for Q1

Q2 Give a psuedocode of finding a simple path connecting two given vertices in an undirected graph in linear time.

Q3 Give a psuedocode of a backtracking algorithm to print out all possible permutation of a given sequence. For example, input is given as “1234”. The 24 output permutations are printed out from “1234” to “4321”.

CX1107 Data Structures and Algorithms
Solution 6: Graph

2020/21 Semester 2

School of Computer Science and Engineering

Nanyang Technological University

Q1 Manually execute breadth-first search on the undirected graph in Figure 6.1, starting from vertex s . Then, use it as an example to illustrate the following properties:

- (a) The results of breadth-first search may depend on the order in which the neighbours of a given vertex are visited.
- (b) With different orders of visiting the neighbours, although the BFS tree may be different, the distance from starting vertex s to each vertex will be the same.

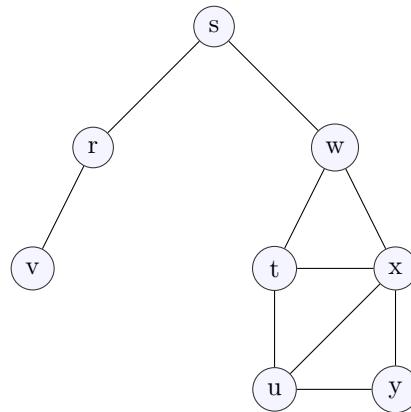


Figure 6.1: Graph for Q1

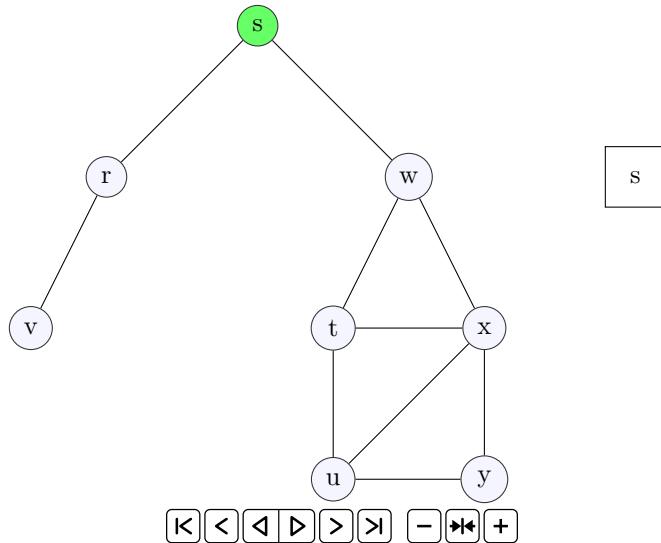


Figure 6.2: Graph for S1

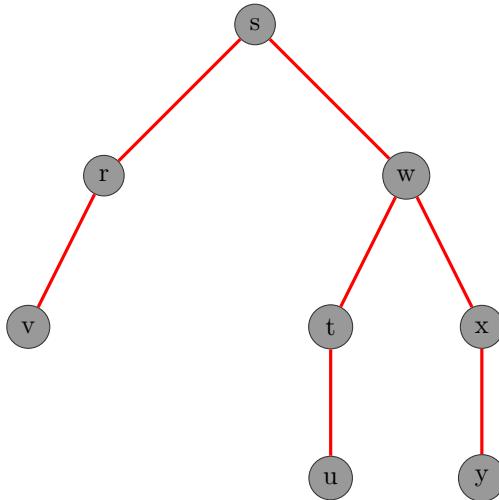


Figure 6.3: Visiting neighbors in alphabetical order

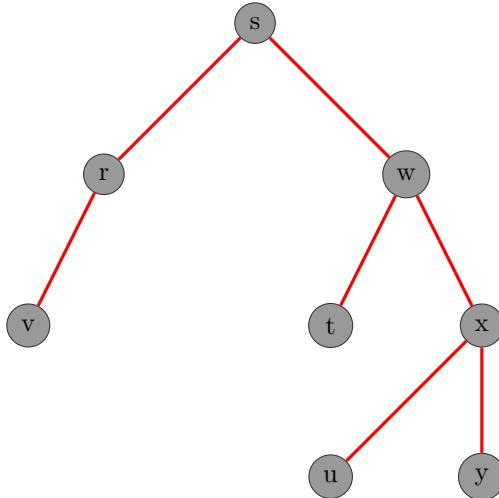


Figure 6.4: Visiting neighbors in reverse alphabetical order

- When the queue is empty, the BFS is finished.
- The edges of BFS tree are shown in red.
- Likewise, a BFS tree can be constructed if the neighbors are visited in the reverse alphabetical order (an exercise for the students).
- The two trees differ in that vertex u is adjacent with t in the left tree, but adjacent with x in the right tree.
- The distance from starting vertex s to each other vertex is equal in the two trees.

Q2 Give a pseudocode of finding a simple path connecting two given vertices in an undirected graph in linear time.

Algorithm 1 Depth First Search (DFS)

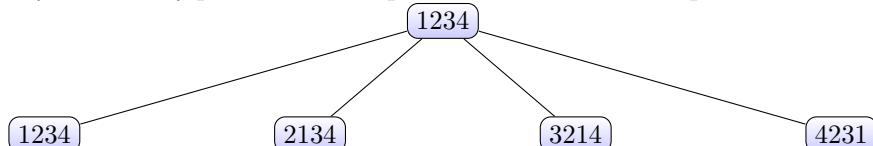
```

S2 function SIMPLEPATH(Graph G, Vertex v,Vertex w)
    create a Stack, S
    push v into S
    mark v as visited
    while S is not empty do
        peek the stack and denote the vertex as x
        if x == w then
            while S is not empty do
                pop a vertex from S
                peek the stack
                print the link
            end while
            return Found
        end if
        if no unvisited vertices are adjacent to x then
            pop a vertex from S
        else
            push an unvisited vertex u adjacent to x
            mark u as visited
        end if
    end while
    return Not Found
end function

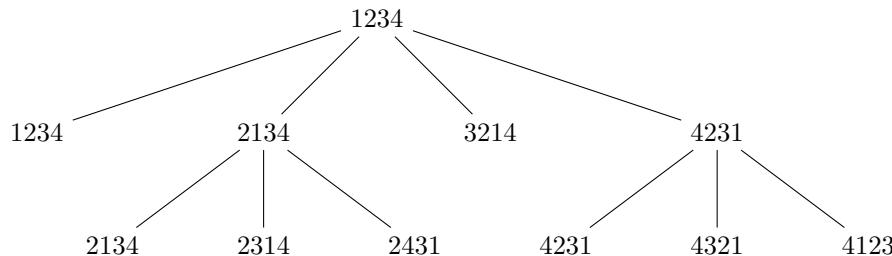
```

Q3 Give a pseudocode of a backtracking algorithm to print out all possible permutation of a given sequence. For example, input is given as “1234”. The 24 output permutations are printed out from “1234” to “4321”.

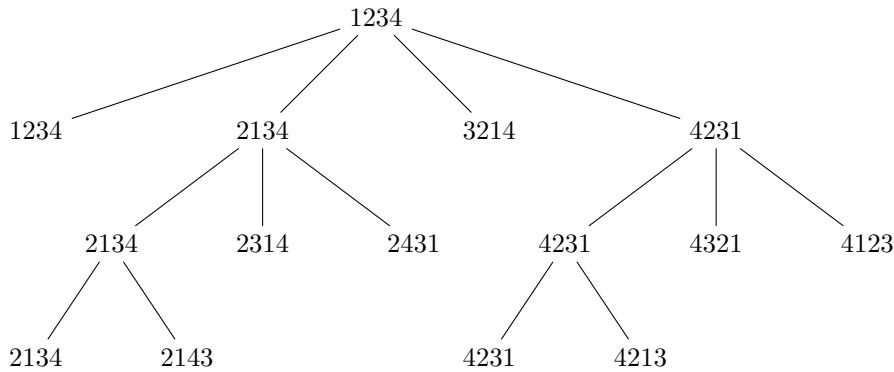
S3 To systematically print out all the permutations, we first swap the first element with each other element.



Next we need to swap the second element with each other element (except the element in the first position). Here we only case the second and the fourth cases.



Next we need to swap the third element with the following element (in the example, we only leave the last element to swap).



You can observe that printing out all the permutation we need to iterative swap one element with each other element and recursively do so on its smaller sequence (reduce by one element) until we reach the last element. The pseudocode is as following:

Algorithm 2 Backtracking algorithm for Permutation

```

function PERMUTATION(char[]seq, sIdx, eIdx)
  if sIdx == eIdx then
    print seq
  else
    for i  $\leftarrow$  sIdx to eIdx do
      swap the sIdxth character and the ith character in seq
      Permutation(seq,sIdx+1,eIdx)
      swap the sIdxth character and the ith character in seq                                 $\triangleright$  backtracking
    end for
  end if
end function
  
```
