Graph theory

Metodický koncept k efektivní podpoře klíčových odborných kompetencí s využitím cizího jazyka ATCZ62 - CLIL jako výuková strategie na vysoké škole

Interreg
EVROPSKÁ UNIE
Rakousko-Česká republika
Evropský fond pro regionální rozvoj

Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

VŠTE

UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Graph theory

- A graph is a pair (V, E), where
  - V is a set of nodes, called vertices
  - E is a collection of pairs of vertices, called edges
  - Vertices and edges are positions and store elements
- Types of edges
  - Directed – ordered pair of vertices (u,v), first vertex u is the origin, second vertex v is the destination
  - Undirected - unordered pair of vertices (u,v)
  - Loops – edge that connects a vertex to itself
  - Multiple edges – between edges (u,v) is more than one edge

Graph theory

- Types of graphs
  - Directed – all edges are direct
  - Undirected – all edges are undirected
  - Multigraph – contains multiple edges
- Terminology
  - End vertices (or endpoints) of an edge
  - Edges incident on a vertex
  - Adjacent vertices
  - Degree of a vertex
  - Parallel edges
  - Self-loop

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Graph theory

- Path
  - sequence of alternating vertices and edges
  - begins with a vertex
  - ends with a vertex
  - each edge is preceded and followed by its endpoints
- Simple path
  - path such that all its vertices and edges are distinct
- Cycle
  - circular sequence of alternating vertices and edges
  - each edge is preceded and followed by its endpoints
- Simple cycle
  - cycle such that all its vertices and edges are distinct

Graph theory

- Electronic circuits
  - Printed circuit board
  - Integrated circuit
- Transportation networks
  - Highway network
  - Flight network
- Computer networks
  - Local area network
  - Internet
  - Web
- Databases
  - Entity-relationship diagram

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

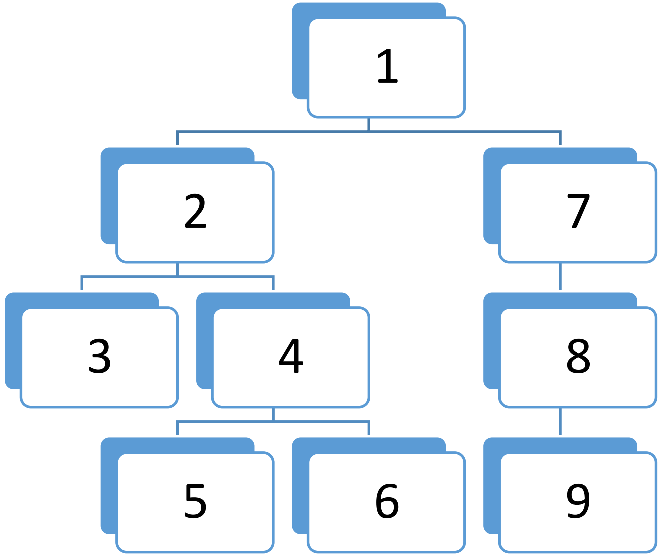
Graph – ADT

- Accessor methods
  - **aVertex()**
  - **incidentEdges(v)**
  - **endVertices(e)**
  - **isDirected(e)**
  - **origin(e)**
  - **destination(e)**
  - **opposite(v,e)**
  - **areAdjecent(v,w)**
- Update methods
  - **insertVertex(o)**
  - **insertEdge(v, w, o)**
  - **insertDirectedEdge(v, w, o)**
  - **removeVertex(v)**
  - **removeEdge(e)**
- Generic methods
  - **numVertices()**
  - **numEdges()**
  - **vertices()**
  - **edges()**

Graph – DFS – depth-first search

- for traversing or searching tree or graph data structures. One starts at the root (selecting some arbitrary node as the root in the case of a graph) and explores as far as possible along each branch before backtracking.

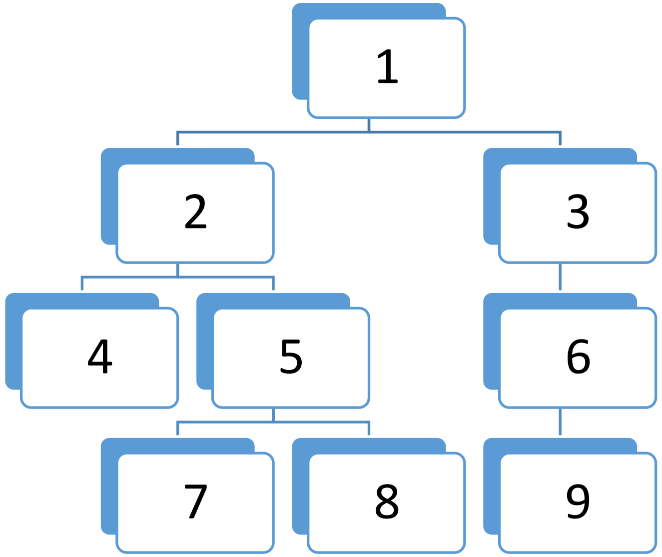
```
procedure DFS-iterative(G,v):
    let S be a stack
    S.push(v)
    while S is not empty
        v = S.pop()
        if v is not labeled as discovered:
            label v as discovered
            for all edges from v to w in G.adjacentEdges(v) do
                S.push(w)}
```



Graph – BFS – Breadth-first search

- traversing or searching tree or graph data structures. It starts at the tree root (or some arbitrary node of a graph, sometimes referred to as a 'search key') and explores the neighbor nodes first, before moving to the next level neighbors.

```
Breadth-First-Search(Graph, root):
  create empty set S
  create empty queue Q
  add root to S
  Q.enqueue(root)
  while Q is not empty:
    current = Q.dequeue()
    if current is the goal:
      return current
    for each node n that is adjacent to current:
      if n is not in S:
        add n to S
        n.parent = current
        Q.enqueue(n)
```



Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
EVROPSKÁ UNIE
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Graph – shortest path

- Dijkstra's algorithm
  - non-negative weights on the edges
  - $O(|V|^2+|E|)$ – V number of vertices, E number of edges
- Bellman-Ford algorithm
  - Graph can have negative edges
  - $O(V \cdot E)$ – slower than Dijsktra's alg.
- Floyd-Warshall algorithm
  - Directed graph with non-negative edges
  - Find shortest path between all vertices
  - Time complexity – $O(V^3)$, memory complexity - $O(V^2)$

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
EVROPSKÁ UNIE
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Graph – shortest path

- Johnson's algorithm
  - find the shortest paths between all pairs of vertices in a sparse, edge weighted, directed graph. It allows some of the edge weights to be negative numbers
  - $O(V^2 \log_2(V) + VE)$