Algorithmen & Datenstrukturen

Übungsstunde 10

Michelle Honegger - CAB G59 - 24.11.2024

Miniquiz9 9

- Passwort: **shortestpath3**
- 10 Fragen
- 10 Minuten
- 0.1 Punkte pro richtige Antwort

Programm

- Miniquiz
- Rückmeldung Serie 8
- Theory Recap
- Prüfungsaufgaben
- Programmieraufgabe
- Besprechung Aufgaben

Rückmeldung Serie 8

- Sehr gut gelöst!
- Counterexamples sollten kurz erklärt werden
- Proofs waren oft etwas knapp
- Graph mit Kreis ≠ Graph ist Kreis

Besprechung Miniquiz

If a directed graph has no directed cycles then there is no back edge in any DFS tree of it.

- Wahr
- Falsch

Decide whether the following are true or false.

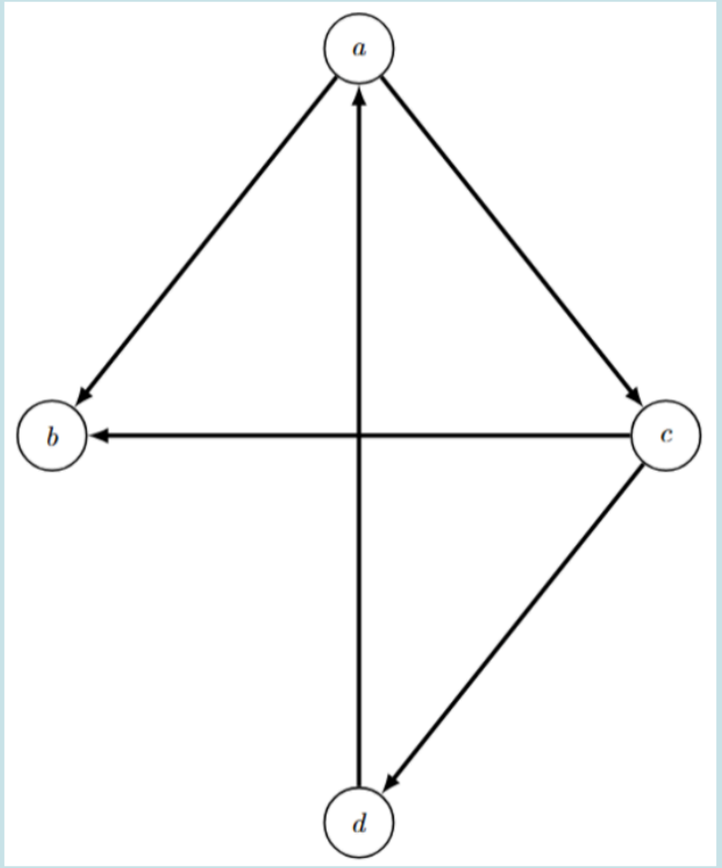
True	False	
○	○	If a directed graph has no directed cycle, then it has a sink.
○	○	If a directed graph has a sink, then it has no directed cycle.

Bewertung: **Teilpunkte**

Let u, v be two vertices connected via an edge in an **undirected graph**. Suppose we compute a DFS tree of this graph. It is possible that $pre(u) < post(u) < pre(v) < post(v)$.

- Wahr
- Falsch

Consider the graph below.



What is a valid adjacency list for vertex c?

- a. $[a]$
- b. $[b, d]$
- c. $[a, b, d]$

Suppose we have a directed graph with 2 different directed cycles. We need to remove at least 2 edges for a topological order to exist in the graph.

- Wahr
- Falsch

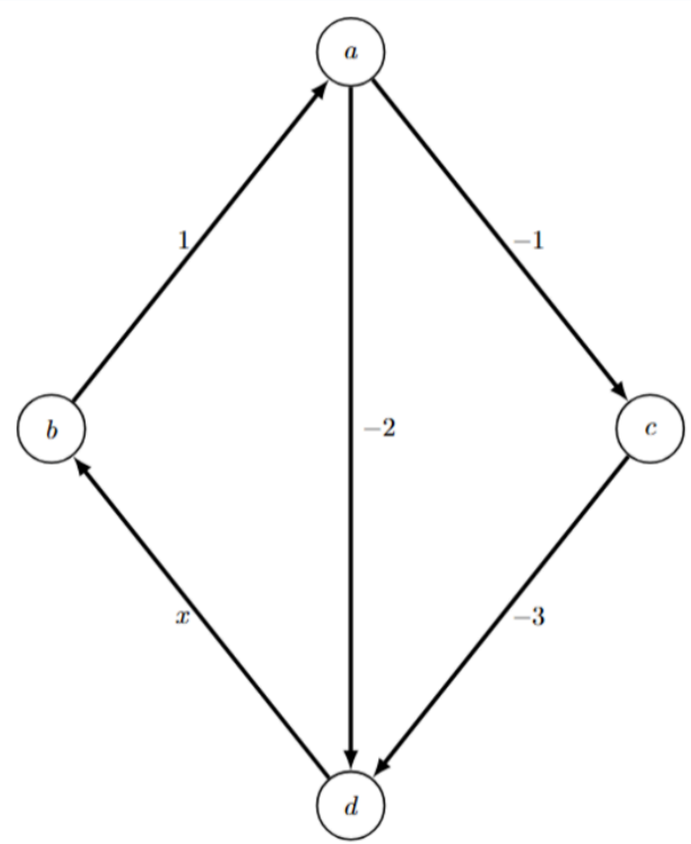
Recall the notation S_i^u used to refer to the set of vertices at distance i from some vertex u, in some directed graph D. Assume that there exists a vertex w with $w \in S_2^u$ and $w \in S_3^v$. Then, the graph distance from u to v in D is at most 5.

- Wahr
- Falsch

Suppose that we run a BFS on a directed graph and after sorting by the enter-time the vertices are d, e, b, a, c, f. What would the order be if we sorted by leave-time instead?

- a. $f, c, a, b, e, d.$
- b. $d, e, b, a, c, f.$
- c. $a, b, c, d, e, f.$
- d. $f, e, d, c, b, a.$
- e. Impossible to specify with the current information.

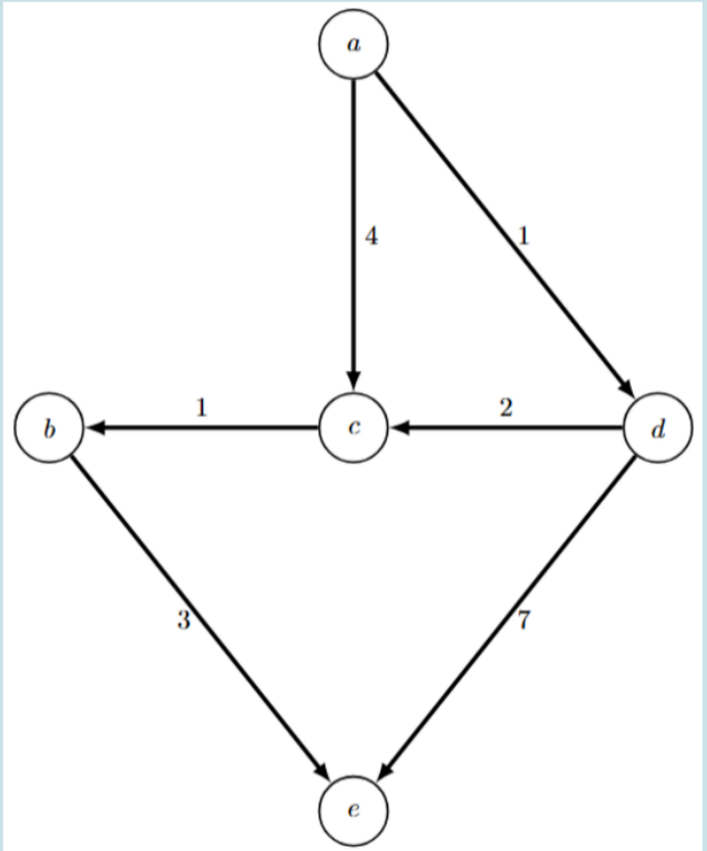
Consider the graph below.



What is the minimum value of x for which there is no negative directed cycle?

Antwort:

Consider the graph below.



What is the length of the shortest path from a to e?

Antwort:

Consider the queue $Q = [3, 5, 2, 4, 1]$. Suppose we carry out the following operations:

1. dequeue
2. dequeue
3. enqueue(1)
4. enqueue(4)

What is the final state of the queue? The enqueue operation adds an element to the right/end of the queue.

- a. $Q = [3, 5, 2, 4, 1]$
- b. $Q = [3, 5, 2, 1, 4]$
- c. $Q = [2, 4, 1, 1, 4]$
- d. $Q = [2, 4, 1, 4, 1]$

Theory Recap

FIFO Queue

- First-in first-out -> Element welches zuerst hinzugefügt wurde, wird als erstes wieder rausgenommen
- Implementierung zB. mit LinkedLists:
  - LinkedList<Integer> q = new LinkedList<Integer>();

BFS

- Knoten mit gleicher Distanz im gleichen Layer vom Suchbaum (Bei Graphen ohne Gewichten)
- Shortest paths von einem Knoten aus zu allen anderen (one-to-all)
- O(n+m)

BFS

```java
public static void bfs(int v, boolean[] visited) {
    LinkedList<Integer> queue = new LinkedList<Integer>();
    queue.add(v);
    visited[v] = true;
    order[v][0] = 0;

    while(!queue.isEmpty()) {
        int x = queue.poll();
        System.out.print(x + " ");
        for(int i = 0; i < adj[x].size(); i++) {
            if(!visited[adj[x].get(i)]) {
                visited[adj[x].get(i)] = true;
                order[adj[x].get(i)][0] = order[x][0]+1;
                queue.add(adj[x].get(i));
            }
        }
    }
}
```

Shortest paths

- Ohne negative Zyklen -> Pfade
- d(s,s) = 0
- Dreiecksungleichung
- Falls v0,v1,...,vl-1,vl günstigst, auch v0,v1,...vl

Shortest Paths

Azyklische Graphen

- In topologischer Reihenfolge Distanz berechnen
  - 0 falls v = s
  - ∞ falls v≠s, $deg_{in}(v) = 0$
  - $\min_{u \to v} d[u] + c(u, v)$, sonst

Dijkstra

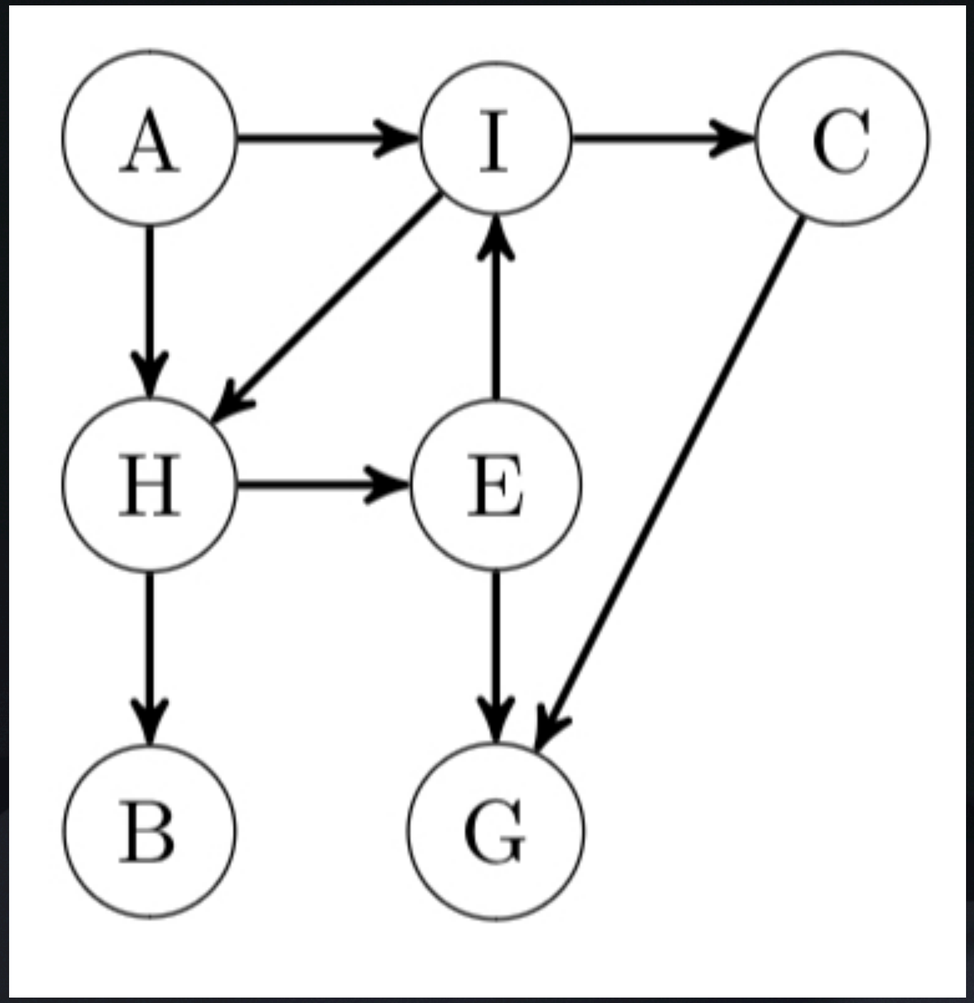
- Graph darf keine Negativen Kantenkosten haben
- Priority-Queue um immer beim Knoten mit kürzester Distanz fortzufahren

Dijkstra

```java
public static void dijkstra(int v) {
    PriorityQueue<Integer> q = new PriorityQueue<>();
    for(int i = 0; i < graph.length; i++) {
        visited[i] = false;
        dists[i] = Integer.MAX_VALUE/2;
    }
    dists[v] = 0;
    q.add(v);
    while(!q.isEmpty()) {
        int curr = q.poll();
        if(visited[curr]) {
            continue;
        }
        visited[curr] = true;
        for(int i = 0; i < dists.length; i++) {
            if(graph[curr][i] < 0) {
                throw new IllegalArgumentException();
            }
            if(curr != i) {
                dists[i] = Math.min(dists[i], dists[curr] + graph[curr][i]);
                q.add(i);
            }
        }
    }
}
```

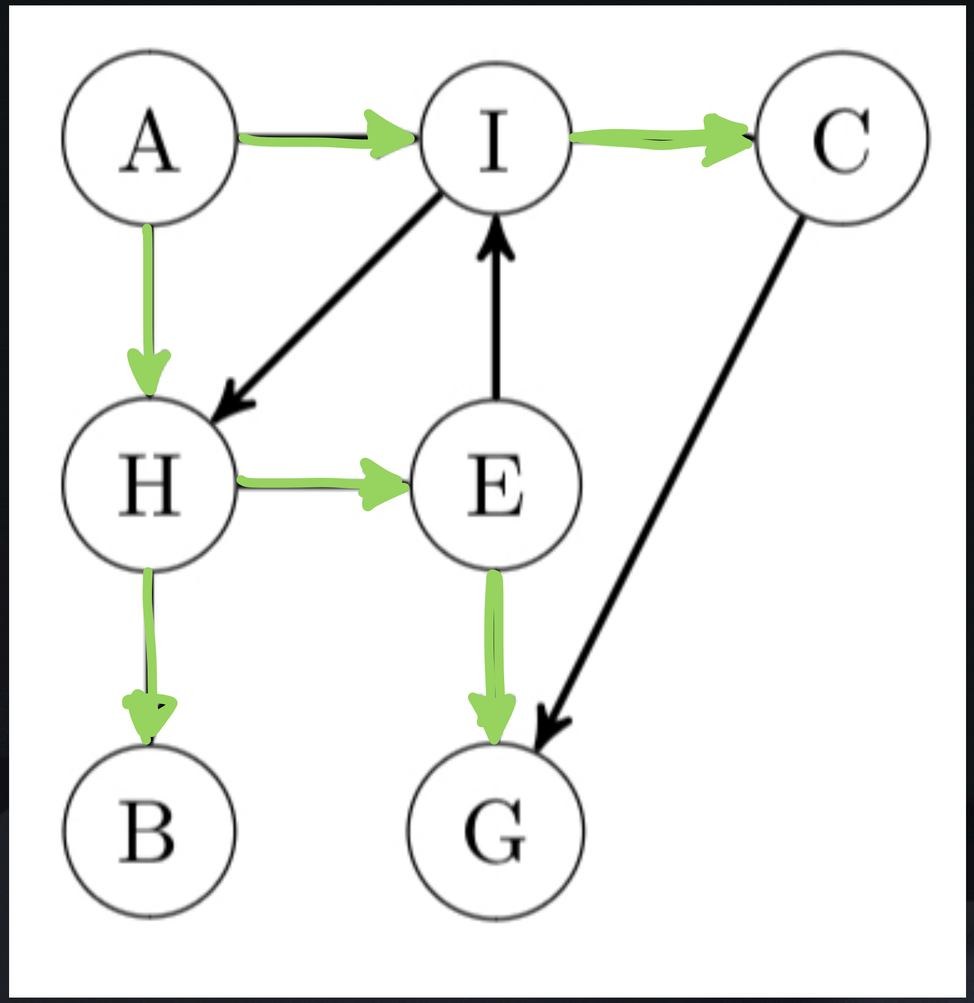
BFS

- Zeichne BFS Baum beginnend bei A
- Wir nehmen immer zuerst den Knoten der lexikografisch zuerst kommt



BFS

- distA[A,B,C,D,E,F,G,H,I] = [0,2,2,∞,2,∞,3,1,1]



Exercise Recommendations

Aufgabenblatt 5

- 10.1 Bonus
- 10.2 Bonus
- 10.3 Typisches Graphen Problem
- 10.4 Mühsam aber eher einfach
- 10.5 Bonus

Leetcode of the Week

- 129. Sum Root to Leaf Numbers
- https://leetcode.com/problems/sum-root-to-leaf-numbers/

Peergrading

- Aufgabe 9.4b
- Grading Scheme in Lösungen beachten