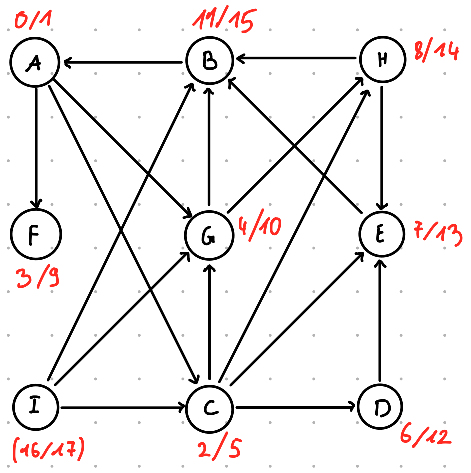
Serie 11

Exercise 11.1



Exercise 11.2

a) $O(|E|^k \cdot |V|^2)$

$\binom{|E|}{k} \leq |E|^k$ $\qquad$ $|E|^k$ Iterationen , jede Iteration benötigt $O(|V|^2)$

$\Rightarrow O(|E|^k \cdot |V|^2)$

b) $O((k \cdot |V|)^2)$

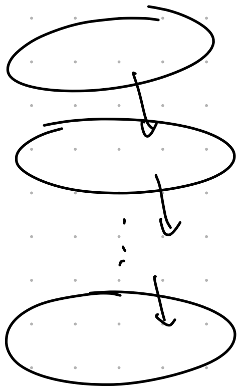
$G' = (V', E', c')$ $\qquad$ $V' = \{v^{(l)} : v \in V, l \in \{0,1,2,\dots,k\}\}$ $\qquad$ $|V'| = (k+1) \cdot |V|$

We define E' and $c' : E' \to \mathbb{R}_{\geq 0}$

(1) $\forall (v,w) \in E,\ 0 \leq l \leq k$: $(v^{(l)}, w^{(l)}) \in E'$ with $c'((v^{(l)}, w^{(l)})) = c((v,w))$

(2) $\forall (v,w) \in E,\ 0 \leq l \leq k-1$: $(v^{(l)}, w^{(l+1)}) \in E'$ with $c'((v^{(l)}, w^{(l+1)})) = 0$

(3) $\forall 0 \leq l \leq k-1$: $(t^{(l)}, t^{(l+1)}) \in E'$ with $c'((t^{(l)}, t^{(l+1)})) = 0$



Dijkstra von $s^{(0)}$

$\Rightarrow$ return$(d[t^{(k)}])$

Exercise 11.4

(a) (1) V is the set of cities

There are directed edges between the cities (in both dir.) if they are connected by a highway.

$c(e)$ Preis, den Highway zu befahren - Geld, was passenger bezahlt

(2) Shortest paths

(3) Bellman-Ford + es gibt keine negativen Zyklen

(b)

(1)

$V' = V \times \{0,1\}$

For every edge $e = (u,v) \in E$ $((u,0),(v,0))$ $((u,1),(v,1))$ in E' if it is a normal highway

if highway in bad condition: $((u,0),(v,1))$ in E'

(2) Shortest paths, mit Bellman-Ford

Exercise 11.3

Exercise 11.3 *Language Hiking.*

Alice loves both hiking and learning new languages. Since she moved to Switzerland, she has always wanted to discover all four language regions of the country in a single hike – but she is not sure whether her week of vacation will be sufficient.

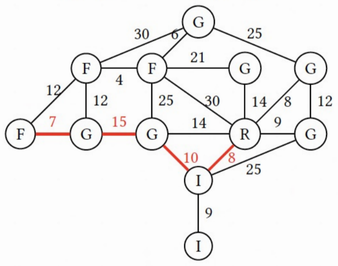
You are given a graph $G = (V, E)$ representing the towns of Switzerland. Each vertex V corresponds to a town, and there is an (undirected) edge $\{v_1, v_2\} \in E$ if and only if there exists a direct road going from town v_1 to town v_2. Additionally, there is a function $w : E \to \mathbb{N}$ such that $w(e)$ corresponds to the number of hours needed to hike over road e, and a function $\ell : V \to \{\text{G}, \text{F}, \text{I}, \text{R}\}$ that maps each

4

town to the language that is spoken there[3]. For simplicity, we assume that only one language is spoken in each town.

Alice asks you to find an algorithm that returns the walking duration (in hours) of the shortest hike that goes through at least one town speaking each of the four languages.

For example, consider the following graph, where languages appear on vertices:

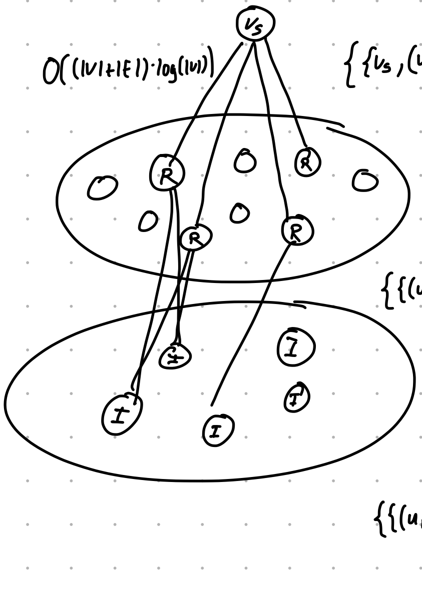


The shortest path satisfying the condition is marked in red. It goes through one R vertex, one I vertex, two G vertices and one F vertex. Your algorithm should return the cost of this path, i.e., 40.

(a) Suppose we know the order of languages encountered in the shortest hike. It first goes from an R vertex to an I vertex, then immediately to a G vertex, and reaches an F vertex in the end, after going through zero, one or more additional G vertices. In other terms, the form of the path is RIGF or RIG...GF. In this case, describe an algorithm which finds the shortest path satisfying the condition, and explain its runtime complexity. Your algorithm must have complexity at most $O((|V| + |E|)\log|V|)$.

Hint: *Consider the new vertex set $V' = V \times \{1,2,3,4\} \cup \{v_s, v_d\}$, where v_s is a 'super source' and v_d a 'super destination' vertex.*

$O((|V|+|E|)\cdot\log(|V|))$



$\{\{v_s,(v,1)\} \mid l(v) = R\}$

$\{\{(u,1),(v,2)\} \mid \{u,v\} \in E, l(v) = I\}$

$\uparrow$ 2 $\shortparallel$ $\uparrow$ 3

3   4

$\{\{(u,3),(v,3)\} \mid \{u,v\} \in E, l(v) = G\}$

$\{\{(v,4),v_d\} \mid v \in V\}$

Now we don't make the assumption in (a). Describe an algorithm which finds the shortest path satisfying the condition. Briefly explain your approach and the resulting runtime complexity. To obtain full points, your algorithm must have complexity at most $O((|V| + |E|) \log |V|)$.

Hint: *Consider the new vertex set* $V' = V \times \{0,1\}^4 \cup \{v_s, v_d\}$, *where* v_s *is a 'super source' and* v_d *a 'super destination' vertex.*

G F I R

$(v, 0, 0, 0, 0)$

$(v, 1, 0, 1, 1)$

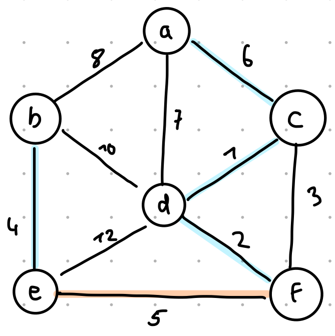
Vorlesung Recap

Boruvka

```
$F \leftarrow \emptyset$
while F nicht Spannbaum:
    $(S_1, \dots, S_k) \leftarrow$ ZHKs von F
    $(e_1, \dots, e_k) \leftarrow$ minimale Kanten an $S_1, \dots, S_k$
    $F \leftarrow F \cup \{e_1, \dots, e_k\}$
```

$O(|V|+|E|)$ $\quad \leq \log|V|$ Iterationen

$\Rightarrow O((|V|+|E|) \cdot \log|V|)$

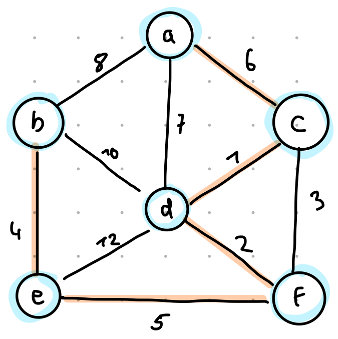


🟦 ≙ 1. Iteration

🟧 ≙ 2. It.

Prim (G, s):

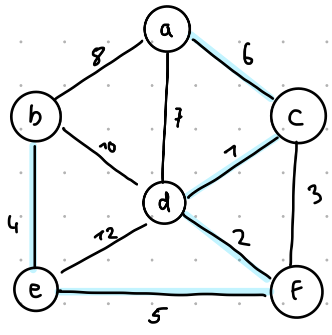
```
$F \leftarrow \emptyset$
$S \leftarrow \{s\}$
while F nicht Spannbaum:
    $u^* v^* \leftarrow$ minimale Kante an S  $(u^* \in S, v^* \notin S)$
    $F \leftarrow F \cup \{u^* v^*\}$ ; $S \leftarrow S \cup \{v^*\}$
```



$O((|V|+|E|) \cdot \log(|V|))$ (wie Dijkstra)

Kruskal

```
$F \leftarrow \emptyset$  (sichere Kanten)
for $uv \in E$, aufst. sortiert
    if u,v in versch. ZHKs von F
    $F \leftarrow F \cup \{uv\}$
```



$$O(\underbrace{|E| \cdot \log|E|}_{\uparrow \text{ Sortieren}} + \underbrace{|V| \cdot \log|V|}_{\uparrow \text{ Union Find}})$$

$|E| \leq O(|V|^2)$

$O(|E| \cdot \log(|V|^2) + |V| \cdot \log(|V|))$

$\leq O((|V|+|E|) \cdot \log(|V|))$

Theory Task T4. / 14 P

Tina wants to travel around Iceland by car in December. There are n towns in Iceland labeled by $1, 2, \dots, n$. For some pairs $i, j \in \{1, 2, \dots, n\}$, towns i and j are connected by a two-way road, represented by $\{i, j\}$. The set of all roads is denoted with $R = \{\{i_1, j_1\}, \{i_2, j_2\}, \dots, \{i_m, j_m\}\}$, with road $\{i_k, j_k\}$ having length L_k. If no road is blocked, the road network is such that Tina can go from town 1 to every other town in Iceland.

However, in winter, the weather conditions in Iceland can get very bad. Unfortunately, a snowstorm takes place before her journey. As a result, some roads are blocked. Let $S \subseteq R$ be the set of roads being blocked.

As format of input, you are given the set $E = \{(i_1, j_1, b_1, L_1), (i_2, j_2, b_2, L_2), \dots, (i_m, j_m, b_m, L_m)\}$, where, for any $k \in \{1, 2, \dots, m\}$, $b_k = 1$ if road $\{i_k, j_k\}$ is in S, and $b_k = 0$ otherwise.

/ 2 P a) Due to the snowstorm, all roads in S are blocked. Write down the pseudocode of an algorithm, which checks whether Tina can reach every town in Iceland starting from town 1. The algorithm should run in time $O(m)$. You don't need to describe in detail how to convert a set of edges to an adjacency list.

/ 4 P b) The government wants to clear some of the blocked roads (i.e., make them passable) such that Tina can reach (via passable roads only) every other town starting from town 1. Describe an algorithm which finds the minimum total length of roads in S to be cleared, such that Tina can arrive at every other town starting from town 1. The algorithm should run in time $O(m \cdot \log(n))$.

You need to address the following aspects in your algorithm description:

- the graph algorithm used to solve this problem;
- the construction of the graph that you run this algorithm on;
- the correctness of the algorithm, i.e why any solution to the original problem gives you a solution to the graph problem and why any solution to the graph problem gives you a solution to the original problem;
- the total running time of your algorithm.

You can directly use the algorithms covered in lecture material, and you can directly use their running time bounds without proof.

/ 4 P c) Due to the effort of the road authority, the roads in S are eventually not blocked anymore, but they are still in unpleasant condition. Now Tina wants to begin her journey and starts from town 1. Suppose Tina's car travels with speed 1 on every road (i.e., it will take L_k units of time to travel through $\{i_k, j_k\}$). However, whenever Tina travels through two roads in S consecutively, she needs to make a stop and spends D amount of time for repairing her car.

For an example, suppose Tina's car travels through roads $e_1 = \{1, 2\}, e_2 = \{2, 4\}, e_3 = \{4, 7\}, e_4 = \{7, 10\}, e_5 = \{10, n\}$. If $e_2, e_3, e_4, e_5 \in S$ and $e_1 \notin S$, then Tina's car needs to stop in towns 7, 10, and it takes her $2D + L(e_1) + L(e_2) + L(e_3) + L(e_4) + L(e_5)$ time to reach town n, where $L(e_1), L(e_2), \dots, L(e_5)$ represent the lengths of roads $e_1, e_2, \dots, e_5$.

Describe an algorithm which finds the shortest time for Tina to reach town n. The algorithm should run in time $O(m \cdot \log(n))$.

You need to address the following aspects in your algorithm description:

- the graph algorithm used to solve this problem;
- the construction of the graph that you run this algorithm on;
- the correctness of the algorithm, i.e why any solution to the original problem gives you a solution to the graph problem and why any solution to the graph problem gives you a solution to the original problem;
- the total running time of your algorithm.

Hint: You want to consider a new graph with vertex set $\{1, 2, \dots, n\} \times \{0, 1\}$, where for each town i, $\{0, 1\}$ records whether town i is reached immediately after going through a road in S.

$V' = V \times \{0,1\}$

Knoten $(u,1)$ $\Rightarrow$ die letzte Kante, um u zu erreichen, war blockiert

Knoten $(u,0)$ $\Rightarrow$ die letzte Kante, um u zu erreichen, war nicht blockiert

Shortest-path problem $\Rightarrow$ Dijkstra $\quad \min\begin{pmatrix}(n,0),\\(n,1)\end{pmatrix}$

```
edgeList ← list of edges, as given in input
visited [1,...,n] ← [false,..., false]
adj [1,...,n] ← adjacency list of tuples

function DFS(u)
    visited [u] ← true
    for each (u,v,b,L) ∈ adj(u)
        if b=0 and !visited [v]
            DFS(v)

DFS(1)
return true if visited = [true,..., true] else return false
```

$c(e) = 0$ if $e \notin S$

$c(e) = L_e$ if $e \in S$

G anpassen: $O(m)$

MST: $O(m \cdot \log(n))$