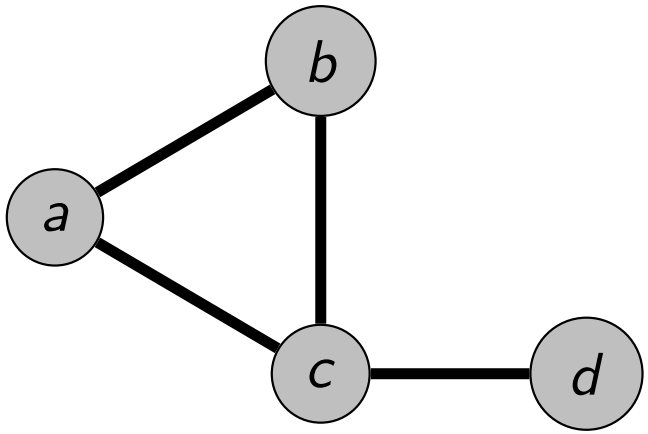
Minimální kostry

Téma přednášky a hlavní příklad jsou udělány podle kapitoly 21 vynikající knihy Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, *Introduction to algorithms* (4th ed), MIT Press, 2022.

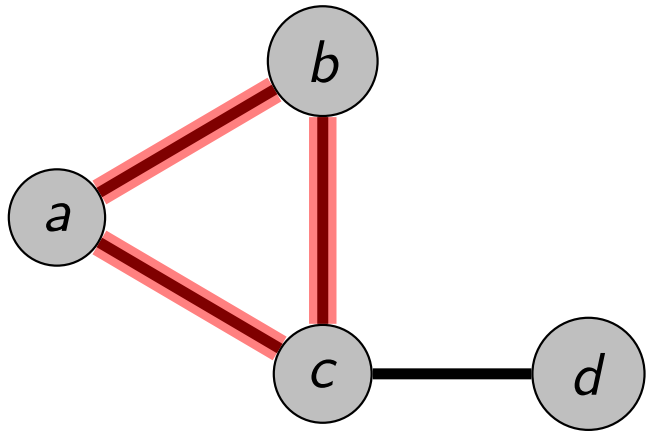
Připomenutí pojmů

1. Neorientovaný graf G je dvojice (V, E), kde V je konečná množina (prvkům V říkáme vrcholy grafu G) a E je množina dvouprvkových podmnožin množiny V (prvkům E říkáme hrany grafu G). Říkáme, že $\{u, v\} \in E$ je hrana mezi vrcholy u a v.
   Například: (V, E), kde $V = \{a, b, c, d\}$ a $E = \Big\{\{a, b\}, \{b, c\}, \{a, c\}, \{c, d\}\Big\}$ je neorientovaný graf s „obrázkem“



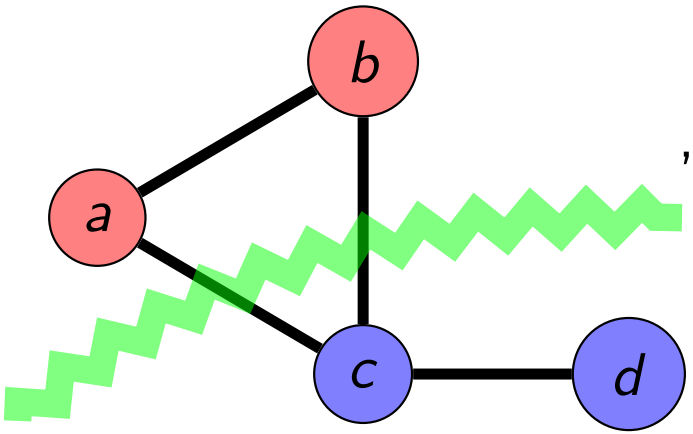
Připomenutí pojmů (pokrač.)

2. Cyklus (také: kružnice) v neorientovaném grafu (V, E) je posloupnost $v_1 v_2 v_3 v_4 \ldots v_k$ vrcholů ve V taková, že $k \geq 3$ a platí: $\{v_1, v_2\} \in E$, $\{v_2, v_3\} \in E$, $\ldots$, $\{v_{k-1}, v_k\} \in E$ a $v_k = v_1$. Číslu k říkáme délka cyklu (a cyklům délky k také říkáme k-cykly).
   Například: posloupnost abc je 3-cyklus v následujícím grafu



Připomenutí pojmů (pokrač.)

3. Řez na množině vrcholů grafu (V, E) je dvojice neprázdných množin S a $V \setminus S$ vrcholů (tj., jde o rozdělení množiny V všech vrcholů na dvě neprázdné podmnožiny: S a $V \setminus S$). Takový řez zapisujeme jako $(S, V \setminus S)$.
Například: $S = \{a, b\}$ a $V \setminus S = \{c, d\}$ je řez

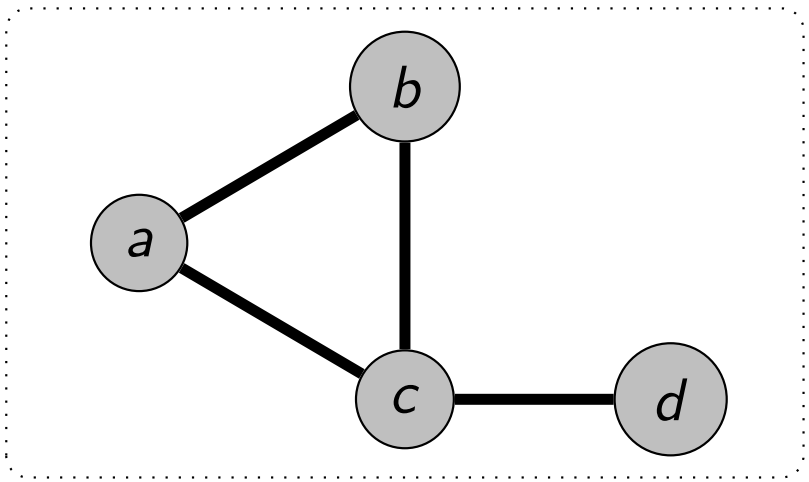


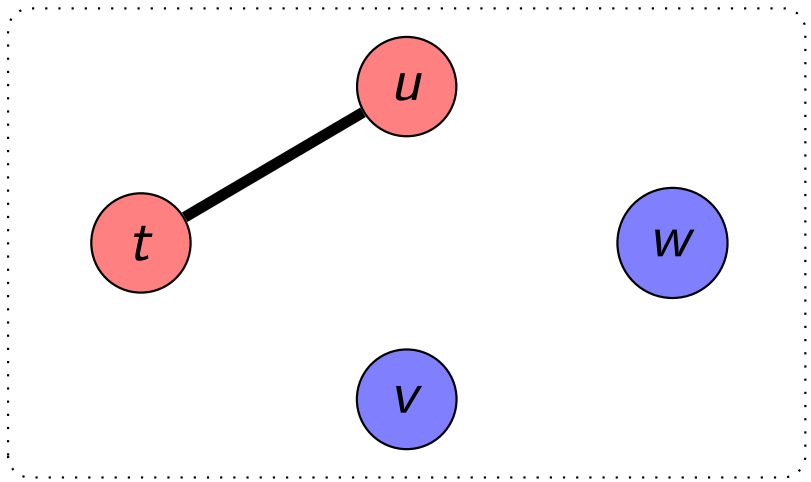
„dělící čára“ řezu

Řez si můžeme představit jako obarvení množiny všech vrcholů přesně dvěma barvami.

Připomenutí pojmů (pokrač.)

4. Graf (V, E) je souvislý, když pro každý řez $(S, V \setminus S)$ existuje alespoň jedna hrana v E, která přes řez přechází, tj., existuje alespoň jedna hrana $\{u, v\} \in E$ tak, že $u \in S$ a $v \in V \setminus S$. Například:

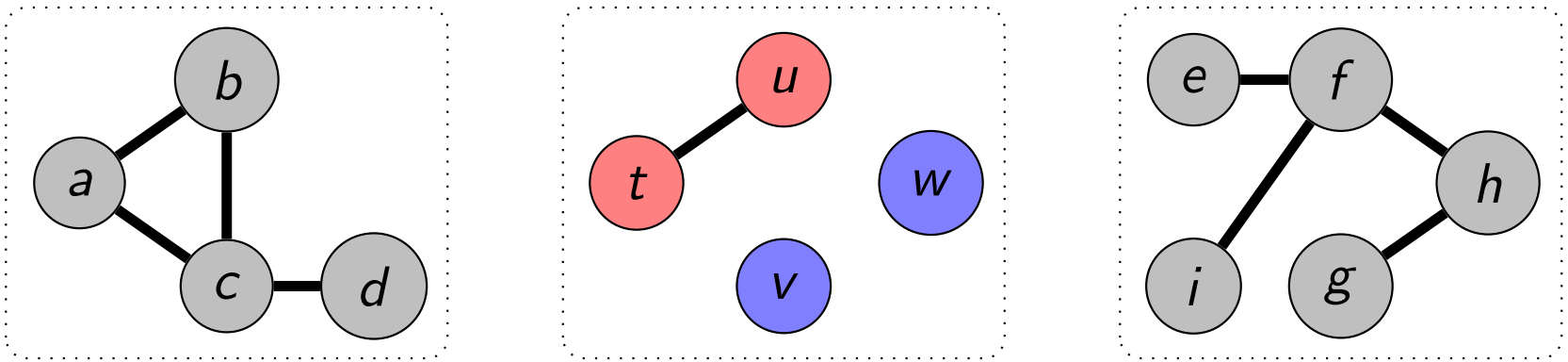




graf nalevo je souvislý, graf napravo souvislý není (je nakreslen i řez, přes který nepřechází žádná hrana).

Připomenutí pojmů (pokrač.)

5. Strom je souvislý graf (V, E) bez cyklů.
   Například:

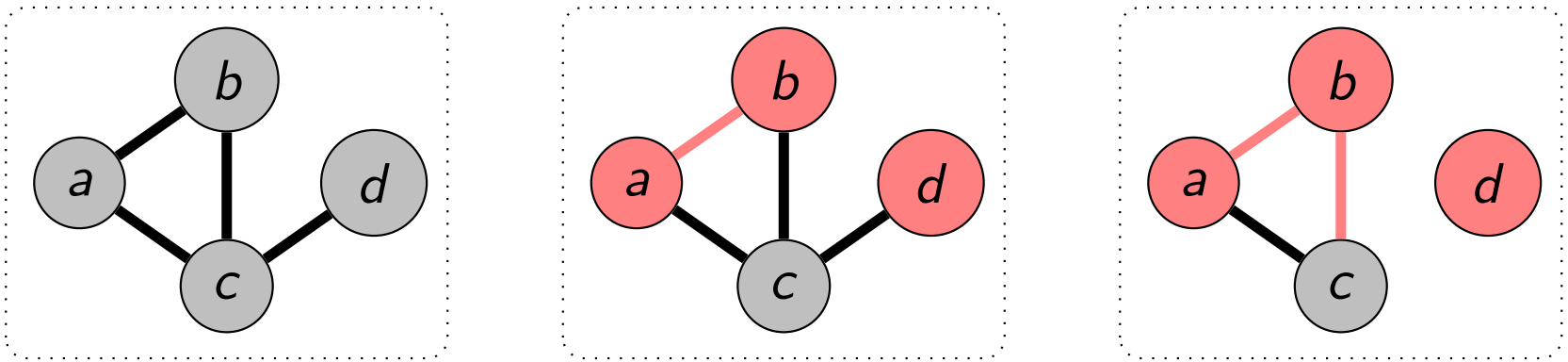


graf nalevo není strom (obsahuje cyklus abc), graf uprostřed není strom (není souvislý) a graf napravo je strom.
Víme: pokud má strom n vrcholů, pak má $n - 1$ hran.

Připomenutí pojmů (pokrač.)

6. Podgraf grafu (V, E) je graf (V', E'), pro který platí dvě podmínky:
   1. $V' \subseteq V$ a $E' \subseteq E$.
   2. Je-li $\{u, v\} \in E'$, pak musí platit $u \in V'$ a $v \in V'$.

   Například (vrcholy ve V' a hrany v E' jsou namalovány červeně):

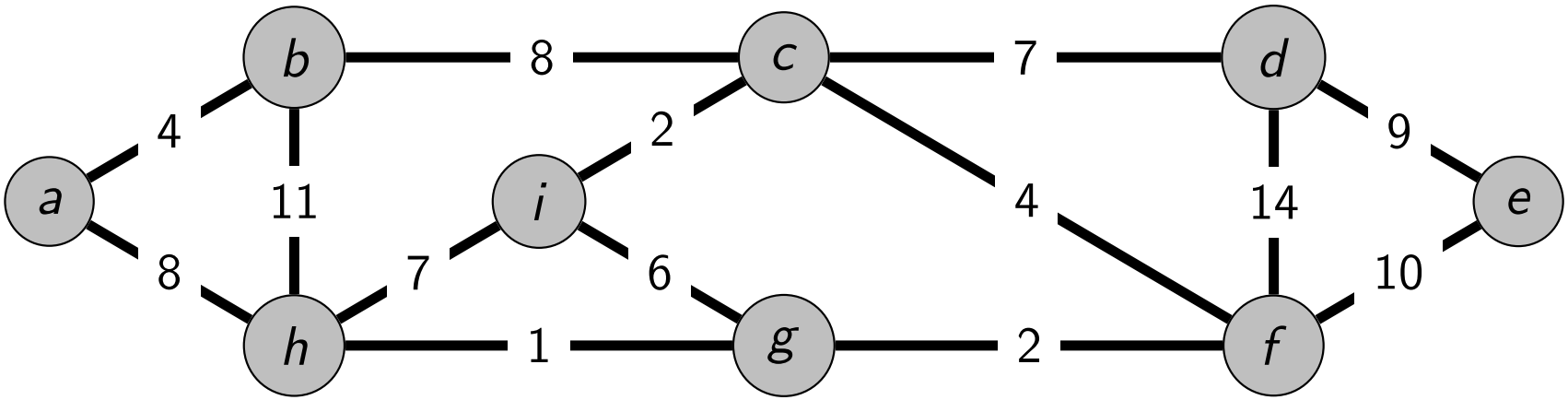


Uprostřed: jde o podgraf. Napravo: nejde o podgraf.

Definice (ohodnocený neorientovaný graf)

Ať $G = (V, E)$ je neorientovaný graf. Funkci $w : E \longrightarrow \mathbb{R}$ říkáme ohodnocení hran grafu G a dvojici (G, w) říkáme ohodnocený graf. Hodnotě $w(\{u, v\})$ říkáme cena (také: váha) hrany $\{u, v\}$.

Příklad ohodnoceného grafu



Například: $w(\{a, h\}) = 8$, tj. hrana $\{a, h\}$ má cenu 8.

Problém minimální kostry

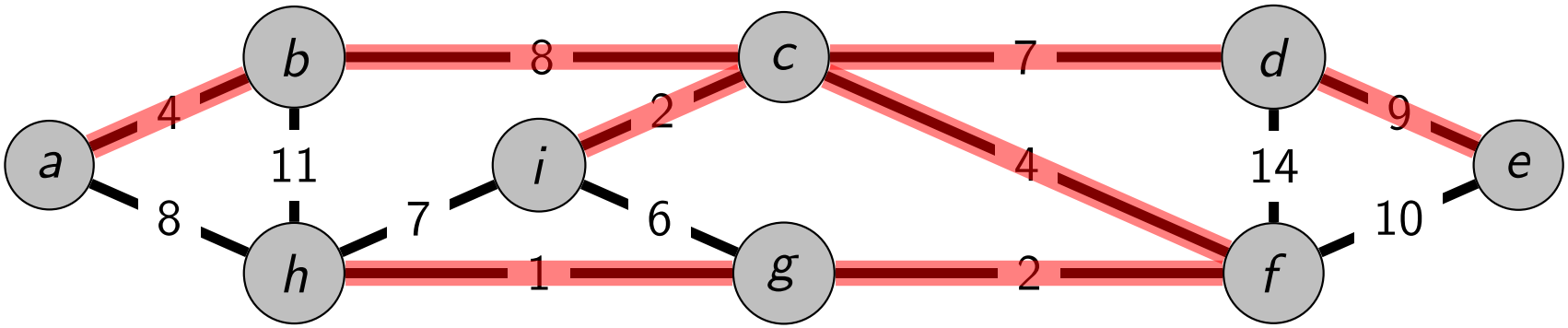
Ať (G, w) je ohodnocený neorientovaný graf. Nalezněte jeho minimální kostru, tj. podgraf T grafu G s následujícími dvěma vlastnostmi:

1. T je kostra grafu G, tj. T je strom a obsahuje všechny vrcholy grafu G.
2. Cena $w(T)$ stromu T, která je definovaná následovně

$$w(T) = \sum_{\{u,v\} \text{ je hrana v } T} w(\{u, v\})$$

je nejmenší možná.

Například toto



je minimální kostra T ohodnoceného grafu (G, w) a platí $w(T) = 4 + 8 + 7 + 9 + 2 + 4 + 1 + 2 = 37$.

Má každý ohodnocený graf (G, w) minimální kostru?

NE! Graf G, který není souvislý, nemůže mít žádnou kostru.

Důvod: Kostra je strom na všech vrcholech grafu G, který je podgraf grafu G. Protože G není souvislý graf, nelze takový strom nalézt.

Má každý souvislý ohodnocený graf (G, w) minimální kostru?

ANO! Graf G, který je souvislý, má alespoň jednu minimální kostru.

Důvod (stupidní algoritmus):[a]

1. Označte jako n počet vrcholů množiny G.
2. Vypište všechny podgrafy grafu G, které mají n vrcholů a $n-1$ hran.
3. U každého grafu z bodu 2 zkontrolujte, zda je souvislý. Pokud ano, nalezli jsme podgraf grafu G, který je strom a obsahuje všechny vrcholy grafu G. Všechny takové stromy vložte do množiny $\mathscr{T}$.
4. U každého stromu T z množiny $\mathscr{T}$ spočtěte $w(T)$ a vyberte strom T_0 takový, že $w(T_0)$ je nejmenší.
5. Strom T_0 je minimální kostra grafu G.

[a] Promyslete, že jde o algoritmus stupidní, nicméně korektní.

Dokázali jsme tvrzení:

Ohodnocený graf (G, w) má minimální kostru právě tehdy, když graf G je souvislý.

Kosmetická vada:

Minimální kostru souvislého ohodnoceného grafu umíme hledat algoritmem.
Námi popsaný algoritmus je velmi komplikovaný[a] a máme pocit, že není efektivní.[b]

---

[a] Například: Jak přesně vypadá algoritmus, který kontroluje souvislost grafu?
[b] Například: Vypisujeme všechny možné podgrafy na n vrcholech, které mají $n - 1$ hran.

Existuje elegantní způsob hledání minimální kostry?

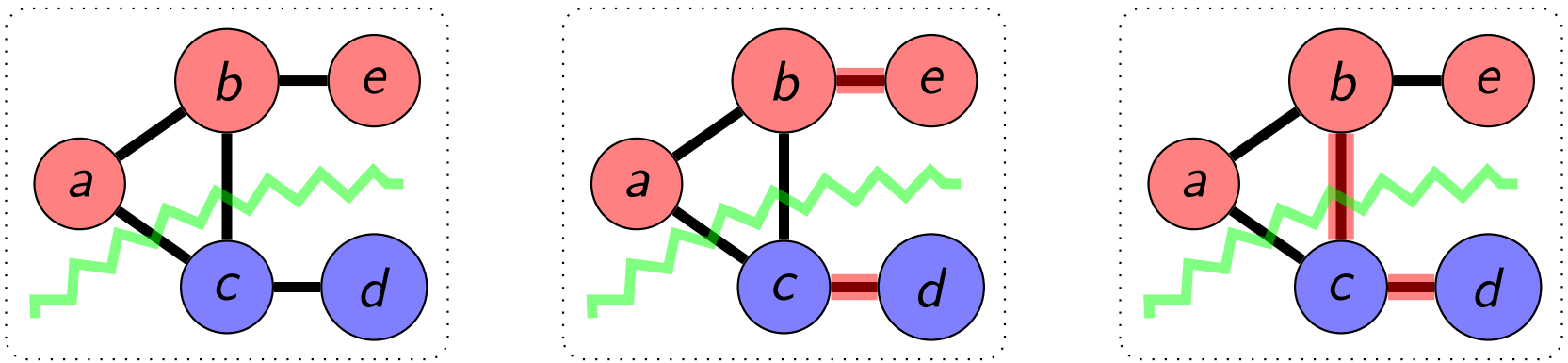
ANO! Ukážeme, že elegantních algoritmů je mnoho.

Navíc: Ukážeme to elegantně. 

Definice (řez respektující podgraf)

Řekneme, že řez $(S, V \setminus S)$ neorientovaného grafu $G = (V, E)$, respektuje podgraf (V', E') grafu G, pokud žádná hrana z E' přes řez $(S, V \setminus S)$ nepřechází.

Například:



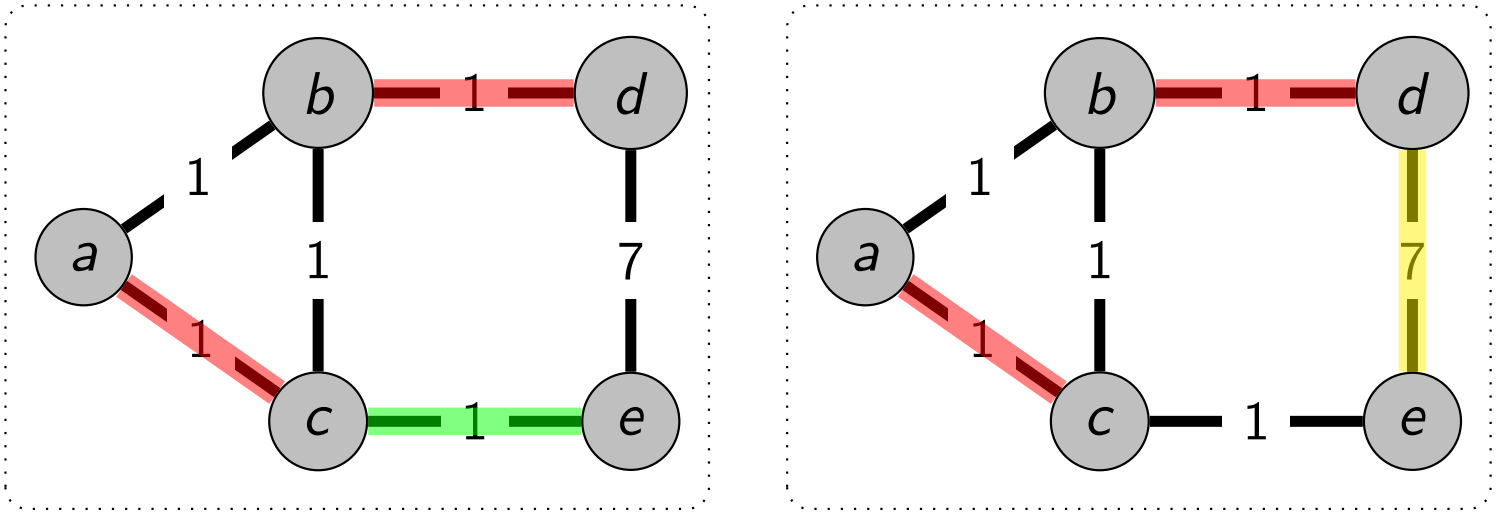
Nalevo: řez $(\{a, b, e\}, \{c, d\})$. Tento řez respektuje podgraf $(\{b, c, d, e\}, \{\{b, e\}, \{c, d\}\})$ (uprostřed) a nerespektuje podgraf $(\{b, c, d\}, \{\{b, c\}, \{c, d\}\})$ (napravo).

Definice (bezpečná hrana)

Ať (G, w) je ohodnocený graf a ať $A = (V', E')$ je podgraf grafu $G = (V, E)$, který je také podgrafem nějaké minimální kostry grafu G. Hrana $\{u, v\} \in E$ je bezpečná pro A, pokud podgraf $A + \{u, v\}$[a] grafu G je opět podgrafem nějaké minimální kostry grafu G.

---

[a] Takto značíme graf A, ke kterému jsme přidali hranu $\{u, v\}$ a vrcholy u a v. Analogicky: Jako $A - \{u, v\}$ značíme graf, ze kterého jsme vyjmuli hranu $\{u, v\}$.



hrany v A, bezpečná pro A, není bezpečná pro A

Obecný greedy algoritmus pro hledání minimální kostry

Ať (G, w) je ohodnocený souvislý neorientovaný graf.

1. Inicialisace: A je prázdný graf.
2. Dokud A není minimální kostra opakujte bod (3).
3. Nalezněte hranu $\{u, v\}$, která je bezpečná pro A. Nové A je graf $A + \{u, v\}$.
4. Konec: A je minimální kostra ohodnoceného grafu (G, w).

Potřebujeme zajistit totální korektnost algoritmu

To znamená, že potřebujeme dokázat dvě věci:

1. Terminace: Algoritmus vždy skončí.
   Variant: Jde o počet vrcholů G nezařazených do A.
2. Parciální korektnost: Pokud algoritmus skončí, pak nalezne minimální kostru.
   Invariant: V každém kroku algoritmu je A podmnožinou nějaké minimální kostry grafu (G, w).[a]

[a] To je lehké: tak jsme definovali bezpečnou hranu.

Jak bezpečné hrany nalézt? (Cut&Paste Argument)

Ať (G, w) je orientovaný graf. Ať A je podgraf grafu G, který je také podgrafem nějaké minimální kostry T. Ať $(S, V \setminus S)$ je řez, respektující podgraf A. Ať $\{u, v\}$ je nejlevnější hrana, která přechází přes řez $(S, V \setminus S)$.[a] Potom je $\{u, v\}$ bezpečná hrana pro podgraf A.

[a] Takové hraně se také říká lehká hrana.

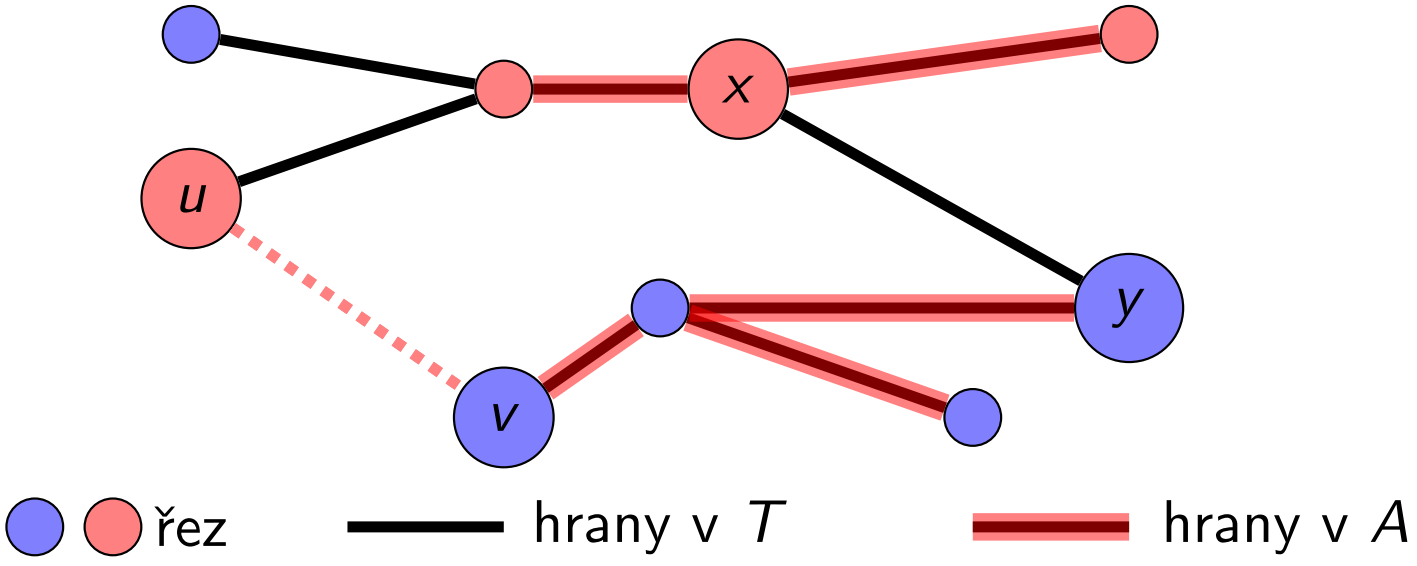
Cut&Paste Argument v programování

Technika Cut&Paste (kterou uvidíme v důkazu) se v programování používá pro důkazy korektnosti greedy algoritmů a algoritmů dynamického programování. Viz např. kapitolu 14 knihy

> Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, *Introduction to algorithms* (4th ed), MIT Press, 2022.

Důkaz

Je-li $\{u, v\}$ v T, je hotovo. Ať tedy $\{u, v\}$ není v T.



řez — hrany v T — hrany v A

ostatní hrany grafu G zakresleny nejsou

Platí: $T' = (T - \{x, y\}) + \{u, v\}$ je opět kostra grafu G.
Platí: $w(\{u, v\}) \leq w(\{x, y\})$, protože $\{u, v\}$ je lehká.
Proto $w(T') = w(T) - w(\{x, y\}) + w(\{u, v\}) \leq w(T)$.
Protože T byla minimální kostra, je i T' minimální kostra.

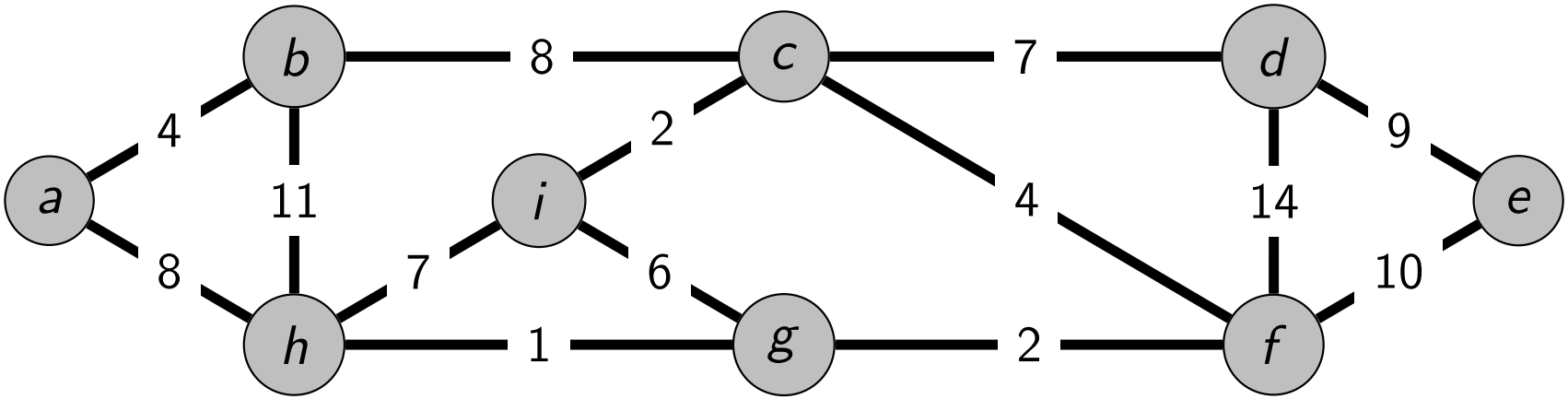
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů.

Příklad



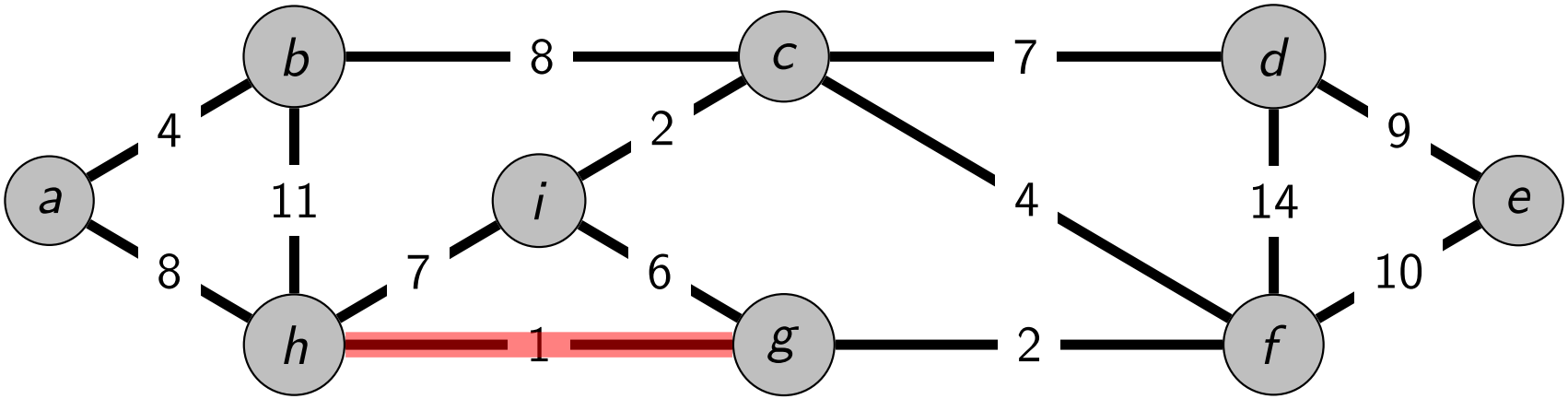
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů.

Příklad



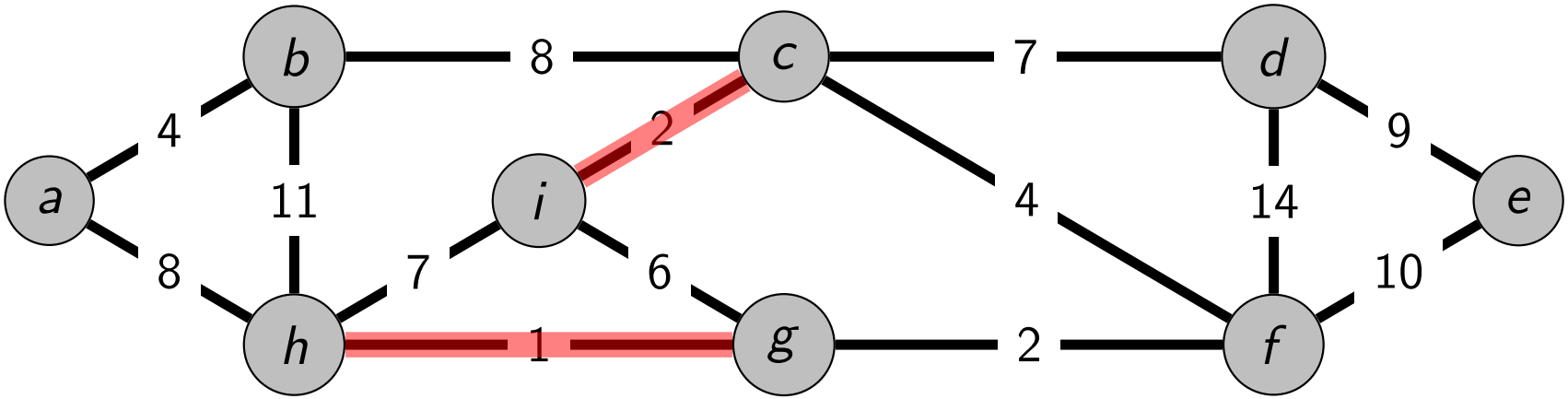
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů.

Příklad



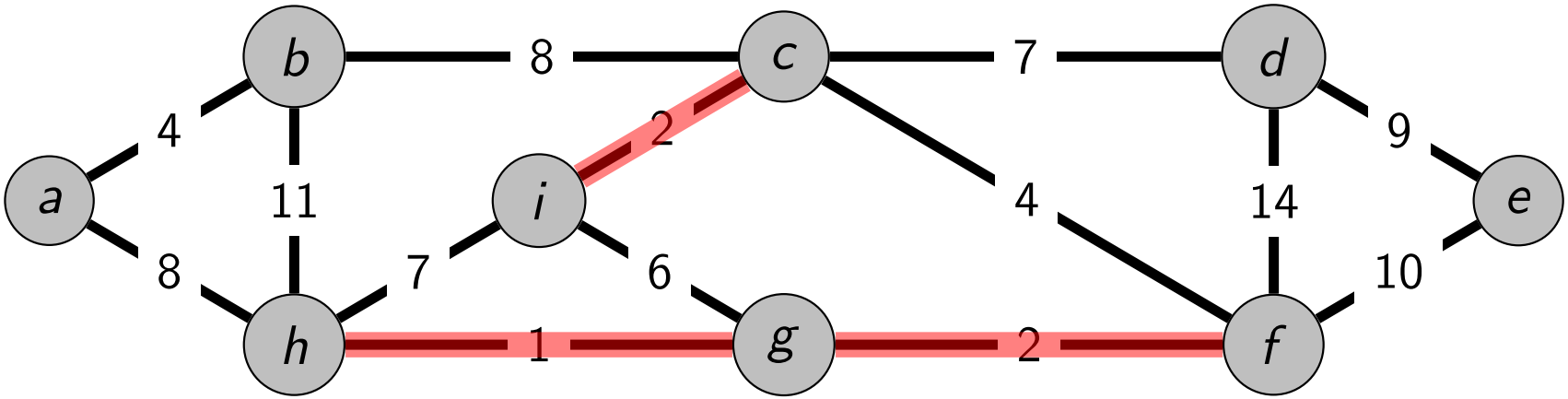
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů.

Příklad



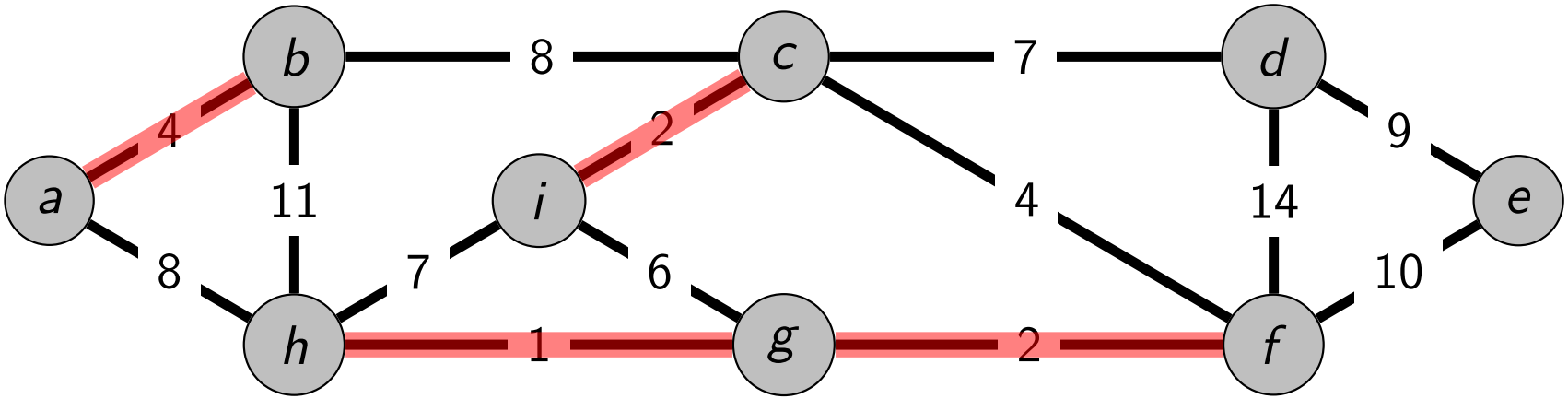
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů. 

Příklad

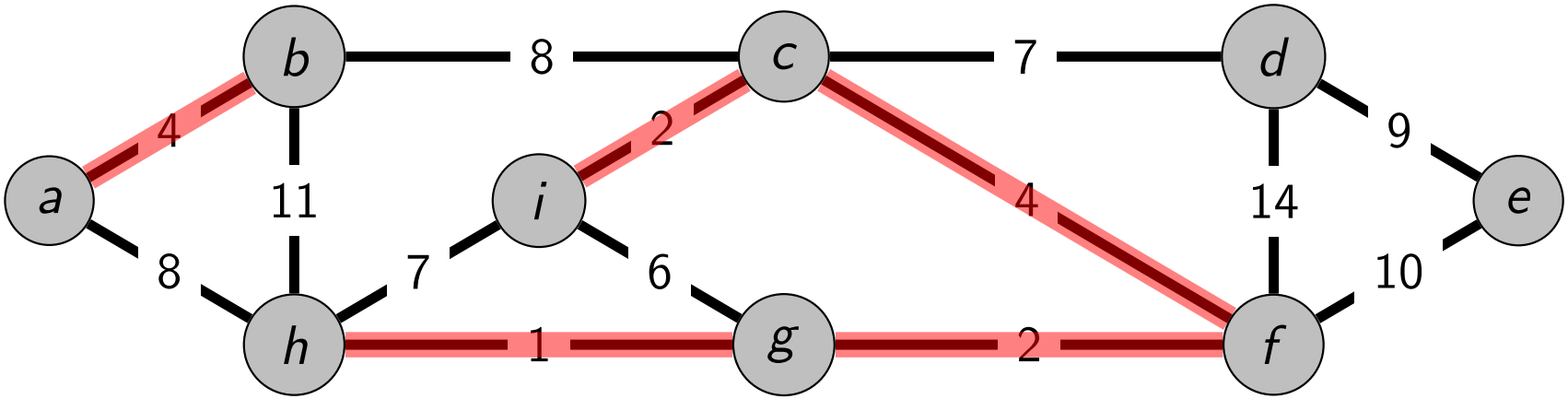


Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

[a] Les je množina stromů.

Příklad

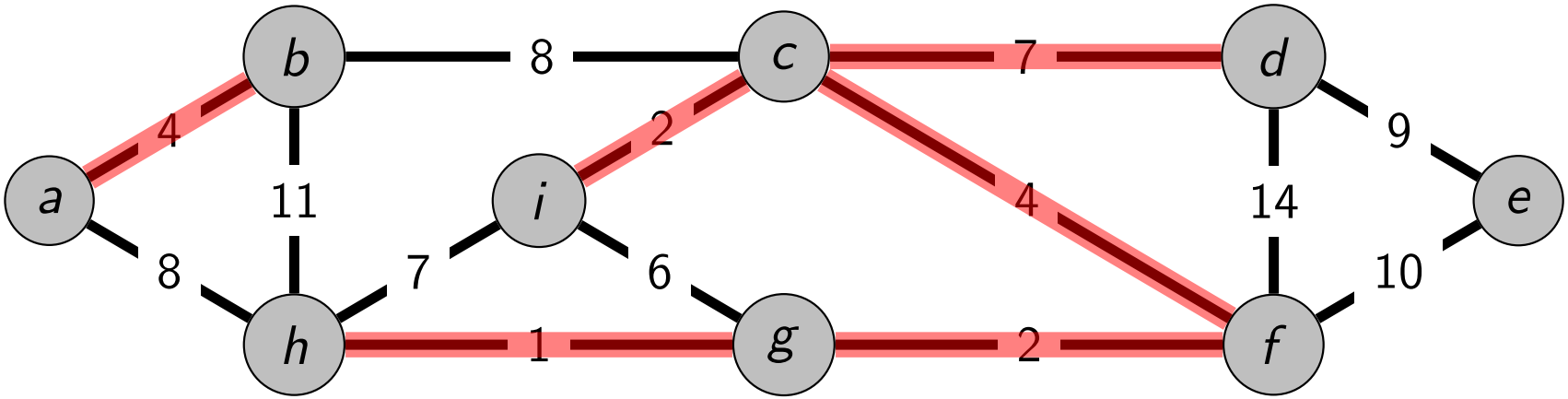


Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

[a] Les je množina stromů. 

Příklad



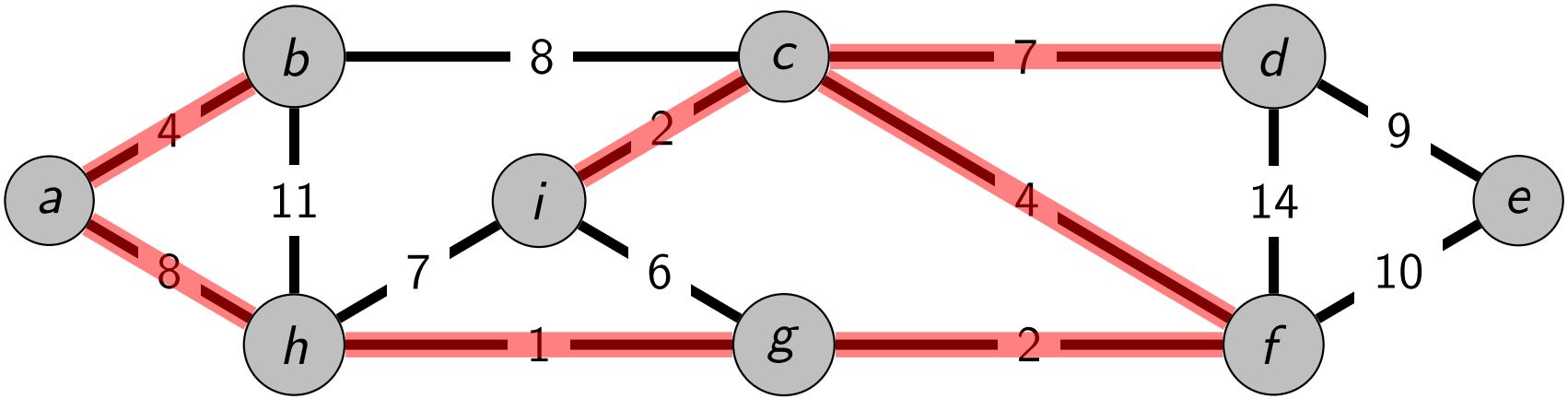
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů. 

Příklad



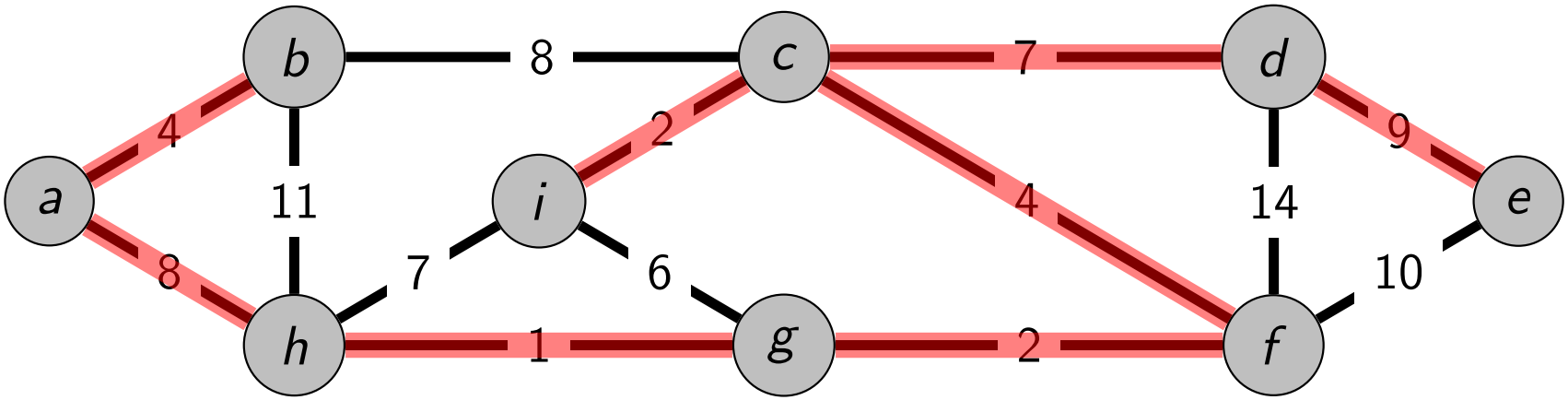
Algoritmus č. 1 (Kruskal-Borůvka)

Minimální kostru budujeme tak, že vybíráme vždy nejlevnější hrany mezi komponentami lesa vznikajícího z lesa všech vrcholů.[a]

---

[a] Les je množina stromů.

Příklad

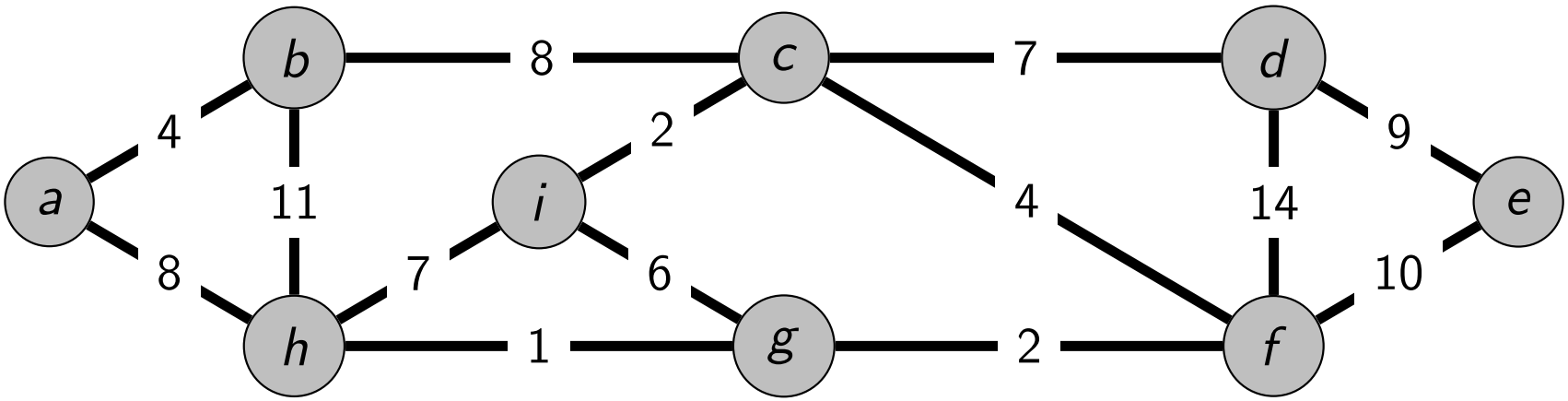


Nalezli jsme minimální kostru T s vahou
$w(T) = 1 + 2 + 2 + 4 + 4 + 7 + 8 + 9 = 37$.

Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

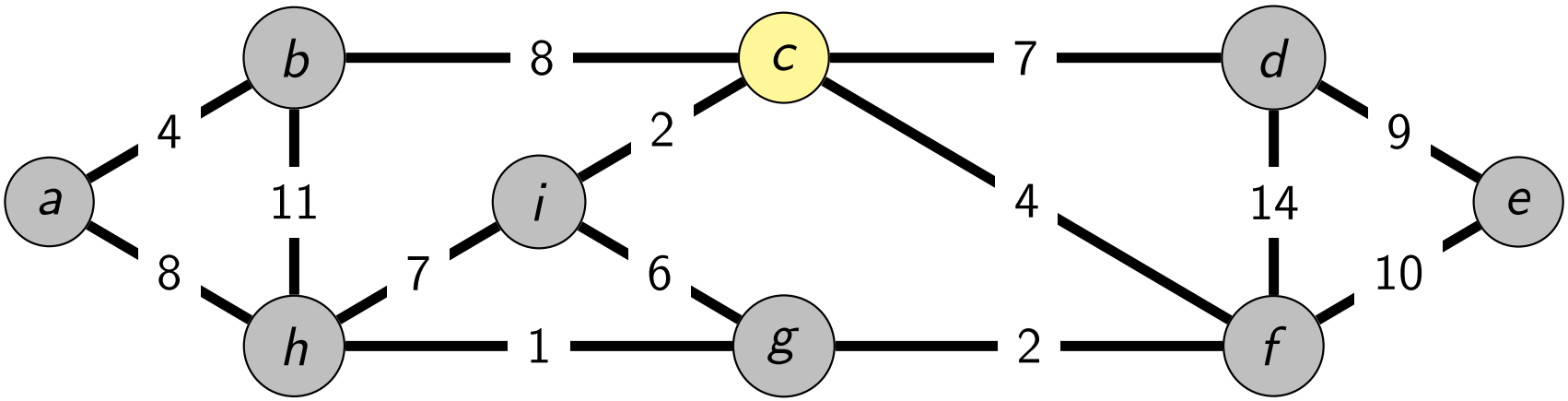
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

Příklad

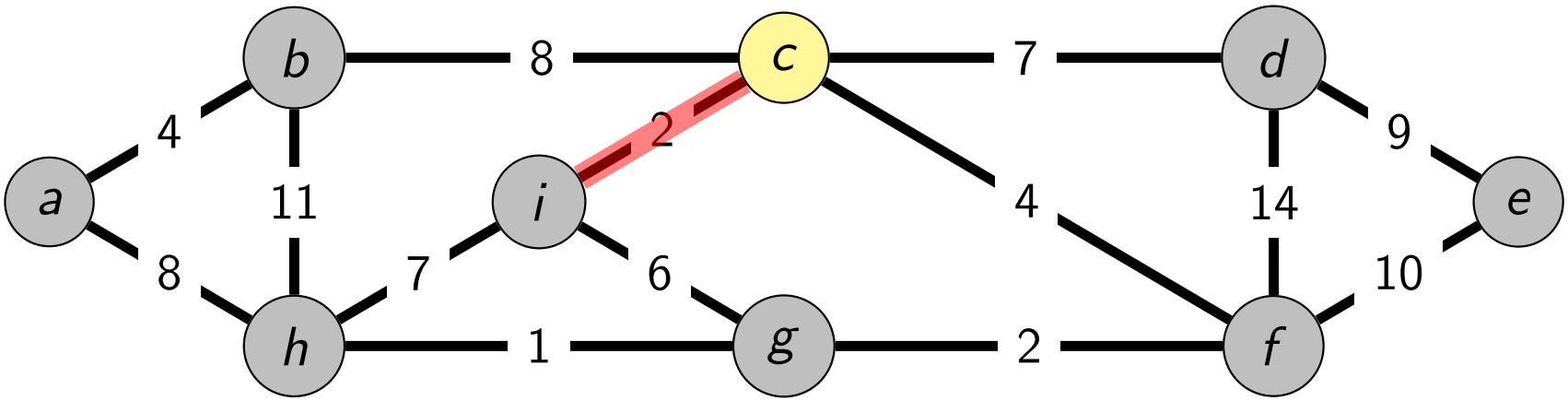


Vybrali jsme vrchol c jako semínko.

Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

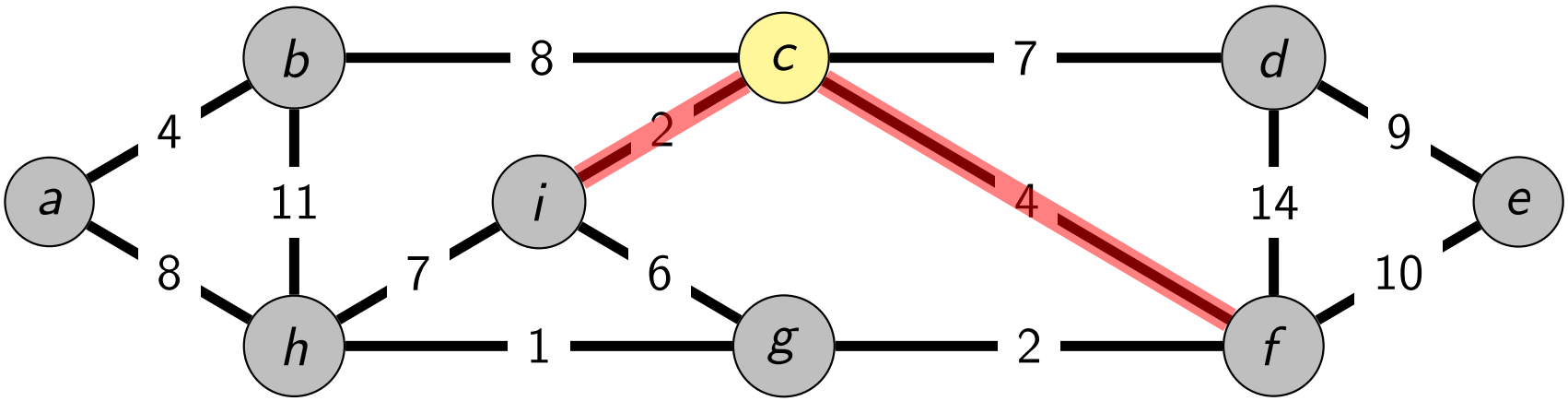
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

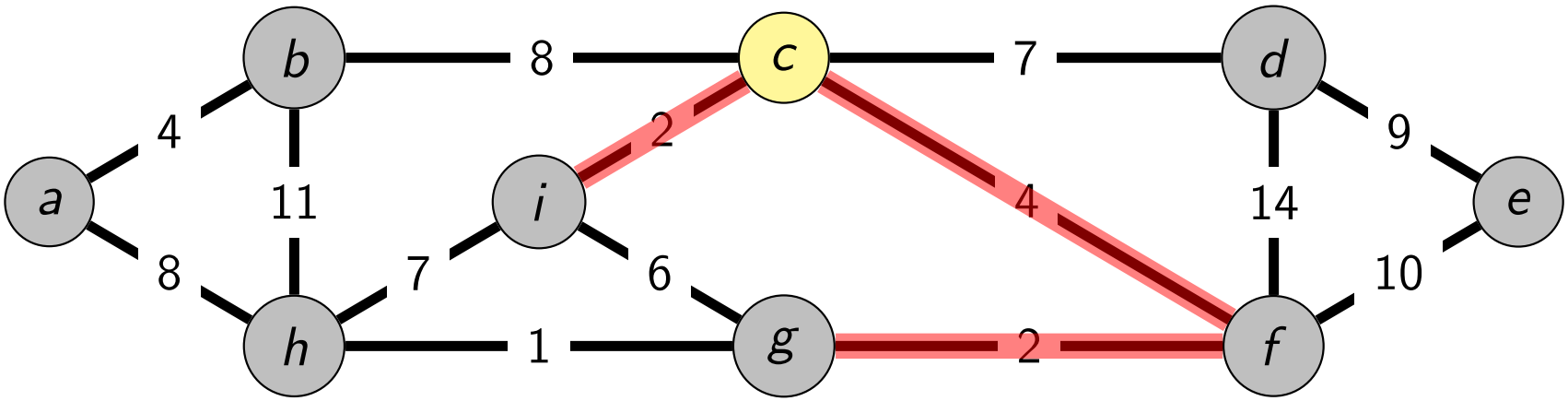
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

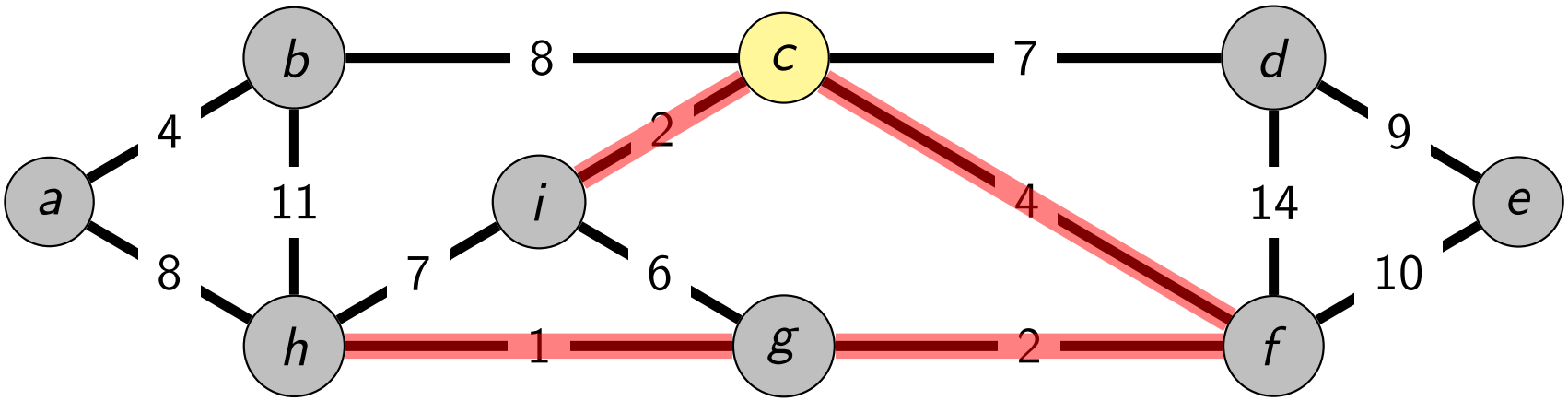
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

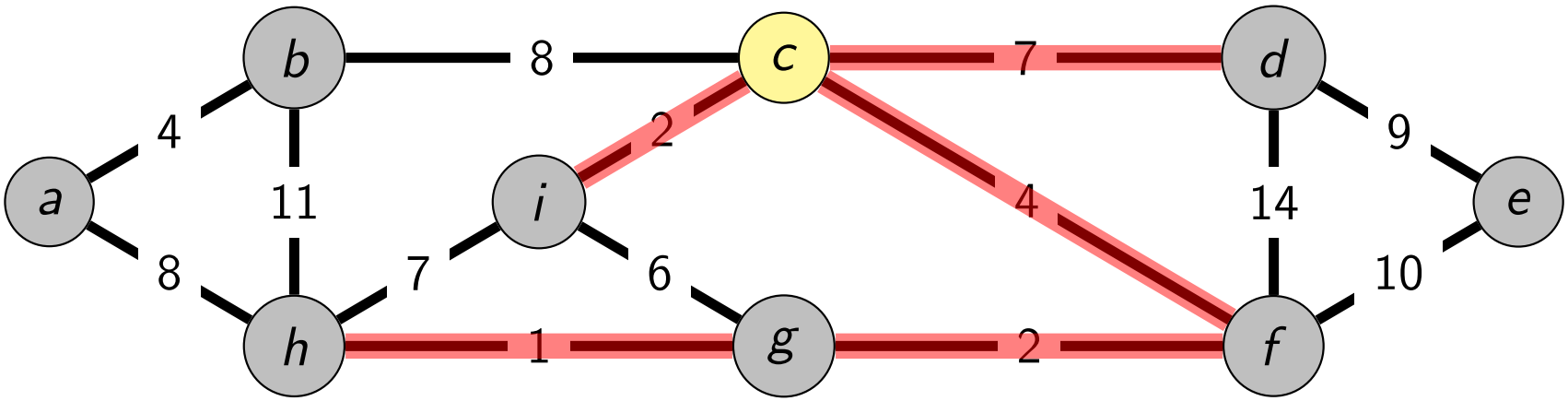
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

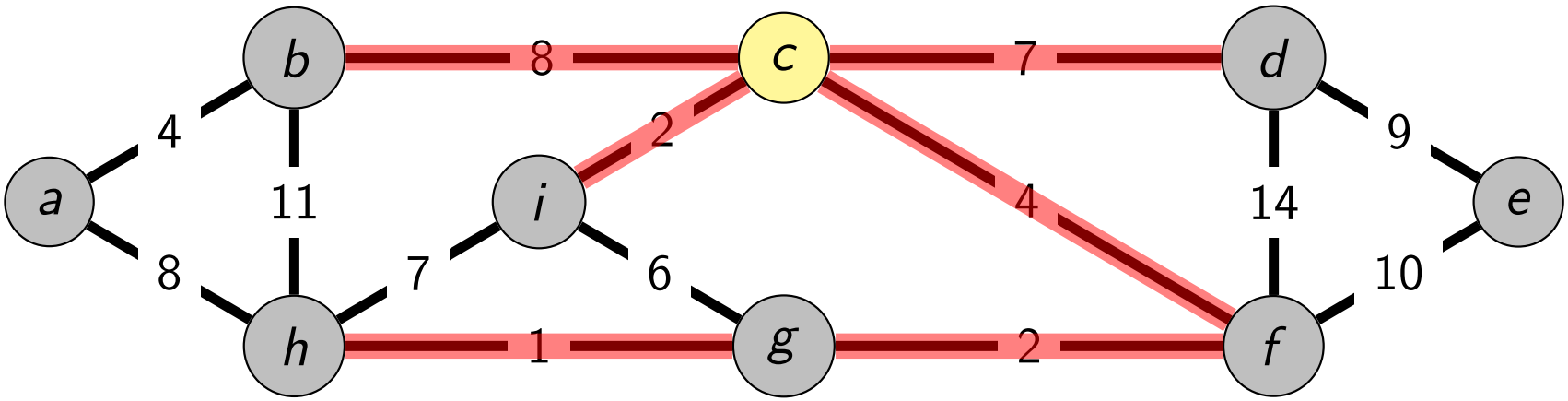
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

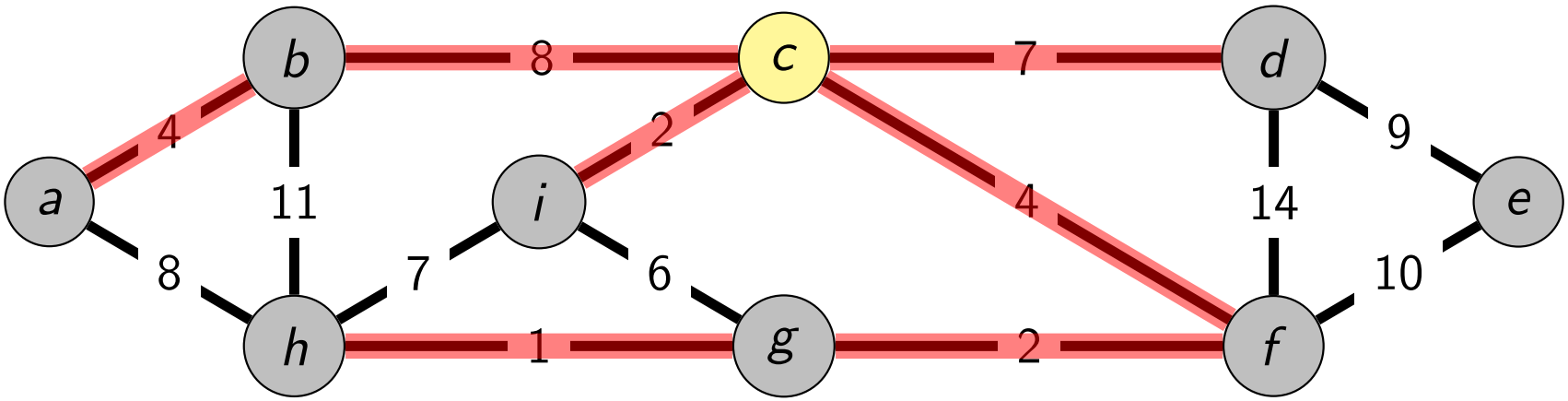
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

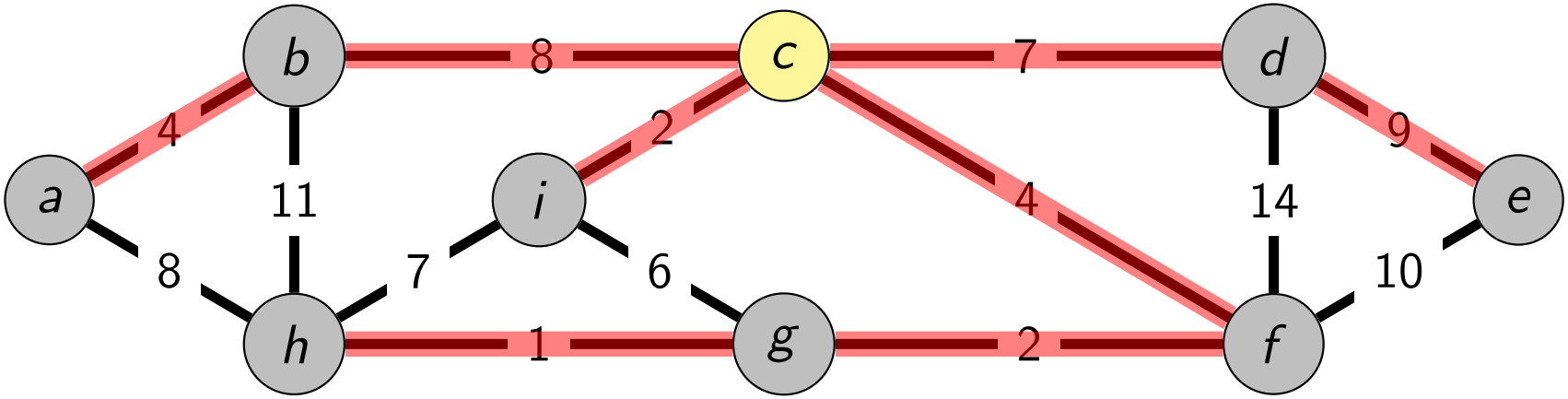
Příklad



Algoritmus č. 2 (Prim-Jarník)

Minimální kostru budujeme ze semínka (pevně zvolený vrchol) a přidáváme vždy nejlevnější hranu, která z této komponenty vede.

Příklad



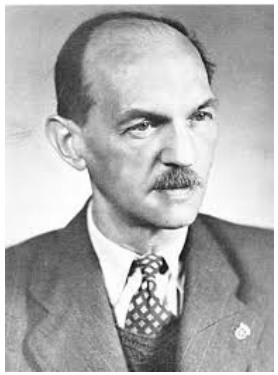
Nalezli jsme minimální kostru T s vahou
$w(T) = 2 + 4 + 2 + 1 + 7 + 8 + 4 + 9 = 37$.

Osoby a obsazení[a]

[a] Copyright Disclaimer under Section 107 of the Copyright Act of 1976

Otakar Borůvka (10.5.1899–22.7.1995)

Joseph Bernard Kruskal, Jr (29.1.1928–19.9.2010)



Vojtěch Jarník (22.12.1897–22.9.1970)

Robert Clay Prim III (25.9.1921–18.11.2021)

Práce, ve kterých se algoritmy objevily

1. Otakar Borůvka, O jistém problému minimálním, Práce moravské přírodovědecké společnosti 3.3 (1926), 37–58.
2. Vojtěch Jarník, O jistém problému minimálním (Z dopisu panu O. Borůvkovi), *Práce moravské přírodovědecké společnosti* 6.4 (1930), 57–63.
3. Joseph Bernard Kruskal, On the shortest spanning subtree of a graph and the traveling salesman problem, *Proc. Amer. Math. Soc.* 7 (1956), 48–50.
4. Robert Clay Prim, Shortest connection networks and some generalizations, *The Bell System Technical Journal* 36.6 (1957), 1389–1401.

Malý slovníček pojmů

česky	anglicky
neorientovaný graf	undirected graph
vrchol, hrana	vertex, edge
cyklus	cycle
řez na vrcholech	vertex cut
souvislý graf[a]	connected graph
strom	tree
ohodnocený graf	weighted graph
(minimální) kostra	(minimum) spanning tree
les	forest
bezpečná hrana	safe edge
lehká hrana	light edge
semínko	seed

[a] Z mně neznámých příčin leckdo používá termín „spojitý graf“. Termín „spojitý graf“ je v teorii grafů NESMYSL!