

Pr: Ypocítějte složitost algoritmu, $N \in \mathbb{N}$, $N = 3^k$ pro $k \in \mathbb{N}$. (k nesmíme)

$i := N$ ($= n$)

Algoritmus skončí, protože i se pořád směruje.

while $i > 0$:

$j := 1$

$j := j/3$

while $j < i$:

$j++$

return

$$g(n) = \sum_{i=1}^{\log_3 N} \frac{n}{3^k} = n \cdot \sum_{i=1}^{\log_3 N} \frac{1}{3^k} \leq n \cdot \sum_{i=1}^{\infty} \frac{1}{3^k} = n \cdot \left[\frac{\frac{1}{3}}{1 - \frac{1}{3}} \right] = n \cdot \frac{1}{2} = \frac{n}{2}, g(n) \in \Theta(n)$$

Vnější cyklus proběhne pro $i = n, n/3, n/9, \dots, n/3^k$, kde $k = \log_3 n$, tedy celkem $(k+1)$ -krát.

Vnitřní cyklus taky proběhne $(k+1)$ -krát, a pokudžé buď hrať a hrať dobu: $n, n/3, n/9, \dots, n/3^k = 1$

Pr: Ypocítějte složitost algoritmu, $N \in \mathbb{N}$

while $N > 1$:

if (N je liché):

For($i=0, i < N, i++$): Algoritmu nekončí, nemá smysl hledat složitost.

print('*')

$N := (N+1)/2$

else:

$N++$

return

Zkusíme pro $n = 4: 4, 5, 3, 2, 3$

Sčítání proběhne v konstantním čase, stejně jako násobení mocninou dvojkdy

Pr: Máme binární čísla a, b a chceme spočítat jejich součin. Ypocítějte složitost:

1) použijeme nový algoritmus: $a \cdot b = \sum_{i=1}^{\ell(a)} b$

$m := 0$

For($i=0, i < \ell(a), i++$):

$m += b$

return m

$\Theta(a) = \Theta(2^{\ell(a)})$ délka a binárně

2) použijeme rekurenci, libovolné 2N cíferné číslo napísáme jako $2^n A + B$, kde A, B jsou N-cíferné. Násobek dvou čísel pak je:

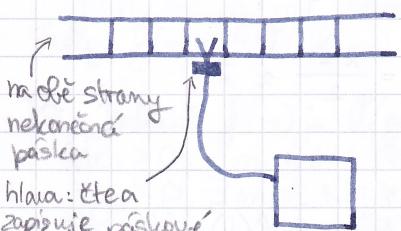
$$M \cdot N = (2^n A + B) \cdot (2^n C + D) = 2^{2n} A \cdot C + 2^n (AD + BC) + BD$$

$$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + 2, a = 4, b = 2, \log_b a = 2$$

Použijeme MT 1, protože $2 \in \Theta(n^{2-\varepsilon})$, $\varepsilon = 1$, proto $T(n) \in \Theta(n^2)$

Přednáška Turingovy stroje: Turingův stroj je sedmice $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$

19.3.2019



Symboly, které se dostran po práci

řídí jednotka: nadívá se u jakém stavu, se po určité jednotce, se posune na jednotku

Q: konečná množina stavek

Σ : vstupních symbolů

Γ : páskových symbolů, $\Sigma \subset \Gamma$, $\delta: Q \times \Gamma \times \{L, R\} \rightarrow Q \times \Gamma \times \{L, R\}$, kde L je posun levá a R je posun doprava

$q_0 \in Q$ je počáteční stav

B: prázdný symbol ("blank"), $B \in \Gamma \setminus \Sigma$

$F \subseteq Q$: množina koncových stavek

(olevo)

Turingův stroj lze zadat tabulkou nebo stavovým diagramem (vlevo)

δ	Σ	Γ
v jakém stavu je aktuální jednotka q_i	Σ_x	Γ_x
		Γ_x
		Γ_x

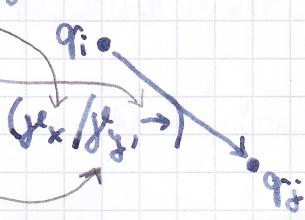
← jazyk symbolu čtu na páse

kam se po páse posun

co zapisu na páse

do jakého stavu se posune aktuální jednotka

$\delta(q_i, \Sigma_x, \Gamma_x) = (q_j, \Gamma_x, \Gamma_x)$



Príklad: Máme bezkontextový jazyk $L = \{0^n 1^n\}^{\text{nz}}$. Chceme mít Turingův stroj, který pro $w \in L$ akonečnou koncovou stavu, a pro $w \notin L$ akonečnou jinde.

Možné představy:

- 1) přejdu slovem, dokud nepřejdu "01". V tom místě se rozdělím, obě hodnoty přepíšu a pak budu ještě se dvěma na stranu a masat slovo od středu.
- 2) na začátku slova ("BO") smažu nulu, přejdu na druhý konec, smažu jedničku a obdobně jako v 1) marnu slovo od středu do středu.
- 3) na začátku slova přepíšu 0 na X, přejdu slovo k první jedničce a tu změním na Y. Pak se vrátím spět až X a přepisuji znova. Slovo teď správne po částech sleva do prava.

Def: Situace Turingova stroje (konfigurace, instantaneous description - ID) plně popisuje obsah pásky, pozici hlavy na pásku, a slav řídící jednotky. Ještě jeou na pásku n k polich symboly $X_1 X_2 X_3 \dots X_k$ a vlevo i doprava i npravo obsahy: Blank, řídící jednotka je ve stavu q a hlava čte symbol X_i , tak je daná situace:

$$X_1 X_2 X_3 \dots X_{i-1} \underbrace{q X_i}_{\substack{\text{Turing machine} \\ \downarrow}} X_{i+1} \dots X_k$$

Stav se píše sleva od čteného symbolu

Def: Počáteční situace TM je $q_0 a_1 a_2 \dots a_n$, kde q_0 je počáteční stav a vstupní slovo je $w \in \Sigma^*$, $w = a_1 a_2 \dots a_n$. Všude jinde jeou blanky, ale ty se nepíšou

Def: Krok Turingova stroje - předpokládejme, že TM je v situaci $X_1 \dots X_{i-1} q X_i \dots X_k$. V jednom kroku přejde TM do následující situace na základě fáchedové funkce S

- posun hlavy doprava: $S(q, X_i) = (p, Y, R)$

$$X_1 \dots X_{i-1} q X_i \dots X_k \vdash X_1 \dots X_{i-1} Y p X_{i+1} \dots X_k$$

pro i=1: $X_1 \dots X_{k-1} q X_k \vdash X_1 \dots X_{k-1} Y p B$ ← hlava se dostala mimo smaky na pásku

- posun hlavy dolera: $S(q, X_i) = (p, Y, L)$

$$X_1 \dots X_{i-1} q X_i \dots X_k \vdash X_1 \dots X_{i-2} p X_{i-1} Y \dots X_k$$

pro i=1: $q X_1 \dots X_k \vdash p B Y X_2 \dots X_k$

Reflexivní uzávěr: \vdash^* je transitiivní a reflexivní uzávěr relace \vdash . \approx provedení několika kroků \vdash , jejichž postupnost je "vypojet"

Jestliže $S(q, X_i)$ není definováno, TM se rozstaní.

Definice: Slovo $w \in \Sigma^*$ je přijímané TM, jestliže $q_0 \overline{a_1 \dots a_n} \vdash^* \alpha p B$ pro nějaký koncový stav $p \in F$ a počítavá slova $\alpha, B \in \Gamma^*$. Jazyk přijímaný TM je množina slov $L(M) = \{w \in \Sigma^* \mid w \text{ je přijímané } M\}$

- pro $w \notin L$ se TM nerastaví, nebo se eastaví neúspěšně.

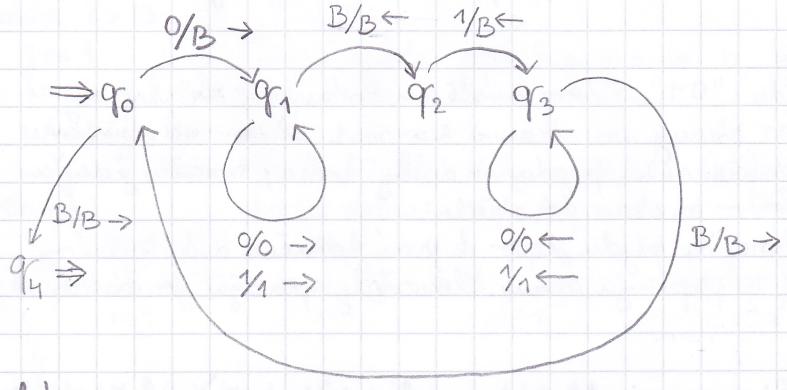
Na rozdíl od automatu nemusíme vstupní slovo přečíst cele. Jakmile slovo dosáhne nějakého koncového stavu, slovo je přijato. Proto s koncových stavů nejde jít nikam dál, nemá to totiž smysl.

Jakmile TM dosáhne koncového stavu, uspěšně se rastaví. Neúspěšně se rastaví, jestliže $S(q, X)$ není definováno (nejde jít dál). Formálně:

uspěšně se rastaví iff $q w \vdash^* \alpha p B$ pro $p \in F$. (TM přijíma w)

neúspěšně se rastaví iff $q w \vdash^* \alpha p X B$, a $S(p, X)$ není definováno, $p \notin F$.

Příklad: Vystrojme TM pro jazyk $L = \{0^n 1^n\}$. Podle přístupu ②.



$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

$$Q = \{q_0, q_1, q_2, q_3, q_4\}$$

$$\Sigma = \{0, 1\}$$

$$\Gamma = \{0, 1, B\}$$

δ je dáná diagramem

$$F = \{q_3\}$$

q_4 : dokonce se

dostanu, když
jsem na levém kraji
a už tam nic nemám

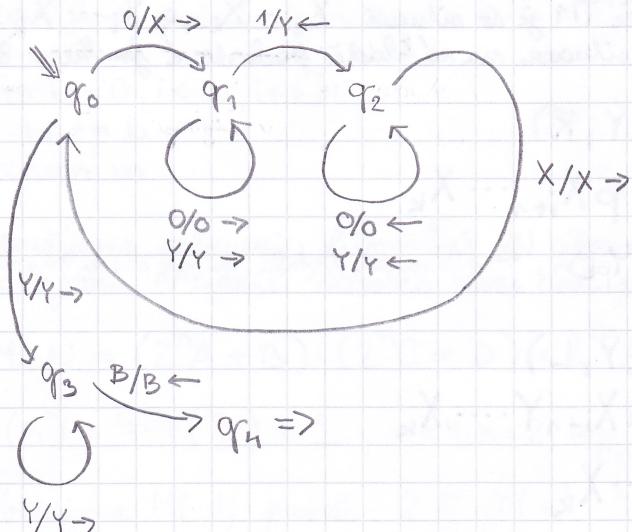
q_1 : čtu celé slovo,
jednou narázem na
konci, mohu klauz na
poslední znak

q_3 : stejně jako q_1 ,
ale jednu druhým
směrem

q_0 : jsem na levém
kraji slova, smažu
nulu a jednu doprava

q_2 : jsem na pravém
kraji, smažu 1 a
jedu doleva

Vystrojme TM pro stejný jazyk přístupem ③



Pokud $w \in 0^k 1^l$ a $k > l$, tak ne
stenu q_1 přechu B (uzala jsem se slova),
a neúspěšně sestavím.

Pokud $l > k$, tak se dostanu do q_3 dílu,
než budou píspomé všechny jednotky. V q_3
přechu 1 a neúspěšně sestavím.

Pokud je slovo kombinace 0 a 1, tak ho
stejně skončí chybou. Slovo začínající jedinou
skončí v n v q_0 a slovo začínající na nulu
píspomé jednotky a pak selže v q_3 .

Def: Je dán rozbarení $f: \Sigma^* \rightarrow \Sigma^*$. Rechneme, že M realizuje rozbarení f, jestliže
pro každé $w \in \Sigma^*$, pro které je $f(w)$ definováno, se M úspěšně sestaví s myšlenkou
 $f(w) = \alpha B$, $q_0 w \xrightarrow{*} \alpha q_F B$. Pro w, pro které $f(w)$ definováno není, se M
sestaví neúspěšně. ≈ stoj přijme w a na páse nustane mezi blanky správný výsledek f(w).

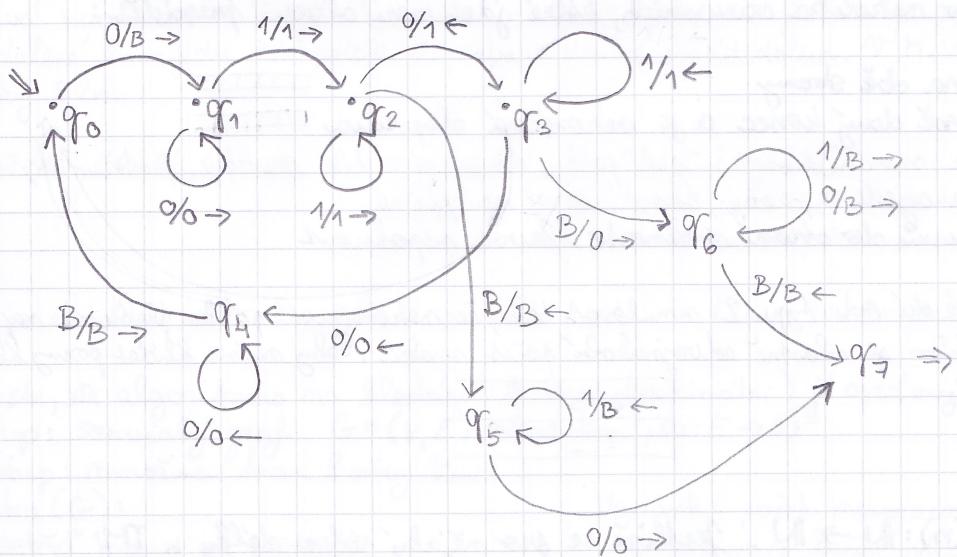
Příklad: $f: \mathbb{N}^2 \rightarrow \mathbb{N}$; $f(m, n) = \max(m, n)$.

Vstup (m, n) zakódujeme jako $0^m 1 0^n$, a budeme chtít výsledek $0^{\max(m, n)}$

Obecný návrh: Budu postupovat obdobně jako v přístupu 3. Která bude 0^m nahrazovat
blanky, dejdu za jedničku a nahradím první nulu za 1 (může i něco jiného, ale musí být
něčím symboly). Následně se na začátek slova a opakuji.

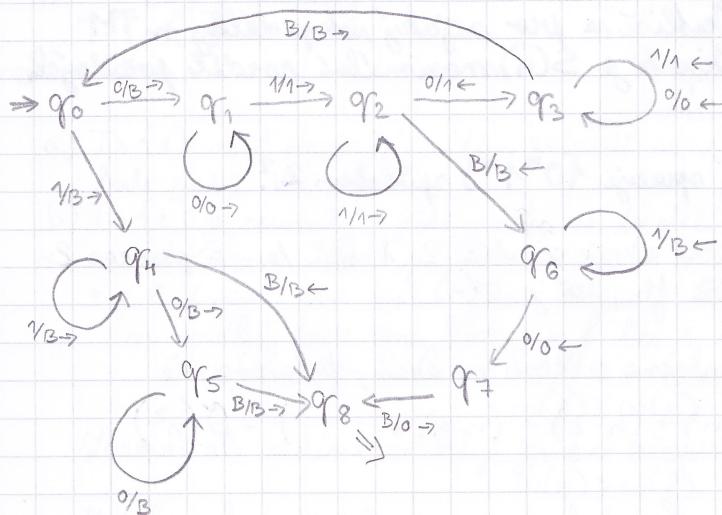
Pokud jednu doprava a hned za 1 je B, smaňu ho, že celé 0ⁿ je odcizené. Odjedu doleva
a nahradím 1 za B.

Pokud jednu doleva a hned za 1 je B, smaňu ho, že $n \geq m$. Zapsu jednu nulu a následně
píšu nahradím B.



< Aakhle jsem to nacvchla ja, mne nezarmicuju...

TM pro stejný jazyk podle tabule, jen když měl dělat $f(m, n) = \max(0, m-n)$



Rozdíl je v tom, že p smazím můly a pravé číslo se obrazí a přechádky jakkoliv vrátí doleva a tam zjistí, že kde číslo 0 (smaz) nebo 1 (skončí).

Můj stroj hledá 0 až při cestě zpět a proto se do q_0 dostane, jen pokud již nejaké 0 nalyavlji.

Def: Jazyk je rozhozený TM, jestliže pro $w \in L$ se TM úspěšně zastaví, a pro $w \notin L$ se zastaví nesplňován. U přijímaného jazyku stačí když se pro $w \notin L$ nezastaví.

Př: Informace musí být uložena ve stavu: $L = \{0^n 1^n, n \geq 1\} \cup \{1^n 0^n, n \geq 1\}$

U takového jazyka by nám stačil jeden stroj, který by si pouze pamatoval, jestli spracovává slovo z první nebo z druhé množiny. Pro sjednocení můžeme uložit informaci v návodu stavu, informace ale musí být konečná.

Př: Informace musí být uložena ve složce na pásku

B	a	a	b	B	B
B	1	1	0	B	B

Páskový symbol je uspořádaná dvojice (k -tice). Můžeme si tím říct, že když pamatovat, jestli v k hlava znak spracovala.

Blank	(B, B)
přečtené a	(a, 1)
— — b	(b, 1)
nepřečtené a	(a, 0)
— — b	(b, 0)

To jde ale snadno přenést na formálnější a méně všeobecný zápis s jedním slopou

B
a
b
c
d

TM se může neplatit v několika variantách, které jsou mezi sebou převádět:

1) pásek je nekonečný na obě strany

2) pásek má vlevo první daný konec a je nekonečná doprava

v tomto případě máme množství číselní polí pásky, ale stroj se hání navrhý, protože když uděláme cyklus a hlava vjede s pásky, skončí stroj neúspěšně

3) stroj má kromě posunu do ohně možnost hlavu nepohnout

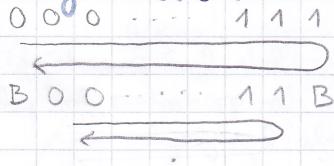
převod $1 \Leftrightarrow 2$: Typ 1 se dá na typu 2 simulovat tak, že nekonečnou pásku přehneme například se vlastní selvojnásobí počet snadk. Taky musíme hledat posun hlavy.

-3 -2 -1 1 2 3	=>	1 2 3 -1 -2 -3
----------------	----	-------------------

Časová složitost TM: $T(n): N \rightarrow N$. Ještě se pro nějaký vstup délky n TM neplatí, pak $T(n)$ není definováno. Jinak je $T(n)$ maximální počet kroků, po nichž dojde k sestavení TM. Maximum se počítá přes všechny vstupy délky n.

Paměťová složitost TM: $S(n): N \rightarrow N$. Ještě se pro nějaký vstup délky n TM neplatí, pak $S(n)$ není definováno. Jinak je $S(n)$ maximální rozměr pořadových čísel polí, které lze k výpočtu použít.

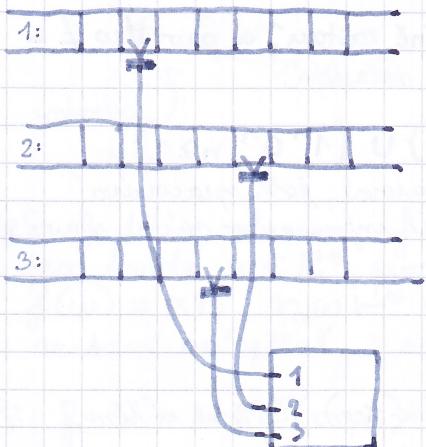
Př: Jaká je časová složitost stroje, který spracuje $\{0^n 1^n\}$ různobarem 2?



pro určení čísel z kraje se vzdálá 2n kroků tam a pak ještě 2n zpátky, takže $4n$, což je $O(n)$

Procházení opakuje celkem n-krať, dostaneme, že
 $T(n) \in O((n) + (n-2) + (n-4) + \dots) = O(n^2)$

Turingův stroj s více páskami



- má k pásek a k hlavám, které se hybnou nezávisle

- Améní se přechodová fce:

$$\delta: Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$$

např. pro $k=3$, $l_i \in \{L, R\}$

$$\delta(q_i, A_i, B_j, C_m) = (p, X, Y, Z, l_1, l_2, l_3)$$

- počáteční situace: první páška obsahuje vstupní slovo, ostatní pásky obsahují B.

- všechny hlavy začínají na stejném indexu (pod sebou)

- v každém kroku se pohnut všechny hlavy

Věta: k-páskový TM lze simulovat na 1-páskovém TM a to tak, že n kroků na k-páskách bude trvat nejdéle n^2 kroků.

Důkaz: Vytvořím jednopáskový stroj M_1 , který bude mít na páse 2k stupňů, a každou pásky ledy vzdílenou dve stupny. První stupň obsahuje znaky B a *, a hvězdička je vzdále tam, kde je hlava v M_k . Druhá stupň obsahuje znaky s pásky.

Hlava na M_1 reaguje na * najít vlevo. Přejde doprava páškou až k poslední *, po cestě přečte k hodnot a vloží krok M_k . Pak jede hlava doprava a soupe hvězdičky podle počtu hlav v M_k , a tuto přepisuje hodnoty stupňů vpravo v M_k . Zastaví se na * najít vlevo.

Casová náročnost: Když M_k udělá n kroků, tak v nejhorším případě bude mít hlavy od sebe oddálené $2n$ - jedna žela pojde doprava a druhá pojde doleva. V M_1 bude oddálenost dvou žel $\leq 2n$.

Pro odávání krovků pak M_1 udělá:

$2n$ kroků doprava

$2n$ kroků zase zpět (vlastně $2n+2$)

$2k$ kroků navíc (pokud se všechny žely posouvaly doprava)

Tuto simulaci udělá M_1 celkem n -krát, tedy $n \cdot (2n + 2n+2 + 2k) \approx O(n^2)$

Dokážte, že algoritmus na hledání kostky (minimální) je správný.

vstup: souvislý graf $G = (V, E)$, ohodnocení $w: E \rightarrow \mathbb{N}$

výstup: možnost hran kostky K_{\min}

Kostra (G):

setřídí hranu od nejobtížnéjší

$T: E$

for e in E :

if $T - \{e\}$ je souvislý

$T := T - \{e\}$

return T

Cvičení
20.3.2019

Variant: počet nespracovaných hran v E , snižuje se v každé iteraci

Invariant: $\exists K_{\min}, K_{\min} \subseteq T$

1) na začátku to určitě platí, protože G je souvislý a $T = E$

2) spracováváme nejobtížnou $e = (x, y)$, $K_{\min} \subseteq T$, máme dvě možnosti:

a) $T - \{e\}$ nemá souvislost, v tom případě se e ponechá v T , a protože se T nemění, bude mít $K_{\min} \subseteq T$.

b) $T - \{e\}$ je souvislý, jde o jednu ze dvou variant

- $e \notin K_{\min}$, v tom případě můžeme e v blízkosti odstranit

- $e \in K_{\min}$, tam máme jinou hranu f , která propojuje x a y v G . Protože K_{\min} je minimální, bude cena kostky s f bez e vysoká než cena K_{\min} :

$$w(K_{\min}) - w(e) + w(f) \geq w(K_{\min})$$

$$w(f) \geq w(e)$$

Zároveň ale můžeme, že když jsme v souvislostech hranách přešli až k f , mít $w(e) \geq w(f)$.

Z toho plyne, že obě cesty jsou shodné, hranu e můžeme odstranit a budeme mít jinou K_{\min} , ale se stejnou cenou.

3) Na konci T je souvislý, a mívá všechny hranach nízme, ne $T - \{e\}$ nemá souvislost. Proto je T kostra, a minimální.

$T := \{\}$

for e in E :

if $T \cup \{e\}$ nemá kružnice:

$T := T \cup \{e\}$

return T

Najde kostru, ale ne minimální
(nějaké hraně byly)

$T := \{\}$

for e in E :

if T obsahuje kružnice:

$T := T - \{f\}$ f je nejobtížná

return T

hranu na kružnici

Variant: počet nespracovaných hran v e

Invariant: Po spracování hran $L = \{e_1, \dots, e_k\}$ je (V, L) minimální les (několik stromů).

1) na začátku to platí, $L = \{\}$ a máme oddělené vrcholy

2) máme $L = \{e_1, \dots, e_k\}, e_{k+1}$, nastanou 2 situace:

a) e_{k+1} měřavice kružnice, tam e_{k+1} může propojit dva stromy



b) e_{k+1} měřavice kružnice, ale v tom případě odebereme nějakou hranu $w(f) \geq w(e_{k+1})$, a tím se sníží cena stromu



3) na konci máme minimální les, který je souvislý, tedy strom, a tím se říká min kostra.