# 1. Prednaska - Zlozitosti

**1.2.1 Symbol $\mathcal{O}$.** Je dána nezáporná funkce $g(n)$. Řekneme, že nezáporná funkce $f(n)$ *je* $\mathcal{O}(g(n))$, jestliže existuje kladná konstanta $c$ a přirozené číslo $n_0$ tak, že

$$f(n) \leq c\,g(n) \quad \text{pro všechny } n \geq n_0.$$

□

$\mathcal{O}(g(n))$ můžeme též chápat jako třídu všech nezáporných funkcí $f(n)$:

$$\mathcal{O}(g(n)) = \{f(n) \mid \exists c > 0, n_0 \in \mathbb{N} \text{ tak, že } f(n) \leq c\,g(n) \ \ \forall n \geq n_0\}.$$

**1.2.2 Symbol $\Omega$.** Je dána nezáporná funkce $g(n)$. Řekneme, že nezáporná funkce $f(n)$ *je* $\Omega(g(n))$, jestliže existuje kladná konstanta $c$ a přirozené číslo $n_0$ tak, že

$$f(n) \geq c\,g(n) \quad \text{pro všechny } n \geq n_0.$$

□

$\Omega(g(n))$ můžeme též chápat jako třídu všech nezáporných funkcí $f(n)$:

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, n_0 \in \mathbb{N} \text{ tak, že } f(n) \geq c\,g(n) \ \ \forall n \geq n_0\}.$$

**1.2.4 Symbol $\Theta$.** Je dána nezáporná funkce $g(n)$. Řekneme, že nezáporná funkce $f(n)$ *je* $\Theta(g(n))$, jestliže existují kladné konstanty $c_1$, $c_2$ a přirozené číslo $n_0$ tak, že

$$c_1\,g(n) \leq f(n) \leq c_2\,g(n) \quad \text{pro všechny } n \geq n_0.$$

□

$\Theta(g(n))$ můžeme též chápat jako třídu všech nezáporných funkcí $f(n)$:

$$\Theta(g(n)) = \{f(n) \mid \exists c_1, c_2 > 0, n_0 \in \mathbb{N} \text{ tak, že } c_1\,g(n) \leq f(n) \leq c_2\,g(n) \ \ \forall n \geq n_0\}.$$

**1.2.6 Symbol malé $o$.** Je dána nezáporná funkce $g(n)$. Řekneme, že nezáporná funkce $f(n)$ *je* $o(g(n))$, jestliže pro každou kladnou konstantu $c$ existuje přirozené číslo $n_0$ tak, že

$$0 \leq f(n) < c\,g(n) \quad \text{pro všechny } n \geq n_0.$$

□

$o(g(n))$ můžeme též chápat jako třídu všech nezáporných funkcí $f(n)$:

$$o(g(n)) = \{f(n) \mid \forall c > 0\ \exists n_0 \in \mathbb{N} \text{ tak, že } 0 \leq f(n) < c\,g(n) \ \ \forall n > n_0\}.$$

**1.2.7 Poznámka.** Fakt, že nezáporná funkce $f(n)$ je $\mathcal{O}(g(n))$, zhruba řečeno znamená, že funkce $f(n)$ neroste asymptoticky více než funkce $g(n)$. Naproti tomu fakt, že nezáporná funkce $f(n)$ je $o(g(n))$, znamená, že funkce $f(n)$ roste asymptoticky méně než funkce $g(n)$.

**1.2.8 Symbol malé $\omega$.** Je dána nezáporná funkce $g(n)$. Řekneme, že nezáporná funkce $f(n)$ *je* $\omega(g(n))$, jestliže pro každou kladnou konstantu $c$ existuje přirozené číslo $n_0$ tak, že

$$0 \leq c\,g(n) < f(n) \quad \text{pro všechny } n \geq n_0.$$

□

$\omega(g(n))$ můžeme též chápat jako třídu všech nezáporných funkcí $f(n)$:

$$\omega(g(n)) = \{f(n) \mid \forall c > 0\ \exists n_0 \in \mathbb{N} \text{ tak, že } 0 \leq c\,g(n) < f(n) \ \ \forall n > n_0\}.$$

**1.2.9 Poznámka.** Fakt, že nezáporná funkce $f(n)$ je $\Omega(g(n))$, zhruba řečeno znamená, že funkce $f(n)$ roste asymptoticky alespoň tak, jako funkce $g(n)$. Naproti tomu fakt, že nezáporná funkce $f(n)$ je $\omega(g(n))$, znamená, že funkce $f(n)$ roste asymptoticky více než funkce $g(n)$.

**1.2.11 Tvrzení.** Jsou dány dvě nezáporné funkce $f(n)$ a $g(n)$. Pak platí

1. $f(n) \in o(g(n))$ právě tehdy, když $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$;
2. $f(n) \in \omega(g(n))$ právě tehdy, když $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$.
3. Jestliže $\lim_{n\to\infty} \frac{f(n)}{g(n)} = a$ pro některé $a \in \mathbb{R}$, $a \neq 0$, pak $f(n) \in \Theta(g(n))$

**1.2.16 Věta (Gauss).** Pro každé $n \geq 1$ platí

$$n^{\frac{n}{2}} \leq n! \leq \left(\frac{n+1}{2}\right)^n.$$

□

Zdůvodněni: Využijeme fakt, že pro každá dvě kladná čísla $a, b$ platí $\frac{a+b}{2} \geq \sqrt{ab}$.

Přepíšeme $(n!)^2$ takto

$$(n!)^2 = n\,(n-1)\,\ldots 2\,1\,1\,2\,\ldots\,(n-1)\,n = \prod_{i=1}^{n}(n-i+1)i.$$

Odtud

$$n! = \prod_{i=1}^{n}\sqrt{(n-i+1)i} \leq \prod_{i=1}^{n}\frac{n+1}{2} = \left(\frac{n+1}{2}\right)^n,$$

protože pro každé $i$ platí $\sqrt{(n-i+1)i} \leq \frac{n-i+1+i}{2}$. Tím jsme dostali horní odhad.

Na druhé straně pro každé $i$ platí $n \leq (n-i+1)i$ a proto je $n^n \leq (n!)^2$. Odmocněním dostaneme dolní odhad, totiž $n^{\frac{n}{2}} \leq n!$.

**1.3.1 Věta — „Master Theorem“.** Jsou dána přirozená čísla $a \geq 1$, $b > 1$ a nezáporná funkce $f(n)$. Předpokládejme, že funkce $T(n)$ je dána na přirozených číslech rekurentním vztahem

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n),$$

kde $\frac{n}{b}$ znamená buď $\lfloor \frac{n}{b} \rfloor$ nebo $\lceil \frac{n}{b} \rceil$.

1. Jestliže $f(n) \in \mathcal{O}(n^{\log_b a - \varepsilon})$ pro nějakou konstantu $\varepsilon > 0$, pak $T(n) \in \Theta(n^{\log_b a})$.
2. Jestliže $f(n) \in \Theta(n^{\log_b a})$, pak $T(n) \in \Theta(n^{\log_b a} \lg n)$.
3. Jestliže $f(n) \in \Omega(n^{\log_b a + \epsilon})$ pro nějakou konstantu $\varepsilon > 0$ a jestliže $a\,f(\frac{n}{b}) \leq c\,f(n)$ pro nějakou konstantu $c < 1$ pro všechna dostatečně velká $n$, pak $T(n) \in \Theta(f(n))$.

**1.3.3 Tvrzení.** Jestliže $f(n) \in \Theta(n^{\log_b a} \lg^k n)$ pro $k \geq 0$, pak pro funkci $T(n)$ danou rovnicí

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n),$$

platí: $T(n) \in \Theta(n^{\log_b a} \lg^{k+1} n)$. □

# 2. Prednaska - Invariant / Variant

**1.3.10 Amortizovaná složitost.** Jedná se o výpočet průměrné složitosti nejhoršího případu pro posloupnost $n$ opakování dané instrukce. Jestliže $n$ opakování v nejhorším případě vyžaduje čas $\mathcal{O}(T(n))$, pak jedno provedení vyžaduje čas $\mathcal{O}(T(n))/n$, a to je amortizovaná složitost jedné instrukce.

Jsou tři základní způsoby, jak amortizovanou složitost zjišťovat.

- První je tzv. *agregační* — postupuje se přímo podle předchozího odstavce.
- Druhá metoda je tzv. *účetní*. Každé provedení instrukce má jistý kredit. Jestliže provedení instrukce nespotřebuje celý kredit, zbývající část kreditu je možno využít v dalších provedení instrukce, které jsou náročnější a na které by jejich kredit nestačil. Podmínkou ale je, aby žádná instrukce v posloupnosti nespotřebovala víc než je součet jejího kreditu a zatím nevyužitých částí kreditů.

**1.4.7 Variant.** Pro důkaz faktu, že se algoritmus na každém vstupu zastaví, je založen na nalezení tzv. *variantu*. Variant je hodnota udaná přirozeným číslem, která se během práce algoritmu snižuje až nabude nejmenší možnou hodnotu (a tím zaručuje ukončení algoritmu po konečně mnoha krocích).

V příkladu 1.4.2 se jednalo o číslo $k$, v příkladu 1.3.7 se jednalo o zbytek $z$ při dělení čísla $r$ číslem $t$.

**1.4.8 Invariant.** *Invariant*, též *podmíněná správnost algoritmu*, je tvrzení, které

- platí před vykonáním prvního cyklu algoritmu, nebo po prvním vykonání cyklu,
- platí-li před vykonáním cyklu, platí i po jeho vykonání,
- při ukončení práce algoritmu zaručuje správnost řešení.

Pro algoritmus pro bublinkové třídění je invariantem tvrzení 1.4.4, pro Euikleidův algoritmus tvrzení 1.4.6.

# 4. Prednaska - Turingov Stroj

**1.5.1 Turingův stroj** si můžeme představit takto: skládá se

- z řídící jednotky, která se může nacházet v jednom z konečně mnoha stavů,
- potenciálně nekonečné pásky (nekonečné na obě strany) rozdělené na jednotlivá pole a
- hlavy, která umožňuje číst obsah polí a přepisovat obsah polí pásky.

Na základě symbolu $X$, který čte hlava na pásce, a na základě stavu $q$, ve kterém se nachází řídící jednotka, se řídící jednotka Turingova stroje přesune do stavu $p$, hlava přepíše obsah čteného pole na $Y$ a přesune se buď doprava nebo doleva (tato akce je popsána tzv. přechodovou funkcí).

**1.5.2 Formální definice.** Turingův stroj je sedmice $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$, kde

- $Q$ je konečná množina stavů,
- $\Sigma$ je konečná množina vstupních symbolů,
- $\Gamma$ je konečná množina páskových symbolů, přitom $\Sigma \subset \Gamma$,
- $B$ je prázdný symbol (též nazývaný *blank*), jedná se o páskový symbol, který není vstupním symbolem, (tj. $B \in \Gamma \setminus \Sigma$),
- $\delta$ je přechodová funkce, tj. parciální zobrazení z množiny $(Q \setminus F) \times \Gamma$ do množiny $Q \times \Gamma \times \{L, R\}$, (zde $L$ znamená pohyb hlavy o jedno pole doleva, $R$ znamená pohyb hlavy o jedno pole doprava),
- $q_0 \in Q$ je počáteční stav a
- $F \subseteq Q$ je množina koncových stavů.

**1.5.3 Situace TM.** *Situace Turingova stroje* (též konfigurace TM, angličtině nazývaná instantaneous description (ID)), plně popisuje obsah pásky, pozice hlavy na pásce a stav, ve kterém se nachází řídící jednotka. Jestliže na pásce jsou v $k$ polích symboly $X_1 X_2 \dots X_k$, všechna pole s větším i menším číslem již obsahují pouze $B$, řídící jednotka je ve stavu $q$ a hlava čte symbol $X_i$, tak danou situaci zapisujeme

$$X_1 X_2 \dots X_{i-1} q X_i X_{i+1} \dots X_k.$$

**1.5.4 Počáteční situace.** Na začátku práce se Turingův stroj nachází v počátečním stavu $q_0$, na pásce má na $n$ polích vstupní slovo $a_1 a_2 \dots a_n$ ($a_i \in \Sigma$), ostatní pole obsahují blank $B$ a hlava čte pole pásky se symbolem $a_1$. Tedy formálně počáteční situaci zapisujeme $q_0 a_1 \dots a_n$.

**1.5.5 Krok Turingova stroje.** Předpokládejme, že se Turingův stroj nachází v situaci $X_1 X_2 \dots$ Pak na základě přechodové funkce TM v jednom kroku přejde do následující situace a to takto:

Jestliže $\delta(q, X_i) = (p, Y, R)$, TM se přesune do stavu $p$, na pásku místo symbolu $X_i$ napíše symbol $Y$ a hlavu posune o jedno pole doprava. Formálně to zapisujeme takto

$$X_1 X_2 \dots X_{i-1} q X_i \dots X_k \vdash X_1 X_2 \dots X_{i-1} Y p X_{i+1} \dots X_k. \tag{1.1}$$

Jestliže $\delta(q, X_i) = (p, Y, L)$, TM se přesune do stavu $p$, na pásku místo symbolu $X_i$ napíše symbol $Y$ a hlavu posune o jedno pole doleva. Formálně to zapisujeme takto

$$X_1 X_2 \dots X_{i-1} q X_i \dots X_k \vdash X_1 \dots X_{i-2} p X_{i-1} Y X_{i+1} \dots X_k. \tag{1.2}$$

Jestliže v případě 1.2 je $i = 1$, pak $q X_1 \dots X_k \vdash p B Y \dots X_k$.

Jestliže $\delta(q, X_i)$ není definováno, TM se zastaví.

**1.5.6 Výpočet Turingova stroje** nad slovem $w = a_1 a_2 \dots a_k$, je posloupnost jeho kroků, která začíná v počáteční situaci $q_0 a_1 \dots a_k$. Formálně se jedná o reflexivní a tranzitivní uzávěr $\vdash^*$ relace $\vdash$ z 1.5.5 (na množině všech situací daného Turingova stroje).

Jestliže během výpočtu Turingova stroje nad slovem $w$ se Turingův stroj dostane do jednoho z koncových stavů $q' \in F$, říkáme, že se TM *úspěšně zastavil*. Obsah pásky při úspěšném zastavení je *výstupem* TM, nad vstupem $w = a_1 a_2 \dots a_n$.

**1.5.7 Definice — jazyk přijímaný TM.** Vstupní slovo $w \in \Sigma^*$ je *přijato* Turingovým strojem $M$, jestliže se Turingův stroj na slově $w$ úspěšně zastaví.

Množina slov $w \in \Sigma^*$, která Turingův stroj přijímá, se nazývá *jazyk přijímaný* $M$ a značíme ji $L(M)$.

**1.5.8 Definice — funkce realizovaná TM.** Je dáno zobrazení $f: \Sigma^* \to \Sigma^*$. Řekneme, že TM $M$ *realizuje* zobrazení $f$, jestliže pro každé $w \in \Sigma^*$, pro které je $f(w)$ definováno, se $M$ úspěšně zastaví s výstupem $f(w)$ (tj. $q_0 w \vdash^* \alpha q_F \beta$, kde $\alpha\beta = f(w)$). Pro $w$, pro něž $f(w)$ není definováno, se $M$ zastaví neúspěšně.

# 5. Prednaska - Turingov Stroj, Nedeterministicky

**1.5.9 Časová složitost Turingova stroje** je parciální zobrazení $T(n)$ z množiny všech přirozených čísel do sebe definované:

Jestliže pro nějaký vstup délky $n$ se Turingův stroj nezastaví, $T(n)$ není definováno. V opačném případě je $T(n)$ rovno maximálnímu počtu kroků, po nichž dojde k zastavení Turingova stroje, kde maximum se bere přes všechny vstupy délky $n$.

**1.5.10 Paměťová složitost Turingova stroje** $S(n)$. Jestliže pro nějaký vstup délky $n$ Turingův stroj použije nekonečnou část pásky (pak se nemůže v konečném čase zastavit), $S(n)$ není definováno. V opačném případě je $S(n)$ rovno největšímu rozdílu pořadových čísel polí, které byly během výpočtu použity, kde maximum se bere přes všechny vstupy délky $n$.

**1.5.12 Definice.** Řekneme, že jazyk $L$ je *přijímán* nějakým Turingovým strojem, jestliže existuje TM $M$ takový, že $L = L(M)$.

Řekneme, že Turingův stroj *rozhoduje* jazyk $L$, jestliže tento jazyk přijímá a navíc se na každém vstupu zastaví.

**1.5.13 Poznámky.**

- Každý jazyk, který je rozhodován Turingovým strojem, je také tímto Turingovým strojem přijímán. Naopak to ale neplatí. Uvidíme, že existují jazyky, které jsou přijímány nějakým Turingovým strojem, ale neexistuje Turingův stroj, který by je rozhodl.

**1.5.16 Turingův stroj s $k$ páskami.** Turingův stroj s $k$ páskami se skládá z řídící jednotky, která se nachází v jednom z konečně mnoha stavů $q \in Q$, množiny vstupních symbolů $\Sigma$, množiny páskových symbolů $\Gamma$, přechodové funkce $\delta$, počátečního stavu $q_0$, páskového symbolu $B$ a množiny koncových stavů $F$. Dále je dáno $k$ pásek a $k$ hlav; $i$-tá hlava vždy čte jedno pole $i$-té pásky. Přechodová funkce $\delta$ je parciální zobrazení, které reaguje na stav, ve kterém se Turingův stroj nachází a na $k$-tici páskových symbolů, kterou jednotlivé hlavy snímají. (Formálně je $\delta$ parciální zobrazení, $\delta: (Q \setminus F) \times \Gamma^k \to Q \times \Gamma^k \times \{L, R\}^k$).

Na začátku:

- Vstupní slovo je na první pásce; kromě vstupního slova obsahují všechna pole pásky blank $B$.
- Všechny ostatní pásky mají ve všech polích blank $B$.
- Řídící jednotka je v počátečním stavu $q_0$.
- První hlava čte první symbol vstupního slova.

**1.5.17 Krok** Turingova stroje s $k$ páskami je určen přechodovou funkcí. Jestliže přechodová funkce je definována, pak (na základě přechodové funkce):

- Řídící jednotka se přesune do nového stavu.
- Každá hlava přepíše obsah pole, které čte (může i stejným symbolem).
- Každá hlava se posune doprava nebo doleva (přechodová funkce udává pohyb každé hlavy nezávisle na pohybech ostatních hlav).

**1.5.18 Jazyk přijímaný Turingovým strojem s $k$ páskami.** Obdobně jako pro Turingův stroj s jednou páskou definujeme:

Turingův stroj se *úspěšně zastaví*, jestliže řídící jednotka vstoupila do koncového stavu. Jestliže Turingův stroj nemá definován následující krok a není v koncovém stavu, říkáme, že se Turingův stroj zastavil *neúspěšně*.

Slovo $w \in \Sigma^*$ je *přijímáno* Turingovým strojem, jestliže se na něm Turingův stroj úspěšně zastaví. Všechna slova přijímaná Turingovým strojem tvoří *jazyk přijímaný* tímto strojem.

Jestliže se navíc Turingův stroj na všech slovech zastaví, říkáme že Turingův stroj jazyk *rozhoduje*.

**1.5.21 Nedeterministický Turingův stroj.** Jestliže pro Turingův stroj (ať již s jednou páskou nebo s více páskami) připustíme, aby v jedné situaci mohl provést několik různých kroků, dostáváme nedeterministický Turingův stroj. Formálně zadefinujeme nedeterministický Turingův stroj (NTM) pouze pro variantu s jednou páskou, která je nekonečná na obě strany.

*Nedeterministický Turingův stroj* je sedmice $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$, kde

- $Q$ je konečná množina stavů,
- $\Sigma$ je konečná množina vstupních symbolů,
- $\Gamma$ je konečná množina páskových symbolů, přitom $\Sigma \subset \Gamma$,
- $B$ je prázdný symbol (též nazývaný *blank*), jedná se o páskový symbol, který není vstupním symbolem, (tj. $B \in \Gamma \setminus \Sigma$),
- $\delta$ je přechodová funkce, tj. parciální zobrazení z množiny $(Q \setminus F) \times \Gamma$ do množiny $\mathcal{P}_f(Q \times \Gamma \times \{L, R\})$ ($\mathcal{P}_f(X)$ je konečná podmnožina množiny $X$),
- $q_0 \in Q$ je počáteční stav a
- $F \subseteq Q$ je množina koncových stavů.

*Krok nedeterministického Turingova kroku* je definován analogicky jako pro (deterministický) Turingův stroj:
Pro $(p, Y, R) \in \delta(q, X_i)$

$$X_1 X_2 \ldots X_{i-1} q X_i \ldots X_k \vdash X_1 X_2 \ldots X_{i-1} Y p X_{i+1} \ldots X_k. \tag{1.3}$$

Pro $(p, Y, L) \in \delta(q, X_i)$

$$X_1 X_2 \ldots X_{i-1} q X_i \ldots X_k \vdash X_1 \ldots X_{i-2} p X_{i-1} Y X_{i+1} \ldots X_k. \tag{1.4}$$

**1.5.22 Jazyk přijímaný nedeterministickým Turingovým strojem** se skládá ze všech slov $w \in \Sigma^*$, pro něž

$$q_0 w \vdash^* Y_1 Y_2 \ldots Y_i q_f Y_{i+1} \ldots Z_m,$$

pro některý koncový stav $q_f$.

Neformálně: slovo $w$ je přijato nedeterministickým Turingovým strojem právě tehdy, když existuje „přijímací výpočet", tj posloupnost kroků, po nichž se stroj dostane do koncového stavu.

Jestliže nedeterministický Turingův stroj $M$ přijímá jazyk $L$ a navíc každý jeho výpočet vždy končí po konečně mnoha krocích, říkáme, že $M$ *rozhoduje* jazyk $L$.

**1.5.23 Věta.** Je-li jazyk $L$ přijímán, resp. rozhodován nedeterministickým Turingovým strojem $M$, pak existuje deterministický Turingův stroj $M_1$ s jednou páskou, který $L$ přijímá, resp. rozhoduje.

# 6. Prednaska - Rozhodovacie ulohy

**1.7.2 Příklad.** *SAT – splňování Booleovských formulí*: Je dána výroková formule $\varphi$ v CNF. Rozhodněte, zda je $\varphi$ splnitelná.

Na danou formuli $\varphi$ je tedy odpověď (tj. řešení) buď „ANO“ nebo „NE“. Všimněte si, že v tomto případě se neptáme po ohodnocení, ve kterém je formule pravdivá – zajímá nás pouze fakt, zda je splnitelná.

**1.7.4 Problém obchodního cestujícího – TSP.** Jsou dána města $1,2,\ldots,n$. Pro každou dvojici měst $i,j$ je navíc dáno kladné číslo $d(i,j)$ (tak zvaná vzdálenost měst $i,j$). *Trasa* je dána permutací $\pi$ množiny $\{1,2,\ldots,n\}$ do sebe. Délka trasy $T$ odpovídající permutaci $\pi$ je

$$d(T) = \sum_{i=1}^{n-1} d(\pi(i),\pi(i+1)) + d(\pi(n),\pi(1)).$$

Neformálně, trasa je pořadí měst, ve kterém má obchodní cestující města projít, a to tak, aby každé město navštívil přesně jednou a vrátil se do toho města, ze kterého vyšel. Cena trasy je pak součtem všech vzdáleností, které při své cestě urazil.

**1.7.5 Rozhodovací verze.**

- Minimální kostra: Je dán neorientovaný graf $G = (V,E)$, ohodnocení $c\colon E \to \mathbb{N}$ a dále číslo $K$. Existuje minimální kostra, jejíž cena je nejvýše $K$?
- Je dána matice délek $\mathbf{A} = (a(i,j))$, výchozí vrchol $r$, cílový vrchol $c$ a číslo $K$. Existuje cesta z vrcholu $r$ do vrcholu $c$ délky nejvýše $K$?
- Kromě čísel $d(i,j)$ z 1.7.4 je dáno číslo $K$. Existuje trasa $\pi$ délky nejvýše $K$?

**1.7.6 Vyhodnocovací verze.**

- Minimální kostra: Je dán neorientovaný graf $G = (V,E)$ a $c\colon E \to \mathbb{N}$. Najděte cenu minimální kostry ohodnoceného grafu.
- Je dána matice délek $\mathbf{A} = (a(i,j))$, výchozí vrchol $r$ a cílový vrchol $c$. Najděte délku nejkratší cesty z vrcholu $r$ do vrcholu $c$.
- Jsou dána čísla $d(i,j)$ a 1.7.4. Najděte cenu optimální trasy, tj. trasy s nejmenší možnou délkou.

**1.7.7 Optimalizační verze.**

- Minimální kostra: Je dán neorientovaný graf $G = (V,E)$ a $c\colon E \to \mathbb{N}$. Najděte minimální kostru ohodnoceného grafu.
- Je dána matice délek $\mathbf{A} = (a(i,j))$, výchozí vrchol $r$, cílový vrchol $c$. Najděte nejkratší cestu z vrcholu $r$ do vrcholu $c$.
- Jsou dána čísla $d(i,j)$ a 1.7.4. Najděte optimální trasu, tj. trasu s nejmenší možnou délkou.

# 7. Prednaska - P, NP, NPC a prevody

**1.8.3 Třída $\mathcal{P}$.** Řekneme, že rozhodovací úloha $\mathcal{U}$ leží ve třídě $\mathcal{P}$, jestliže existuje deterministický Turingův stroj, který rozhodne jazyk $L_{\mathcal{U}}$ a pracuje v polynomiálním čase; tj. funkce $T(n)$ je $\mathcal{O}(p(n))$ pro nějaký polynom $p(n)$.

**1.8.4 Příklady.**

- Minimální kostra v grafu. Je dán neorientovaný graf $G$ s ohodnocením hran $c$. Je dáno číslo $k$. Existuje kostra grafu ceny menší nebo rovno $k$?
- Nejkratší cesty v acyklickém grafu. Je dán acyklický graf s ohodnocením hran $a$. Jsou dány vrcholy $r$ a $c$. Je dáno číslo $k$. Existuje orientovaná cesta z vrcholu $r$ do vrcholu $c$ délky menší nebo rovno $k$?
- Toky v sítích. Je dána síť s horním omezením $c$, dolním omezením $l$, se zdrojem $z$ a spotřebičem $s$. Dále je dáno číslo $k$. Existuje přípustný tok od $z$ do $s$ velikosti alespoň $k$?
- Minimální řez. Je dána síť s horním omezením $c$, dolním omezením $l$. Dále je dáno číslo $k$. Existuje řez, který má kapacitu menší nebo rovnu $k$?

**1.8.5 Třída $\mathcal{NP}$.** Řekneme, že rozhodovací úloha $\mathcal{U}$ leží ve třídě $\mathcal{NP}$, jestliže existuje nedeterministický Turingův stroj, který rozhodne jazyk $L_{\mathcal{U}}$ a pracuje v polynomiálním čase.

**1.8.9 Příklady $\mathcal{NP}$ úloh.**

- Kliky v grafu. Je dán neorientovaný graf $G$ a číslo $k$. Existuje klika v grafu $G$ o alespoň $k$ vrcholech?
- Nejkratší cesty v obecném grafu. Je dán orientovaný graf s ohodnocením hran $a$. Jsou dány vrcholy $r$ a $v$. Je dáno číslo $k$. Existuje orientovaná cesta z vrcholu $r$ do vrcholu $v$ délky menší nebo rovno $k$?
- $k$-barevnost. Je dán neorientovaný graf $G$. Je graf $G$ $k$-barevný?
- Problém batohu. Je dáno $n$ předmětů $1, 2, \ldots, n$. Každý předmět $i$ má cenu $c_i$ a váhu $w_i$. Dále jsou dána čísla $A$ a $B$. Je možné vybrat předměty tak, aby celková váha nepřevýšila $A$ a celková cena byla alespoň $B$? Přesněji, existuje podmnožina předmětů $I \subseteq \{1, 2, \ldots, n\}$ taková, že

$$\sum_{i \in I} w_i \leq A \quad \text{a} \quad \sum_{i \in I} c_i \geq B?$$

**1.9.1 Redukce a polynomiální redukce úloh.** Jsou dány dvě rozhodovací úlohy $\mathcal{U}$ a $\mathcal{V}$. Řekneme, že úloha $\mathcal{U}$ se *redukuje* na úlohu $\mathcal{V}$, jestliže existuje algoritmus (program pro RAM, Turingův stroj) $M$, který pro každou instanci $I$ úlohy $\mathcal{U}$ zkonstruuje instanci $I'$ úlohy $\mathcal{V}$ a to tak, že

$$I \text{ je ANO-instance } \mathcal{U} \text{ iff } I' \text{ je ANO-instance } \mathcal{V}.$$

Fakt, že úloha $\mathcal{U}$ se redukuje na úlohy $\mathcal{V}$ značíme

$$\mathcal{U} \lhd \mathcal{V}.$$

Jestliže navíc algoritmus $M$ pracuje v polynomiálním čase, říkáme, že $\mathcal{U}$ se *polynomiálně* redukuje na $\mathcal{V}$ a značíme

$$\mathcal{U} \lhd_p \mathcal{V}.$$

**1.9.3** $\mathcal{NP}$ **úplné úlohy.** Řekneme, že rozhodovací úloha $\mathcal{U}$ je $\mathcal{NP}$ *úplná*, jestliže

1. $\mathcal{U}$ je ve třídě $\mathcal{NP}$;
2. každá $\mathcal{NP}$ úloha se polynomiálně redukuje na $\mathcal{U}$.

Třída všech $\mathcal{NP}$ úplných úloh se značí $\mathcal{NPC}$.

Zhruba řečeno, $\mathcal{NP}$ úplné úlohy jsou ty „nejtěžší" mezi všemi $\mathcal{NP}$ úlohami.

**1.9.4 Tvrzení.** Jsou dány dvě $\mathcal{NP}$ úlohy $\mathcal{U}$ a $\mathcal{V}$, pro které platí $\mathcal{U} \lhd_p \mathcal{V}$. Pak

1. jestliže $\mathcal{V}$ je ve třídě $\mathcal{P}$, pak také $\mathcal{U}$ je ve třídě $\mathcal{P}$;
2. jestliže $\mathcal{U}$ je $\mathcal{NP}$ úplná úloha, pak také $\mathcal{V}$ je $\mathcal{NP}$ úplná úloha.

**1.9.5 Tvrzení.** Kdyby některá $\mathcal{NP}$ úplná úloha patřila do třídy $\mathcal{P}$ (tj. byla by polynomiálně řešitelná), pak $\mathcal{P} = \mathcal{NP}$. Jinými slovy, každá $\mathcal{NP}$ úloha by byla polynomiálně řešitelná.

**1.9.6** $\mathcal{NP}$ **obtížné úlohy.** Jestliže o některé úloze $\mathcal{U}$ pouze víme, že se na ní polynomiálně redukuje některá $\mathcal{NP}$ úplná úloha, pak říkáme, že $\mathcal{U}$ je $\mathcal{NP}$ těžká, nebo též $\mathcal{NP}$ obtížná. Poznamenejme, že to vlastně znamená, že $\mathcal{U}$ je alespoň tak těžká jako všechny $\mathcal{NP}$ úlohy.

**1.9.7 Cookova věta.** Úloha $SAT$, splňování formulí v konjunktivním normálním tvaru, je $\mathcal{NP}$ úplná úloha.

# 10. Prednaska - co-NP, PSPACE, NPSPACE

## 1.11 Třída co-$\mathcal{NP}$

**1.11.1 Pozorování.** Je-li jazyk $L$ ve třídě $\mathcal{P}$, pak i jeho doplněk $\overline{L}$ patří do třídy $\mathcal{P}$. Obdobné tvrzení se pro jazyky třídy $\mathcal{NP}$ neumí dokázat.

**1.11.2 Definice.** Jazyk $L$ patří do třídy co-$\mathcal{NP}$, jestliže jeho doplněk patří do třídy $\mathcal{NP}$.

**1.11.3 Příklady.**

- Jazyk $USAT$, který je doplňkem jazyka $SAT$ splnitelných booleovských formulí, leží ve třídě $co-\mathcal{NP}$. (Jazyk $USAT$ se skládá ze všech nesplnitelných booleovských formulí a ze všech slov, které neodpovídají booleovské formuli.)
- Jazyk $TAUT$, který se skládá ze všech slov odpovídajících tautologii výrokové logiky, patří do třídy co-$\mathcal{NP}$.

## 1.12 Třídy $\mathcal{PSPACE}$ a $\mathcal{NPSPACE}$

**1.12.1** Je dán Turingův stroj $M$ (deterministický nebo nedeterministický). Připomeňme, že $M$ pracuje s paměťovou složitostí $p(n)$ právě tehdy, když pro každé slovo délky $n$ nepoužije paměťovou buňku větší než $p(n)$. Uvědomte si: jestliže Turingův stroj přijímá jazyk s polynomiální časovou složitostí, pak se musí zastavit na **každém** vstupu (ať už leží v přijímané jazyce, nebo ne); tj. Turingův stroj jazyk rozhoduje. Podobné tvrzení neplatí pro Turingův stroj, který nějaký jazyk přijímá s polynomiální paměťovou složitostí; ano, Turingův stroj se na slově, které nepřijímá může zacyklit.

**1.12.2 Třída $\mathcal{PSPACE}$.** Jazyk $L$ patří do třídy $\mathcal{PSPACE}$ jestliže existuje deterministický Turingův stroj $M$, který přijímá jazyk $L$ a pracuje s polynomiální paměťovou složitostí.

**1.12.3 Tvrzení.** Platí

$$\mathcal{P} \subseteq \mathcal{PSPACE}.$$

**1.12.4 Třída $\mathcal{NPSPACE}$.** Jazyk $L$ patří do třídy $\mathcal{NPSPACE}$ jestliže existuje nedeterministický Turingův stroj $M$, který přijímá jazyk $L$ a pracuje s polynomiální paměťovou složitostí.

**1.12.5 Tvrzení.** Platí

$$\mathcal{NP} \subseteq \mathcal{NPSPACE}.$$

**1.12.6 Věta.** Je dán Turingův stroj $M$ (deterministický nebo nedeterministický), který přijímá jazyk $L$ s paměťovou složitostí $p(n)$ (kde $p$ je nějaký polynom). Pak existuje konstanta $c$ taková, že $M$ přijme slovo $w$ délky $n$ po nejvýše $c^{p(n)+1}$ krocích.

**1.12.8 Věta.** Je-li jazyk $L$ ve třídě $\mathcal{PSPACE}$ ($\mathcal{NPSPACE}$), pak $L$ je rozhodován deterministickým (nedeterministickým) Turingovým strojem $M$ s polynomiální paměťovou složitostí, který se vždy zastaví po nejvýše $c^{q(n)}$ krocích, kde $q(n)$ je vhodný polynom a $c$ konstanta.

**1.12.10 Savitchova věta.** Platí

$$\mathcal{PSPACE} = \mathcal{NPSPACE}.$$

**1.12.12 Důsledek.** Platí

$$\mathcal{P} \subseteq \mathcal{NP} \subseteq \mathcal{PSPACE}.$$

# 11. Prednaska - Random Turingov Stroj, RP, co-RP, ZPP

**1.14.1 Randomizovaný Turingův stroj.** RTM je, zhruba řečeno, Turingův stroj $M$ se dvěma nebo více páskami, kde první páska má stejnou roli jako u deterministického Turingova stroje, ale druhá páska obsahuje náhodnou posloupnost 0 a 1, tj. na každém políčku se 0 objeví s pravděpodobností $\frac{1}{2}$ a 1 také s pravděpodobností $\frac{1}{2}$.

Na začátku práce:

- stroj $M$ se nachází v počátečním stavu $q_0$;
- první páska obsahuje vstupní slovo $w$, zbytek pásky pak blanky $B$;
- druhá páska obsahuje náhodnou posloupnost 0 a 1;
- případné další pásky obsahují $B$;
- všechny hlavy jsou nastaveny na prvním políčku dané pásky.

Na základě stavu $q$, ve kterém se stroj $M$ nachází, a na základě obsahu políček, které jednotlivé hlavy čtou, přechodová funkce $\delta$ určuje, zda se $M$ zastaví nebo přejde do nového stavu $p$, přepíše obsah první pásky (**nikoli ale obsah druhé pásky**) a hlavy posune doprava, doleva nebo zůstanou stát (posuny hlav jsou nezávislé).

Formálně, je-li $M$ ve stavu $q$, hlava na první pásce čte symbol $X$, na druhé pásce je číslo $a$ a

$$\delta(q, X, a) = (p, Y, D_1, D_2), \quad q, p \in Q, a \in \{0, 1\}, X, Y \in \Gamma, D_1, D_2 \in \{L, R, S\},$$

pak $M$ se přesune do stavu $p$, na první pásku napíše $Y$ a $i$-tá hlava se posune doprava pro $D_i = R$, doleva pro $D_i = L$ nebo zůstane na místě pro $D_i = S$.

Jestliže $\delta(q, X, a)$ není definováno, $M$ se zastaví.

$M$ se úspěšně zastaví právě tehdy, když se přesune do koncového (přijímacího) stavu $q_f$.

**1.14.4 Třída $\mathcal{RP}$.** Jazyk $L$ patří do třídy $\mathcal{RP}$ právě tehdy, když existuje RTM $M$ takový, že:

1. Jestliže $w \notin L$, stroj $M$ se ve stavu $q_f$ zastaví s pravděpodobností 0.
2. Jestliže $w \in L$, stroj $M$ se ve stavu $q_f$ zastaví s pravděpodobností, která je alespoň rovna $\frac{1}{2}$.
3. Existuje polynom $p(n)$ takový, že každý běh $M$ (tj. pro jakýkoli obsah druhé pásky) trvá maximálně $p(n)$ kroků, kde $n$ je délka vstupního slova.

Miller-Rabinův test prvočíselnosti je příklad algoritmu, který splňuje všechny tří podmínky (utvoříme-li k němu odpovídající RTM) a proto jazyk $L$, který se skládá ze všech složených čísel, patří do třídy $\mathcal{RP}$.

**1.14.5 Turingův stroj typu Monte-Carlo.** RTM splňující podmínky 1 a 2 z předchozí definice 1.14.4, se nazývá RTM typu *Monte-Carlo*.

Uvědomte si, že RTM typu Monte-Carlo obecně nemusí pracovat v polynomiálním čase.

**1.14.7 Třída $\mathcal{ZPP}$.** Jazyk $L$ patří do třídy $\mathcal{ZPP}$ právě tehdy, když existuje RTM $M$ takový, že:

1. Jestliže $w \notin L$, stroj $M$ se úspěšně zastaví (tj. zastaví se ve stavu $q_f$) s pravděpodobností 0.
2. Jestliže $w \in L$, stroj $M$ se úspěšně zastaví (tj. zastaví se ve stavu $q_f$) s pravděpodobností 1.
3. Střední hodnota počtu kroků $M$ v jednom běhu je $p(n)$, kde $p(n)$ je polynom a $n$ je délka vstupního slova.

To znamená: $M$ neudělá chybu, ale nezaručujeme vždy polynomiální počet kroků při jednom běhu, pouze střední hodnota počtu kroků je polynomiální.

**1.14.8 Turingův stroj typu Las-Vegas.** RTM splňující podmínky z předchozí definice 1.14.7, se nazývá typu *Las-Vegas*.

**1.14.9 Tvrzení.** Jestliže jazyk $L$ patří do třídy $\mathcal{ZPP}$, pak i jeho doplněk $\overline{L}$ patří do třídy $\mathcal{ZPP}$.

Stejný RTM $M$ typu Las-Vegas slouží „k přijetí“ jak jazyka $L$, tak i jeho doplňku $\overline{L}$; stačí koncové (přijímající) stavy RTM $M$ prohlásit za nekoncové a ze všech nekoncových stavů $M$ udělat koncové.

**1.14.10 Poznámka.** Pro jazyky ze třídy $\mathcal{RP}$ se tvrzení obdobné 1.14.9 neumí dokázat. To motivuje následující třídu jazyků.

**1.14.11 Třída co-$\mathcal{RP}$.** Jazyk $L$ patří do třídy co-$\mathcal{RP}$ právě tehdy, když jeho doplněk $\overline{L}$ patří do třídy $\mathcal{RP}$.

**1.14.12 Věta.**

$$\mathcal{ZPP} = \mathcal{RP} \cap \text{co-}\mathcal{RP}.$$

**Nástin důkazu.** Ukážeme nejprve $\mathcal{RP} \cap \text{co-}\mathcal{RP} \subseteq \mathcal{ZPP}$.

Předpokládejme, že jazyk $L$ leží v obou třídách $\mathcal{RP}$ i co-$\mathcal{RP}$. Existují proto dva RTM $M_1$ a $M_2$ typu Monte Carlo pracující v polynomiálním čase a takové, že

$M_1$ — pro jazyk $L$;

$M_2$ — pro jazyk $\overline{L}$.

Označme $p(n)$ ten větší z polynomů, které určují počet kroků $M_1$ a $M_2$. Setrojíme RTM $M$ typu Las-Vegas pro jazyk $L$ takto: Pro dané vstupní slovo $w$

1. $M$ nechá pracovat $M_1$ po dobu $p(n)$ kroků. Jestliže $M_1$ úspěšně skončí, $M$ také skončí úspěšně.
2. $M$ nechá pracovat $M_2$ po dobu $p(n)$ kroků. Jestliže $M_2$ úspěšně skončí, $M$ skončí ale neúspěšně.
3. Jestliže $M$ neskončí ani v kroku 1 ani v kroku 2, $M$ pokračuje opět krokem 1.

Dá se dokázat, že RTM $M$ je typu Las-Vegas.

Nyní ukážeme, že $\mathcal{ZPP} \subseteq \mathcal{RP} \cap \text{co-}\mathcal{RP}$.

Předpokládejme, že jazyk $L$ leží ve třídě $\mathcal{ZPP}$, existuje tedy pro něj RTM $M_1$ typu Las-Vegas. Označme $p(n)$ polynom, který udává střední hodnotu počtu kroků RTM $M_1$ pro vstupní slovo délky $n$. Vytvoříme RTM $M$ typu Monte Carlo pracující polynomiálním čase pro jazyk $L$.

$M$ nechá na vstupu $w$ pracovat RTM $M_1$ po dobu $2p(n)$. Jestliže $M_1$ úspěšně skončí, $M$ úspěšně skončí; ve všech ostatních případech RTM $M$ skončí neúspěšně.

Dá se dokázat, že $M$ splňuje všechny podmínky pro RTM typu Monte Carlo. Protože pracuje v čase $2p(n)$, jedná se o polynomiální RTM typu Monte Carlo. Proto je jazyk $L$ ve třídě $\mathcal{RP}$.

Protože třída $\mathcal{ZPP}$ je uzavřena na doplňky, je každý jazyk ze třídy $\mathcal{ZPP}$ také ve třídě co-$\mathcal{RP}$.

**1.14.13 Věta.** Platí

$$\mathcal{P} \subseteq \mathcal{ZPP}, \quad \mathcal{RP} \subseteq \mathcal{NP}, \quad \text{co-}\mathcal{RP} \subseteq \text{co-}\mathcal{NP}.$$

# 12. Prednaska - R, RE, Kod TM

**1.15.1 Rekursivní jazyky.** Řekneme, že jazyk $L$ je *rekursivní*, jestliže existuje Turingův stroj $M$, který rozhoduje jazyk $L$.

Připomeňme, že Turingův stroj $M$ rozhoduje jazyk $L$ znamená, že jej přijímá a na každém vstupu se zastaví (buď úspěšně nebo neúspěšně).

Třída rekursivních jazyků se často značí $R$.

**1.15.2 Rekursivně spočetné jazyky.** Řekneme, že jazyk $L$ je *rekursivně spočetný*, jestliže existuje Turingův stroj $M$, který tento jazyk přijímá.

Jinými slovy, $M$ se pro každé slovo $w$, které patří do $L$, úspěšně zastaví a pro slovo $w$, které nepatří do $L$ se buď zastaví neúspěšně nebo se nezastaví vůbec.

Třída rekursivně spočetných jazyků se často značí $RS$.

Každý rekursivní jazyk je též rekursivně spočetný. V dalším textu ukážeme, že naopak to neplatí, tj. existují rekursivně spočetné jazyky, které nejsou rekursivní.

**1.15.4 Tvrzení.** Jestliže jazyk $L$ je rekursivní, pak je rekursivní i jeho doplněk $\overline{L}$.

**1.15.5 Tvrzení.** Jestliže jazyk $L$ i jeho doplněk $\overline{L}$ jsou oba rekursivně spočetné, pak $L$ je rekursivní.

**1.15.6 Tvrzení.** Pro jazyk $L$ může nastat jedna z následujících možností:

1. $L$ i $\overline{L}$ jsou oba rekursivní.
2. Jeden z $L$ a $\overline{L}$ je rekursivně spočetný a druhý není rekursivně spočetný.
3. $L$ i $\overline{L}$ nejsou rekursivně spočetné.

**1.15.7 Kód Turingova stroje.** Každý Turingův stroj $M$ lze zakódovat jako binární slovo. Mějme Turingův stroj $M$ s množinou stavů $Q = \{q_1, q_2, \ldots, q_n\}$, množinou vstupních symbolů $\Sigma = \{0, 1\}$, množinou páskových symbolů $\Gamma = \{X_1, X_2, \ldots, X_m\}$, kde $X_1 = 0$, $X_2 = 1$ a $X_3 = B$. Dále počáteční stav je stav $q_1$, koncový stav je $q_2$. Označme $D_1$ pohyb hlavy doprava a $D_2$ pohyb hlavy doleva. (Tj. $D_1 = R$ a $D_2 = L$.)

Jeden přechod stroje M

$$\delta(q_i, X_j) = (q_k, X_l, D_r)$$

zakódujeme slovem

$$w = 0^i 1 0^j 1 0^k 1 0^l 1 0^r.$$

které nazýváme *Kód Turingova stroje* $M$, značíme jej $\langle M \rangle$, je

$$\langle M \rangle = 111\, w_1\, 11\, w_2\, 11 \ldots 11\, w_p\, 111,$$

Kde $w_1, \ldots, w_p$ jsou slova odpovídající všem přechodům stroje $M$.

**1.15.9 Diagonální jazyk $L_d$.** Nejprve uděláme následující úmluvu. Jestliže binární slovo $w$ nemá tvar z 1.15.7, považujeme ho za kód Turingova stroje $M$, který nepřijímá žádné slovo (neudělá nikdy žádný krok). Tj. $L(M) = \emptyset$.

Jazyk $L_d$ se skládá ze všech binárních slov $w$ takových, že Turingův stroj s kódem $w$ nepřijímá slovo $w$. (Tedy $L_d$ obsahuje i všechna slova $w$, která neodpovídají kódům nějakého Turingova stroje, ovšem obsahuje i další binární slova.)

**1.15.11 Univerzální jazyk.** *Univerzální jazyk* $L_{UN}$ je množina slov tvaru $\langle M\rangle\, w$, kde $\langle M\rangle$ je kód Turingova stroje a $w \in \{0,1\}^*$ je binární slovo takové, že $w \in L(M)$.

**1.15.12 Univerzální Turingův stroj.** Popíšeme, velmi zhruba, Turingův stroj, který přijímá univerzální jazyk $L_{UN}$. Tomuto Turingovu stroji se také říká *univerzální Turingův stroj* a značíme ho $U$.

Univerzální Turingův stroj $U$ má 4 pásky. První páska obsahuje vstupní slovo $\langle M\rangle\, w$, druhá páska simuluje pásku Turingova stroje $M$ a třetí páska obsahuje kód stavu, ve kterém se Turingův stroj $M$ nachází. Dále má $U$ ještě čtvrtou, pomocnou pásku.

Na začátku práce Turingova stroje $U$ je na první pásce vstupní slovo $\langle M\rangle\, w$, ostatní pásky obsahují pouze $B$, blanky. Připomeňme, že kód Turingova stroje získáme takto. Předpokládejme, že Turingův stroj $M$ se skládá z $(Q, \{0,1\}, \{0,1,B\}, \delta, q_1, \{q_2\})$, kde $Q = \{q_1, q_2, \ldots, q_n\}$. Označme 0 jako $X_1$, 1 jako $X_2$, $B$ jako $X_3$, pohyb doprava $R$ jako $D_1$, pohyb doleva $L$ jako $D_2$. Pak jednotlivé přechody $\delta(q_i, X_j) = (q_k, X_l, D_m)$ kódujeme

$$t = 0^i 1 0^j 1 0^k 1 0^l 1 0^m, \quad \text{kde } 1 \le i,k \le n, 1 \le j,l \le 3, 1 \le m \le 2.$$

Turingův stroj $M$ má kód

$$111\, t_1\, 11\, t_2\, 11 \ldots 11\, t_r\, 111.$$

Turingův stroj $U$ nejprve zkontroluje, že vstup je opravdu kódem Turingova stroje $M$ následovaný binárním slovem. Jestliže není, $U$ se neúspěšně zastaví.

V případě, že vstupní slovo je tvaru kód Turingova stroje $M$ následovaný binárním slovem $w$, $U$ přepíše slovo $w$ na druhou pásku a na třetí pásku napíše 0. To je proto, že Turingův stroj je na začátku práce ve stavu $q_1$ kódovaném jako 0.

Nyní Turingův stroj $U$ simuluje kroky Turingova stroje $M$ s tím, že kdykoli se stroj $M$ dostane do stavu $q_2$ (koncový „přijímací“ stav $M$), $U$ se úspěšně zastaví. Toto poznáme tak, že na třetí pásce se objeví 00 předcházené a následované $B$, blanky.

Poznamenejme, že je třeba ještě řada dalších technických detailů. Např. při přepisování slova $w$ na druhou pásku to děláme tak, že za 0 ve vstupním slově $w$ na pásku napíšeme 10, za 1 ve $w$ na druhou pásku zapíšeme 100. Je-li na druhou pásku potřeba (vzhledem k přechodové funkci Turingova stroje $M$) na druhou pásku napsat $B$, napíšeme 1000. Čtvrtá páska slouží k tomu, abychom na druhou pásku byli schopni vždy napsat stav pásky TM $M$.

**1.15.13 Důsledek.** Univerzální jazyk $L_{UN}$ je rekursivně spočetný.

# 13. Prednaska - Nerozhodnutelne, PCP

**1.15.14 Tvrzení.** Univerzální jazyk $L_{UN}$ není rekursivní.

Kdyby totiž $L_{UN}$ byl rekursivní, existoval by Turingův stroj $M$, který rozhodne $L_{UN}$. Tj. $M$ se vždy zastaví; na slovech z jazyka $L_{UN}$ se úspěšně zastaví, na slovech neležících v $L_{UN}$ se neúspěšně zastaví. Na základě tohoto Turingova stroje $M$ bychom byli schopni rozhodnout diagonální jazyk $L_d$, o kterém víme, že není ani rekursivně spočetný, viz 1.15.10.

**1.15.15 Redukce.** Připomeňme definici redukce z 1.9.1.

Jsou dány dvě rozhodovací úlohy $\mathcal{U}$ a $\mathcal{V}$. Řekneme, že úloha $\mathcal{U}$ se *redukuje* na úlohu $\mathcal{V}$, jestliže existuje algoritmus (program pro RAM, Turingův stroj) $\mathcal{A}$, který pro každou instanci $I$ úlohy $\mathcal{U}$ zkonstruuje instanci $I'$ úlohy $\mathcal{V}$ a to tak, že

$$I \text{ je ANO instance } \mathcal{U} \text{ iff } I' \text{ je ANO instance } \mathcal{V}.$$

Fakt, že úloha $\mathcal{U}$ se redukuje na úlohu $\mathcal{V}$ značíme

$$\mathcal{U} \lhd \mathcal{V}.$$

Jsou dány dva jazyky $L_1 \subseteq \Sigma^*$, $L_2 \subseteq \Gamma^*$. Řekneme, že jazyk $L_1$ se *redukuje* na jazyk $L_2$, jestliže existuje algoritmus (program pro RAM, Turingův stroj) $\mathcal{A}$, který pro každé slovo $w \in \Sigma^*$ zkonstruuje slovo $A(w) \in \Gamma^*$ a to tak, že

$$w \in L_1 \text{ iff } A(w) \in L_2.$$

Fakt, že jazyk $L_1$ se redukuje na jazyk $L_2$ značíme

$$L_1 \lhd L_2.$$

**1.15.16 Tvrzení.** Jsou dány dvě úlohy $\mathcal{U}$ a $\mathcal{V}$ takové, že $\mathcal{U} \lhd \mathcal{V}$ . Pak platí:

1. Jestliže $\mathcal{V}$ je rozhodnutelná, pak i $\mathcal{U}$ je rozhodnutelná.
2. Jestliže $\mathcal{U}$ je nerozhodnutelná, pak i $\mathcal{V}$ je nerozhodnutelná.
3. Jestliže jazyk úlohy $\mathcal{U}$ není rekursivně spočetný, pak i jazyk úlohy $\mathcal{V}$ není rekursivně spočetný.

**1.15.17 Tvrzení.** Jsou dány jazyky

$$L_e = \{M \mid L(M) = \emptyset\}, \quad L_{ne} = \{M \mid L(M) \neq \emptyset\}.$$

Pak jazyk $L_{ne}$ je rekursivně spočetný, ale ne rekursivní. Jakyk $L_e$ není ani rekursivně spočetný.

**1.16.2 Postův korespondenční problém (PCP).** Jsou dány dva seznamy slov $A, B$ nad danou abecedou $\Sigma$.

$$A = (w_1, w_2, \ldots, w_k), \quad B = (x_1, x_2, \ldots, x_k),$$

kde $w_i, x_i \in \Sigma^*$, $i = 1, 2, \ldots, k$. Řekneme, že dvojice $A, B$ *má řešení*, jestliže existuje posloupnost $i_1, i_2, \ldots, i_r$ indexů, tj $i_j \in \{1, 2, \ldots, k\}$, taková, že

$$w_{i_1} w_{i_2} \ldots w_{i_r} = x_{i_1} x_{i_2} \ldots x_{i_r}.$$

Otázka: Existuje řešení dané instance?

**1.16.3 Příklady.**

1. Jsou dány seznamy

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $A$ | 011 | 0 | 101 | 1010 | 010 |
| $B$ | 1101 | 00 | 01 | 00 | 0 |