1.10 Třída co-$\mathcal{NP}$

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.

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

1.10.2 Příklady.

- Jazyk $USAT$, který je doplňkem jazyka SAT splnitelných booleovských formulí, leží ve třídě co-$\mathcal{NP}$. (Jazyk $USAT$ se skládá ze všech nesplnítelných 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-$\mathcal{NP}$.

1.10.3 Otázka, zda co-$\mathcal{NP} = \mathcal{NP}$, je otevřená.

1.10.4 Tvrzení. co-$\mathcal{NP} = \mathcal{NP}$ právě tehdy, když existuje $\mathcal{NP}$ úplná úloha, jejíž doplněk je ve třídě $\mathcal{NP}$.

Nástin důkazu: Kdyby co-$\mathcal{NP} = \mathcal{NP}$, pak samozřejmě platí, že každý doplněk $\mathcal{NP}$ úplné úlohy je ve třídě $\mathcal{NP}$.

Předpokládejme, že existuje $\mathcal{NP}$ úplná úloha $\mathcal{U}$, jejíž doplněk patří $\overline{L_\mathcal{U}}$ do třídy $\mathcal{NP}$. Ukážeme, že v tomto případě co-$\mathcal{NP} = \mathcal{NP}$.

Uvažujme dvě $\mathcal{NP}$ úlohy $\mathcal{V}$ a $\mathcal{W}$ takové, že $L_\mathcal{V}$ se polynomiálně redukuje na $L_\mathcal{W}$. To znamená, že existuje polynomiální algoritmus $\mathcal{A}$, který pro každé slovo w vytvoří slovo x tak, že

$$w \in L_\mathcal{V} \quad \text{iff} \quad x \in L_\mathcal{W}.$$

Tedy

$$w \notin L_\mathcal{V} \quad \text{iff} \quad x \notin L_\mathcal{W}.$$

To znamená, že algoritmus $\mathcal{A}$ je polynomiální redukcí jazyka $\overline{L_\mathcal{V}}$ na $\overline{L_\mathcal{W}}$.

a) co-$\mathcal{NP} \subseteq \mathcal{NP}$. Uvažujme libovolný jazyk $L_\mathcal{V}$ ze třídy co-$\mathcal{NP}$. Pak jazyk $\overline{L_\mathcal{V}}$ patří do třídy $\mathcal{NP}$ a proto se polynomiálně redukuje na $\mathcal{U}$. Máme tedy polynomiální redukci $\overline{L_\mathcal{V}}$ na $L_\mathcal{U}$ a tedy i polynomiální redukci $L_\mathcal{V}$ na $\overline{L_\mathcal{U}}$. Protože $\overline{L_\mathcal{U}}$ patří do třídy $\mathcal{NP}$, patří do třídy $\mathcal{NP}$ i jazyk $L_\mathcal{V}$.

b) $\mathcal{NP} \subseteq$ co-$\mathcal{NP}$. Uvažujme libovolný jazyk $L_\mathcal{V}$ ze třídy $\mathcal{NP}$. Protože $L_\mathcal{U}$ je $\mathcal{NP}$-úplný jazyk, existuje polynomiální redukce jazyka $L_\mathcal{V}$ na $L_\mathcal{U}$. Z výše uvedeného vyplývá, že je to také polynomiální redukce $\overline{L_\mathcal{V}}$ na $\overline{L_\mathcal{U}}$. Navíc, je jazyk $\overline{L_\mathcal{U}}$ ve třídě $\mathcal{NP}$‘tedy i jazyk $\overline{L_\mathcal{V}}$ je ve třídě $\mathcal{NP}$. Odtud dostáváme, že jazyk $L_\mathcal{V}$ patří do třídy co-$\mathcal{NP}$.

1.11 Třídy $\mathcal{P}SPACE$ a $\mathcal{NP}SPACE$

1

1.11.1 Je dán Turingův stroj M (determininstický nebo nedeterministický). Připomeňme, že M pracuje s paměťovou složitostí $p(n)$ právě tehdy, když pro každé slovo délky n nepoužije paměťovou buňku větší než $p(n)$.

1.11.2 Třída $\mathcal{P}SPACE$. Jazyk L patří do třídy $\mathcal{P}SPACE$ právě tehdy, když existuje deterministický Turingův stroj M, který přijímá jazyk L a pracuje s polynomiální paměťovou složitostí.

1.11.3 Tvrzení. Platí

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

1.11.4 Třída $\mathcal{NP}SPACE$. Jazyk L patří do třídy $\mathcal{NP}SPACE$ právě tehdy, když existuje nedeterministický Turingův stroj M, který přijímá jazyk L a pracuje s polynomiální paměťovou složitostí.

1.11.5 Tvrzení. Platí

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

1.11.6 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.

1.11.7 Myšlenka důkazu věty 1.11.6. Konstantu c volíme tak, abychom měli zajištěno, že Turingův stroj M má při práci se vstupem délky n méně než $c^{p(n)+1}$ různých situací. Zajímají nás totiž pouze takové výpočty, ve kterých se situace neopakují. Označme t počet páskových symbolů Turingova stroje M a označme s počet stavů M. Pak M má $p(n)\, s\, t^{p(n)}$ různých situací.

Položme $c = t + s$. Z binomické věty vyplývá, že

$$c^{p(n)+1} = (t+s)^{p(n)+1} = t^{p(n)+1} + p(n)\, t^{p(n)}\, s + \ldots.$$

Odtud $c^{p(n)+1} \geq p(n)\, t^{p(n)}\, s$.

1.11.8 Věta. Je-li jazyk L ve třídě $\mathcal{P}SPACE$ ($\mathcal{NP}SPACE$)), 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.

1.11.9 Myšlenka důkazy věty 1.11.8. Předpokládejme, že $L \in \mathcal{P}SPACE$. Pak existuje Turingův stroj M_1, který přijímá jazyk L s paměťovou složitostí $p(n)$ ($p(n)$ je vhodný polynom). Víme (z věty 1.11.6), že existuje konstanta c taková, že Turingův stroj M_1 potřebuje nejvýše $c^{p(n)+1}$ kroků.

Vytvoříme Turingův stroj M_2, který bude mít dvě pásky: první páska bude simulovat M_1, druhá bude počítat kroky na první pásce. Jestliže počet kroků překročí $c^{p(n)+1}$, Turingův stroj M_2 se neúspěsně zastaví.

Hledaný Turingův stroj M je Turingův stroj s jednou páskou, který simuluje Turingův stroj M_2. Turingův stroj M pracuje v s časovou složitostí $\mathcal{O}(c^{2p(n)})$, tedy v maximálně $d\, c^{2p(n)}$ krocích. Nyní stačí položit $q(n) = 2p(n) + \log_c d$ nebo jakýkoli polynom větší.

1.11.10 Savitchova věta. Platí

$$\mathcal{P}SPACE = \mathcal{NP}SPACE.$$

1.11.11 Nástin myšlenky důkazu Savitchovy věty. Zřejmě $\mathcal{P}SPACE \subseteq \mathcal{NP}SPACE$. Důkaz opačné inkluze $\mathcal{NP}SPACE \subseteq \mathcal{P}SPACE$ spočívá v tom, že jsme schopni nedeterministický Turingův stroj pracující s paměťovou složitostí $p(n)$ simulovat deterministickým Turingovým strojem, který pracuje s paměťovou složitostí $\mathcal{O}([p(n)]^2)$.

Je dán nedeterministický Turingův stroj M, který přijímá jazyk L s polynomiální paměťovou složitostí $p(n)$. Konstrukce deterministického Turingova stroje přijímajícího stejný jazyk jako M s polynomiální paměťovou složitostí je založena na rekursivní proceduře *dostup*(I, J, m), kde I a J jsou situace a m je číslo. Výstup procedury *dostup*(I, J, m) je buď 1, jestliže Turingův stroj se z situace I do situace J dostane v nejvýše m krocích, 0 v opačném případě. Procedura *dostup*(I, J, m) pro každou situaci K rekursivně zavolá procedury *dostup*$(I, K, m/2)$ a *dostup*$(K, J, m/2)$.

Pro vstup w voláme proceduru *dostup*(I_0, J, m), kde I_0 je počáteční situace M, J je přijímající situace M a $m = \log_2 c^{p(n)+1}$ (c je konstanta z 1.11.6). Dá se dokázat, že pro vykonání procedury *dostup*(I, J, m) deterministickým Turingovým strojem stačí paměťová složitost $\mathcal{O}([p(n)]^2)$. (Uvědomte si, že nám nezáleží na tom, jak dlouho deterministický Turingův stroj pracuje, zajímáme se pouze o paměťové nároky.)

1.11.12 Důsledek. Platí

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