Jméno: **Souřadnice:**

0007 *list* 1 *učo* *body*

Oblast strojově snímatelných informací. Své UČO vyplňte zleva dle přiloženého vzoru číslic. Jinak do této oblasti nezasahujte. 0 1 2 3 4 5 6 7 8 9

(a) Která z následujících tvrzení jsou pravdivá? Svou odpověď zdůvodněte.

(i) Množina $A_1 \subseteq \mathbb{N}$ je rekurzivní právě tehdy, když existuje totálně vyčislitelná funkce f taková, že $x \in A_1 \Leftrightarrow \forall x' < x : f(x) > f(x')$.

(ii) Množina $A_2 \subseteq \mathbb{N}$ obsahující ty $i \in \mathbb{N}$, že se i-tý while-program zastaví pro nějaký vstup, je rekurzivní.

(iii) Množina $A_2 \subseteq \mathbb{N}$ z předchozího bodu je rekurzivně spočetná.

(b) Definujte, co znamená, že rozhodovací problém je polynomiálně redukovatelný na jiný rozhodovací problém. Platí, že každý NP-úplný problém je polynomiálně redukovatelný na každý PSPACE-úplný problém? Svou odpověď zdůvodněte.

(a) (i) Není pravdivé. Množina $A_1 = \{1\}$ je rekurzivní, ale libovolná funkce f splňuje $\forall x' < 0 : f(0) > f(x')$.

Na dalším listu je uvedeno nesprávné řešení této části příkladu, za které bychom také dali plný počet bodů - nebylo naším úmyslem vytvořit „chyták“.

(ii) Není pravdivé. Množina A_2 je netriviální množina, která respektuje funkce, a není tedy rekurzivní dle První Riceovy věty.

(iii) Je pravdivé. Následující algoritmus se zastaví právě tehdy, když $i \in A_2$. Postupně pro $k = 1, 2, \ldots$ budeme simulovat prvních k kroků výpočtu i-tého programu na vstupech $1, \ldots, k$. Pokud pro nějaké k a nějaký vstup simulace skončí, pak se popisovaný algoritmus zastaví. Pokud takové k nikdy nenalezneme, pak se popisovaný algoritmus nikdy nezastaví. To ale nastane jen tehdy, pokud se i-tý program pro žádný vstup nezastaví. Existence výše popsaného algoritmu dokazuje, že A_2 je rekurzivně spočetná.

(b) Problém P je polynomiálně redukovatelný na problém Q, pokud existuje polynomiální algoritmus A, t.ž. $x \in P$ právě tehdy, když $A(x) \in Q$ pro všechny možné vstupy x, kde $A(x)$ je výstup algoritmu A pro x.

Ano, platí. Nechť P je PSPACE-úplný problém. Protože každý problém ze třídy PSPACE je na P polynomiálně redukovatelný, každý NP-úplný problém patří do NP dle definice NP-úplnosti a NP $\subseteq$ PSPACE, je každý NP-úplný problém polynomiálně redukovatelný na P.

Jméno: **Souřadnice:**

0007 *list* 1 *učo* *body*

Oblast strojově snímatelných informací. Své UČO vyplňte zleva dle přiloženého vzoru číslic. Jinak do této oblasti nezasahujte.

0123456789

(a) Která z následujících tvrzení jsou pravdivá? Svou odpověď zdůvodněte.

(i) Množina $A_1 \subseteq \mathbb{N}$ je rekurzivní právě tehdy, když existuje totálně vyčislitelná funkce f taková, že $x \in A_1 \Leftrightarrow \forall x' < x : f(x) > f(x')$.

(ii) Množina $A_2 \subseteq \mathbb{N}$ obsahující ty $i \in \mathbb{N}$, že se i-tý while-program zastaví pro nějaký vstup, je rekurzivní.

(iii) Množina $A_2 \subseteq \mathbb{N}$ z předchozího bodu je rekurzivně spočetná.

(b) Definujte, co znamená, že rozhodovací problém je polynomiálně redukovatelný na jiný rozhodovací problém. Platí, že každý NP-úplný problém je polynomiálně redukovatelný na každý PSPACE-úplný problém? Svou odpověď zdůvodněte.

(a) (i) *Toto řešení je chybné – viz komentář na předchozím listu.*
Je pravdivé. Ekvivalenci dokazujeme jako dvě implikace.

A_1 je rekurzivní $\Rightarrow$ existence funkce f ze zadání. Nechť $f(x)$ je počet hodnot $y \leq x$, t.ž. $y \in A_1$. Funkci f je součtem prvních x hodnot charakteristické funkce množiny A_1 a je tedy totálně vyčislitelná. Pokud $x \notin A_1$, pak $f(x) = f(x-1)$ a tedy neplatí $\forall x' < x : f(x) > f(x')$. Pokud $x \in A_1$, pak $f(x) > f(x')$ pro všechna $x' < x$. *(Tento argument selhává pro $x = 0$, pokud $0 \notin A_1$.)*

Existence funkce f ze zadání $\Rightarrow$ A_1 je rekurzivní. Následující algoritmus počítá charakteristickou funkci množiny A_1. Postupně pro $x' = 0, \ldots, x-1$ vypočteme $f(x')$ a ověříme, že $f(x') < f(x)$. Pokud test někdy selže, pak vystoupíme 0. Jinak vystoupíme 1.

(ii) Není pravdivé. Množina A_2 je netriviální množina, která respektuje funkce, a není tedy rekurzivní dle První Riceovy věty.

(iii) Je pravdivé. Následující algoritmus se zastaví právě tehdy, když $i \in A_2$. Postupně pro $k = 1, 2, \ldots$ budeme simulovat prvních k kroků výpočtu i-tého programu na vstupech $1, \ldots, k$. Pokud pro nějaké k a nějaký vstup simulace skončí, pak se popisovaný algoritmus zastaví. Pokud takové k nikdy nenalezneme, pak se popisovaný algoritmus nikdy nezastaví. To ale nastane jen tehdy, pokud se i-tý program pro žádný vstup nezastaví. Existence výše popsaného algoritmu dokazuje, že A_2 je rekurzivně spočetná.

(b) Problém P je polynomiálně redukovatelný na problém Q, pokud existuje polynomiální algoritmus A, t.ž. $x \in P$ právě tehdy, když $A(x) \in Q$ pro všechny možné vstupy x, kde $A(x)$ je výstup algoritmu A pro x.

Ano, platí. Nechť P je PSPACE-úplný problém. Protože každý problém ze třídy PSPACE je na P polynomiálně redukovatelný, každý NP-úplný problém patří do NP dle definice NP-úplnosti a NP $\subseteq$ PSPACE, je každý NP-úplný problém polynomiálně redukovatelný na P.

Oblast strojově snímatelných informací, nezasahujte.

Jméno: **Souřadnice:**

0007 *list* 2 *učo* *body*

Oblast strojově snímatelných informací. Své UČO vyplňte zleva dle přiloženého vzoru číslic. Jinak do této oblasti nezasahujte. 0123456789

(a) Vysvětlete, co znamená, že množina $B \subseteq \mathbb{N}$ respektuje funkce.

(b) Nechť $f : \mathbb{N} \to \mathbb{N} \cup \{\perp\}$ je funkce, t.ž. $f(x) \neq \perp$ pro konečně mnoho $x \in \mathbb{N}$, a nechť C je množina těch $i \in \mathbb{N}$, že funkce φ_i je rozšířením f.

(i) Pro které funkce f je množina C rekurzivní?

(ii) Pro které funkce f je množina C rekurzivně spočetná?

(iii) Pro které funkce f je množina $\overline{C}$ rekurzivně spočetná?

Své odpovědi zdůvodněte.

(a) Množina B respektuje funkce, pokud pro každé dva indexy i a j, t.ž. $\varphi_i = \varphi_j$, platí, že buď jsou oba v B nebo není ani jeden v B.

(b) Protože $f(x)$ je různá od $\perp$ pouze pro konečně mnoho hodnot $x \in \mathbb{N}$, je funkce f vyčíslitelná a tedy existuje α, t.ž. $f = \varphi_\alpha$. Dále zvolme β, t.ž. funkce φ_β cyklí na všech vstupech.

(i) Právě tehdy, když $f = \varphi_\beta$. Pokud $f = \varphi_\beta$, pak $C = \mathbb{N}$ a C je tedy rekurzivní. Pokud $f \neq \varphi_\beta$, pak $\alpha \in C$ a $\beta \notin C$, a C není rekurzivní dle První Riceovy věty, neboť respektuje funkce a není triviální.

(ii) Vždy. Nechť $x_1, \ldots, x_k$ jsou právě ty hodnoty, pro které $f(x) \neq \perp$. Následující algoritmus se zastaví na vstupu i právě tehdy, když $i \in C$: postupně pro $j = 1, \ldots, k$ simulujeme i-tý program pro vstup x_j a pokud simulace skončí a výsledek není $f(x_j)$, tak se algoritmus zacyklí. Pokud simulace skončí pro všech k hodnot $x_1, \ldots, x_k$ a výsledek je vždy $f(x_j)$, pak se algoritmus zastaví. Právě popsaný algoritmus se na vstupu i zastaví právě tehdy, když $i \in C$. Platí tedy, že množina C je vždy rekurzivně spočetná.

(iii) Právě tehdy, když $f = \varphi_\beta$. Protože množina C je rekurzivně spočetná, je množina $\overline{C}$ rekurzivně spočetná právě tehdy, když C je rekurzivní, což jsme vyřešili v první části této podotázky.

Jméno: **Souřadnice:**

0007 *list* 3 *učo* *body*

Oblast strojově snímatelných informací. Své UČO vyplňte zleva dle přiloženého vzoru číslic. Jinak do této oblasti nezasahujte. 0 1 2 3 4 5 6 7 8 9

(a) Definujte prostorovou složitost nedeterministického algoritmu.

(b) Zformulujete a dokažte větu z přednášky, která hovoří o inkluzi třídy NSPACE($f(n)$) pro $f(n) \in \Omega(\log n)$ ve třídě problémů řešitelných deterministicky v omezeném čase.

(a) Prostorová složitost nedeterministického algoritmu je funkce $f : \mathbb{N} \to \mathbb{N}$, t.ž. $f(n)$ se rovná nejmenšímu čislu K, pro které platí, že každá větev výpočtu algoritmu pro libovolný vstup velikost n pracuje v prostoru nejvýše K.

(b) Zadání popisuje větu, která říká, že NSPACE(f) $\subseteq$ TIME $\left(2^{O(f)}\right)$. Zvolme problém z NSPACE(f) a nedeterministický algoritmus A, který jej řeší v prostoru $O(f)$. Nyní popíšeme deterministický algoritmus, který zvolený problém řeší v čase $2^{O(f)}$. Pro daný vstup velikosti n a vhodně zvolené číslo s sestavíme pomocný orientovaný graf, jehož vrcholy jsou možné stavy výpočtu algoritmu A, který používá prostor nějvýše s. Takových stavů je nejvýše $2^{O(s)}$: pokud uvažujeme výpočetní mode Turingových strojů, je stav výpočtu určen stavem stroje, pozicí hlav na vstupní pásce (n možností) a pracovní pásce (s možností) a obsahem pracovní pásky (3^s možností v případě binární abecedy). Z vrcholu odpovídajícího stavu s vede hrana do vrcholu odpovídající stavu s' právě tehdy, když ze stavu s lze přejít v jednom kroku do stavu s'. Pokud je s zvolené tak, že prostorová složitost algoritmu pro danný vtsup je nejvýše s, pak stačí v tomto pomocném grafu ověřit existenci cesty z počátečního stavu výpočtu do některého z přijímajících stavů. To lze snadno udělat v čase v polynomiálním ve velikosti pomocného grafu, tj. v v čase $2^{O(s)}$.

Zbývá tedy popsat, jak nalézt takové číslo s, které zároveň není o mnoho větší než $f(n)$. Nejdříve zvolíme $s = 1$ a sestavíme pomocný graf pro tuto hodnotu s. Pokud v něm existuje cesta z počátečního stavu výpočtu do stavu, ze kterého lze přejít do stavu používajícího prostor $s + 1$ (toto lze vyhodnotit v čase $2^{O(s)}$), pak zjevně $s < f(n)$ a v takovém případě hodnotu s zdvojnásobíme a postup opakujeme. Zastavíme se nejpozději, když $s \geq f(n)$. Postup popsaný v prvním odstavci pak použijeme pro tuto hodnotu s. Vzhledem k popsanému postupu bude tedy nakonci platit, že $s \leq 2f(n)$ a tedy $s \in O(f)$. Celková časová složitost právě popsaného algoritmu je tedy $2^{O(s)}$.