Teoretická informatika (TIN) – 2022/2023
Úkol 3

(max. zisk 5 bodů – 10 bodů níže odpovídá 1 bodu v hodnocení předmětu)

1. Dokažte redukcí, že následující jazyk L není ani částečně rozhodnutelný:

$$L = \{\langle M_1 \rangle \sharp \langle M_2 \rangle \mid M_1 \text{ a } M_2 \text{ jsou Turingovy stroje takové, že } L(M_1) \leq L(M_2)\}$$

(Připomínáme, že $L(M_1) \leq L(M_2)$ znamená, že $L(M_1)$ se redukuje na $L(M_2)$.)

15 bodů

2. Pro danou konečnou podmnožinu M přirozených čísel a číslo k je úkolem rozhodnout, zda existuje rozklad R množiny M, který má maximálně k tříd, a kde každé dva různé prvky i, j stejné třídy r jsou nesoudělné. Dokažte, že se jedná o NP-úplný problém. Čísla jsou kódována binárně.[1]

Inspirujte se zde: `https://en.wikipedia.org/wiki/List_of_NP-complete_problems`

15 bodů

3. Uvažujte *množinový term* t definovaný gramatikou

$$t ::= \emptyset \mid (t \cup t) \mid \mathsf{add\_max}(t) \mid \mathsf{remove\_min}(t)$$

Hodnotu $h(t)$ termu t (slova z jazyka gramatiky) definujeme induktivně jako množinu přirozených čísel:

$$h(\emptyset) = \emptyset \qquad h(\mathsf{add\_max}(t)) = h(t) \cup \{\max(h(t)) + 1\} \quad \text{(vložení maxima plus 1)}$$
$$h(t \cup t') = h(t) \cup h(t') \qquad h(\mathsf{remove\_min}(t)) = h(t) \setminus \{\min(h(t))\} \quad \text{(odebrání minima)}$$

($\max(\emptyset)$ definujeme jako 0 a $\min(\emptyset)$ jako ∞)

(a) Navrhněte pseudokód algoritmu, který vyhodnotí daný term t, tj., vrátí $h(t)$.
Algoritmus bude term vyhodnocovat rekurzivně od listů ke kořeni.
Reprezentujte hodnoty termů, množiny čísel, jako dvousměrně vázané seznamy čísel. Vysvětlete, jak implementovat operace tak, aby pro konečné množiny S, S' mělo vyhodnocení operací $\mathsf{add\_max}(S)$ a $\mathsf{remove\_min}(S)$ konstantní časovou složitost, a aby vyhodnocení sjednocení S a S' mělo složitost lineární k maximu množiny s menším maximem. (Máte k dispozici operace získání odkazu na první/poslední/další prvek dvousměrného seznamu, vložení prvku před/za odkazovanou pozici, které mají konstantní časovou složitost. Nepopisujte jejich implementaci na úrovni manipulace s ukazateli.)

(b) Ukažte, že Váš algoritmus má lineární časovou složitost (běží v čase $O(|t|)$[2]).
Čas výpočtu $h(t)$ bude součtem času potřebného k vyhodnocení všech výskytů operací v t.
Budete potřebovat argumentovat, že čas vyhodnocení každého sjednocení $h(t \cup t')$ na základě $h(t)$ a $h(t')$ se amortizuje časem potřebným pro výpočet $h(t)$ a $h(t')$.[3]

Zvolte vhodnou úroveň abstrakce algoritmů a důkazu složitosti. Vágní nebo zbytečně technický popis nebude akceptován.

20 bodů

---

[1] Můžete předpokládat, že hodnota n-tého nejmenšího prvočísla je polynomiální k n.
Můžete také použít následující dodatečné předpoklady: (1) výpočet prvních n prvočísel je polynomiální k n, (2) faktorizace binárně zapsaného čísla je v NP. Předpoklady (1) a (2) byste ale měli být schopni zdůvodnit. Pokud je nepoužijete, tj., nebudete je pouze předpokládat ale zdůvodníte je (nebo příklad správně vyřešíte bez nich), bude to oceněno odpuštěním až cca 10 bodů v případě chyb jinde.

[2] $|t|$ je jednoduše délka slova t.

[3] Uvědomte si, že naivním součtem složitostí nejhoršího případů jednotlivých operací bychom dostali příliš velkou, kvadratickou, složitost vyhodnocení $h(t)$: čas potřebný k vyhodnocení sjednocení je lineární k velikosti operandů, velikost operandu je ohraničena počtem výskytů $\mathsf{add\_max}$ v t, vyhodnocení jednoho sjednocení je tedy v $O(|t|)$. Sjednocení se vyskytuje maximálně $|t|$-krát, dohromady je tedy čas všech sjednocení v $O(|t|)^2$.