Teoretická informatika (3. ročník)

Algorimus A*, problém SAT, algoritmus DPLL **(varianta A)**

JMÉNO: ______________________ TŘÍDA: ________ DATUM: __________

Body	25 – 21	20 – 17	16 – 13	12 – 10	9 – 0
Známka	1	2	3	4	5

1. **(5 bodů) Algoritmus A***

   (a) (4 body) Vyberte **pravdivá tvrzení** o algoritmu A*.

   - √ **Po skončení algoritmu jsou nalezeny nejkratší cesty mezi počátečním a ostatními vrcholy v grafu.**
   - ◯ V každé iteraci vnějšího cyklu vybírá vrcholy podle f-skóre, které odpovídá vzdálenosti od počátečního vrcholu v_0, tj. $f(v) = D(v)$.
   - ◯ Algoritmus **vždy** uzavře všechny vrcholy v grafu.
   - ◯ Po skončení algoritmu jsou nalezeny nejkratší cesty **mezi všemi dvojicemi** vrcholů v grafu.
   - √ **Algoritmus vždy aktualizuje vzdálenosti do všech vrcholů sousedících s právě vybraným (uzavíraným) vrcholem, je-li nová vzdálenost kratší než původní.**
   - ◯ Algoritmus lze aplikovat na libovolný graf (tj. bez ohledu na *počet vrcholů*, *hrany*, ani jejich *váhy*).
   - √ **Algoritmus lze aplikovat na grafy, kde jsou váhy hran *nezáporné*.**
   - √ **Heuristická funkce ψ musí splňovat trojúhelníkovou nerovnost.**

   (b) (1 bod) K čemu slouží *heuristická funkce*?

   Řešení: Heuristická funkce slouží pro výpočet odhadu vzdálenosti libovolného vrcholu od cílového.

2. **(15 bodů) Problém SAT**

   (a) (1 bod) Co je to problém SAT?

   Řešení: Jedná se o problém, zda pro danou logickou formuli φ existuje takové ohodnocení proměnných, aby byla pravdivá, tj. $\varphi(\dots) = 1$.

   (b) (3 body) Vyberte všechny logické formule, které jsou **splnitelné** a ke každé **uveďte libovolné splňující ohodnocení** (pokud existuje).

   - √ $\neg x_1 \Rightarrow x_2$
   - √ $(x_1 \Leftrightarrow x_2) \wedge x_3$
   - ◯ $(x_1 \Rightarrow x_2) \wedge \neg(\neg x_1 \vee x_2)$
   - √ $(\neg x_1 \wedge \neg x_2) \vee (x_3 \Leftrightarrow x_1)$
   - ◯ $(x_1 \Leftrightarrow x_2) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee x_2)$
   - ◯ $\neg x \Rightarrow \neg(\neg x)$

(c) (2 body) Co je to **literál** a **klauzule** ve formuli v *konjunktivní normální formě (CNF)*?

Řešení:

- *Literál* je libovolná proměnná, nebo její negace.
- *Klauzule* je výraz sestavený z literálů spojených disjunkcí $\vee$.

(d) (4 body) Vyberte všechny logické formule, které jsou v *CNF*.

○ $(x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2)$
√ $\neg x_1 \vee \neg x_2 \vee \neg x_3$
○ $(\neg x_1 \vee x_4) \wedge (x_1 \vee (\neg x_2 \wedge x_3))$
○ $\neg(x_1 \vee x_2)$
√ $\neg x$
√ $x_2 \wedge (x_1 \vee x_2) \wedge x_3$
√ $x_1 \vee \neg x_2$
○ $(\neg x_1 \Rightarrow x_2) \wedge (\neg x_1 \Leftrightarrow \neg x_2)$

(e) (5 bodů) Převeďte formuli $\psi(x, y) = (\neg x_1 \wedge x_2) \Leftrightarrow (x_1 \Rightarrow \neg x_2)$ do CNF. Uveďte tabulku pravdivostních hodnot.

Řešení: Tabulka pravdivostních hodnot pro ψ:

x_1	x_2	$\neg x_1 \wedge x_2$	$x_1 \Rightarrow \neg x_2$	$\psi(x, y)$
0	0	0	1	0
0	1	1	1	1
1	0	0	1	0
1	1	0	0	1

$\psi'(x, y) = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2)$

3. **(5 bodů) Algoritmus DPLL**

(a) (2 body) K čemu slouží algoritmus DPLL? Co je jeho **vstupem**?

Řešení: Algoritmus DPLL slouží k rozhodnutí, zda je formule splnitelná. Vstupem je formule v CNF.

(b) (3 body) Máme formuli $\varphi(x, y, z) = (y \vee \neg z) \wedge (x \vee \neg y) \wedge (\neg y \vee \neg x \vee \neg z) \wedge (\neg x \vee y)$. Napište výslednou formuli φ' po aplikaci procedury **pure literal elimination**.

Řešení: $\varphi'(x, y, z) = (x \vee \neg y) \wedge (\neg x \vee y)$

4. **(Bonus)** *Odvoďte* časovou složitost algoritmu DPLL. (Stačí odvodit danou *rekurentní rovnici* a pak uvést její řešení v $\mathcal{O}$-notaci. Řešit ji nemusíte.)

Řešení: V nejhoriším případě algoritmus eliminuje pouze jeden literál z formule φ, kde počet literálů označíme n. Při každém volání funkce strávíme konstantní čas c a následně rekurzivně voláme dvakrát funkci na vstup velikosti $n - 1$. Označíme-li počet kroků algoritmu $T(n)$ pro formuli φ, pak platí

$$T(n) = c + 2 \cdot T(n - 1).$$

Lze ukázat, že $T(n) = \mathcal{O}(2^n)$.