NTIN090 — Základy složitosti a vyčíslitelnosti
6. cvičení

Petr Kučera

5. ledna 2023

1. Uvažme rozhodovací verzi problému Obchodního cestujícího

Problém 1: Obchodní cestující

Instance: Množina měst $C = \{c_1, \ldots, c_n\}$, hodnoty $d(c_i, c_j) \in \mathbb{N}$ přiřazující každé dvojici měst vzdálenost a přirozené číslo D.

Otázka: Existuje permutace měst $c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}$, pro kterou platí, že

$$\left(\sum_{i=1}^{n-1} d(c_{\pi(i)}, c_{\pi(i+1)})\right) + d(c_{\pi(n)}, c_{\pi(1)}) \leq D?$$

a jeho optimalizační verzi

Problém 2: Obchodní cestující

Instance: Množina měst $C = \{c_1, \ldots, c_n\}$, hodnoty $d(c_i, c_j) \in \mathbb{N}$ přiřazující každé dvojici měst vzdálenost.

Cíl: Nalézt permutace měst $c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}$, pro kterou je

$$\left(\sum_{i=1}^{n-1} d(c_{\pi(i)}, c_{\pi(i+1)})\right) + d(c_{\pi(n)}, c_{\pi(1)})$$

minimální.

Předpokládejme, že rozhodovací verzi problému Obchodního cestujícího umíme rozhodnout v polynomiálním čase. To znamená, že máme funkci `rozhodniOC`(C, d, D), kde $C = \{c_1, \ldots, c_n\}$ je množina měst, $d(c_i, c_j) \in \mathbb{N}$ určuje vzdálenosti mezi nimi a D je limit na vzdálenost a která rozhodne, zda existuje pořadí měst, při němž navštíví obchodní cestující všechna města, vrátí se do svého domovského města a nacestuje přitom vzdálenost nejvýš D. Popište, jak za tohoto předpokladu vyřešit v polynomiálním čase optimalizační verzi Obchodního cestujícího, tedy popište algoritmus, který nalezne pořadí měst, při němž nacestuje obchodní cestující nejkratší vzdálenost. Algoritmus bude pracovat v polynomiálním čase za předpokladu, že volání funkce `rozhodniOC`(C, d, D) proběhne v polynomiálním čase.

Řešení: Algoritmus nejprve určí hodnotu D_{opt} optimálního řešení binárním prohledáváním na intervalu $0, \ldots, \sum_{i=1}^{n} \sum_{j=i+1}^{n} d(c_i, c_j)$ za využití volání funkce `rozhodniOC`(C, d, D). Díky binárnímu vyhledávání je počet volání funkce `rozhodniOC`(C, d, D) polynomiální v délce binárního zápisu vah $d(c_i, c_j)$.

Když už známe D_{opt}, můžeme najít konkrétní cyklus následovně. V cyklu zkoušíme deaktivovat hrany tak, že postupně pro každou hranu $\{c_i, c_j\}$ nastavíme hodnotu $d(c_i, c_j) = D_{\text{opt}} + 1$ a položíme dotaz funkcí $\texttt{rozhodniOC}(C, d, D_{\text{opt}})$. Pokud je odpovědí, že v s upravenými vahami existuje řešení, pak necháme upravenou vzdálenost $d(c_i, c_j)$, jinak vrátíme tuto vzdálenost na původní hodnotu. Poté, co probereme všechny hrany, hrany s vahou $d(c_i, c_j) \le D_{\text{opt}}$ tvoří cyklus, který je hledaným řešením.

2. Rozmyslete si, jak obtížné je o dané formuli φ zodpovědět následující otázky v závislosti na tom, je-li φ KNF, DNF, či obecná výroková formule.
   (a) $(\exists \mathbf{a})[\varphi(\mathbf{a}) = 1]$?
   (b) $(\exists \mathbf{a})[\varphi(\mathbf{a}) = 0]$?
   (c) $(\forall \mathbf{a})[\varphi(\mathbf{a}) = 1]$?
   (d) $(\forall \mathbf{a})[\varphi(\mathbf{a}) = 0]$?

Řešení:

	KNF	DNF	Formule
$(\exists \mathbf{a})[\varphi(\mathbf{a}) = 1]$	NP-úplné	P	NP-úplné
$(\exists \mathbf{a})[\varphi(\mathbf{a}) = 0]$	P	NP-úplné	NP-úplné
$(\forall \mathbf{a})[\varphi(\mathbf{a}) = 1]$	P	co-NP-úplné	co-NP-úplné
$(\forall \mathbf{a})[\varphi(\mathbf{a}) = 0]$	co-NP-úplné	P	co-NP-úplné

3. Ukažte, že pro každé m existuje formule φ v KNF s m klauzulemi taková, že libovolná formule ψ v DNF, která je ekvivalentní s φ, obsahuje alespoň 2^m termů.

Řešení: $\varphi = \bigwedge_{i=1}^{m} (x_i \vee y_i)$

4. (a) Pro každou z následujících formulí popište její ekvivalentní KNF
      1. $v \iff \neg x$
      2. $v \iff x \vee y$
      3. $v \iff x \wedge y$
      4. $v \iff (x \implies y)$
      5. $v \iff (x \iff y)$

   (b) Uvažme formuli
   $$\varphi \equiv [(x \implies y) \wedge (y \implies z)] \iff (x \implies z) \tag{1}$$
   (c) Je formule φ splnitelná?
   (d) Je formule φ tautologie?
   (e) Rozmyslete si podrobněji následující obecný postup, který k libovolné formuli φ zkonstruuje v polynomiálním čase formuli ψ v KNF, která je splnitelná, právě když φ je splnitelná.
      1. Sestrojíme strom odvození formule φ.
      2. S každým uzlem stromu asociujeme novou proměnnou.
      3. Sémantiku dané proměnné definujeme jednou z ekvivalencí výše podle toho, jaká logická spojka je v daném uzlu.
      4. Přepíšeme jednotlivé ekvivalence do KNF.
      5. ψ sestrojíme jako konjunkci zkonstruovaných KNF podformulím, k nimž přidáme proměnnou kořene stromu jako jednotkovou klauzuli.

Ukažte korektnost tohoto postupu a demonstrujte jej na formuli (1). Vznikne tímto postupem formule ψ, která je tautologie?

Řešení:

(a)
1. $(v \vee x) \wedge (\neg v \vee \neg x)$
2. $(\neg v \vee x \vee y) \wedge (v \vee \neg x) \wedge (v \vee \neg y)$
3. $(\neg v \vee x) \wedge (\neg v \vee y) \wedge (v \vee \neg x \vee \neg y)$
4. Vznikne aplikací předchozích bodů s využitím toho, že $(x \implies y) \equiv (\neg x \vee y)$.
5. Vznikne aplikací předchozích bodů s využitím toho, že $(x \iff y) \equiv (x \vee \neg y) \wedge (\neg x \vee y)$.

(b) Formule φ je splnitelná.

(c) Formule φ je tautologie.

(d) Výsledná formule je splnitelná, ale není to tautologie.

5. Popište jednoduchý algoritmus pro 3-SAT, který v každém kroku vybere klauzuli $(l_1 \vee l_2 \vee l_3)$ a větví na tři podpřípady dle částečných ohodnocení $\{l_1\}$, $\{\neg l_1, l_2\}$ a $\{\neg l_1, \neg l_2, l_3\}$. Odhadněte složitost tohoto algoritmu v závislosti na počtu proměnných n ve formuli.