1.3.7 3-dimenzionální párovaní. Jsou dány tři konečné množiny X, Y a W o stejném počtu prvků q, které jsou po dvou disjunktní. Dále je dána množina tříprvkových množin $\mathcal{S} \subseteq \{\{x, y, w\} \,|\, x \in X, y \in Y, w \in W\}$.

Existuje $H \subseteq \mathcal{S}$ taková, že H je rozklad množiny $X \cup Y \cup W$? Jinými slovy, podmnožina H množiny $\mathcal{S}$, která má q prvků a každé dvě množiny z H jsou disjunktní?

1.3.8 Tvrzení. Úloha 3-dimenzionálního párovnání je $\mathcal{NP}$-úplná úloha.

Zdůvodnění: Úloha 3-dimenzionálního párování leží ve třídě $\mathcal{NP}$; ano, máme-li podmnožinu H, je možné ověřit, že má pozadované vlastnosti v polynomiálním čase.

Ukážeme, že se 3-CNF SAT polynomiálně redukuje na úlohu 3-dimenzionálního párování.

Vezměme libovolnou formuli $\varphi = C_1 \vee \ldots \vee C_k$, kde každá klauzule se skládá ze tří literálů. Označme $\{u_1, \ldots, u_n$ všechny logické proměnné obsažené ve formuli φ.

Definujeme množiny W, X, Y a $\mathcal{S}$ takto:

- $W = \{u_i[j], \overline{u}_i[j] \,|\, i = 1, \ldots, n, j = 1, \ldots, k\}$.
- $X = \{a_i[j], s[j], z[s] \,|\, i = 1, \ldots, n, j = 1, \ldots, k, s = 1, \ldots, k(n-1)\}$.
- $Y = \{b_i[j], t[j], v[s] \,|\, i = 1, \ldots, n, j = 1, \ldots, k, s = 1, \ldots, k(n-1)\}$.
- $\mathcal{S}$ se skládá z množin T_i^{true}, T_i^{false} pro každou logickou proměnnou u_i, množin K_j pro každou klauzuli C_j a množiny Q, kde
  - $T_i^{true} = \{\{\overline{u}_i[j], a_i[j+1], b[j]\} \,|\, j = 1, \ldots, k-1\} \cup \{\overline{u}_i[k], a_i[1], b_i[k]\}$.
  - $T_i^{false} = \{\{u_i[j], a_i[j], b[j]\} \,|\, j = 1, \ldots, k\}$.
  - $K_j = \{\{l[j], s[j], t[j]\} \,|\, l \text{ je literál } C_j\}$.
  - $Q = \{\{u_i[j], z[s], v[s]\}, \{\overline{u}_i[j], z[s], v[s]\} \,|\, i = 1, \ldots, n, j = 1, \ldots, k, s = 1, \ldots, k(n-1)\}$.

Dá se dokázat, že pro W, x, Y lze z $\mathcal{S}$ vybrat H s vlastnostmi z 1.3.7 právě tehdy, když je formule φ splnitelná.

1.3.9 Heuristiky. Jestliže je třeba řešit problém, který je $\mathcal{NP}$ úplný, musíme pro větší instance opustit myšlenku přesného nebo optimálního řešení a smířit se s tím, že získáme „dostatečně přesné“ nebo „dostatečně kvalitní“ řešení. K tomu se používají heuristické algoritmy pracující v polynomiálním čase. Algoritmům, kde umíme zaručit „jak daleko“ je nalezené řešení od optimálního, se také říká aproximační algoritmy.

1.3.10 Tvrzení. Kdyby existovala konstanta r a polynomiální algoritmus $\mathcal{A}$ takový, že pro každou instanci obchodního cestujícího I najde trasu délky $D \leq r\, OPT(I)$, kde $OPT(I)$ je délka optimální trasy instance I, pak

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

1.3.11 Zdůvodnění tvrzení 1.3.10. Za předpokladu tvrzení 1.3.10 bychom uměli polynomiálně vyřešit problém existence hamiltonovské kružnice. Naznačíme odpovídající převod.

Je dán neorientovaný graf $G = (V, E)$, $V = \{1, 2, \ldots, n\}$, a ptáme se, zda v něm exituje hamiltonovská kružnice. Zkonstruujeme instanci obchodního cestujícího takto: Pro města $\{1, 2, \ldots, n\}$ položíme

$$d(i,j) = \begin{cases} 1, & \{i,j\} \in E \\ r\,n + 1, & \{i,j\} \notin E \end{cases}$$

Trasa v instanci popsané výše může mít délku n, jestliže je tvořena všemi hranami délky 1. V tomto případě jsou všechny hrany hranami grafu G a trasa představuje hamiltonovskou kružnici. Nebo musí trasa mít délku alespoň $n - 1 + n\,r + 1 = n\,r + n$. To je v případě, že aspoň jedna spojnice v trase není tvořena hranou grafu G.

Tedy jestliže algoritmus $\mathcal{A}$ najde trasu délky jiné než n, pak v grafu G neexistuje hamiltonovská kružnice. Takto bychom polynomiálním algoritmem byli schopni rozhodnout existenci hamiltonovské kružnice. Protože existence hamiltonovské kružnice je $\mathcal{NP}$ úplný problém, platilo by $\mathcal{P} = \mathcal{NP}$.

1.3.12 Trojúhelníková nerovnost. Řekneme, že instance obchodního cestujícího splňuje trojúhelníkovou nerovnost, jestliže pro každá tři města i, j, k platí:

$$d(i,j) \leq d(i,k) + d(k,j).$$

1.3.13 Tvrzení. Jestliže instance I obchodního cestujícího splňuje trojúhelníkovou nerovnost, pak existuje polynomiální algoritmus $\mathcal{A}$, který pro I najde trasu délky D, kde $D \leq 2\,OPT(I)$ ($OPT(I)$ je délka optimální trasy v I).

1.3.14 Slovní popis algoritmu z tvrzení 1.3.13. Instanci I považujeme za úplný graf G s množinou vrcholů $V = \{1, 2, \ldots, n\}$ a ohodnocením d.

1. V grafu G najdeme minimální kostru (V, K).
2. Kostru (V, K) prohledáme do hloubky z libovolného vrcholu.
3. Trasu T vytvoříme tak, že vrcholy procházíme ve stejném pořadí jako při prvním navštívení během prohledávání grafu. T je výstupem algoritmu.

Zřejmě platí, že délka kostry K je menší než $OPT(I)$. Ano, vynecháme-li z optimální trasy některou hranu, dostaneme kostru grafu G. Protože K je minimální kostra, musí být délka K menší než $OPT(I)$ (předpokládáme, že vzdálenosti měst jsou kladné). Vzhledem k platnosti trojúhelnikové nerovnosti, je délka T menší nebo rovna dvojnásobku délky kostry K.

1.3.15 Christofidesův algoritmus. Jestliže instance I obchodního cestujícího splňuje trojúhelníkovou nerovnost, pak následující algoritmus najde trasu T délky D takovou, že $D \leq \frac{3}{2}\,OPT(I)$.

Instanci I považujeme ze úplný graf G s množinou vrcholů $V = \{1, 2, \ldots, n\}$ a ohodnocením d.

1. V grafu G najdeme minimální kostru (V, K).
2. Vytvoříme úplný graf H na množině všech vrcholů, které v kostře (V, K) mají lichý stupeň.
3. V grafu H najdeme nejlevnější perfektní párování P.
4. Hrany P přidáme k hranám K minimální kostry. Graf $(V, P \cup K)$ je eulerovský graf. V grafu $(V, P \cup K)$ sestrojíme uzavřený eulerovský tah.
5. Trasu T získáme z eulerovského tahu tak, že vrcholy navštívíme v pořadí, ve kterém jsme do nich poprvé vstoupili při tvorbě eulerovského tahu.

Platí, že délka takto vzniklé trasy je maximálně $\frac{3}{2}$ krát větší než délka optimální trasy.

1.3.16 Poznámka. Odhad délky trasy, kterou jsme získali v 1.3.14, i odhad pro trasu získanou Christofidesovým algoritmem není možné zlepšit.