10. Převody problémů *(zapsali Martin Chytil, Vladimír Kudelas)*

Na této přednášce se budeme zabývat rozhodovacími problémy a jejich obtížností. Za jednoduché budeme trochu zjednodušeně považovat ty problémy, na něž známe algoritmus pracující v polynomiálním čase.

Definice: *Rozhodovací problém* je takový problém, jehož výstupem je vždy ANO, nebo NE. [Formálně bychom se na něj mohli dívat jako na množinu L vstupů, na které je odpověď ANO, a místo $L(x) =$ ANO psát prostě $x \in L$.]

Příklad: Je dán bipartitní graf G a $k \in \mathbb{N}$. Existuje v G párování, které obsahuje alespoň k hran?

To, co bychom ve většině případů chtěli, je samozřejmě nejen zjistit, zda takové párování existuje, ale také nějaké konkrétní najít. Všimněme si ale, že když umíme rozhodovat existenci párování v polynomiálním čase, můžeme ho polynomiálně rychle i najít:

Mějme černou skříňku (fungující v polynomiálním čase), která odpoví, zda daný graf má nebo nemá párování o k hranách. Odebereme z grafu libovolnou hranu a zeptáme se, jestli i tento nový graf má párovaní velikosti k. Když má, pak tato hrana nebyla pro existenci párování potřebná, a tak ji odstraníme. Když naopak nemá (hrana patří do každého párování požadované velikosti), tak si danou hranu poznamenáme a odebereme nejen ji a její vrcholy, ale také hrany, které do těchto vrcholů vedly. Toto je korektní krok, protože v původním grafu tyto vrcholy byly navzájem spárované, a tedy nemohou být spárované s žádnými jinými vrcholy. Na nový graf aplikujeme znovu tentýž postup. Výsledkem je množina hran, které patří do hledaného párování. Hran, a tedy i iterací našeho algoritmu, je polynomiálně mnoho a skříňka funguje v polynomiálním čase, takže celý algoritmus je polynomiální.

A jak náš rozhodovací problém řešit? Nejsnáze tak, že ho převedeme na jiný, který už vyřešit umíme. Tento postup jsme (právě u hledání párování) už použili v kapitole o Dinicově algoritmu. Vytvořili jsme vhodnou síť, pro kterou platilo, že v ní existuje tok velikosti k právě tehdy, když v původním grafu existuje párování velikosti k.

Takovéto převody mezi problémy můžeme definovat obecně:

Definice: Jsou-li A, B rozhodovací problémy, pak říkáme, že A lze *redukovat* (neboli převést) na B (píšeme $A \to B$) $\Leftrightarrow$ existuje funkce f spočitatelná v polynomiálním čase taková, že pro $\forall x : A(x) = B(f(x))$.

Všimněte si, že $A \to B$ také znamená, že problém B je alespoň tak těžký jako problém A (tím myslíme, že pokud lze B řešit v polynomiálním čase, lze tak řešit i A): Nechť problém B umíme řešit v čase $\mathcal{O}(b^k)$, kde b je délka jeho vstupu. Nechť dále funkce f převádějící A na B pracuje v čase $\mathcal{O}(a^\ell)$ pro vstup délky a. Spustíme-li tedy $B(f(x))$ na nějaký vstup x problému A, bude mít $f(x)$ délku $\mathcal{O}(a^\ell)$, takže $B(f(x))$ poběží v čase $\mathcal{O}(a^{k\ell})$, což je polynomiální v délce vstupu a.

Pozorování: Převoditelnost je:

- reflexivní (úlohu můžeme převést na tu stejnou identickým zobrazením): $A \to A$,
- tranzitivní: Je-li $A \to B$ funkcí f, $B \to C$ funkcí g, pak $A \to C$ složenou funkcí $g \circ f$ (složení dvou polynomiálních funkcí je zase polynomiální funkce, jak už jsme zpozorovali v předchozím odstavci).

Podívejme se nyní na převody mezi dalšími zajímavými problémy:

1. problém: SAT

Splnitelnost logických formulí, tj. dosazení *true* či *false* za proměnné v logické formuli tak, aby formule dala výsledek *true*.

Zaměříme se na speciální formu zadání formulí, *konjunktivní normální formu* (CNF).

$$(\ldots \vee \ldots \vee \ldots \vee \ldots) \wedge (\ldots \vee \ldots \vee \ldots \vee \ldots) \wedge \ldots$$

Vstup: Formule φ v konjunktivní normální formě.

Výstup: $\exists$ dosazení *true* a *false* za proměnné takové, že hodnota formule $\varphi(\ldots) =$ *true*.

Pro formuli platí následující podmínky:

- *formule* je zadána pomocí *klauzulí* oddělených $\wedge$,
- každá *klauzule* je složená z *literálů* oddělených $\vee$,
- každý *literál* je buďto proměnná nebo její negace.

Ukážeme, že stačí vyřešit jednodušší problém 3-SAT.

2. problém: 3-SAT

Definice: 3-SAT je takový SAT, v němž každá klauzule obsahuje nejvýše tři literály.

Převod 3-SAT na SAT: Vstup není potřeba nijak upravovat, 3-SAT splňuje vlastnosti SATu, proto 3-SAT $\to$ SAT (3-SAT je alespoň tak těžký jako SAT)

Převod SAT na 3-SAT: Musíme formuli převést tak, abychom neporušili splnitelnost.

Trik pro dlouhé klauzule: Každou klauzuli

$$(\alpha \vee \beta), \text{ tž. } |\alpha| + |\beta| \geq 4$$

přepíšeme na:

$$(\alpha \vee x) \wedge (\beta \vee \neg x),$$

kde x je nová proměnná, kterou nastavíme tak, abychom neovlivnili splnitelnost formule.

Platí-li:

- $\alpha \Rightarrow x = 0$ (zajistí splnění druhé poloviny nové formule),
- $\beta \Rightarrow x = 1$ (zajistí splnění první poloviny nové formule),
- $\alpha, \beta / \neg\alpha, \neg\beta \Rightarrow x = 0/1$ (je nám to jedno, celkové řešení nám to neovlivní).

Tento trik opakujeme tak dlouho, dokud je to třeba.

Nabízí se otázka, proč můžeme proměnnou x nastavit, jak se nám zlíbí. Vysvětlení je prosté, proměnná x nám původní formuli nijak neovlivní, protože se v ní nevyskytuje, proto ji můžeme nastavit tak, jak chceme.

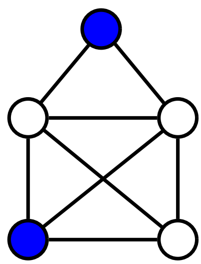
Poznámka: U 3-SAT lze vynutit právě tři literály, pro krátké klauzule použijeme následující trik:

$$(\alpha) \to (\alpha \vee x) \wedge (\alpha \vee \neg x).$$

3. problém: Hledání nezávislé množiny v grafu

Existuje nezávislá množina vrcholů z G velikosti alespoň k?

Definice: *Nezávislá množina* (NzMna) budeme říkat každé množině vrcholů grafu takové, že mezi nimi nevede žádná hrana.



Příklad nezávislé množiny

Vstup: Neorientovaný graf G, $k \in \mathbb{N}$.

Výstup: $\exists A \subseteq V(G)$, $|A| \geq k$: $\forall u, v \in A \Rightarrow uv \notin E(G)$?

Poznámka: Každý graf má minimálně jednu nezávislou množinu, a tou je prázdná množina. Proto je potřeba zadat i minimální velikost hledané množiny.

Ukážeme, jak na tento probém převést 3-SAT.

Převod: Z každé klauzule vybereme jeden literál tak, abychom v různých klauzulích nevybírali konfliktně, tj. x a $\neg x$.

Příklad: $(x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee \neg y \vee p)$.

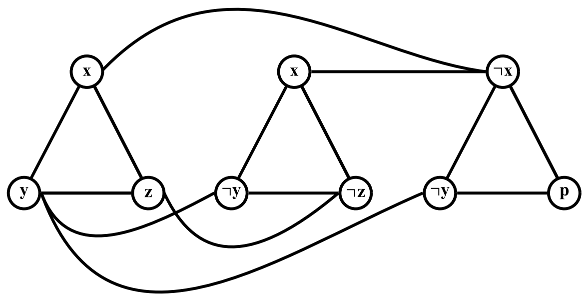
Pro každou klauzuli sestrojíme graf (trojúhelník) a přidáme „konfliktní“ hrany, tj. x a $\neg x$.

Princip je takový, že z každé klauzule si vybereme literál, který danou klauzuli splní, a to tak, aby literály, které si vybereme, nekolidovaly. Kolize ošetříme hranami mezi proměnnými a jejich negacemi.

Existuje nezávislá množina velikosti rovné počtu klauzulí? Pokud ano, tak dostaneme seznam proměnných, pomocí kterých splníme danou formuli.

Převod NzMna na SAT: Máme proměnné $v_1, \ldots, v_n$ pro vrcholy.

Nyní ukážeme, jak převést problém hledání nezávislé množiny, na SAT.



Ukázka převodu 3-SAT na nezávislou množinu

- Pořídíme si proměnné $v_1, \ldots, v_n$ odpovídající vrcholům grafu. Proměnná v_i bude indikovat, zda se i-tý vrchol vyskytuje v nezávislé množině.
- Pro každou hranu $ij \in E(G)$ přidáme klauzuli $(\neg v_i \vee \neg v_j)$. Tyto klauzule nám ohlídají, že vybraná množina je vskutku nezávislá.
- Ještě potřebujeme zkontrolovat, že je množina dostatečně velká, takže si její prvky očíslujeme čísly od 1 do k. Očíslování popíšeme maticí proměnných x_{ij}, přičemž x_{ij} bude pravdivá právě tehdy, když v pořadí i-tý prvek nezávislé množiny je vrchol v_j.
- Přidáme tedy klauzuje, které nám řeknou, že vybrané do nezávislé množiny jsou právě ty vrcholy, které jsou touto maticí očíslované: $\forall i, j, x_{ij} \Rightarrow v_j$.
- Ještě potřebujeme zajistit, aby byla v každém řádku i sloupci nejvýše jedna jednička: $\forall j, i, i^{'}, i \neq i^{'} : x_{ij} \Rightarrow \neg x_{i^{'}j}$ a $\forall i, j, j^{'}, j \neq j^{'} : x_{ij} \Rightarrow \neg x_{ij^{'}}$.
- A nakonec si ohlídáme, aby v každém řádku byla alespoň jedna jednička, klauzulí $\forall i : x_{i1} \vee x_{i2} \vee \ldots \vee x_{in}$.

Příklad matice: Jako příklad použijeme nezávislou množinu z ukázky nezávislé množiny. Nechť jsou vrcholy grafu očíslované zleva a ze zhora. Hledáme nezávislou množinu velikosti 2. Matice pak bude vypadat následovně:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Vysvětlení: Jako první vrchol množiny bude vybrán vrchol v_1, proto v prvním řádku a v prvním sloupci bude 1. Jako druhý (k-tý) vrchol množiny bude vybrán vrchol v_4, proto na druhém (k-tém) řádku a ve čtvrtém sloupci bude 1. Na ostatních místech bude 0.

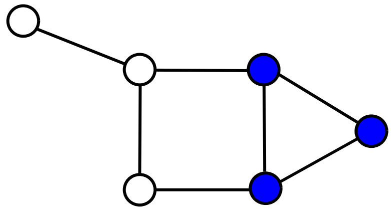
4. problém: Klika

Vstup: Graf $G, k \in N$.

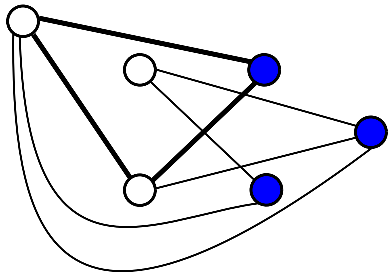
Výstup: $\exists$ úplný podgraf grafu G na k vrcholech?

Převod: Prohodíme v grafu G hrany a nehrany $\Rightarrow$ hledání nezávislé množiny.

Důvod: Pokud existuje úplný graf na k vrcholech, tak v „invertovaném“ grafu tyto vrcholy nejsou spojeny hranou, tj. tvoří nezávislou množinu.



Příklad kliky



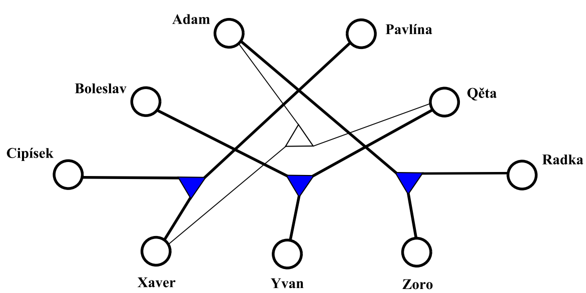
Prohození hran a nehran

5. problém: 3D párování (3D matching)

Vstup: Tři množiny, např. K (kluci), H (holky), Z (zvířátka) a množina kompatibilních trojic (těch, kteří se spolu snesou).

Výstup: Perfektní podmnožina trojic - tj. taková podmnožina trojic, která obsahuje všechna K, H a Z.

Ukážeme, jak tento problém převést na 3,3-SAT (ovšem to až na další přednášce).



Ukázka 3D párování

6. problém: 3,3-SAT

Definice: 3,3-SAT je speciální případ 3-SATu, kde každá proměnná se vyskytuje v maximálně třech literálech.

Převod 3-SAT na 3,3-SAT: Pokud se proměnná x vyskytuje v $k > 3$ literálech, tak nahradíme výskyty novými proměnnými $x_1, \dots, x_k$ a přidáme klauzule:

$$(\neg x_1 \vee x_2)(\neg x_2 \vee x_3)(\neg x_3 \vee x_4)\dots(\neg x_{k-1} \vee x_k)(\neg x_k \vee x_1),$$

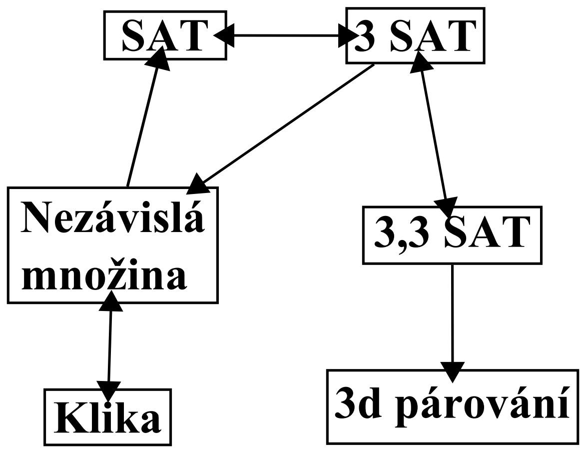
což odpovídá:

$$(x_1 \Rightarrow x_2)(x_2 \Rightarrow x_3)(x_3 \Rightarrow x_4)\dots(x_{k-1} \Rightarrow x_k)(x_k \Rightarrow x_1).$$

Tímto zaručíme, že všechny nové proměnné budou mít stejnou hodnotu.

Mimochodem, můžeme rovnou zařídit, že každý literál se vyskytuje nejvíce dvakrát (tedy že každá proměnná se vyskytuje alespoň jednou pozitivně a alespoň jednou negativně). Pokud by se nějaká proměnná nějaká proměnná objevila ve třech stejných literálech, můžeme na ni také použít náš trik a nahradit ji třemi proměnnými. V nových klauzulích se pak bude vyskytovat jak pozitivně, tak negativně.

Závěr: Obrázek ukazuje problémy, jimiž jsme se dnes zabývali, a vztahy mezi těmito problémy.



Převody mezi problémy