České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra teoretické informatiky

Ondřej Suchý

MI-PAM, Přednáška č. 2

Vylepšený prohledávací strom pro Vrcholové pokrytí, předzpracování, kernelizace

Letní semestr 2016/17 1. března 2017 (Verze dokumentu: 10. 3. 2017 17:00)

Domácí úkol. 1 Na minulé přednášce jsme zadali úkol s PlusFPT, potřebovat ho budeme ale v následující variantě:

> Označme PlusPlusFPT třídu všech parametrizovaných problémů, pro které existuje algoritmus pracující v čase $f''(k) + (k + |x|)^{c''}$. Ukažte, že pro každý parametrizovaný problém L platí, že L je v FPT právě tehdy, když je v PlusPlusFPT.

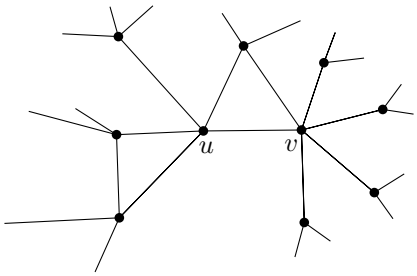
I za tuto variantu můžete dostat **do 8.3.2017** 2 body, ovšem pokud vyřešíte obě varianty, dostanete celkem jen 3, nikoliv 4.

1 Zpět k vrcholovému pokrytí

Připomeňme si $O(2^k \cdot n^2)$ algoritmus pro Vrcholové pokrytí:

```
Pokrytí (G = (V, E), k){
If k < 0 return NO;
If E(G) = ∅ return YES;
If k = 0 return NO;
Nechť {u, v} ∈ E(G) je libovolná hrana;
return (Pokrytí(G \ u, k − 1) || Pokrytí(G \ v, k − 1));
}
```

- V první větvi rekurze zkoumáme **všechna** možná řešení obsahující u – tedy i to, které obsahuje zároveň v.
- V druhé větvi se tedy můžeme omezit na ta řešení, která určitě neobsahují u.



- Když u v řešení není – nedokážeme říct něco víc, než že tam musí být v?
- Ano! Musí tam být všichni sousedé u.

Poslední řádek algoritmu tedy můžeme přepsat na

`return (Pokrytí(`$G \setminus u, k-1$`) || Pokrytí(`$G \setminus N(u), k - \deg(u)$`));`

Jak to změní časovou složitost?

Pokud první volání je s parametrem k a každá rekurze proběhne nad vrcholem stupně $\deg(u)$, pak pro počet listů $T(k)$ ve stromě rekurzivních volání platí předpis:

$$\begin{aligned} T(k) &\leq T(k-1) + T(k - \deg(u)) && k \geq 1 \\ T(k) &= 1 && k \leq 0 \end{aligned}$$

Říká se, že algoritmus má *branching (větvící) vektor* $(1, \deg(u))$ ... to o kolik se sníží parametr v jednotlivých větvích.

My ale máme v průběhu různé stupně.

Představme si, že bychom chtěli indukcí dokázat, že $T(k) \leq \lambda^k$ pro nějaké $\lambda > 1$.
Co musí λ splňovat?
$T(0) \leq 1 = \lambda^0$ - splněno vždy
(*Ano, tady tak trochu podvádíme, potřebovali bychom základní krok pro všechny $k \leq 0$. Lze to vyřešit, ale zbytečně by se to tím teď komplikovalo. Kdyžtak vysvětlím.*)

Indukční krok:

- Víme pro všechna $k' < k$, že $T(k') \leq \lambda^{k'}$.
- Chceme ukázat, že $T(k) \leq \lambda^k$.
- Víme, že $T(k) \leq T(k-1) + T(k - \deg(u)) \leq \lambda^{k-1} + \lambda^{k-\deg(u)}$.
- Potřebujeme, aby z toho bylo jasné, že $T(k) \leq \lambda^k$, tedy aby platilo $\lambda^{k-1} + \lambda^{k-\deg(u)} \leq \lambda^k$.
- λ tedy zvolíme nejmenší možné tak, aby $\lambda^{k-1} + \lambda^{k-\deg(u)} \leq \lambda^k$ platilo pro všechny stupně $\deg(u)$, které používáme.
- Nerovnost lze upravit na tvar $1 \geq \lambda^{-1} + \lambda^{-\deg(u)}$
- Vyřešíme pro každý stupeň nerovnost s rovností a nejhorší λ, které nám vyjde určuje časovou složitost algoritmu.

Máme-li odhad na počet listů, pak odhad na počet všech volání je $2\times$ počet listů, protože strom rekurze je binární. Pro odhad složitosti pak stačí vynásobit časem stráveným uvnitř každého volání – obvykle polynomiální.

Povšimneme si, že λ musí být tím větší, čím menší je stupeň vrcholu na kterém se větvíme.

Algoritmus tedy poběží rychleji, když budeme nejdříve brát vrcholy co největšího stupně (na ty s malým stupněm často ani nedojde).

Pokud jsme takto donuceni vzít vrchol stupně 2, znamená to, že celý graf je maximálního stupně 2. Jak ale takový graf vypadá?

Ano, je to disjunktní sjednocení cest a kružnic.

V takovém grafu přece dokážeme vyřešit Vrcholové pokrytí v polynomiálním čase (dokonce lineárním), takže není potřeba se dál větvit.

Nejhorší případ je tedy, že se větvíme na vrcholu stupně 3, branching vektor $(1, 3)$ (Na grafech maximálního stupně 3 je Vrcholové pokrytí skutečně stále ještě NP-úplné.).

Získáme tedy rovnici $1 = \lambda^{-1} + \lambda^{-3}$ jejíž jediným (reálným) řešením je $\lambda \doteq 1.4656$.

Algoritmus běží v čase $O(1.4656^k \cdot k \cdot n) = O(1.466^k \cdot n)$.

Domácí úkol. 5 *(4 body)* Vymyslete algoritmus, který rozhodne 3-SAT s n proměnými v čase $O((2-\varepsilon)^n)$ pro nějaké $\varepsilon > 0$.

Domácí úkol. 6 Pro nějakou pevně zvolenou konstantu d uvažte následující problém:

> d-Hitting Set
> **Vstup:** Množina prvků U, systém jejích podmnožin $\mathcal{F} \subseteq 2^U$ takový, že každá množina je velikosti nejvýše d, k nezáporné celé.
> **Otázka:** Existuje S podmnožina U taková, že každá množina A z $\mathcal{F}$ obsahuje aspoň jeden prvek z S (tedy $S \cap A \neq \emptyset$)?

Ukažte, že tento problém je FPT vzhledem k velikosti řešení k.
(3 body, ***pouze do 8.3.****)*

2 Předzpracování

Co je to předzpracování dat?

- např. seřazení.

Chtěli bychom najít takový algoritmus, který nás v polynomiálním čase zbaví nezajímavých částí instance a zachová odpověď. Zároveň bychom chtěli nějak zaručit, že funguje efektivně, tj. zejména, že něco opravdu dělá a někdy pomůže. Jak na to?

Může existovat algoritmus, který v **polynomiálním čase** z instance (např.) Vrcholového pokrytí (G, k) vyrobí jinou instanci (G', k') tak, že

- (G', k') je ekvivalentní s (G, k) *(viz následující definice)*
- a $|V(G')| < |V(G)|$?

Definice. Instance (x, k) a (x', k') parametrizovaného problému L jsou *ekvivalentní*, pokud $(x, k) \in L \Leftrightarrow (x', k') \in L$.

Ne, pokud P≠NP, takový algoritmus existovat nemůže! Iterací takového algoritmu bychom se mohli v polynomiálním čase dostat vždy ke grafu s jedním vrcholem, pro který bychom snadno v konstantním čase rozhodli. To znamená, že bychom vyřešili NP-úplný problém (např. Vrcholového pokrytí) v polynomiálním čase.

Ukážeme si, jak definovat efektivní předzpracování pomocí parametrizovaných problémů tak, aby skutečně mohlo existovat.

Budeme hledat části instance, které se dají snadno zjednodušit, nahradit něčím menším, nebo smazat.

Představme si například, že máme instanci Vrcholového pokrytí, takovou, že v grafu je izolovaný vrchol.

Pravidlo. 1 pro Vrcholové pokrytí.
Je-li v izolovaný v G pak odeber v z G.
Jinými slovy: Pokračuj s instancí (G', k') takovou, že $G' = G - v$ a $k' = k$.

Pozorování. *Výsledná instance je ekvivalentní s původní.*

Definice. Řekneme, že pravidlo je *korektní*, pokud je výsledná instance ekvivalentní s původní instancí.

Co lze říci, pokud máme instanci (G, k) Vrcholového pokrytí, takovou, že v G je vrchol stupně $\geq k + 1$?

Takový vrchol vždy musí být v řešení.

Pravidlo. 2 pro Vrcholové pokrytí.
Je-li v stupně aspoň $k + 1$ v G pak odeber v z G a sniž k o jedna.
Jinými slovy: Pokračuj s instancí (G', k') takovou, že $G' = G - v$ a $k' = k - 1$.

Lemma. *Pravidlo 2 je korektní.*

Důkaz. "⇒" – pokud je (G, k) yes-instance, pak i (G', k'):
Nechť S je vrcholové pokrytí velikosti $\leq k$ v G. Protože $|N(v)| \geq k+1$, existuje $u \in N(v) \setminus S$. Protože hrana $\{u, v\}$ musí být pokryta, platí $v \in S$. Označme $S' = S \setminus \{v\}$, máme $|S'| \leq k - 1 = k'$. Protože S je pokrytí G, obsahuje $G \setminus S$ jen izolované vrcholy a tedy $G' \setminus S' = G \setminus S$ také a S' je pokrytí G'.

"⇐": Nechť S' je vrcholové pokrytí G' velikosti $\leq k'$ a označme $S = S' \cup \{v\}$. Pak $|S| \leq k$ a protože S' je pokrytí G', obsahuje $G' \setminus S'$ jen izolované vrcholy a tedy $G \setminus S = G' \setminus S'$ také a S je pokrytí G. □

Definice. Řekneme, že je instance *redukovaná* vzhledem k nějaké sadě pravidel, pokud žádné z těchto pravidel nelze na tuto instanci použít (případně jejich použití instanci nezmění).

Věta (Buss ≤ 1993)**.** *Je-li (G, k) redukovaná vzhledem k Pravidlům 1 a 2 a má-li G více než k^2 hran nebo více než $k^2 + k$ vrcholů, pak (G, k) je no-instance.*

Důkaz. Předpokládejme, že (G, k) je yes-instance. Ukážeme $|E(G)| \leq k^2$ a $|V(G)| \leq k^2 + k$. Nechť S je nějaké vrcholové pokrytí velikosti $\leq k$ pro G. Každý vrchol z S má stupeň nejvýše k, protože Pravidlo 2 nelze použít ((G, k) redukovaná). Tedy S je incidentní s nejvýše k^2 hranami a každá hrana G musí být incidentní s něčím z S. Tedy $|E(G)| \leq k^2$. Vrcholy z S navíc sousedí s nejvýše k^2 vrcholy a každý vrchol z $V \setminus S$ musí sousedit s něčím z S protože není izolovaný (Pravidlo 1 nelze použít). Tedy $|V| = |V \setminus S| + |S| \leq k^2 + k$. □

Definice. *Kernelizace* pro parametrizovaný problém L je algoritmus A, který na vstup dostane instanci (x, k) problému L a vrátí instanci (x', k') takovou, že:

- (x', k') je ekvivalentní s (x, k),
- A běží v čase polynomiálním vzhledem k $(|x| + k)$
- a $|x'| \leq g(k), k' \leq g(k)$ pro nějakou $g(k)$, která závisí pouze na k.

Instance (x', k') se nazývá *jádro (kernel)* problému a $g(k)$ jeho velikost.

Poznámka. *Když řekneme*

- polynomiální kernel *myslíme tím, že jeho velikost $g(k)$ je polynomiální.*
- lineární kernel *myslíme tím, že jeho velikost $g(k)$ je lineární (ve starší literatuře někdy ovšem i pokud jen $|V(G)| = O(k)$).*

Čas je polynomiální vždy!

Příklad. Uvažme následující algoritmus:

1. Aplikuj Pravidla 1 a 2 dokud je možné některé z nich aplikovat.
2. Pokud má G více než k^2 hran nebo více než $k^2 + k$ vrcholů, vrať nějakou pevně zvolenou no-instanci konstantní velikosti.
3. Jinak vrať (G, k).

Tento algoritmus je kernelizací pro Vrcholové pokrytí a velikost kernelu je $O(k^2)$ (Přesněji k zápisu výsledku můžeme potřebovat až $\Theta(k^2 \log k)$ bitů, ale tento drobný rozdíl se většinou příliš nezdůrazňuje.).

Domácí úkol. 7 Uvažte následující problém:

> Point Line Cover
> **Vstup**: n bodů v rovině, k nezáporné celé.
> **Otázka**: Existuje k přímek v rovině tak, že každý ze zadaných bodů leží na alespoň jedné z těchto přímek?

Ukažte, že tento problém má kernel s $O(k^2)$ body (nejedná se o kernel v pravém slova smyslu, neboť neomezujeme délky zápisu (souřadnice) jednotlivých bodů).
(3 body)

Domácí úkol. 8 Uvažte následující problém:

> d-Bounded Degree Deletion
> **Vstup**: Graf G, k nezáporné celé, d nezáporné celé.
> **Otázka**: Lze z G smazat nejvýše k vrcholů tak, aby ve výsledném grafu byl maximální stupeň nejvýše d?

Ukažte, že tento problém má kernel velikosti $(k+d)^{O(1)}$.
(4 body)

Domácí úkol. 9 Uvažte následující problém:

> Dělení Množin
> **Vstup**: Množina prvků U, systém jejích podmnožin $\mathcal{F} \subseteq 2^U$, k nezáporné celé.
> **Otázka**: Existuje obarvení množiny U pomocí dvou barev takové, že alespoň k množin z $\mathcal{F}$ obsahuje prvky obou barev?

Ukažte, že tento problém má kernel s $2k$ množinami a universem U s $O(k^2)$ prvky (nebo alespoň jednu z podmínek za méně bodů).
(5 bodů)

Věta. *Rozhodnutelný parametrizovaný problém je v FPT právě tehdy, když pro něj existuje kernelizace.*

Důkaz. "$\Leftarrow$": Provedu kernelizaci, získám instanci (x', k'). Spustím algoritmus (nějaký existuje, problém je rozhodnutelný) na (x', k'). Algoritmus skončí za čas, který závisí na $|x'|$ a k', ale $|x'| \leq g(k)$ a $k' \leq g(k)$, takže vlastně závisí jen na k. Celkový čas je $h(g(k), g(k)) + (k + |x|)^{O(1)}$ takže problém náleží PlusPlusFPT `:-)`.
"$\Rightarrow$": $\in$ FPT $\Rightarrow$ existuje algoritmus v čase $f(k) \cdot |x|^c$. Pustíme ho na čas $|x|^{c+1}$:

- Když odpoví, tak vrátíme nějakou konstantně velkou instanci se stejnou odpovědí.
- Když to nestihne, tak $f(k) \cdot |x|^c > |x|^{c+1}$, tedy $|x| < f(k)$ a (x, k) je kernel. □

Poznámka. *Výsledný kernel nejspíše nebude polynomiální.*
$\Rightarrow$Zajímavé kernely jsou ty polynomiální $\Rightarrow$„Má kernel“= obvykle „má polynomiální kernel“.