Základy složitosti a vyčíslitelnosti

NTIN090

Petr Kučera

2021/22 (11. přednáška)

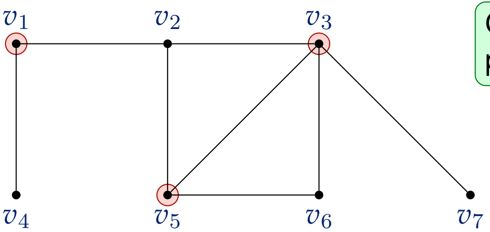
Parametrizované algoritmy

Vrcholové pokrytí

Vrcholové pokrytí (VP)

Instance: Neorientovaný graf $G = (V, E)$ a celé číslo $k \geq 0$.

Otázka: Existuje množina vrcholů $S \subseteq V$ velikosti nejvýš k, která obsahuje alespoň jeden vrchol z každé hrany $\{u, v\} \in E$ (tedy $\{u, v\} \cap S \neq \emptyset$)?



Graf s vrcholovým pokrytím velikosti $k = 3$

Jednoduchý algoritmus pro VRCHOLOVÉ POKRYTÍ

$G - u$ graf G po odstranění vrcholu u a incidentních hran

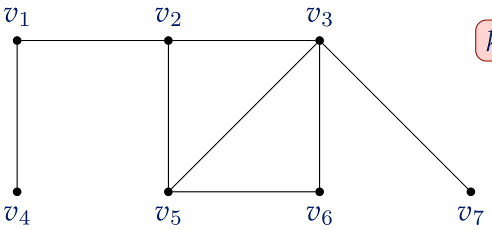
Funkce `SearchVP` (G, k)

Vstup: Graf $G = (V, E)$, $k \geq 0$
Výstup: Vrcholové pokrytí $S \subseteq V$, $|S| \leq k$ nebo NONE

if $E = \emptyset$ **then return** $\emptyset$
if $k = 0$ **then return** NONE
Vyber libovolnou hranu $\{u, v\} \in E$
if `SearchVP` $(G - u, k - 1)$ vrátí množinu S' **then**
    return $S = S' \cup \{u\}$
else if `SearchVP` $(G - v, k - 1)$ vrátí množinu S'' **then**
    return $S = S'' \cup \{v\}$
return NONE

Příklad práce `SearchVP` (G, k)

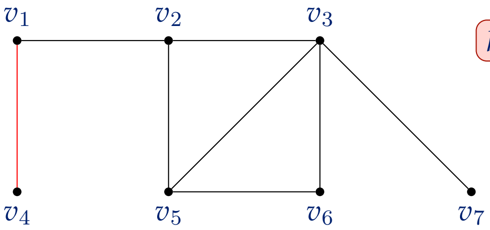
Uvažme následující graf



$k = 3$

Příklad práce `SearchVP` (G, k)

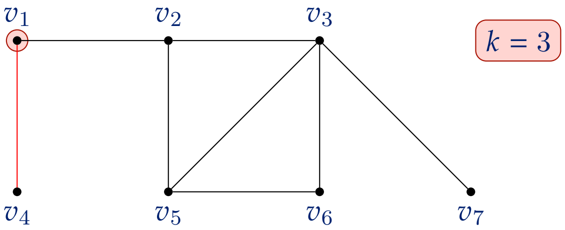
Vybereme hranu $\{v_1, v_4\}$



$k = 3$

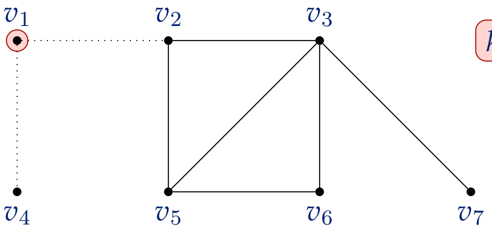
Příklad práce `SearchVP`(G, k)

Vybereme v_1 pro větvení



Příklad práce `SearchVP`(G, k)

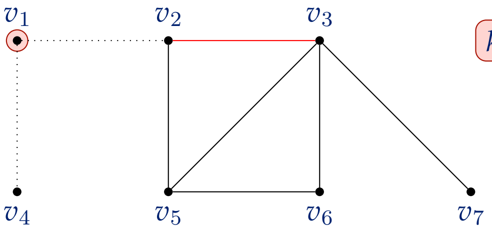
Smažeme v_1 a snížíme k o 1



$k = 2$

Příklad práce `SearchVP` (G, k)

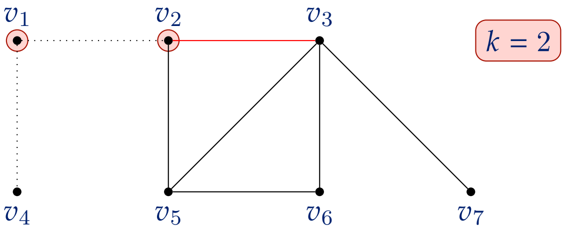
Vybereme hranu $\{v_2, v_3\}$



$k = 2$

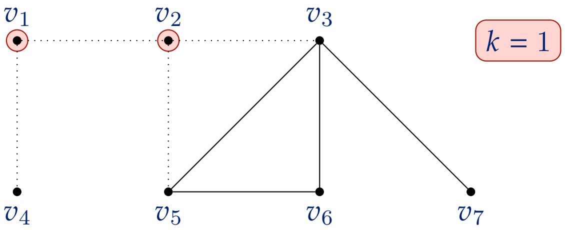
Příklad práce `SearchVP`(G, k)

Vybereme v_2 pro větvení



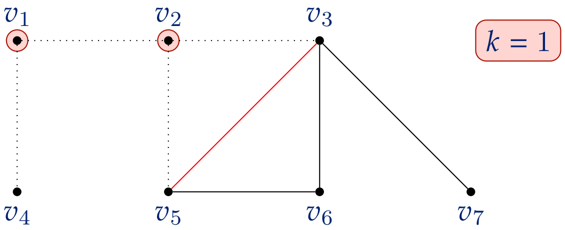
Příklad práce `SearchVP` (G, k)

Smažeme v_2 a snížíme k o 1



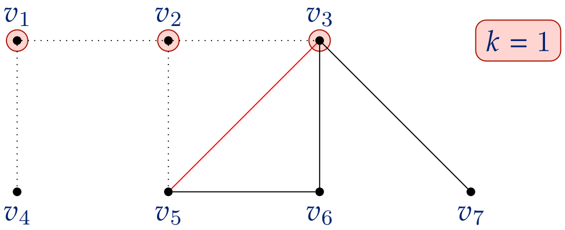
Příklad práce `SearchVP` (G, k)

Vybereme hranu $\{v_3, v_5\}$



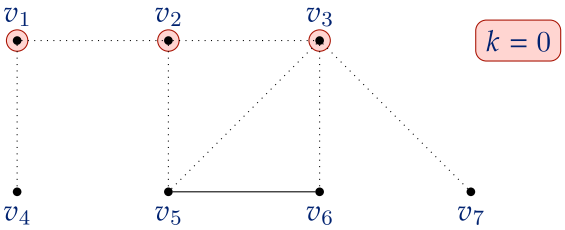
Příklad práce `SearchVP` (G, k)

Vybereme v_3 pro větvení



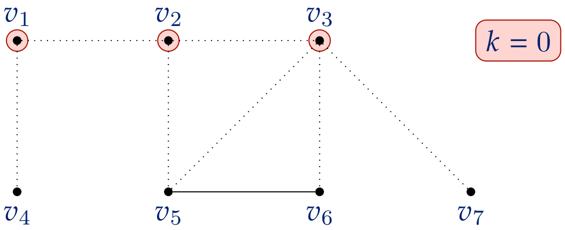
Příklad práce `SearchVP`(G, k)

Smažeme v_3 a snížíme k o 1



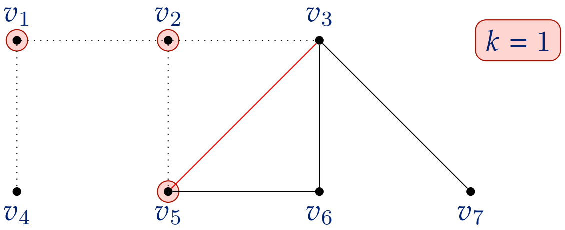
Příklad práce `SearchVP`(G, k)

$k = 0$ a v grafu zbývají hrany, je třeba vrátit se o krok zpět



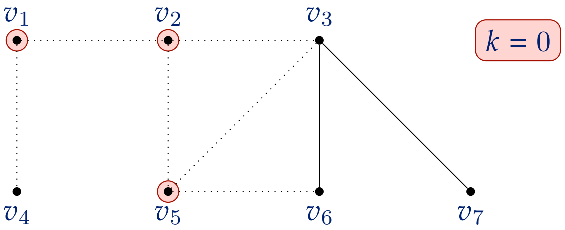
Příklad práce `SearchVP` (G, k)

Po návratu o krok zpět vybereme v_5



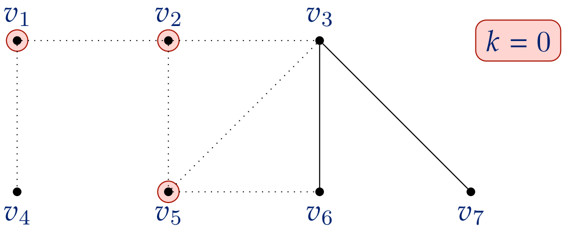
Příklad práce `SearchVP` (G, k)

Smažeme v_5 a snížíme k o 1



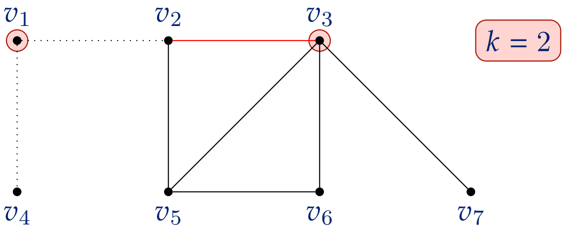
Příklad práce `SearchVP`(G, k)

I tato větev je neúspěšná, nyní je potřeba vrátit se o dva kroky zpět



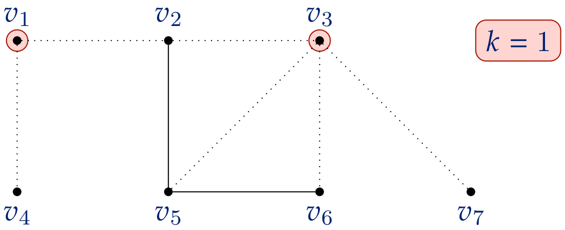
Příklad práce `SearchVP`(G, k)

Po návratu o krok zpět vybereme v_3



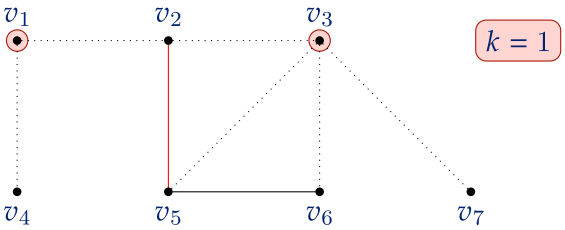
Příklad práce `SearchVP`(G, k)

Smažeme v_3 a snížíme k o 1



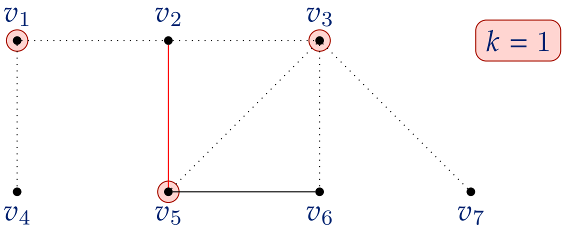
Příklad práce `SearchVP`(G, k)

Vybereme hranu $\{v_2, v_5\}$



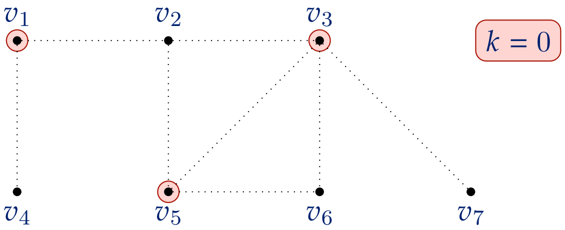
Příklad práce `SearchVP` (G, k)

Vybereme v_5 pro větvení



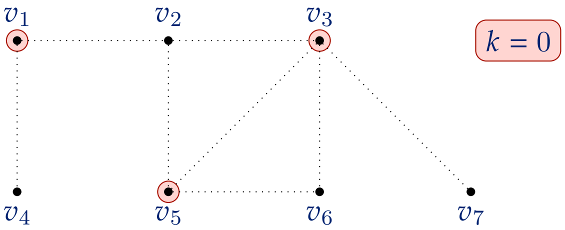
Příklad práce `SearchVP`(G, k)

Smažeme v_5 a snížíme k o 1



Příklad práce `SearchVP`(G, k)

Našli jsme vrcholové pokrytí velikosti 3



Korektnost `SearchVP`(G, k)

Lemma

`SearchVP`(G, k) najde vrcholové pokrytí G velikosti nejvýš k, právě když takové vrcholové pokrytí existuje.

Důkaz.

Indukcí podle k… □

Korektnost `SearchVP`(G, k) (základní krok indukce)

Lemma

`SearchVP`(G, k) *najde vrcholové pokrytí* G *velikosti nejvýš* k*, právě když takové vrcholové pokrytí existuje.*

$$\boxed{k = 0}$$

- $E = \emptyset \implies S = \emptyset$ je pokrytí velikosti 0
- $E \neq \emptyset \implies G$ nemá pokrytí velikosti 0

Korektnost `SearchVP`(G, k) (indukční krok)

Lemma

`SearchVP`(G, k) *najde vrcholové pokrytí* G *velikosti nejvýš* k*, právě když takové vrcholové pokrytí existuje.*

$$\boxed{k > 0}$$

- Uvažme libovolnou hranu $\{u, v\} \in E$
- Každé vrcholové pokrytí S obsahuje u nebo v
- G má vrcholové pokrytí velikosti nejvýš k, právě když
  - $G - u$ má vrcholové pokrytí velikosti nejvýš $k - 1$ nebo
  - $G - v$ má vrcholové pokrytí velikosti nejvýš $k - 1$

Časová složitost `SearchVP` (G, k)

Lemma

Předpokládejme, že $n = |V|$, $m = |E|$ a G je reprezentován seznamem sousedů. Pak `SearchVP` (G, k) pracuje v čase $O(2^k(m + n))$.

Důkaz.

- Hloubka rekurze je k
- Dvě rekurzivní volání v každé instanci
- Funkce `SearchVP()` je tedy volána nejvýš 2^k-krát
- Čas $O(m + n)$ stačí k výběru hrany a odstranění vrcholu □

- Exponenciální v parametru k, ale nikoli ve velikosti grafu G

VRCHOLOVÉ POKRYTÍ je řešitelné s pevným parametrem

Parametrizovaný problém

Definice

- **Parametrizovaný problém** je jazyk $L \subseteq \Sigma^* \times \mathbb{N}$, kde Σ je konečná abeceda
- k je **parametrem** instance $\langle I, k\rangle \in \Sigma^* \times \mathbb{N}$
- **Velikost instance** $\langle I, k\rangle$ definujeme jako

$$|\langle I, k\rangle| = |I| + k$$

Příklad

- Je-li $\langle G, k\rangle$ instance Vrcholového pokrytí, pak k je její parametr
- Můžeme uvažovat i jiné parametry
  - $|V| - k$
  - stromová šířka G

Problémy řešitelné s pevným parametrem

Definice

Parametrizovaný problém $L \subseteq \Sigma^* \times \mathbb{N}$ je řešitelný s pevným parametrem (fixed parameter tractable, FPT), právě když jej lze rozhodnout algoritmem $\mathcal{A}$, který pracuje v čase

$$O(f(k) \cdot |I|^c)$$

pro nějakou algoritmicky vyčíslitelnou funkci f a konstantu c.

- $\mathcal{A}$ zveme parametrizovaným algoritmem pro L
- FPT označuje třídu problémů řešitelných s pevným parametrem

Věta

Vrcholové pokrytí s parametrem k patří do FPT

O^*-notace

Definice

$O^*(f(k))$ označuje třídu funkcí $O(f(k) \cdot n^c)$ pro libovolnou konstantu c.

- Předpokládejme, že algoritmus $\mathcal{A}$ má časovou složitost $O(f(k) \cdot n^c)$
- Chceme zachytit závislost na parametru k
- Pomíjíme při tom polynomiální faktor

Příklad

`SearchVP`(G, k) pracuje v čase $O(2^k(n + m)) = O^*(2^k)$

Možné parametrizace VRCHOLOVÉHO POKRYTÍ

Obecné grafy

Parametr	Horní odhad
k	$O(1{,}28^k + kn)$
$n - k$	$\mathrm{W}[1]$-úplné (nejspíš není v FPT)
stromová šířka w grafu G	$2^w \cdot n$

Rovinné grafy

Parametr	Horní odhad
k	$O(c^{\sqrt{k}} + kn)$, $c \leq 2^{4\sqrt{3}} \approx 121{,}8$

- Rovinný graf G má vrcholové pokrytí velikosti nejvýš $\frac{3}{4}n$.
- Lze uvažovat parametr $\frac{3}{4}n - k$.
- Parametrizace nad/pod zaručenou hodnotu

Závislost času na funkci f

k	$f(k) = 2^k$	$f(k) = 1{,}32^k$	$f(k) = 1{,}28^k$
10	$\approx 10^3$	≈ 16	≈ 12
20	$\approx 10^6$	≈ 258	≈ 140
30	$\approx 10^9$	≈ 4140	≈ 1650
40	$\approx 10^{12}$	$\approx 6{,}6 \cdot 10^4$	$\approx 2{,}0 \cdot 10^4$
50	$\approx 10^{15}$	$\approx 1{,}1 \cdot 10^6$	$\approx 2{,}3 \cdot 10^5$
75	$\approx 10^{22}$	$\approx 1{,}1 \cdot 10^9$	$\approx 1{,}1 \cdot 10^8$
100	$\approx 10^{30}$	$\approx 1{,}1 \cdot 10^{12}$	$\approx 5{,}3 \cdot 10^{10}$
500	$\approx 10^{130}$	$\approx 1{,}4 \cdot 10^{60}$	$\approx 4{,}1 \cdot 10^{53}$

Kernelizace

Idea kernelizace

Chceme v polynomiálním čase zredukovat instanci $\langle I, k\rangle$ parametrizovaného problému L na instanci $\langle I', k'\rangle$, pro kterou platí

- $\langle I', k'\rangle$ je ekvivalentní $\langle I, k\rangle$

$$\langle I, k\rangle \in L \iff \langle I', k'\rangle \in L$$

- Velikost $\langle I', k'\rangle$ závisí jen na k
  - Pro nějakou algoritmicky vyčíslitelnou funkci $g(k)$ platí

$$|I'| + k' \leq g(k)$$

Výsledkem je kernel (jádro) velikosti $g(k)$.

Kernel

Definice

- $\mathcal{A}$ je kernelizační algoritmus (kernel) pro problém L, pokud
  - $\mathcal{A}$ pracuje v polynomiálním čase
  - Pro instance $\langle I, k\rangle$ a $\langle I', k'\rangle = \mathcal{A}(I, k)$ platí

$$\langle I, k\rangle \in L \iff \langle I', k'\rangle \in L$$

  - Existuje vyčíslitelná funkce $g(k)$ taková, že pro každé instance $\langle I, k\rangle$ a $\langle I', k'\rangle = \mathcal{A}(I, k)$ platí

$$|I'| + k' \leq g(k)$$

- Funkce $g(k)$ je velikostí kernelu

- Obvykle platí $k' \leq k$

Kernel pro jazyk L

$$\langle I, k\rangle$$

$$\downarrow$$

Kernelizační algoritmus — Polynomiální čas

$$\downarrow$$

$$\langle I', k'\rangle \qquad |I'| + k' \leq g(k)$$

$$\langle I, k\rangle \in L \iff \langle I', k'\rangle \in L$$

Redukční pravidla pro VRCHOLOVÉ POKRYTÍ

- Popíšeme redukční pravidla pro VRCHOLOVÉ POKRYTÍ
- Kernelizační algoritmus pak aplikuje tato pravidla, dokud je to možné
- Jazyk asociovaný s problémem VRCHOLOVÉHO POKRYTÍ označíme

$$\mathrm{VP} = \{\langle G, k\rangle \mid G \text{ má vrcholové pokrytí velikosti nejvýš } k\}$$

- Pro popis pravidel předpokládáme instanci $\langle G, k\rangle$

Izolované vrcholy

Pravidlo VP1

Pokud G obsahuje izolovaný vrchol v, smaž v z grafu G.

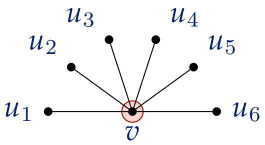
- Výsledkem je nová instance $\langle G - v, k\rangle$
- Izolované vrcholy nemohou pokrýt žádnou hranu, tedy

$$\boxed{\langle G - v, k\rangle \in \mathrm{VP} \iff \langle G, k\rangle \in \mathrm{VP}}$$

Vrcholy s velkým stupněm

Pravidlo VP2

Pokud G obsahuje vrchol v stupně alespoň $k+1$, smaž v a sniž hodnotu k o 1.



Pokud $k < 6$, pak $v \in S$

- Dostaneme instanci $\langle G - v, k - 1\rangle$
- Uvažme vrcholové pokrytí S
- Je-li $v \notin S$, pak všichni sousedi v patří do S
- Je-li $|S| \leq k$ a $\deg(v) \geq k+1$, pak $v \in S$

$$\langle G - v, k - 1\rangle \in \mathrm{VP} \iff \langle G, k\rangle \in \mathrm{VP}$$

Redukovaná instance

Lemma

Předpokládejme, že na instanci $\langle G, k\rangle \in \mathrm{VP}$ *nelze aplikovat žádné z pravidel VP1 and VP2, pak* $|V| \leq k^2 + k$ *a* $|E| \leq k^2$.

- $\langle G, k\rangle \in \mathrm{VP}$, tedy G má vrcholové pokrytí S velikosti nejvýš k
- VP1 nelze použít $\implies$ G nemá izolované vrcholy
- Každý vrchol $v \in V \setminus S$ má souseda v S
- VP2 nelze použít $\implies$ každý vrchol $v \in V$ má stupeň nejvýš k
- Tedy $|V \setminus S| \leq k|S|$
- Protože $|S| \leq k$, platí $|V| - k \leq |V \setminus S|$, tedy

$$|V| \leq |V \setminus S| + k \leq k|S| + k \leq k^2 + k$$

Redukovaná instance (hrany)

- Každá hrana je pokryta vrcholem z S
- Každý vrchol v S pokrývá nejvýš k hran, tedy

$$|E| \leq k|S| \leq k^2$$

Poslední pravidlo a kernel

Pravidlo VP3

Předpokládejme, že žádné z pravidel VP1 a VP2 nelze aplikovat na $\langle G, k\rangle$. Předpokládejme, že

$$k < 0 \quad \text{nebo} \quad |V| > k^2 + k \quad \text{nebo} \quad |E| > k^2.$$

Pak G nemá vrcholové pokrytí velikosti nejvýš k.

Kernelizační algoritmus pro Vrcholové pokrytí

Kernelizační algoritmus pro Vrcholové pokrytí

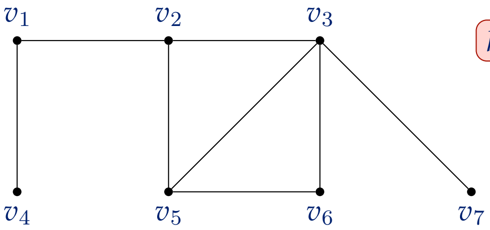
Vstup: Graf $G = (V, E)$ a celé číslo k
Výstup: Kernel pro $\langle G, k \rangle$

```
repeat
    while G obsahuje izolovaný vrchol v do          // Pravidlo VP1
        G ← G − v
    while v G je vrchol v stupně deg(v) > k do      // Pravidlo VP2
        G ← G − v
        k ← k − 1
until hodnota k se nezměnila
if k < 0 or |V| > k² + k or |E| > k² then         // Pravidlo VP3
    return negativní instanci (např. graf s jednou hranou a k = 0)
return ⟨G, k⟩
```

Algoritmus vrátí kernel s $O(k^2)$ vrcholy a $O(k^2)$ hranami.

Příklad kernelu VRCHOLOVÉHO POKRYTÍ

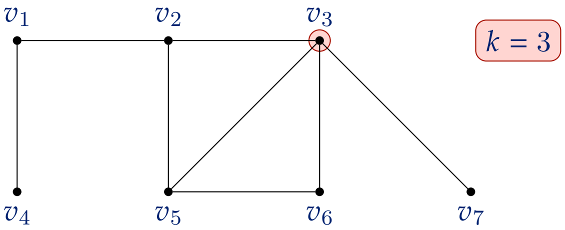
Uvažme následující graf



$k = 3$

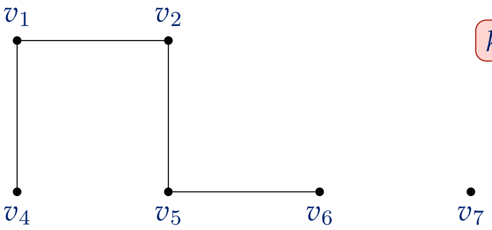
Příklad kernelu Vrcholového pokrytí

Vrchol v_3 má $4 \geq k + 1$ sousedů



Příklad kernelu Vrcholového pokrytí

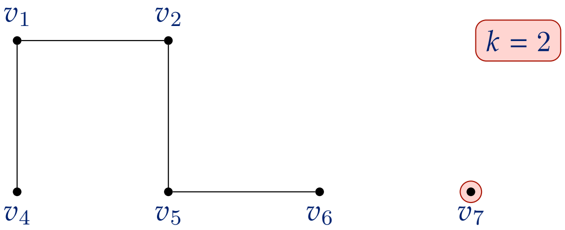
Smažeme v_3 a snížíme k o 1



$k = 2$

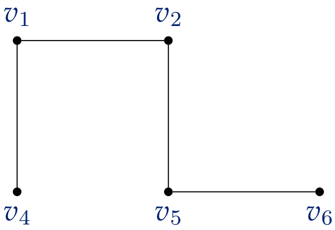
Příklad kernelu Vrcholového pokrytí

Vrchol v_7 je izolovaný



Příklad kernelu VRCHOLOVÉHO POKRYTÍ

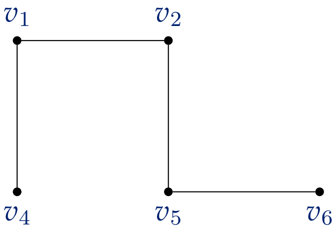
Smažeme v_7



$k = 2$

Příklad kernelu Vrcholového pokrytí

Další pravidlo nelze aplikovat, máme kernel



$k = 2$

$|V| = 5 \leq 6 = k^2 + k$ a $|E| = 4 = k^2$

Parametrizované algoritmy a kernely

Lemma

Rozhodnutelný parametrizovaný problém L je v FPT, právě když má kernel.

Důkaz.

Důkaz ve dvou krocích

1. Kernel pro L implikuje $L \in \mathrm{FPT}$
2. $L \in \mathrm{FPT}$ implikuje existenci kernelu

$\square$

Kernel pro L implikuje $L \in \mathrm{FPT}$

- Nechť L má kernelizační algoritmus $\mathcal{A}$ s velikostí $g(k)$

Parametrizovaný algoritmus pro L
Vstup: Instance $\langle I, k\rangle$ problému L
1 $\langle I', k'\rangle \leftarrow \mathcal{A}(I, k)$
2 Vyřeš $\langle I', k'\rangle$ hrubou silou

- Krok 1 pracuje v polynomiálním čase
- Krok 2 lze provést, protože L je rozhodnutelný
- Složitost kroku 2 závisí jen na k
  - Platí $|I'| + k' \leq g(k)$
- Složitost algoritmu je tedy $O^*(f(k))$ pro nějakou vyčíslitelnou funkci $f(k)$

$L \in \mathrm{FPT}$ implikuje kernel pro L

- Nechť $\mathcal{A}$ je algoritmus rozhodující, zda $\langle I, k \rangle \in L$ v čase $f(k) \cdot |I|^c$

Kernelizační algoritmus pro L

Vstup: Instance $\langle I, k \rangle$ problému L

1 Proveď nejvýš $|I|^{c+1}$ kroků výpočtu $\mathcal{A}(I, k)$

2 **if** $\mathcal{A}$ přijal během $|I|^{c+1}$ kroků **then**

3   **return** triviální instanci $\langle I', k' \rangle \in L$

4 **if** $\mathcal{A}$ odmítl během $|I|^{c+1}$ kroků **then**

5   **return** triviální instanci $\langle I'', k'' \rangle \notin L$

// Výpočet $\mathcal{A}(I, k)$ neskončil během $|I|^{c+1}$ kroků

6 **return** $\langle I, k \rangle$

- Algoritmus pracuje v polynomiálním čase
- Zbývá odhadnout velikost instance $\langle I, k \rangle$ vrácené v kroku 6

Analýza kernelu

- Předpokládejme, že algoritmus dosáhl kroku 6
- Výpočet $\mathcal{A}(I, k)$ tedy neskončil během $|I|^{c+1}$ kroků
- Z toho plyne, že

$$|I|^{c+1} \leq f(k) \cdot |I|^c$$

- Tedy $|I| \leq f(k)$
- Dostáváme, že $|\langle I, k\rangle| = |I| + k \leq f(k) + k$

$\langle I, k\rangle$ je tedy kernel velikosti $g(k) \leq f(k) + k$

Prohledávání s omezenou hloubkou

Prohledávání s omezenou hloubkou

Idea

Pro danou instanci $\langle I, k \rangle$ problému L

- Rozděl $\langle I, k \rangle$ do malé množiny podúloh
- Rekurzivně vyřeš podúlohy
- Hloubka rekurze je omezena v závislosti na parametru k

- Dělení na podúlohy by mělo proběhnout v polynomiálním čase
- Rekurzivní volání tvoří strom prohledávání
- Složitost $O^*(f(k))$ je pak rovna velikosti tohoto stromu
  - Počet uzlů
  - Celkový počet rekurzivních volání

Příklad

Jednoduchý algoritmus pro VRCHOLOVÉ POKRYTÍ se složitostí $O(2^k \cdot (m + n))$.

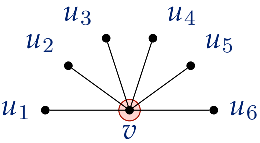
Podúlohy Vrcholového pokrytí

- Uvažme graf $G = (V, E)$
- Označme množinu sousedů vrcholu $v \in V$ pomocí

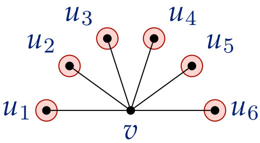
$$N(v) = \{u \in V \mid \{u, v\} \in E\}$$

- Pro každý vrchol v a minimální vrcholové pokrytí S platí

$$\boxed{\text{Buď } v \in S \text{ nebo } N(v) \subseteq S}$$



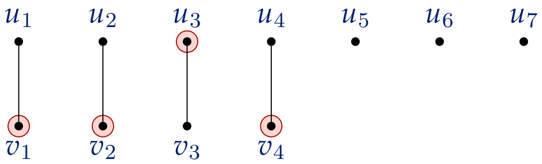
nebo



Základní případ

- Předpokládejme, že $G = (V, E)$ je graf s maximálním stupněm 1
- Pak každá komponenta G je buď
  - izolovaný vrchol, nebo
  - hrana
- Právě $|E|$ vrcholů je potřeba k pokrytí hran G
- V polynomiálním čase lze rozhodnout, zda $\langle G, k\rangle \in \mathrm{VP}$

$\langle G, k\rangle \in \mathrm{VP}$, právě když $k \geq |E|$



Funkce `SearchVP2`(G, k)

Vstup: Graf $G = (V, E)$, $k \geq 0$

Výstup: Vrcholové pokrytí $S \subseteq V$, $|S| \leq k$ nebo NONE

if $k \leq 1$ **then**
  Vyřeš základní případ
$d \leftarrow \max_{v \in V} \deg v$ `// Maximální stupeň` G
if $d \leq 1$ **then**
  Vyřeš základní případ
$v \leftarrow$ nějaký vrchol s maximálním stupněm `//` $\deg(v) = d \geq 2$
if `SearchVP2`($G - v$, $k - 1$) vrátí množinu S' **then**
  return $S = S' \cup \{v\}$
if `SearchVP2`($G - (N(v) \cup \{v\})$, $k - d$) vrátí množinu S' **then**
  return $S = S' \cup N(v)$
return NONE

Korektnost `SearchVP2(`G, k`)`

`SearchVP2(`G, k`)` najde vrcholové pokrytí G velikosti nejvýš k, právě když existuje.

- Plyne z předchozích úvah

Časová složitost `SearchVP2`(G, k)

- Nechť $n = |V|$ a $m = |E|$
- Základní případy lze vyřešit v čase $O(m + n)$
- Podúlohy lze zkonstruovat v čase $O(m + n)$
- Rekurzivní volání `SearchVP2`(G, k) tvoří strom prohledávání $\mathcal{T}$
- Označme $\tau(k)$ počet uzlů stromu $\mathcal{T}$

Časová složitost `SearchVP2`(G, k) je

$$O(\tau(k)(m + n))$$

Počet uzlů stromu $\mathcal{T}$

- Označme ℓ počet listů $\mathcal{T}$
- Vnitřní uzly stromu mají 2 syny, tedy

$$\tau(k) \leq 2\ell - 1$$

- Definujme rekurentní funkci

$$T(i) = \begin{cases} T(i-1) + T(i-2) & i \geq 2 \\ 1 & \text{jinak} \end{cases}$$

- $\mathcal{T}$ má nejvýš $T(k)$ listů

Časová složitost `SearchVP2(`G, k`)` je

$$O(T(k)(m+n))$$

Počet listů $\mathcal{T}$

Jaká je hodnota $T(k)$?

Krátká odpověď: $T(k) \leq 1.6181^k$

- Důkaz indukcí dle k

První krok Pro $k \leq 1$ platí $T(k) = 1 \leq 1.6181^k$

Indukční krok Pro $k > 1$ platí

$$\begin{aligned} T(k) = T(k-1) + T(k-2) &\leq 1.6181^{k-1} + 1.6181^{k-2} \\ &\leq 1.6181^{k-2}(1.6181 + 1) \leq 1.6181^{k-2}1.6181^2 \\ &\leq 1.6181^k \end{aligned}$$

Proč $T(k) = 1.6181^k$?

- Hledáme řešení typu

$$T(k) \leq c \cdot \lambda^k$$

- Pak hledáme konstanty c a λ splňující

$$T(k) = T(k-1) + T(k-2) \leq c \cdot \lambda^{k-1} + c \cdot \lambda^{k-2} \leq c \cdot \lambda^k$$

- Potřebujeme tedy vyřešit

$$c \cdot \lambda^{k-1} + c \cdot \lambda^{k-2} \leq c \cdot \lambda^k$$

- To je ekvivalentní

$$\lambda + 1 \leq \lambda^2$$

Řešením je zlatý řez

- Řešíme rovnici

$$\lambda^2 = \lambda + 1$$

- Největší kořen tohoto výrazu je zlatý řez

$$\lambda = \frac{1+\sqrt{5}}{2} < 1.6181$$

Časová složitost `SearchVP2(`G, k`)` je

$$O(1.6181^k(m+n))$$

Přidáme kernel

- Vrcholové pokrytí má kernel s $O(k^2)$ vrcholy a hranami
- Kernel lze zkonstruovat v čase $O(n^c)$ pro nějakou konstantu c
  - Lepší kernel s nejvýš $2k$ vrcholy lze nalézt v čase $n\sqrt{m}$
- Použijeme následující postup
  1. Zkonstruuj kernel
  2. Aplikuj `SearchVP2(`G, k`)` na zkonstruovaný kernel

Časová složitost `SearchVP2(`G, k`)` je

$$O(1.6181^k k^2 + n^c)$$

Vylepšený základní případ

Lemma

Pro grafy s nejvyšším stupněm 2 lze najít minimální vrcholové pokrytí v polynomiálním čase.

Důkaz.

- Každá komponenta G je buď cesta nebo cyklus
- V obou případech lze minimální vrcholové pokrytí nalézt v polynomiálním čase □

V druhém rekurzivním volání lze předpokládat, že maximální stupeň G je alespoň 3.

Funkce `SearchVP3`(G, k)

Vstup: Graf $G = (V, E)$, $k \geq 0$

Výstup: Vrcholové pokrytí $S \subseteq V$, $|S| \leq k$ nebo NONE

if $\boxed{k \leq 2}$ **then**
  Vyřeš základní případ
$d \leftarrow \max_{v \in V} \deg v$ // `Maximální stupeň` G
if $\boxed{d \leq 2}$ **then**
  Vyřeš základní případ
$v \leftarrow$ nějaký vrchol s maximálním stupněm // $\boxed{\deg(v) = d \geq 3}$
if `SearchVP3`($G - v$, $k - 1$) vrátí množinu S' **then**
  return $S = S' \cup \{v\}$
if `SearchVP3`($G - (N(v) \cup \{v\})$, $k - d$) vrátí množinu S' **then**
  return $S = S' \cup N(v)$
return NONE

Časová složitost `SearchVP3`(G, k)

- Odhad provedeme podobně jako v případě funkce `SearchVP2`(G, k)
- Počet listů $\mathcal{T}$ lze odhadnout shora funkcí $T(i)$ definované pomocí

$$T(i) = \begin{cases} T(i-1) + T(i-3) & i \geq 3 \\ 1 & \text{otherwise} \end{cases}$$

- Hledáme řešení nerovnosti

$$\lambda^{k-1} + \lambda^{k-3} \leq \lambda^k$$

- Řešením je největší kořen rovnice

$$\lambda^3 = \lambda^2 + 1$$

- Největší kořen této rovnice je $\lambda \leq 1.4656$

Časová složitost `SearchVP3`(G, k)

Časová složitost `SearchVP3`(G, k) je

$$O(1.4656^k k^2 + n^c)$$

- Použijeme-li lepší kernel, lze ukázat

Věta

Vrcholové pokrytí lze vyřešit v čase $O(n\sqrt{m} + 1.4656^k k^{O(1)})$

Poznámky k parametrizovaným algoritmům

- Parametrizace umožňuje lepší porozumění některým problémům
  - Které části problému jsou těžké
  - Kterých parametrů je třeba si všímat
- Lepší algoritmy
  - Lze použít k řešení instancí s malými hodnotami parametrů
  - Dokazatelné horní odhady složitosti
    - Narozdíl od většiny heuristických algoritmů
- Jemnější rozlišení složitosti NP-úplných problémů
  - Některé NP-úplné problémy mají dobré parametrizované algoritmy
  - Jiné nikoli

Introduction to Parameterized Algorithms — NTIN103