VCS – Přednáška 7

Definition 1. *Nechť M je deterministický Turingův stroj, který zastaví pro každý vstup.* Časová složitost *TS M je funkce $f : \mathbb{N} \to \mathbb{N}$, kde maximální délka výpočtu nad slovem délky n.*

Pokud $f(n)$ je časová složitost TS M, říkáme, že TS M rozhoduje v čase $f(n)$.

Definition 2. *Nechť f a g jsou dvě funkce $f, g : \mathbb{N} \to \mathbb{R}^+$. Říkáme, že $f(n) = \mathcal{O}(g(n))$, pokud existují kladná čísla c a n_0 tak, že pro každé přirozené číslo $n \geq n_0$*

$$f(n) \leq c \cdot g(n)$$

Pokud $f(n) = \mathcal{O}(g(n))$, říkáme, že $g(n)$ je asymptotická horní hranice *pro $f(n)$.*

Definition 3. *Nechť $t : \mathbb{N} \to \mathbb{N}$ je funkce. Třída časové složitosti* $\mathrm{TIME}(t(n))$ *je množina*

$$\mathrm{TIME}(t(n)) = \{L \mid L \text{ je jazyk rozhodovaný TS s časovou slož. } \mathcal{O}(t(n))\}$$

Theorem 1. *Nechť $t(n)$ je časová složitost vícepáskového TS. Pak existuje ekvivalentní TS s jednou páskou s časovou složitostí $\mathcal{O}(t^2(n))$.*

Definition 4. *Nechť M je nedeterministický Turingův stroj, který zastaví pro každý vstup.* Časová složitost *TS M je funkce $f : \mathbb{N} \to \mathbb{N}$, kde maximální délka výpočtu nad slovem délky n v kterékoli větvi výpočtu.*

Pokud $f(n)$ je časová složitost TS M, říkáme, že TS M nedeterministicky rozhoduje v čase $f(n)$.

Theorem 2. *Nechť $t(n)$ je funkce t.ž. $t(n) \geq n$. Ke každému nedeterministickému TS s jednou páskou s časovou složitostí $t(n)$ existuje equivalentní deterministický TS s jednou pásku s časovou složitosti $2^{\mathcal{O}(t(n))}$.*

1 Třída $\mathcal{P}$

$$\mathcal{P} = \bigcup_k \mathrm{TIME}(n^k)$$

.

Příklad problému v $\mathcal{P}$

$$\mathrm{PATH} = \{[G, p, q] \mid G \text{ je orientovaný graf a existuje v něm cesta z } p \text{ do } q\}$$

2 Třída $\mathcal{NP}$

Definition 5. Verifier *pro jazyk A je algoritmus V, kde*

$$A = \{w \mid V \text{ přijímá } \langle w, c\rangle \text{ pro nějaký řetězec } c\}.$$

Čas verifieru měříme pouze podle délky w. Jazyk A se nazývá polynomicky ověřitelný, *pokud pro něj existuje verifier s polynomickou časovou složitostí.*

Definition 6. *Třída $\mathcal{NP}$ je třída polynomicky ověřitelných jazyků.*

Třídu $\mathcal{NP}$ bychom alternativně mohli definovat podobným způsobem, jako jsme definovali třídu $\mathcal{P}$. Zavedeme třídu časové složitosti NTS

$$\mathrm{NTIME}(t(n)) = \{L \mid L \text{ je jazyk rozhodovaný NTS s časovou slož. } \mathcal{O}(t(n))\}$$

a $\mathcal{NP}$je pak definujeme následovně.

Definition 7.

$$\mathcal{NP} = \bigcup_k \mathrm{NTIME}(n^k)$$

.

Theorem 3. *Definice 6 a 7 jsou ekvivalentní.*

Důkaz. Ukážeme, že

– pokud je jazyk L ověřitelný v polynomickém čase, pak je L nedeterministicky rozhodovaný v polynomickém čase. Předpokládáme tedy, že L má NTS M který ho nedeterministicky rozhoduje v polynomickém čase.
  Verifikátor V pro vstupní slovo $[w, c]$
  1. Simuluje činnost NTS M pro w deterministicky – pokaždé, když se má rozhodnout mezi více možnými konfiguracemi, které lze v jednom kroku odvodit, rozhodne se podle dalšího znaku z certifikátu c.
  2. Pokud certifikát nemá dostatek znaků nebo NTS M zamítne w, verifikátor V zamítne $[w, c]$. Pokud NTS M přijme w, verifikátor V přijímá $[w, c]$.

  Verifikátor V tedy prochází jednu větev výpočtu NTS M pro w a používá certifikát c jako navigaci. Pokud NTS M v nějaké větvi výpočtu w skončí v přijímající konfiguraci, musí existovat c, který tam simulaci navede. Pokud naopak žádná taková větev neexistuje, nebude existovat ani takový certifikát.
– pokud je jazyk L nedeterministicky rozhodovaný v polynomickém čase, pak je L ověřitelný v polynomickém čase. Předpokládáme tedy, že L má verifikátor V pracující v polynomickém čase. Sestrojíme nedeterministický stroj M: NTS M pro vstupní slovo w
  1. nedeterministicky určí certifikát c,
  2. spustí V pro $[w, c]$; Pokud V přijme $[w, c]$, M přijme w, pokud V zamítne $[w, c]$, M zamítne w.

  NTS M tedy nedeterministicky vygeneruje certifikát pro verifikátor v.

Příklad jazyka v $\mathcal{NP}$

V této části si představíme dva jazyky, které spadají do třídy $\mathcal{NP}$.

Jazyk HAMPATH

$$\text{HAMPATH} = \{[G, p, q] \mid G \text{ je or. graf, který má hamiltonovskou cestu z } p \text{ do } q\}$$

NTS M_{HAMPATH} pro vstupní slovo $[G, p, q]$:

– Nedeterministicky vybere sekvenci uzlů začínající p a končící q.
– Zjistí, je tato sekvence cesta v orientovaném grafu. Pokud ano, přijme, pokud ne, zamítne.

Jazyk k-CLIQUE
Klika grafu je podgraf, ve kterém jsou všechny uzly vzájemně spojeny hranami. Jazyk k-CLIQUE je definován následovně:

$$k\text{-CLIQUE} = \{[G, k] \mid G \text{ je neorientovaný graf, který má kliku o } k \text{ uzlech}\}$$

NTS $M_{k\text{-CLIQUE}}$ pro vstupní slovo $[G, k]$:

– Nedeterministicky vybere k uzlů.
– Zjistí, jestli tyto uzly tvoří kliku (tedy jestli je každý s každým spojen hranou). Pokud ano, přijme, pokud ne, zamítne.

3 $\mathcal{P}$ vs. $\mathcal{NP}$

To, že $\mathcal{P} \subseteq \mathcal{NP}$ je zjevné už z definice těchto dvou tříd a z faktu, že deterministický TS je speciální případ nedeterministického TS. Jak je to s opačnou inkluzí není známo.
Co by znamenalo rozřešení $\mathcal{P}$ vs. $\mathcal{NP}$?

– $\mathcal{P} = \mathcal{NP}$ – výsledek by měl ohromné praktické následky, pokud by důkaz vedl k výkonným metodám pro řešení některých problémů, které jsou v $\mathcal{NP}$. Je také možné že důkaz by nevedl k takovýmto výkonným metodám, protože by třeba neměl konstruktivní charakter, nebo by řád ohraničujícího polynomu by byl příliš velký na to, aby byly metody použity v praxi. Následky by byly pozitivní i negativní, protože protože $\mathcal{NP}$-úplné problémy se používají v mnoha oblastech. Například mnoho kryptografických metod je založeno na tom, že řešit některé problémy je obtížné. Pokud by existovala výkonná metoda na řešení takového problému, muselo by se šifrování založené na těchto metodách upravit nebo úplně nahradit. Na druhou strany by to mělo ohromné pozitivní následky v podobě nových výpočetních postupů.
– $\mathcal{P} \neq \mathcal{NP}$ Pokud by se ukázal, že $\mathcal{P} \neq \mathcal{NP}$ nepřineslo by to praktické výpočetní výhody, jako by tomu bylo v případě $\mathcal{P} = \mathcal{NP}$. N adruhou stranu by to přineslo výrazný posun do teorie složitosti a dalo základ novým směrům výzkumu. Existoval by způsob, jak formálně ukázat, že mnoho běžných problémů nelze efektivně řešit, takže pozornost vědců by se přesunula k částečným řešením nebo k řešením jiných problémů. Vzhledem k tomu, že se obecně věří, že $\mathcal{P} \neq \mathcal{NP}$, toto přesunutí pozornosti již nastalo.

4 $\mathcal{NP}$-úplné problémy

Definition 8. *Nechť $f : \Sigma^* \to \Sigma^*$ je funkce. Říkáme, že (deterministický) TS* vyčísluje funkci f v polynomickém čase*, pokud pro každé vstupní slovo $w \in \Sigma^*$ zapíše v polynomickém čase na pásku $f(w)$ a skončí. Funkce f se nazývá* vyčíslitelná v polynomickém čase*, pokud existuje (deterministický) TS, který ji vyčísluje v polynomickém čase.*

Definition 9. *Nechť $f : \Sigma^* \to \Sigma^*$ je funkce vyčíslitelná v polynomickém čase a nechť $L_1, L_2 \subseteq \Sigma^*$ jsou jazyky. Říkáme, že f redukuje L_1 na L_2 v polynomickém čase, pokud pro $winL_1$ právě když $f(w) \in L_2$. Funkci f pak říkáme* redukce jazka L_1 na L_2 v polynomickém čase.
Pokud existuje redukce jazka L_1 na L_2 v polynomickém čase, říkáme, že L_1 je redukovatelný na L_2 v polynomickém čase. *Tento fakt zapisujeme $L_1 \leq_p L_2$.*

Pokud bychom používali pojmosloví problémů, mohli bychom napsat, že redukce je funkce vyčíslitelná v polynomickém čase konvertující instance problému P_1 odpovědí „ano“ na instance problému P_2 s odpovědí „ano“ a instance problému P_1 odpovědí „ne“ na instance problému P_2 s odpovědí „ne“. Všimněte si, že pojmy z definic 8 a 9 se liší od pojmů z definic **??** a **??** jen ve vyčíslitelnosti v polynomickém čase. Jinak je vše stejné. Platí také, že redukovatelnost v polynomickém čase je tranzitivní.

Lemma 1. *Pokud $L_1 \leq_p L_2$ a $L_2 \leq_p L_3$ pak $L_1 \leq_p L_3$.*

S pojmem redukovatelnosti v polynomickém čase už můžeme přejít k $\mathcal{NP}$-úplným problémům.

Definition 10. *Jazyk B nazýváme $\mathcal{NP}$-úplný, pokud*

- *$B \in \mathcal{NP}$,*
- *Každý jazyk $A \in \mathcal{NP}$ je redukovatelný v polynomickém čase na jazyk B. Tedy $A \leq_p B$*

O tom, že existuje alespoň jeden $\mathcal{NP}$-úplný problém mluví tzv. Cookova věta[1].

Theorem 4 (Cookova věta). *Jazyk* $\mathrm{SAT} = \{\varphi \mid \varphi$ *je splnitelná Booleovská formule*$\}$ *je $\mathcal{NP}$-úplný.*

Důkaz. Nejdříve ukážeme, že $\mathrm{SAT} \in \mathcal{NP}$. To uděláme tak, že sestrojíme NTS, který nedeterministicky rozhoduje SAT v polynomickém čase:

NTS M_{SAT} pro vstupní slovo $[\varphi]$, kde φ je booleovská formule:

1. nedeterministicky určí pravdivostní hodnotu pro každou výrokovou proměnnou, která se ve formuli vyskytuje. Nedeterministicky se pak vytvoří ohodnocení e.
2. Zjistí hodnotu $|\varphi|_e$; přijmi, pokud $|\varphi|_e = 1$, jinak zamítni.

Je snadné ověřit, že takto zkonstruovaný stroj nedeterministicky rozhoduje SAT v polynomickém čase.

Dále musíme ukázat, že SAT je $\mathcal{NP}$-těžký; tedy ukázat, že každý jazyk $A \in \mathcal{NP}$ se dá redukovat v polynomickém čase na SAT. Protože $A \in \mathcal{NP}$, musí existovat NTS M, který nedeterministicky rozhoduje A v čase $p(n)$, kde p je polynom. Předpokládáme, že NTS M se nikdy nepokusí o pohyb hlavy přes levý okraj pásky. Slova $w \in \Sigma_M^*$ budeme v polynomickém čase redukovat na formule φ a to tak, že φ je splnitelná právě když $w \in L(M)$.

Budeme používat následující výrokové proměnné:

- q_{lk}; význam této výrokové proměnné je, že NTS M se nachází v kroku k ve stavu q_l.
- t_{ijk}; význam této výrokové proměnné je, že NTS M má v k-tém kroku výpočtu zapsán na j-tém políčku pásky symbol $t_i \in \Sigma_M$.
- h_{jk}; význam této výrokové proměnné je, že NTS M má v k-tém kroku výpočtu hlavu umístěnou nad j-tým políčkem pásky.

Formule φ_w, kterou budeme konstruovat má následující tvar:

$$A \wedge B \wedge C \wedge D \wedge E \wedge F \wedge G \wedge H$$

Subformule označené písmeny A–H mají následující význam:

- A – každém kroku je řídící jednotka NTS M je v právě jednom stavu,
- B – každém kroku je hlava NTS M právě nad jedním políčkem pásky,
- C – každém kroku je na každém políčku pásky NTS M právě jeden symbol z Γ_M,
- D – přechodová funkce δ_M,
- E – změna na pásce je jen na místě, kde je hlava,

[1] Cookova věta se týká problému splnitelnosti Booleovských formulí, v této větě a jejím důkazu se používá pojmů z výrokové logiky. Tyto pojmy jsou studentům známy z kurzu XXXX

– F – výpočet končí přijímající konfigurací,
– H – výpočet začíná iniciální konfigurací.

Budeme používat následujících zkratek:

– $\bigvee_{0\leq i\leq m} x_i$ znamená, že alespoň jedno x_i je pravdivé; je to zkratka pro disjunkci $x_0 \vee x_1 \vee \cdots \vee x_m$. Obdobně symbol $\bigwedge$ budeme používat pro zkrácený zápis konjunkce.
– $x \equiv y$ znamená, že x má pravdivostní stejnou hodnotu jako y; zkracuje se tak zápis $(x \wedge y) \vee (\neg x \wedge \neg y)$.
– $x \to y$ pro implikaci; jedná se o zkratku za $\neg x \vee y$.
– $\bigotimes_{0\leq i\leq m} x_i$ znamená, že právě jedno x_i je pravdivé; je to zkratka pro $\bigvee_{0\leq i\leq m} x_i \wedge \bigwedge_{\substack{0\leq i,j\leq m\\ i\neq j}} \neg x_i \vee \neg x_j$.

Subformule A, B a C:

$$\bigwedge_{0\leq k<p(n)} \bigotimes_{q_l\in Q} q_{lk} \tag{A}$$

$$\bigwedge_{0\leq k<p(n)} \bigotimes_{0\leq j<p(n)} h_{jk} \tag{B}$$

$$\bigwedge_{\substack{0\leq j<p(n)\\ 0\leq k<p(n)}} \bigotimes_{t_i\in\Gamma} t_{ijk} \tag{C}$$

Subformule D: Tuto subformuli rozebereme o něco podrobněji. Uvažujme obecné pravidlo (prozatím deterministické) přechodové funkce $\delta(q_l, t_i) = (q_{l'}, t_{i'}, S)$. Toto pravidlo můžeme přímo přepsat na formuli

$$(q_{lk} \wedge t_{ijk} \wedge h_{jk}) \to (q_{l'(k+1)} \wedge t_{i'j(k+1)} \wedge h_{(j\pm1)(k+1)}).$$

Tato formule znamená pokud je stroj v kroku k ve stavu q_l a hlava je nad j-tým políčkem pásky a na tomto políčku je zapsán symbol t_i, pak v následujícím kroku (tj. kroku $k+1$) bude stroj ve stavu $q_{l'}$, na j-tém políčku pásky bude zapsán symbol $t_{i'}$ a hlava se bude nalézat buďto na $j-1$-tém nebo na $j+1$-tém políčku pásky (podle toho, jestli S je pohyb doleva nebo doprava). Toto musí platit v každém kroku a pro každé políčko pásky, proto formuli rozšíříme na formuli

$$\bigwedge_{\substack{0\leq k<p(n)-1\\ 0\leq k<p(n)}} (q_{lk} \wedge t_{ijk} \wedge h_{jk}) \to (q_{l'(k+1)} \wedge t_{i'j(k+1)} \wedge h_{(j\pm1)(k+1)}).$$

Tuto formuli ještě rozšíříme tak, aby zahrnovala všechna pravidla:

$$\bigwedge_{\substack{q_l\in Q\\ t_i\in\Gamma}} \bigwedge_{\substack{0\leq k<p(n)-1\\ 0\leq k<p(n)}} (q_{lk} \wedge t_{ijk} \wedge h_{jk}) \to (q_{l'(k+1)} \wedge t_{i'j(k+1)} \wedge h_{(j\pm1)(k+1)}).$$

Zbývá upravit formuli tak, aby zohledňovala nedeterminismus; tedy fakt, že pro stejnou kombinaci q_l, t_i může přechodová funkce nabízet různé trojice $(q_{l'}, t_{i'}, S)$. Z těchto musíme vybrat právě jednu. Tomu by odpovídala úprava podformule za $\to$ na $\bigotimes_{(q_{l'},t_{i'},S)\in\delta(q_l,t_i)}(q_{l'(k+1)} \wedge t_{i'j(k+1)} \wedge h_{(j\pm1)(k+1)})$. Takovou formuli můžeme zjednodušit a nevybírat právě jednu trojici ale alespoň jednu:

$$\bigwedge_{\substack{q_l\in Q\\ t_i\in\Gamma}} \bigwedge_{\substack{0\leq k<p(n)-1\\ 0\leq k<p(n)}} \bigvee_{(q_{l'},t_{i'},S)\in\delta(q_l,t_i)} (q_{lk} \wedge t_{ijk} \wedge h_{jk}) \to (q_{l'(k+1)} \wedge t_{i'j(k+1)} \wedge h_{(j\pm1)(k+1)}). \tag{D}$$

Toto zjednodušení si můžeme dovolit, protože pokud by byly vybrány dvě nebo více trojic, nemohla by být splnitelná alespoň jedna z subformulí A, B a C, které zajišťují, že se stroj vždy nalézá ve formálně správné konfiguraci.

Subformule E:

$$\bigwedge_{\substack{0\leq k<p(n)-1\\ 0<j\leq p(n)\\ t_i\in\Gamma}} t_{ijk} \equiv t_{ij(k+1)} \vee h_{jk} \tag{E}$$

Subformule F: Uvažujme $q_1 = q_+$, subformule F je pak jednoduše:

$$q_{1p(n)} \tag{F}$$

Subformule H: Označme $t_{w_i} \in \Gamma$ i-tý symbol ze vstupního slova w a uvažujme $t_0 = \text{␣}$ a q_0 je počáteční stav. Subformuli H můžeme zapsat takto:

$$\left(\bigwedge_{0\le j<n} t_{w_i j 0}\right) \wedge \left(\bigwedge_{n\le j<p(n)} t_{0j0}\right) \wedge h_{00} \wedge q_{00} \tag{H}$$

Formule φ_w je splnitelná, pokud existuje přijímající výpočet NTS M nad slovem w. Potřebujeme ukázat, že tuto formuli lze z w vytvořit v polynomickém čase. Zápis takové formule je přímo úměrný délce této formule a ta je přímo úměrná počtu výskytů výrokových proměnných:

Po rozepsání $\bigotimes_{\substack{q_l\in Q\\0\le k<p(n)}} q_{lk}$ na $\bigvee_{\substack{0\le k\le p(n)\\q_l\in Q}} q_{lk} \wedge \bigwedge_{\substack{0\le k\le p(n)\\q_l\ne q_{l'}\in Q}} \neg q_{lk} \vee \neg q_{l'k}$ vidíme, že v první části této formule máme $|Q|\cdot p(n)$ výskytů proměnných q_{lk} a v druhé části máme $p(n)\cdot|Q|(|Q|-1)$. Celkově tedy $p(n)\cdot|Q|^2 = O(p(n))$ výskytů. Obdobně v subformulích B a C máme $\mathcal{O}(p^3(n))$ výskytů výrokové proměnných. V subformulích D a E je $\mathcal{O}(p^2(n))$ výskytů, v subformulí F $\mathcal{O}(1)$ výskytů a v subformuli H máme $\mathcal{O}(p(n))$ výskytů. V součtu máme tedy $\mathcal{O}(p^3(n))$ výskytů výskytů výrokových proměnných.

Theorem 5. *Jazyk B je $\mathcal{NP}$-úplný, pokud*

- *$B \in \mathcal{NP}$,*
- *Existuje $\mathcal{NP}$-úplný jazyk A, který je redukovatelný v polynomickém čase na jazyk B. Tedy $A \le_p B$*

Důkaz. Plyne přímo z definice $\mathcal{NP}$-úplného problému a tranzitivity redukovatelnosti v polynomickém čase.

Jazyk 3SAT (problém splnitelnosti booleovských formulí v 3CNF) Formule je v *konjunktivní normální formě* (CNF), pokud má tvar $k_1 \wedge k_2 \wedge \cdots \wedge k_n$, kde $k_1, k_2, \ldots, k_n$, jsou tzv. *klauzule*, které mají tvar disjunkce *literálů*. Literál je pak buďto výroková proměnná nebo její negace. Formule je v *3-konjunktivní normální formě* (3CNF), pokud každá její klauzule má právě tři literály.

Jazyk 3SAT je definován následovně:

$$\text{3SAT} = \{[\varphi] \mid \varphi \text{ je splnitelná booleovská formule v 3CNF}\}$$

Theorem 6. 3SAT *je NP-úplný.*

Důkaz. Tvrzení nebudeme dokazovat celé. Namísto toho upravíme důkaz Cookovy věty tak, aby vytvářená formule φ_w byla v 3CNF. Musíme tedy zařídit, aby každá ze subformulí A–H byla v 3CNF. Po rozepsání zkratek, které jsme používali, jde vidět, že každá subformule je v CNF, ale není v 3CNF. Musíme zařídit, aby klauzule obsahovaly právě tři literály. Pokud klauzule obsahuje jeden literál nebo dva literály, přidáme do ní opakovaně jeden z nich. Pokud obsahuje více než tři literály (označme tento počet n) rozdělíme ji na klauzuli, která má 3 literály a klauzuli, která má $n-1$ literálů pomocí nové výrokové proměnné:

$$(l_1 \vee l_2 \vee l_3 \vee \cdots \vee l_n) \mapsto (l_1 \vee l_2 \vee x) \wedge (\neg x \vee l_3 \vee \cdots \vee l_n)$$

Opakováním tohoto kroku dostaneme formuli v 3CNF formě. Je zjevné, že počet pomocných výrokových proměnných je lineárně závislý na počtu výskytů původních výrokových proměnných. Délka formule i čas potřebný na její vytvoření zůstává polynomiální.

Theorem 7. k-CLIQUE*je $\mathcal{NP}$-úplný.*

Důkaz. To, že k-CLIQUE $\in \mathcal{NP}$ už jsme ukázali výše.

Zbývá tedy dokázat, že k-CLIQUE je $\mathcal{NP}$-těžký. Dokážeme to tak, že ukážeme 3SAT $\le_p$ k-CLIQUE.

TS $R_{\text{3SAT}\to k\text{-CLIQUE}}$ pro vstupní slovo $[\varphi]$, kde φ je Booleovská formule v 3CNF:

1. Vytvoří trojici uzlů pro každou klauzuli v φ, každý uzel v této trojici představuje jeden literál z klauzule a je tímto literálem označen.
2. Vytvoř hrany mezi každými dvěma uzly až na následující výjimky:
   - uzly z jedné trojice nebudou spojeny hranou,
   - uzly které jsou označené výrokovou proměnnou a její negací nebudou spojeny.
3. zapiš kód $[G, k]$, kde G je vytvořený graf a k je počet klauzulí ve φ

Nyní potřebujeme ukázat dvě věci:

- Že jde o správně vytvořenou redukci, tedy $[\varphi] \in$ 3SAT právě když $[G, k] \in k$-CLIQUE: Aby φ byla při ohodnocení e pravdivá, je potřeba, aby alespoň jeden literál z každé klauzule byl při tomto ohodnocení pravdivý. Vybereme-li z každé klauzule jeden literál, který je při ohodnocení e pravdivý (tedy celkem k literálů), tvoří uzly odpovídající těmto literálům k-kliku v grafu G. Tím je ukázáno, že Pokud $[\varphi] \in$ 3SAT pak $[G, k] \in k$-CLIQUE.
  Abychom mohli ukázat i že, pokud $[\varphi] \notin$ 3SAT pak $[G, k] \notin k$-CLIQUE, musíme se blíže podívat na to, jak je graf G zkonstruován. Uzly v trojici, která odpovídá jedné klauzuli, nejsou spojeny – to způsobuje, že jakákoli k-klika musí obsahovat právě jeden uzel z každé trojice. Uzly, které představují výrokovou proměnnou a její negaci, nejsou spojeny – to způsobuje, že žádná k-klika nebude obsahovat uzly, které by nemohly být při stejném pravdivostním ohodnocení oba pravdivé. Pokud by existovala k-klika v grafu G, dalo by se podle ní najít ohodnocení e, při kterém by φ byla pravdivá – stačí ohodnotit výrokové proměnné tak, aby všechny literály, kterými jsou označeny uzly v klice, byly pravdivé.

– Že redukce pracuje v polynomickém čase. Počet uzlů, které se mají vypsat, odpovídá počtu výskytů výrokových symbolů, tedy $\mathcal{O}(n)$, počet hran které se mají vytvořit je $\mathcal{O}(n^2)$. Redukce tedy probíhá v polynomickém čase.