Ukážeme si lineární algoritmus, který pro pevné k rozhodne, zda vstupní graf má stromovou šířku nejvýše k, a je-li tomu tak, také vrátí příslušný stromový rozklad. Poznamenejme, že je-li k součástí vstupu, pak rozhodnout zda graf má stromovou šířku nejvýše k je NP-úplné [1]. Popsaný algoritmus je motivován Bodlaenderovým algoritmem [3]. Bez důkazu budeme používat následující tvrzení.

Věta 1 (Bodlaender a Kloks [2])**.** *Nechť $K \geq k \geq 0$ jsou celá čísla. Pak existuje algoritmus, který jako vstup bere graf G a jeho stromový rozklad šířky nanejvýš K, a rozhodne zda G má stromovou šířku nanejvýš k. Je-li tomu tak, pak také vrátí stromový rozklad G šířky nejvýše k. Časová složitost algoritmu je $O(|V(G)|)$.*

Algoritmus z Věty 1 je založen na dynamickém programování zpracovávajícím rekurzivně vstupní stromový rozklad, je však poměrně netriviální. Multiplikativní konstanta algoritmu skrytá v O-notaci závisí exponenciálně na K.

Dále budeme potřebovat několik jednoduchých pozorování. Připomeňme, že G je *k-degenerovaný*, pokud každý jeho podgraf obsahuje vrchol stupně nejvýše k.

Lemma 2. *Platí následující tvrzení:*

(a) Má-li graf stromovou šířku nejvýše k, pak je k-degenerovaný.

(b) Je-li H minor G, pak stromová šířka H je menší nebo rovna stromové šířce G.

(c) Každý k-degenerovaný graf na n vrcholech má nejvýše kn hran.

(d) Je-li graf G na n vrcholech k-degenerovaný a $k \geq 1$, pak G obsahuje nejvýše $2^k n$ klik.

Důkaz. Ukážeme si pouze (d), ostatní tvrzení si laskavý čtenář snadno rozmyslí. Tvrzení budeme dokazovat indukcí. Je-li $n = 1$, pak G obsahuje dvě kliky (prázdnou a jednovrcholovou), a $2^k n \geq 2$. Můžeme tedy předpokládat, že $n > 1$ a tvrzení (e) platí pro grafy s $n-1$ vrcholy. Jelikož G je k-degenerovaný, obsahuje vrchol v stupně nejvýše k. Každá klika v G je buď podgrafem $G-v$ nebo obsahuje v. Z indukce $G-v$ obsahuje nejvýše $2^k(n-1)$ klik. Každá klika obsahující v se skládá z vrcholu v a z nějaké podmnožiny jeho okolí, a těchto podmnožin je nejvýše 2^k. Proto G obsahuje nejvýše $2^k(n-1) + 2^k = 2^k n$ klik. □

Pro pevnou konstantu k lze o grafu G na n vrcholech v čase $O(n)$ rozhodnout, zda je k-degenerovaný nebo ne: je-li $|E(G)| > kn$, pak G není k-degenerovaný. Jinak si budeme udržovat frontu vrcholů stupně nejvýše k. Z ní vždy odebereme vrchol a smažeme ho z grafu. Pro sousedy odebraného vrcholu pak ověříme, zda jsme jejich stupeň snížili na k a v tom případě je přidáme do fronty. Odebereme-li takto postupně všechny vrcholy grafu, pak G je k-degenerovaný. Jinak nám nakonec zůstane podgraf G s minimálním stupněm alespoň $k+1$, který ukazuje, že G není k-degenerovaný.

Ukažme si nyní kvadratický algoritmus, který pro vstupní graf G rozhodne, zda jeho stromová šířka je nejvýše $k \geq 1$, a pokud ano, vrátí příslušný rozklad:

Algoritmus 1.

- *Pokud G není k-degenerovaný, pak jeho stromová šířka je větší než k (dle Lemma 2(a)).*
- *Pokud G nemá žádné hrany, pak má stromovou šířku* 0 *a triviálně určíme nějaký jeho stromový rozklad.*
- *Jinak uvažme libovolné neprázdné párování $M \subseteq G$ a jako G_1 označme minor G vzniklý kontrakcí hran v M. Rekurzivně se zavoláme na G_1.*
  - *Pokud G_1 má stromovou šířku větší než k, pak i G má stromovou šířku větší než k (Lemma 2(b)).*
  - *Jinak buď (T, f) stromový rozklad G_1 šířky nejvýše k. Pro uzel $v \in V(T)$ definujme $f'(v)$ jako bramboru získanou z $f(v)$ dekontrakcí hran M (tj. vrcholy vzniklé kontrakcí hran v M nahradíme odpovídajícími dvojicemi vrcholů G). Pak (T, f') je stromový rozklad G šířky nanejvýš $2k+1$. Na G s tímto stromovým rozkladem poté použijeme algoritmus z Věty 1.*

Je-li G graf na n vrcholech, pak s ním až na rekurzivní volání strávíme pouze čas $O(n)$ (je trochu potřeba si rozmyslet, jak v tomto čase vyrobit minor G_1; zde se využije toho, že po prvním kroku víme, že G je k-degenerovaný, a tedy má pouze $O(n)$ hran). Jelikož M je neprázdné párování, graf G_1 má nejvýše $n-1$ vrcholů, a proto je časová složitost popsaného algoritmu $O(n + (n-1) + (n-2) + \ldots) = O(n^2)$.

Kdybychom ale měli štěstí a M samo mělo lineární velikost (tedy $|V(M)| \geq \varepsilon n$ pro nějakou konstantu ε), pak by G_1 měl pouze $(1-\varepsilon/2)n$ hran. Kdybychom dokázali obdobně graf zmenšit v každém kroku, dostávali bychom časovou složitost $n + (1-\varepsilon/2)n + (1-\varepsilon/2)^2 n + (1-\varepsilon/2)^3 n + \ldots = \frac{2}{\varepsilon} n = O(n)$.

Zvolme si tedy M jako maximální párování (na inkluzi, tedy ne nutně největší možné). Co budeme dělat, když párování M není dost velké? Povšimneme si, že $G - V(M)$ je nezávislá množina, jelikož jinak bychom mohli přidat další hranu do M ($V(M)$ je tedy *vrcholové pokrytí*), a aplikujeme následující tvrzení.

Lemma 3. *Pro každé $k \geq 1$ existuje následující algoritmus. Vstupem je k-degenerovaný graf G na n vrcholech a množina $X \subseteq V(G)$ taková, že $G - X$ je nezávislá množina. Výstupem je buď minor G, který není k-degenerovaný, nebo navzájem disjunktní množiny $A_1, A_2, \ldots \subseteq V(G) \setminus X$ velikosti alespoň k takové, že $\sum_i |A_i| \geq n - (k + 1 + (k-1)2^k)|X|$ a pro každé i, všechny vrcholy v A_i mají stejnou množinu sousedů a těchto sousedů je nejvýše k. Časová složitost algoritmu je $O(n)$.*

Důkaz. Nechť v_1, v_2, $\ldots$, v_m jsou vrcholy $V(G) \setminus X$ v libovolném pořadí. Postupně definujme množiny $B_j, C_j \subseteq V(G)$ a grafy G_j pro $j = 0, 1, 2, \ldots, m$, kde G_j je minor G. Položme $G_0 = G$ a $B_0 = C_0 = \emptyset$. Uvažme nyní $j \geq 1$. Jako N_j označme libovolnou podmnožinou sousedů v_j velikosti $\min(\deg(v_j), k+1)$. Zkontrolujeme, zda N_j indukuje kliku v G_{j-1}. Pokud ne, pak zvolme libovolně $x_j, y_j \in N_j$ tž. $x_j y_j \notin E(G_{j-1})$, a položíme $G_j = G_{j-1}/v_j x_j$, $B_j = B_{j-1} \cup \{v_j\}$ a $C_j = C_{j-1}$. Pokud N_j je klika v G_{j-1} a $|N_j| = k+1$, pak G obsahuje ne-k-degenerovaný graf K_{k+2} jako minor a algoritmus ukončíme. Poslední případ je, že N_j je klika v G_{j-1} a $|N_j| \leq k$, potom položíme $G_j = G_{j-1}$, $B_j = B_{j-1}$ a $C_j = C_{j-1} \cup \{v_j\}$.

Definujme $H = G_m$, $B = B_m$ a $C = C_m$. Povšimněme si, že platí následující vlastnosti:

- $V(H) = X \cup C$
- H je minor G
- okolí každého vrcholu z C je v H klika a má velikost nanejvýš k
- $H - C$ má alespoň $|B|$ hran (když přidáváme v_j do B_j, pak také kontrahujeme hranu mezi v_j a x_j; $G_j - B$ proto obsahuje hranu $x_j y_j$, a má tedy alespoň o jednu hranu víc než G_{j-1})

Nyní ověříme, zda $H - C$ je k-degenerovaný. Pokud ne, G má ne-k-degenerovaný minor a skončíme. Jinak (Lemma 2(c)) má $H - C$ nejvýše $k|X|$ hran, a proto $|B| \leq k|X|$ a $|C| = n - |X| - |B| \geq n - (k+1)|X|$. Dále (Lemma 2(d)), $H - C$ obsahuje nejvýše $2^k|X|$ klik.

Pro každou kliku K v $H - C$ si definujme C_K jako množinu vrcholů v C, jejichž okolí je rovno K. Položme

$$A = C \setminus \bigcup_{|C_K| < k} C_K.$$

Jelikož tím pro každou kliku v $H - C$ odstraníme nejvýše $k - 1$ vrcholů, máme $|A| \geq |C| - (k-1)2^k|X| \geq n - (k + 1 + (k-1)2^k)|X|$. Množinu A si nyní rozložme na maximální navzájem disjunktní podmnožiny A_1, A_2, ... takové, že všechny vrcholy v A_i mají stejnou množinu sousedů. Z konstrukce A máme $|A_i| \geq k$ pro každé i. Jelikož $\bigcup_i A_i = A$, máme $\sum_i |A_i| = |A| \geq n - (k + 1 + (k-1)2^k)|X|$, a množiny A_i tedy splňují všechny podmínky zadání.

Snadno si rozmyslíme, že tento algoritmus lze implementovat v lineárním čase (zde využíváme fakt, že G je k-degenerovaný, a má tedy pouze lineární počet hran; dále pak díky volbě N_j tak, že $|N_j| \leq k + 1$, lze rozhodnout zda v_j patří do B nebo C v konstantním čase). $\square$

Dále použijeme následující pozorování.

Lemma 4. *Nechť G je graf na n vrcholech a $A_1, A_2, \ldots \subseteq V(G)$ jsou disjunktní podmnožiny velikosti alespoň k takové, že $A = \bigcup_i A_i$ je nezávislá množina a pro každé i, všechny vrcholy v A_i mají stejnou množinu sousedů a těchto sousedů je nejvýše k. Nechť H je graf získaný z $G - A$ tak, že pro každé i přidáme kliku na sousedy vrcholů v A_i. Pak H je minor G a z libovolného stromového rozkladu H šířky nanejvýš k lze v lineárním čase vyrobit stromový rozklad G šířky nanejvýš k.*

Důkaz. Pro každé A_i si vybereme libovolné párování M_i mezi A_i a jeho sousedy, které pokrývá všechny sousedy (máme $|A_i| \geq k$ a sousedů A_i je nejvýše k, proto takové párování existuje). Označme $M = \bigcup_i M_i$. Pak H vznikne z G kontrakcí hran M a smazáním zbylých vrcholů A, a je to tedy minor G.

Nechť (T, f) je stromový rozklad H šířky nejvýše k. Okolí každého $a \in A$ indukuje kliku v H, a tedy existuje uzel $u_a \in V(T)$ takový, že $f(u_a)$ obsahuje všechny sousedy a. Nechť strom T' je vytvořen z T tak, že pro každé $a \in A$ přidáme nový uzel v_a a hranu $u_a v_a$. Definujme f' tak, že $f'(u) = f(u)$ pro $u \in V(T)$ a $f'(v_a)$ je tvořeno vrcholem a a jeho okolím pro každé $a \in A$. Pak (T', f') je stromový rozklad G šířky nejvýše k. $\square$

Lineární algoritmus, který pro vstupní graf G rozhodne, zda jeho stromová šířka je nejvýše $k \geq 1$, a pokud ano, vrátí příslušný rozklad, je nyní již jednoduchý. Označme

$$\varepsilon = \frac{1}{2(k + 1 + (k-1)2^k)}.$$

Algoritmus 2.

- *Pokud G není k-degenerovaný, pak jeho stromová šířka je větší než k.*
- *Hladovým algoritmem najdeme maximální (ne nutně největší) párování M v G.*
- *Jestliže $|V(M)| > \varepsilon n$, pak postupujeme stejně jako v Algoritmu 1.*
- *Jinak aplikujeme algoritmus z Lemma 3 s $X = V(M)$.*
  - *Dostaneme-li ne-k-degenerovaný minor G, pak stromová šířka G je větší než k.*
  - *Jinak dostaneme množiny $A_1, A_2, \ldots$, jejichž sjednocení A má velikost alespoň $n - (k+1+(k-1)2^k)|V(M)| \geq n/2$. Nechť H je graf získaný z $G - A$ tak, že pro každé i přidáme kliku na sousedy vrcholů v A_i. Rekurzivně se zavoláme na H.*
    - *Pokud H má stromovou šířku větší než k, pak G má stromovou šířku větší než H (dle Lemma 4 je H minor G).*
    - *Jinak dostaneme stromový rozklad H šířky nejvýše k, který přepracujeme na rozklad G dle Lemma 4.*

Průběh algoritmu až na rekurzivní volání zabere čas $O(n)$. Rekurzivně se voláme na graf velikosti nejvýše $(1-\varepsilon/2)n$ když $|V(M)| > \varepsilon n$ a nejvýše $n/2 \leq (1-\varepsilon/2)n$ když $|V(M)| \leq \varepsilon n$. Celková časová složitost je tedy $n + (1-\varepsilon/2)n + (1-\varepsilon/2)^2 n + \ldots = \frac{2}{\varepsilon}n = O(n)$.

Reference

[1] S. Arnborg, D. Corneil, A. Proskurowski: Complexity of finding embeddings in a k-tree, *SIAM Journal on Matrix Analysis and Applications* **8** (1987), 277–284.

[2] H. Bodlaender, T. Kloks: Efficient and Constructive Algorithms for the Pathwidth and Treewidth of Graphs, *J. Algorithms* **21** (1996), 358–402.

[3] H. Bodlaender: A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth, *SIAM J. Comput.* **25** (1996), 1305–1317.