

1 Kapitola 1

2 Složitost

3 V této kapitole ukážeme několik výsledků ohledně časové a prostorové slo-
4 žitosti. Problém klastrové rovinnosti patří do třídy NP z pohledu časové
5 složitosti a z hlediska prostorového se dá řešit v prostoru $\mathcal{O}(n)$ na determi-
6 nistickém stroji, kde n je počet vrcholů.

7 Jako výchozí model uvažujeme RAM s logaritmickou velikostí paměťo-
8 vých buněk vzhledem k velikosti vstupu a jednotkovou cenou za aritmetické
9 operace s čísly, případně jeho nedeterministickou verzi nRAM. Tedy polyno-
10 miálně velká čísla lze uložit a operovat s nimi v konstantním prostoru a čase.
11 Je to standardní model u tohoto typu problémů, díky němuž lze říci, že graf
12 s n vrcholy a m hranami je uložen v prostoru $\mathcal{O}(n + m)$.

13 1.1 Datová reprezentace

14 Nejprve uvedeme možnosti reprezentace klastrového grafu a ujasníme si,
15 vzhledem k čemu budeme vztahovat příslušnou složitost. Pro reprezentaci
16 klastrové hierarchie se nabízí dvě možnosti.

17 1. Seznamy vrcholů

18 2. Strom, kde listy představují vrcholy a vnitřní uzly představují klastry

- 19 • Zde předpokládejme, že kořen tohoto stromu reprezentuje klastr
20 obsahující všechny vrcholy.

21 V této kapitole budeme několikrát hovořit o maximálních podklastrech
 22 (vzhledem na inkluzi) v nějakém klastru K . Proto si zavedeme následující
 23 definici.

24 **Definice 1.1.** Pod *maximálním podklastrem vzhledem ke klastru K* myslíme
 25 klaster takový, že je přímým potomkem K v klastrové hierarchii.

26 Nejprve musíme určit, kolik klastrů se v klastrové hierarchii může nachá-
 27 zet. Pro zjednodušení budeme předpokládat, že v klastrové hierarchii máme
 28 vždy klaster obsahující všechny vrcholy a každá jednovrcholová množina je
 29 též klastrem v klastrové hierarchii.

30 **Tvrzení 1.2.** *Maximální počet klastrů v grafu G s n ($n \geq 1$) vrcholy je $2n - 1$.*

31 *Důkaz.* Důkaz indukcí podle n :

32 Základ indukce: $n=1$

33 Zjevně platí.

34 Indukční předpoklad: Tvrzení platí pro $|V| < n$.

35 Indukční krok:

36 Díky předpokladům víme, že klaster, který obsahuje aspoň dva vrcholy, má
 37 aspoň dva maximální podklastry.

38 Máme graf s n vrcholy. Podle předpokladu máme klaster K obsahující všechny
 39 vrcholy. Ten obsahuje k vzájemně disjunktních maximálních podklastrů. Ve-
 40 likost i -tého klastru nechť je k_i . Každý z těchto klastrů obsahuje méně než n
 41 vrcholů. Platí pro ně tedy indukční předpoklad. Máme tedy:

$$42 \text{ počet klastrů } \leq 1 + \sum_{i=1}^k (2 * k_i - 1) = 1 + 2 * \sum_{i=1}^k k_i - k = 2n - k + 1$$

43 K maximalizování dojde pokud bude vždy $k = 2$, což odpovídá situaci, kde
 44 každý klaster, který není listem, má právě dva maximální podklastry. \square

45 Velikost grafu na vstupu je $\mathcal{O}(n+m+|\mathcal{C}|)$, \mathcal{C} je klastrová hierarchie a $|\mathcal{C}|$ je
 46 velikost její reprezentace. První varianta reprezentace klastrové hierarchie má
 47 za následek, že klastrová hierarchie zabírá prostor až $\mathcal{O}(n^2)$. Příkladem takové
 48 klastrové hierarchie je graf, kde klastry jsou postupně do sebe vnořené. První
 49 klaster obsahuje všechny vrcholy, druhý o vrchol méně, třetí o další vrchol,
 50 atd.. Druhá varianta reprezentace naproti tomu dává prostor $\mathcal{O}(n)$. Nejvíce
 51 nám tedy o časové a prostorové složitosti problému prozradí, když budeme
 52 složitost vyjadřovat vzhledem k počtu vrcholů vstupního grafu. Dostali jsme,
 53 že klastrový graf (G, \mathcal{C}) , kde G je rovinný graf, lze reprezentovat v prostoru
 54 $\mathcal{O}(n)$.

55 Dále v textu budeme pracovat výhradně se stromovou reprezentací klastrové
56 hierarchie.

57 1.2 Časová složitost

58 Hlavním výsledkem této části je lineární nedeterministický algoritmus pro
59 klastrovou rovinnost.

60 **Tvrzení 1.3.** *Problém rozhodnutí existence rovinného klastrového nakreslení*
61 *patří do třídy NP.*

62 *Důkaz.* Využíváme toho, že ekvivalentním problémem ke klastrové rovin-
63 nosti je existence saturátoru. Ten nám zajistí, že klastry jsou souvislé. Saturá-
64 tor dostaneme jako certifikát. Vzhledem k tomu, že klastrů je polynomiálně
65 mnoho, tak ověření saturátoru se dá provést v polynomiálním čase (například
66 otestováním souvislosti každého klastru zvlášť). Dále jsou algoritmy testující
67 klastrovou rovinnost v polynomiálním čase (viz věta 1.6), pokud klastry jsou
68 souvislé. \square

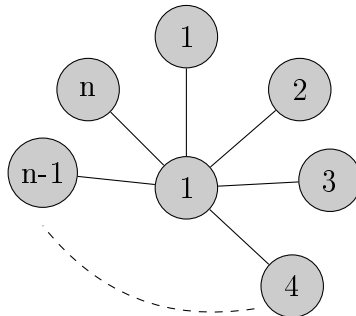
69 **Tvrzení 1.4.** *Pro problém klastrové rovinnosti je nedeterministický algorit-*
70 *mus, jehož časová složitost je $\mathcal{O}(n)$.*

71 *Důkaz.* Důkaz tohoto tvrzení je pouze doplněním důkazu, že klastrová rovin-
72 nost je v NP. Pro důkaz je třeba ukázat, že umíme ověřit souvislost všech
73 klastrů v čase $\mathcal{O}(n)$, a pak že klastrová rovinnost se dá otestovat v lineárním
74 čase, pokud jsou klastry souvislé. Druhá část viz věta 1.6.

75 Prosté otestování všech klastrů zvlášť na souvislost vede na algoritmus s
76 časovou složitostí $\mathcal{O}(n^2)$, protože klastrů je až lineárně mnoho a jejich celková
77 velikost je až kvadratická. Pro zlepšení půjdeme cestou, kdy budeme testo-
78 vat souvislost klastrů od nejmenších k největším. A po otestování klastru
79 na souvislost daný klastř zkontrahujeme do jediného vrcholu, abychom při
80 testování nadklastrů nemuseli opětovně procházet přes vrcholy otestovaného
81 klastru.

82 Při testování klastru na souvislost použijeme klasický algoritmus na tes-
83 tování souvislosti. Hrany, které vedou ven z klastru, si při průchodu jen za-
84 pamatujeme, a po doběhnutí testu je aktualizujeme, tedy nasměrujeme je
85 do nového vrcholu vzniklého kontrakcí klastru. Časová složitost pro jeden
86 klastř C je $\mathcal{O}(n_C + m_C + \text{počet hran ven z klastru})$, kde n_C je počet vrcholů
87 klastru a m_C je počet hran mezi vrcholy klastru. Problémem je, že tohle stále

88 vede na algoritmus s kvadratickou časovou složitostí (viz obrázek 1.1). Je to
 89 způsobeno tím, že se hrany můžou aktualizovat příliš často.



Obrázek 1.1: Klastrový graf, pro který poběží kvadraticky dlouho vzhledem k počtu vrcholů algoritmus s aktualizací hran. Je to hvězda, kde i -tý klastr je tvořen vrcholy s čísly nejvýše i . Důvodem neefektivity je to, že kontrakce vždy zasáhne středový vrchol hvězdy a všechny hrany se musí přesměrovat do nového vrcholu vzniklého kontrakcí.

90 Nyní uvedeme algoritmus s lineární časovou složitostí. Ten vychází z před-
 91 chozího pokusu, kde jsme si zdánlivě nepomohli. Pro zlepšení musíme dosáhnout toho, že hrany opakovaně nenavštěvujeme. To provedeme následovně,
 92 pro každý klastr K budeme mít pomocný graf (označme jej G_K), kde vrcholy
 93 představují maximální podklastry daného klastru. Hrany v těchto pomocných
 94 grafech představují hrany jdoucí mezi klastry. Každé hraně $\{x, y\}$ v grafu G
 95 odpovídá hrana v právě jednom pomocném grafu. Abychom mohli určit, do
 96 kterého pomocného grafu hrana patří, tak potřebujeme určit nejmenší klastr,
 97 který sdílí vrcholy příslušné hrany. Navíc také potřebujeme znát podklastry,
 98 kam vrcholy patří. To je ale problém nejmenšího společného předka v zako-
 99 řeněném stromu, kdy potřebné dotazy se provádí v konstantním čase a s
 100 lineárním předvýpočtem a využívající lineární prostor. Jednoduchou úpra-
 101 vou získáme i ty potřebné informace (ty podklastry). (viz [?] a [?])

103 Náš algoritmus tedy napřed provede předvýpočet potřebný pro problém
 104 hledání minimálního společného předka, kde se hrany rozdělí do pomocných
 105 grafů. Následně se pro každý pomocný graf provede test souvislosti. První
 106 část algoritmu pracuje v čase $\mathcal{O}(n)$ díky tomu, že umístění hrany do po-
 107 mocného grafu umíme provést v konstantním čase a hran je pouze $\mathcal{O}(n)$.
 108 Druhá část algoritmu pracuje v čase $\sum_{C \in \mathcal{C}} (n_{G_C} + m_{G_C}) = \sum_{C \in \mathcal{C}} n_{G_C} + \sum_{C \in \mathcal{C}} m_{G_C} \leq$

109 počet klastrů $+ m = \mathcal{O}(n) + \mathcal{O}(n) = \mathcal{O}(n)$, zde n_{G_C} značí počet vrcholů po-
110 mocného grafu a m_{G_C} počet jeho hran. \square

111 1.3 Prostorová složitost

112 Z výsledků o časové složitosti můžeme říci, že můžeme klastrovou rovin-
113 nost rozhodovat v nedeterministickém prostoru o velikosti $\mathcal{O}(n)$. Ze Savit-
114 chovy věty (viz [?]) plyne, že v deterministickém prostoru stačí nejvýše
115 prostor velikosti $\mathcal{O}(n^2)$. Lepšího výsledku ve smyslu, že potřebujeme méně
116 prostoru, dosáhneme využitím vztahu tříd NTIME a DSPACE, který je
117 $NTIME(t(n)) \subseteq DSPACE(t(n))$. Jelikož máme nedeterministický algo-
118 ritmus pro klastrovou rovinnost pracující v lineárním čase, tak díky přede-
119 šlému víme, že existuje deterministický algoritmus využívající pouze lineárně
120 mnoho prostoru.

121 **Tvrzení 1.5.** *Klastrová rovinnost lze rozhodovat na RAMu s lineárně ome-*
122 *zeným prostorem.*

123 *Důkaz.* \square

124 **Tvrzení 1.6.**