

離散最適化基礎論 第 12 回
木分解構成アルゴリズム

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2017 年 2 月 3 日

最終更新 : 2017 年 2 月 10 日 09:10

- | | | |
|---|-----------------|---------|
| 1 | 離散最適化における木分解の役割 | (10/7) |
| ★ | 休講 (国内出張) | (10/14) |
| 2 | 木に対するアルゴリズム設計 | (10/23) |
| 3 | 道幅と道分解 | (10/30) |
| 4 | 道分解を用いたアルゴリズム設計 | (11/4) |
| ★ | 休講 (海外出張) | (11/11) |
| 5 | 木分解と木幅 | (11/18) |
| ★ | 休講 (調布祭) | (11/25) |
| 6 | 木幅の性質 | (12/2) |

- | | | |
|----|---------------------|---------|
| 7 | 木分解を用いたアルゴリズム設計 | (12/9) |
| 8 | 木分解を用いたアルゴリズム設計：連結性 | (12/16) |
| ★ | 休講 (天皇誕生日) | (12/23) |
| ★ | 冬季休業 | (12/30) |
| 9 | 木幅と論理：単項二階論理 | (1/6) |
| ★ | 休講 (センター試験準備) | (1/13) |
| 10 | 木幅と論理：オートマトン | (1/20) |
| 11 | 木幅と論理：アルゴリズム設計 | (1/27) |
| 12 | 木分解構成アルゴリズム | (2/3) |
| 13 | 固定パラメータ・アルゴリズムと木幅 | (2/10) |
| ★ | 期末試験 | (2/17) |

注意：予定の変更もありうる

主題

離散最適化のトピックの1つとして

グラフの木分解を取り上げ、

- ▶ 木分解とは何か？
- ▶ 木分解がなぜ役に立つのか？
- ▶ 木分解がどう役に立つのか？

について、**数理的**側面と**計算的**側面の双方を意識して講義する

なぜ講義で取り扱う？

- ▶ 「離散最適化の神髄」だから

この講義のキーワード (と荒っぽい説明)

グラフの木幅	グラフの「木っぽさ」を表す尺度 (の1つ)
グラフの木分解	グラフを「木っぽく」表した構造
動的計画法	木分解上の効率的アルゴリズム
オートマトン	動的計画法に基づくアルゴリズムの解釈
Courcelle の定理	上記と論理学に基づく『メタアルゴリズム』

次の主題が有機的に結びつく面白い話題

- ▶ グラフ
- ▶ アルゴリズム
- ▶ オートマトン
- ▶ 論理 (特に, 有限モデル理論)

ポイント

効率的アルゴリズムが設計できる背景に「美しい数理構造」がある

この講義では, その一端に触れたい

今日の目標

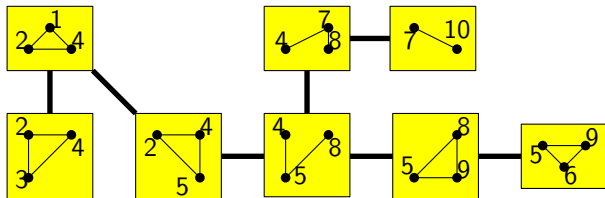
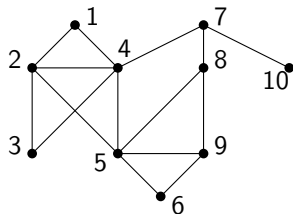
- ▶ 木分解を構成することについて何が知られているか、概観する
- ▶ 木分解を構成する指数時間厳密アルゴリズムを理解する
 - ▶ その設計原理が他の問題にも使えるようになる

- ① 木分解の復習
- ② 木分解を構成するアルゴリズム
- ③ 木分解の構成：指数時間厳密アルゴリズム — 準備
- ④ 木分解の構成：指数時間厳密アルゴリズム
- ⑤ 今日のまとめ

木分解 (tree decomposition) とは？

無向グラフ $G = (V, E)$ の木分解とは木 \mathcal{T} で、

- (T1) \mathcal{T} の節点はどれも V の部分集合
- (T2) 各辺 $\{u, v\} \in E$ に対して、 $u, v \in X$ となる \mathcal{T} の節点 X が存在する
- (T3) 各頂点 $v \in V$ に対して、 \mathcal{T} の節点で v を含むものは \mathcal{T} の (連結で非空な) 部分木を誘導する



木分解の節点を**バッグ** (bag) と呼ぶことがある

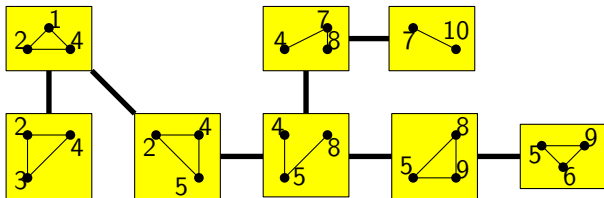
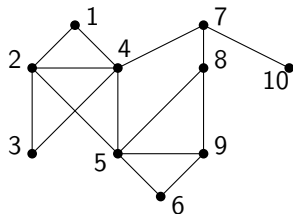
グラフの木幅とは？

- ▶ 無向グラフ G の木分解 \mathcal{T} の幅 (width)

$$\text{tw}(\mathcal{T}) = \max\{|S| - 1 \mid S \text{ は } \mathcal{T} \text{ の節点}\}$$

- ▶ 無向グラフ G の木幅 (treewidth)

$$\text{tw}(G) = \min\{\text{tw}(\mathcal{T}) \mid \mathcal{T} \text{ は } G \text{ の木分解}\}$$

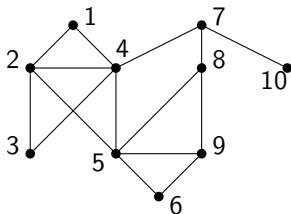


$$\text{tw}(G) = 2$$

- ① 木分解の復習
- ② 木分解を構成するアルゴリズム
- ③ 木分解の構成：指数時間厳密アルゴリズム — 準備
- ④ 木分解の構成：指数時間厳密アルゴリズム
- ⑤ 今日のまとめ

木幅を計算する問題

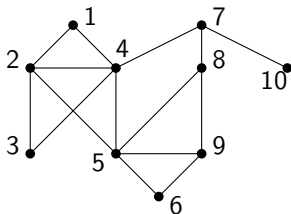
- ▶ 入力：無向グラフ G
- ▶ 出力： G の木幅 $tw(G)$



2

木幅を計算する問題

- ▶ 入力：無向グラフ G
- ▶ 出力： G の木幅 $tw(G)$



→ 2

木幅を計算する問題は NP 困難

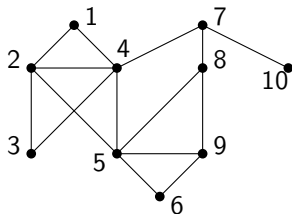
- ▶ つまり、多項式時間では解けない

(Arnborg, Proskurowski '89)

($P \neq NP$ の仮定の下で)

木幅を計算する問題

- ▶ 入力：無向グラフ G
- ▶ 出力： G の木幅 $tw(G)$



2

未解決問題

平面的グラフの木幅は多項式時間で計算できるか？

平面的グラフ：平面上に辺交差なく描けるグラフ

NP 困難問題へのアプローチ

NP 困難問題をどのように「解く」のか？

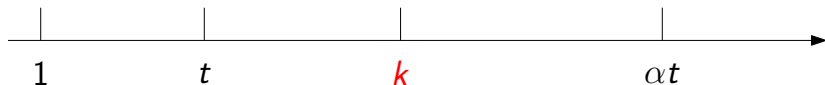
- ▶ **近似アルゴリズム** (approximation algorithm)
 - ▶ 多項式時間で動作することは保証
 - ▶ 出力の質を妥協
- ▶ **厳密アルゴリズム** (exact algorithm)
 - ▶ 多項式時間で動作することは妥協
 - ▶ 出力の質を保証

これらの考え方が組み合わせられることもある

木幅の近似とは？

実数 k が木幅 t の α 近似であるとは、次を満たすこと

$$t \leq k \leq \alpha t$$



- ▶ α を近似比 (approximation ratio) と呼ぶ
- ▶ $\alpha \geq 1$ が小さければ小さいほど、よい近似

多項式時間で達成できる近似比

($n = |V|$)

- ▶ $O(\log n)$ (Bodlaender, Gilbert, Hafsteinsson, Kloks '95)
- ▶ $O(\log \text{tw}(G))$ (Amir '01, '10)
- ▶ $O(\sqrt{\log \text{tw}(G)})$ (Feige, Hajiaghayi, Lee '05, '08)

未解決問題

定数近似比を多項式時間で達成できるか？

指数時間計算量の改善の歴史

($n = |V|$)

▶ $O(2^n \text{poly}(n))$

(割と簡単)

▶ $O(1.8899^n \text{poly}(n))$

(Fomin, Kratsch, Todinca Villanger '08)

▶ $O(1.7549^n \text{poly}(n))$

(Fomin, Villanger '12)

「指数の底」を小さくすることに注力されている

固定パラメータ・アルゴリズムの研究もされている

定理 (Bodlaender '96)

入力として与えられたグラフ G に対して、 $t^{O(t^3)}n$ 時間で次ができる

- ▶ $\text{tw}(G) \leq t$ ならば、幅 t の木分解を構成する
- ▶ $\text{tw}(G) > t$ ならば、「 $\text{tw}(G) > t$ 」であると教えてくれる

ただし、 n は G の頂点数

これは Courcelle の定理や動的計画法の文脈で重要

- ▶ Courcelle の定理や動的計画法を適用するとき、木分解が必要だから

定理 (Bodlaender '96)

入力として与えられたグラフ G に対して、 $t^{O(t^3)}n$ 時間で次ができる

- ▶ $\text{tw}(G) \leq t$ ならば、幅 t の木分解を構成する
- ▶ $\text{tw}(G) > t$ ならば、「 $\text{tw}(G) > t$ 」であると教えてくれる

ただし、 n は G の頂点数

第7回講義より

無向グラフ $G = (V, E)$ の最大独立集合の要素数は、
 G の素敵な木分解 \mathcal{T} が与えられていれば、
 $O(2^t t^2 |V|)$ 時間で計算できる

($t = \text{tw}(\mathcal{T})$)

この2つをまとめると次がいえる

G の木幅が t であるとき、 $t^{O(t^3)}n$ 時間で
 G の最大独立集合の要素数が分かる (t を事前に知る必要はない)

固定パラメータ近似アルゴリズムの研究もされている

最近の定理

(Bodlaender, Drange, Dregi, Fomin, Lokshtanov, Pilipczuk '16)

入力として与えられたグラフ G に対して、 $2^{O(t)}n$ 時間で次ができる

- ▶ $tw(G) \leq t$ ならば、幅 $5t + 4$ 以下の木分解を構成する
- ▶ $tw(G) > t$ ならば、「 $tw(G) > t$ 」であると教えてくれる

ただし、 n は G の頂点数

これを使えば次が導ける

帰結

G の木幅が t であるとき、 $2^{O(t)}n$ 時間で
 G の最大独立集合の要素数が分かる (t を事前に知る必要はない)

指数時間計算量の改善の歴史

($n = |V|$)

▶ $O(2^n \text{poly}(n))$

(割と簡単)

▶ $O(1.8899^n \text{poly}(n))$

(Fomin, Kratsch, Todinca Villanger '08)

▶ $O(1.7549^n \text{poly}(n))$

(Fomin, Villanger '12)

「指数の底」を小さくすることに注力されている

今から行うこと

$O(2^n \text{poly}(n))$ 時間厳密アルゴリズムの紹介

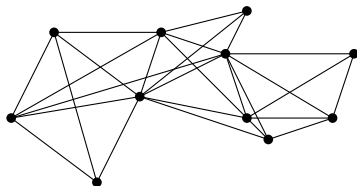
- ① 木分解の復習
- ② 木分解を構成するアルゴリズム
- ③ 木分解の構成：指数時間厳密アルゴリズム — 準備
- ④ 木分解の構成：指数時間厳密アルゴリズム
- ⑤ 今日のまとめ

無向グラフ $G = (V, E)$ **k 木 (k-tree) とは？** G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3 木

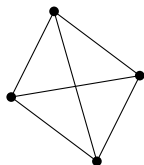


無向グラフ $G = (V, E)$ k 木 (k -tree) とは？ G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3木

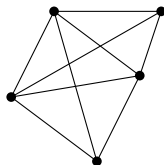


無向グラフ $G = (V, E)$ k 木 (k -tree) とは？ G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3 木

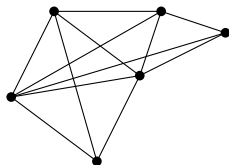


無向グラフ $G = (V, E)$ **k 木 (k-tree) とは？** G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ ，完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3 木

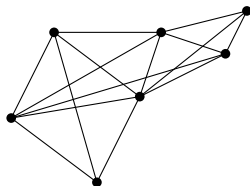


無向グラフ $G = (V, E)$ k 木 (k -tree) とは？ G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ ，完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3木

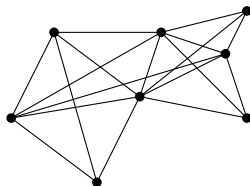


無向グラフ $G = (V, E)$ **k 木 (k-tree) とは？** G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3 木

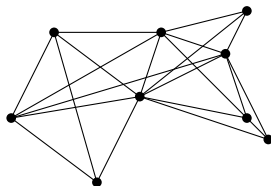


無向グラフ $G = (V, E)$ **k 木 (k-tree) とは？** G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3 木

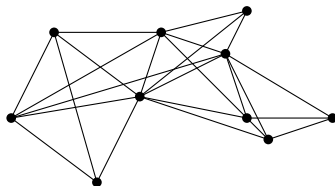


無向グラフ $G = (V, E)$ **k 木 (k-tree) とは？** G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

例：3 木

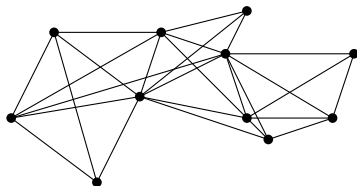


無向グラフ $G = (V, E)$ **k 木 (k-tree) とは？** G が k 木であるとは、次のいずれかを満たすこと

- ▶ G は頂点数 $k + 1$ の完全グラフである
- ▶ k 木 $G' = (V', E')$ と、頂点 $v \in V - V'$ 、完全部分グラフを誘導する G' の頂点部分集合 $\{w_1, \dots, w_k\}$ が存在して、

$$V = V' \cup \{v\}, \quad E = E' \cup \{\{v, w_i\} \mid i \in \{1, \dots, k\}\}$$

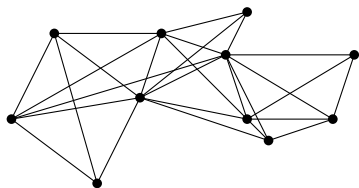
例：3 木



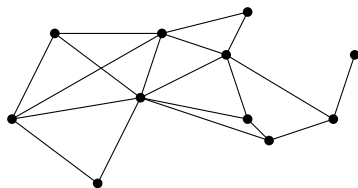
無向グラフ $G = (V, E)$

部分 k 木 (partial k -tree) とは？

G が部分 k 木であるとは、それが k 木の部分グラフであること



3木



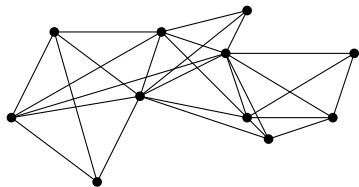
部分 3木

無向グラフ G

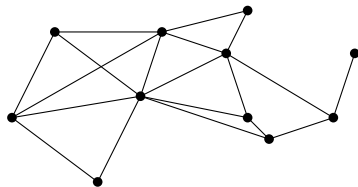
目標とする定理

任意の自然数 $k \geq 1$ に対して

$$G \text{ は部分 } k \text{ 木} \quad \Leftrightarrow \quad \text{tw}(G) \leq k$$



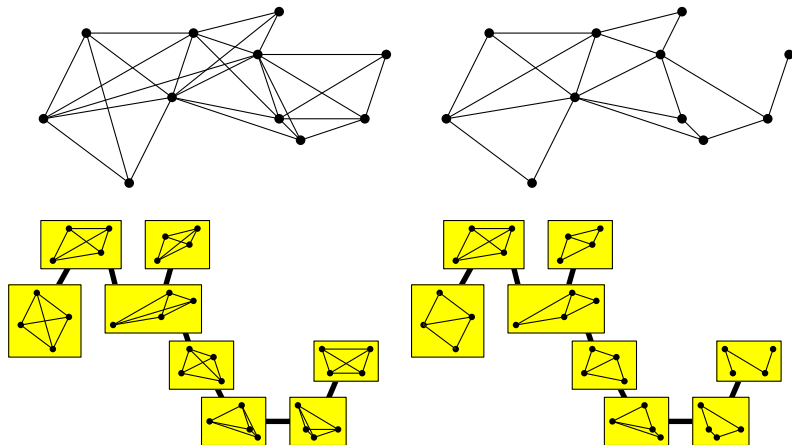
3木



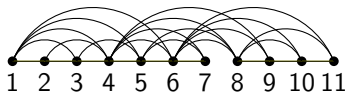
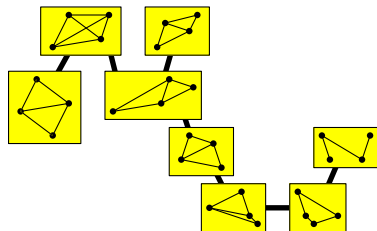
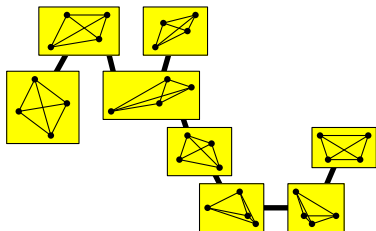
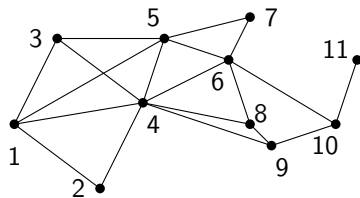
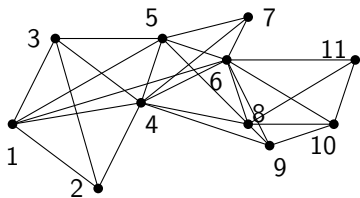
部分 3木

つまり, $\text{tw}(\text{部分 } 3 \text{ 木}) \leq 3$

復習：証明から得られる木分解



証明から得られる線形順序



無向グラフ $G = (V, E)$

線形順序から得られる木幅

- ▶ 頂点を1列に並べ、線形順序 $L: v_1, v_2, \dots, v_n$ を得る
- ▶ 各添え字 i に対して、次を考える

$$\deg_L(v_i) := |\{j \mid j < i, \{v_j, v_i\} \in E\}|$$

つまり、 v_i よりも「前」にある v_j の隣接頂点数

- ▶ その最大値が線形順序から得られる木分解 \mathcal{T}_L の幅

$$\text{tw}(\mathcal{T}_L) = \max_i \deg_L(v_i)$$

つまり、次が成り立つ

$$\text{tw}(G) = \min_L \text{tw}(\mathcal{T}_L)$$

$\therefore \text{tw}(\mathcal{T}_L)$ を最小化する線形順序 L を見つければよい

頂点集合上の線形順序 $L: v_1, v_2, \dots, v_n$ に対して
 L から得られる木分解を \mathcal{T}_L とすると

$$\begin{aligned} \text{tw}(\mathcal{T}_L) &= \max_i \deg_L(v_i) \\ \therefore \text{tw}(G) &= \min_L \text{tw}(\mathcal{T}_L) \end{aligned}$$

$\text{tw}(\mathcal{T}_L)$ を最小化する線形順序 L を求めればよい

簡単なアルゴリズム

- 1 すべての線形順序 L を考える
- 2 各 L に対して $\text{tw}(\mathcal{T}_L)$ を計算し、最大値を出力

計算量: $O(n!m)$

$(n = |V|, m = |E|)$

目標: この計算量を改善する

- ① 木分解の復習
- ② 木分解を構成するアルゴリズム
- ③ 木分解の構成：指数時間厳密アルゴリズム — 準備
- ④ 木分解の構成：指数時間厳密アルゴリズム
- ⑤ 今日のまとめ

- ▶ 頂点を1つずつ追加して、線形順序 L を作っていく
- ▶ しかし、 v_i を追加するとき、 v_1, \dots, v_{i-1} がどのように並んでいるか、
ということは、 $\deg_L(v_i)$ の計算に無関係

⇒ つまり、順序 v_1, \dots, v_{i-1} を覚える必要はない

- ▶ 集合 $\{v_1, \dots, v_{i-1}\}$ を覚えておけば十分である

⇒ 再帰式

記法

任意の $S \subseteq V$ と $v \in V - S$ に対して

$$f(v, S) = \min \left\{ \max_{u \in S \cup \{v\}} \deg_L(u) \mid \begin{array}{l} L \text{ は } S \cup \{v\} \text{ 上の線形順序で} \\ v \text{ が最後に来るもの} \end{array} \right\}$$

記法の気持ち

- ▶ v : 今から追加する頂点 (v_i)
- ▶ S : 今まで追加された頂点の集合 ($\{v_1, \dots, v_{i-1}\}$)

$$S \subseteq V, v \in V - S$$

再帰式

$S = \emptyset$ のとき

$$f(v, S) = 0$$

$S \neq \emptyset$ のとき

$$f(v, S) = \max \left\{ \min_{u \in S} f(u, S - \{u\}), |\{w \mid w \in S, \{w, v\} \in E\}| \right\}$$

この再帰式に従って、 $f(v, S)$ を $|S|$ が小さい方から順に計算していけばよい

木幅を計算する厳密アルゴリズム

入力：無向グラフ $G = (V, E)$

- 1 任意の集合 $S \subseteq V$ に対して, $|S|$ が小さい方から順に
- 2 任意の頂点 $v \in V - S$ に対して
- 3 $S = \emptyset$ ならば, $f(v, S) = 0$
- 4 $S \neq \emptyset$ ならば,

$$f(v, S) = \max \left\{ \min_{u \in S} f(u, S - \{u\}), |\{w \mid w \in S, \{w, v\} \in E\}| \right\}$$

- 5 $\min_{v \in V} f(v, V - \{v\})$ を出力

各 $f(v, S)$ の計算にかかる時間 $\leq |S| + \deg(v)$ なので、

$$\begin{aligned}
 \text{計算量} &\leq \sum_{S \subseteq V} \sum_{v \in V} (|S| + \deg(v)) \\
 &= \sum_{S \subseteq V} \left(|S| + \sum_{v \in V} \deg(v) \right) \\
 &= \sum_{S \subseteq V} (|S| + 2m) \\
 &= \sum_{i=0}^n \binom{n}{i} (i + 2m) \\
 &\leq 2^n (n + 2m) = O(2^n (n + m))
 \end{aligned}$$

つまり、このアルゴリズムの計算量は $O(2^n \text{poly}(n))$

木幅を計算する厳密アルゴリズム

入力：無向グラフ $G = (V, E)$

- 1 任意の集合 $S \subseteq V$ に対して, $|S|$ が小さい方から順に
- 2 任意の頂点 $v \in V - S$ に対して
- 3 $S = \emptyset$ ならば, $f(v, S) = 0$
- 4 $S \neq \emptyset$ ならば,

$$f(v, S) = \max \left\{ \min_{u \in S} f(u, S - \{u\}), |\{w \mid w \in S, \{w, v\} \in E\}| \right\}$$

- 5 $\min_{v \in V} f(v, V - \{v\})$ を出力

- ▶ このように、「すべての部分集合の上で行う動的計画法」を **Held-Karp 型動的計画法**, **Bellman-Held-Karp 型動的計画法**, **部分集合上動的計画法** (dynamic programming over subsets) と呼ぶ

- ① 木分解の復習
- ② 木分解を構成するアルゴリズム
- ③ 木分解の構成：指数時間厳密アルゴリズム — 準備
- ④ 木分解の構成：指数時間厳密アルゴリズム
- ⑤ 今日のまとめ

今日の目標

- ▶ 木分解を構成することについて何が知られているか、概観する
- ▶ 木分解を構成する指数時間厳密アルゴリズムを理解する
 - ▶ その設計原理が他の問題にも使えるようになる

- ▶ 演習問題をやる
 - ▶ 相談推奨 (ひとりでやらない)
- ▶ 質問をする
 - ▶ 教員は巡回
- ▶ 退室時, 小さな紙に感想など書いて提出する ← 重要
 - ▶ 内容は何でも OK
 - ▶ 匿名で OK

- ① 木分解の復習
- ② 木分解を構成するアルゴリズム
- ③ 木分解の構成：指数時間厳密アルゴリズム — 準備
- ④ 木分解の構成：指数時間厳密アルゴリズム
- ⑤ 今日のまとめ