離散最適化基礎論 第 2 回
木に対するアルゴリズム設計

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2016 年 10 月 21 日

最終更新：2016 年 10 月 21 日　16:34

スケジュール 前半 (予定)

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)

注意：予定の変更もありうる

スケジュール 後半 (予定)

7	木分解を用いたアルゴリズム設計：頂点集合の選択・分割	(12/9)
8	木分解を用いたアルゴリズム設計：辺集合の選択・分割	(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：準備	(2/3)
13	木分解構成アルゴリズム	(2/10)
⋆	期末試験	(2/17?)

注意：予定の変更もありうる

概要

主題

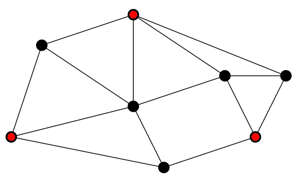
離散最適化のトピックの 1 つとして
グラフの木分解を取り上げ，
- 木分解とは何か？
- 木分解がなぜ役に立つのか？
- 木分解がどう役に立つのか？

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

テーマ：離散最適化問題の解きやすさ と 解きにくさ

最大独立集合問題：隣接しない頂点をできるだけたくさん選ぶ



これは NP 困難問題 (解きにくい問題)

事実

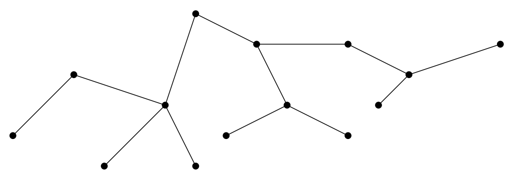
グラフが木 (tree) ならば，簡単に解ける (次回)

直感？

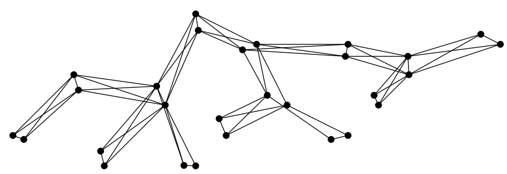
グラフが木に近ければ，簡単に解けそう？

木に近い，とは？

これは木



これは木に近い？ ⇝ 「木っぽさ」を表す尺度を考える必要あり



木幅 — 「木っぽさ」を表す尺度

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』

木幅と木分解の面白さ

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

今日の目標
木に対して，効率的アルゴリズムが設計できるようになる
- 最大独立集合問題
- 最小支配集合問題
- 最大マッチング問題

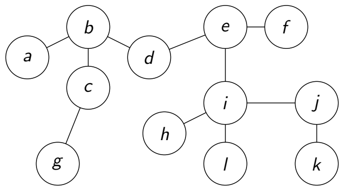
キーワード：再帰，動的計画法

目次

木

無向グラフ $G = (V, E)$

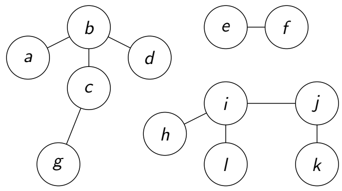
木 (tree) とは？
G が木であるとは，次の 2 つの条件を満たすこと
- G は連結である
- G は閉路を部分グラフとして含まない



森

無向グラフ $G = (V, E)$

森 (forest) とは？
G が森 (または林) であるとは，次の条件を満たすこと
- G は閉路を部分グラフとして含まない

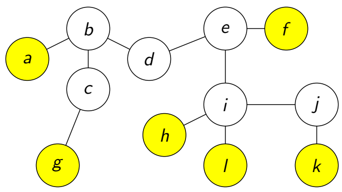


木は森であり，森の各連結成分は木である

木の重要な性質：木と葉

木 $G = (V, E)$，$|V| \geq 2$

木の重要な性質 (1)：木は葉を持つ
G には次数 1 の頂点が 2 つ以上存在する

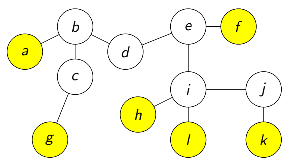


木における次数 1 の頂点を葉 (leaf) と呼ぶ

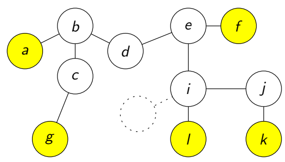
木の重要な性質：葉の除去

木 $G = (V, E)$，$|V| \geq 2$，その葉 $v \in V$

木の重要な性質 (2)：木から葉を除去しても木
G から v を除去したグラフ $G - v$ も木



G

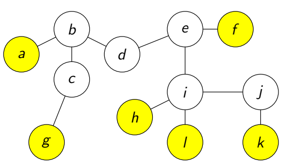


$G - h$

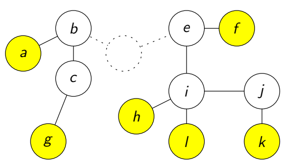
木の重要な性質：葉以外の除去

木 $G = (V, E)$，$|V| \geq 2$，葉ではない頂点 $v \in V$

木の重要な性質 (4)：葉以外の頂点を除去すると非連結
G から v を除去したグラフ $G - v$ は非連結



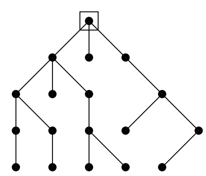
G



$G - d$

根付き木

木の頂点 1 つを特別視して，根付き木 (rooted tree) と見なすことがある



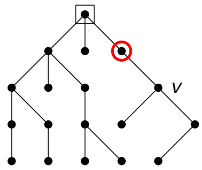
根 (root)：特別視した頂点
- 根付き木 T の根 r を強調するため，
  根付き木を (T, r) と表記することもある

根付き木の用語：親

根付き木 T の頂点 v

親とは？

根ではない頂点 v の親 (parent) とは，v に隣接する頂点で，根に近いもの



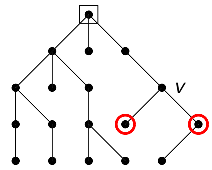
根ではない頂点 v の親はただ 1 つ (根の親は存在しない)

根付き木の用語：子

根付き木 T の頂点 v

子とは？

頂点 v の子 (child) とは，v に隣接する頂点で，根から遠いもの



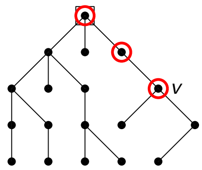
$C(v) = v$ の子全体の集合

根付き木の用語：祖先

根付き木 T の頂点 v

祖先とは？

頂点 v の祖先 (ancestor) とは，v と根を結ぶ道の上にある頂点

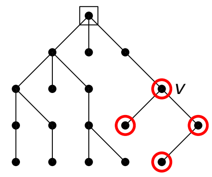


根付き木の用語：子孫

根付き木 T の頂点 v

子孫とは？

頂点 v の子孫 (descendant) とは，v を祖先として持つ頂点



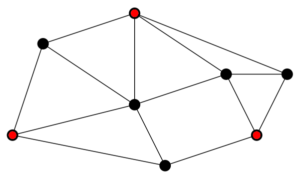
目次

独立集合

$G = (V, E)$ 無向グラフ

独立集合とは？

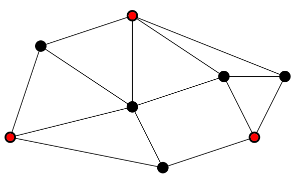
G の独立集合 (independent set) とは，
頂点部分集合 $I \subseteq V$ で，I のどの 2 頂点も隣接しないもの



最大独立集合問題

最大独立集合問題とは？

- 入力：無向グラフ $G = (V, E)$
- 出力：G の最大独立集合 (の要素数)



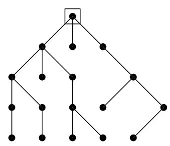
これは NP 困難問題 (解きにくい問題)

木における最大独立集合問題

目標

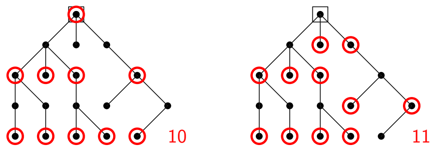
入力が木である場合に限って，最大独立集合問題を効率的に解く

入力として与えられる木を $T = (V, E)$ として，根 $r \in V$ を 1 つ決める



木における最大独立集合問題：よくある間違い

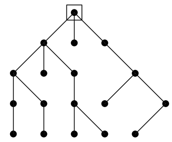
効率的解法：よくある間違い
奇数段目と偶数段目を見て，大きい方を最大独立集合とする



もっとちゃんと考えないといけない

木における最大独立集合問題：基本的な考え方

木 $T = (V, E)$，根 $r \in V$



基本的な考え方：場合分け
X は T の最大独立集合
⇒ 次のどちらかは正しい
1. $r \notin X$　　(X は根を持たない)
2. $r \in X$　　(X は根を持つ)

木における最大独立集合問題：アルゴリズム設計方針
次の 2 つを計算する
1. r を持たない独立集合の中で，要素数最大のもの
2. r を持つ独立集合の中で，要素数最大のもの

そのどちらかが，最大独立集合

木における最大独立集合問題：基本的な考え方 — まとめ

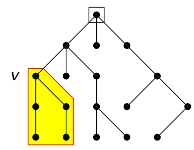
アルゴリズム設計方針 (再掲)
次の 2 つを計算する
1. r を持たない独立集合の中で，要素数最大のもの
2. r を持つ独立集合の中で，要素数最大のもの

そのどちらかが，最大独立集合

記法
$s(r) = T$ の独立集合の最大要素数 (T の最大独立集合の要素数)
$s_1(r) = T$ の独立集合で，r を持たないものの最大要素数
$s_2(r) = T$ の独立集合で，r を持つものの最大要素数

このとき，$s(r) = \max\{s_1(r), s_2(r)\}$

木における最大独立集合問題：基本的な考え方 — 一般化



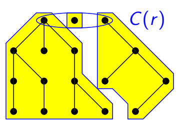
記法：頂点 $v \in V$ に対して
$s(v)$　=　v を根とする T の部分木の独立集合の最大要素数

木における最大独立集合問題：根を持たない独立集合

解く部分問題 (1)
r を持たない独立集合の中で，要素数最大のもの

- X は r の子を根とする部分木の独立集合の合併　($\because$ X は r を持たない)
- つまり，

$$s_1(r) = \sum_{u \in C(r)} s(u)$$



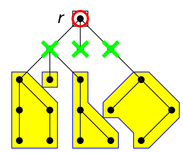
木における最大独立集合問題：根を持つ独立集合

解く部分問題 (2)
r を持つ独立集合の中で，要素数最大のもの

- X は r の子をどれも持てない　($\because$ X は独立集合)
- $\therefore$ X は r の孫 (子の子) を根とする部分木の独立集合と $\{r\}$ の合併
- つまり，

$$s_2(r) = 1 + \sum_{u \in GC(r)} s(u)$$

$GC(r)$ は r の孫全体の集合



木における最大独立集合問題：ここまでのまとめ

ここまでのまとめ

$$s(r) = \max\{s_1(r), s_2(r)\},$$
$$s_1(r) = \sum_{u \in C(r)} s(u),$$
$$s_2(r) = 1 + \sum_{u \in GC(r)} s(u)$$

つまり，

$$s(r) = \max\left\{\sum_{u \in C(r)} s(u), 1 + \sum_{u \in GC(r)} s(u)\right\}$$

要点：$s(r)$ は r の子と孫 u に対する $s(u)$ から簡単に計算できる

木における最大独立集合問題：再帰式

先ほどの式は，根 r 以外の頂点でも同様に成り立つ．つまり，

再帰式
任意の頂点 v に対して

$$s(v) = \max\left\{\sum_{u \in C(v)} s(u), 1 + \sum_{u \in GC(v)} s(u)\right\}$$

この式にしたがって，すべての頂点 v に対して $s(v)$ を計算すればよい

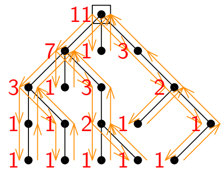
木における最大独立集合問題：アルゴリズム

木の後行順 (post-order) に従って，計算を進めていく

再帰式

任意の頂点 v に対して

$$s(v) = \max\left\{ \sum_{u \in C(v)} s(u), 1 + \sum_{u \in GC(v)} s(u) \right\}$$



$s(r) = 11$

木における最大独立集合問題：アルゴリズム — 計算量解析

木の後行順 (post-order) に従って，計算を進めていく

再帰式

任意の頂点 v に対して

$$s(v) = \max\left\{ \sum_{u \in C(v)} s(u), 1 + \sum_{u \in GC(v)} s(u) \right\}$$

- 木の後行順は $O(|V|)$ 時間で見つけられる
- 各頂点 v に対して，$s(v)$ の値は高々2 回参照される
- $\therefore$ 再帰式の計算にかかる時間 $= O(|V|)$
- $\therefore$ このアルゴリズムの計算量 $= O(|V|)$

線形時間アルゴリズム

木における最大独立集合問題：まとめ

アルゴリズム：動的計画法

1. 根 r を任意に定める
2. 木の後行順 (post-order) に従って，各頂点 v に対して次を計算

$$s(v) = \max\left\{ \sum_{u \in C(v)} s(u), 1 + \sum_{u \in GC(v)} s(u) \right\}$$

3. $s(r)$ を出力

計算量

$O(|V|)$ 時間

「動的計画法」については次を参照 (東京理科大学 小林先生)

- `http://mathopt.sakura.ne.jp/dp.html`

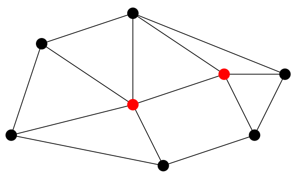
目次

支配集合

$G = (V, E)$ 無向グラフ

支配集合とは？

G の支配集合 (dominating set) とは，頂点部分集合 $D \subseteq V$ で，
$V - D$ のどの頂点も D のある頂点に隣接するもの

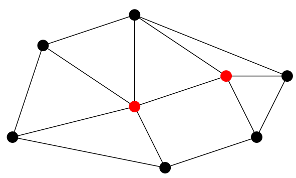


D の頂点 v は，v と v の隣接頂点を支配する，という

最小支配集合問題

最小支配集合問題とは？

- 入力：無向グラフ $G = (V, E)$
- 出力：G の最小支配集合 (の要素数)



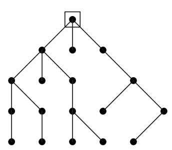
これは NP 困難問題 (解きにくい問題)

木における最小支配集合問題

目標

入力が木である場合に限って，最小支配集合問題を効率的に解く

入力として与えられる木を $T = (V, E)$ として，根 $r \in V$ を 1 つ決める



また，再帰式を導出したい

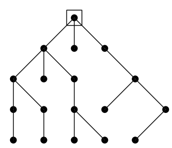
木における最小支配集合問題：基本的な考え方

木 $T = (V, E)$，根 $r \in V$

基本的な考え方：場合分け

X は T の最小支配集合
$\Rightarrow$ 次のどちらかは正しい

1. $r \notin X$ (X は根を持たない)
2. $r \in X$ (X は根を持つ)



木における最小支配集合問題：アルゴリズム設計方針

次の 2 つを計算する

1. r を持たない支配集合の中で，要素数最小のもの
2. r を持つ支配集合の中で，要素数最小のもの

そのどちらかが，最小支配集合

木における最小支配集合問題：基本的な考え方 — まとめ

アルゴリズム設計方針 (再掲)

次の 2 つを計算する

1. r を持たない支配集合の中で，要素数最小のもの
2. r を持つ支配集合の中で，要素数最小のもの

そのどちらかが，最小支配集合

記法

v を根とする部分木を $T(v)$ と書くと

$s(v) = T(v)$ の支配集合の最小要素数 ($T(v)$ の最小支配集合の要素数)
$s_1(v) = T(v)$ の支配集合で，v を持たないものの最小要素数
$s_2(v) = T(v)$ の支配集合で，v を持つものの最小要素数

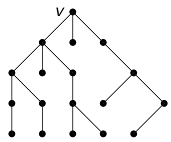
このとき，$s(v) = \min\{s_1(v), s_2(v)\}$ であり，$s(r)$ を最終的に求めたい

木における最小支配集合問題：根を持たない支配集合

解く部分問題 (1)

v を持たない支配集合の中で，要素数最小のもの

- このとき
$$s_1(v) = \sum_{u \in C(v)} s(u)$$
となるかどうかわからない !! (なぜか？)



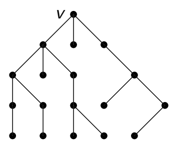
$v \notin X$ ならば，ある $u \in C(v)$ に対して，$u \in X$ とならなければならない

木における最小支配集合問題：根を持たない支配集合 (続)

解く部分問題 (1)

v を持たない支配集合の中で，要素数最小のもの

- このとき $s_1(v) = \min\limits_{u \in C(v)} \left\{ s_2(u) + \sum\limits_{w \in C(v)-\{u\}} s(w) \right\}$



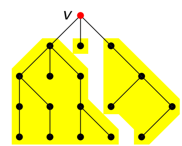
これは $O(|C(v)|)$ 時間で計算できる
(各 $u \in C(v)$ に対して $s(u), s_2(u)$ が分かっていれば)

木における最小支配集合問題：根を持つ支配集合

解く部分問題 (2)

v を持つ支配集合の中で，要素数最小のもの

- このとき，$s_2(v) = 1 + \sum\limits_{u \in C(v)} s(u)$ であるとは限らない



$v \in X$ であるので，部分木において $C(v)$ の頂点を支配する必要はない

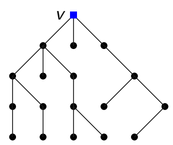
木における最小支配集合問題：基本的な考え方 — 再考

記法

v を根とする部分木を $T(v)$ と書くと

$s(v) = T(v)$ の支配集合の最小要素数 ($T(v)$ の最小支配集合の要素数)
$s_1(v) = T(v)$ の支配集合で，v を持たないものの最小要素数
$s_2(v) = T(v)$ の支配集合で，v を持つものの最小要素数
$s_3(v) = T(v)$ の頂点部分集合で，
$T(v) - v$ の支配集合となるものの最小要素数

このとき，$s(v) = \min\{s_1(v), s_2(v)\}$ であり，$s(r)$ を最終的に求めたい

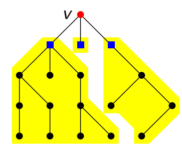


木における最小支配集合問題：根を持つ支配集合 (再考)

解く部分問題 (2)

v を持つ支配集合の中で，要素数最小のもの

- このとき，
$$s_2(v) = 1 + \sum_{u \in C(v)} s_3(u)$$

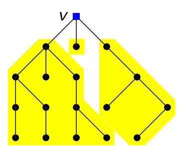


木における最小支配集合問題：根が既に支配されている場合

解く部分問題 (3)

v 以外の頂点を支配する頂点部分集合で，要素数最小のもの

- このとき，
$$s_3(v) = \min \left\{ \sum_{u \in C(v)} s(u), 1 + \sum_{u \in C(v)} s_3(u) \right\}$$



木における最小支配集合問題：ここまでのまとめ

再帰式

$$\begin{aligned}
s(v) &= \min\{s_1(v), s_2(v)\}, \\
s_1(v) &= \min_{u \in C(v)} \left\{ s_2(u) + \sum_{w \in C(v)-\{u\}} s(w) \right\}, \\
s_2(v) &= 1 + \sum_{u \in C(v)} s_3(u), \\
s_3(v) &= \min \left\{ \sum_{u \in C(v)} s(u), 1 + \sum_{u \in C(v)} s_3(u) \right\}
\end{aligned}$$

$s_1(v), s_2(v), s_3(v), s(v)$ を木の後行順に沿って計算すればよい
($s(v)$ だけの式にする必要はない)

木における最小支配集合問題：計算量解析

再帰式

$$s(v) = \min\{s_1(v), s_2(v)\},$$

$$s_1(v) = \min_{u \in C(v)} \left\{ s_2(u) + \sum_{w \in C(v)-\{u\}} s(w) \right\},$$

$$s_2(v) = 1 + \sum_{u \in C(v)} s_3(u), \qquad s_3(v) = \min\left\{ \sum_{u \in C(v)} s(u), 1 + \sum_{u \in C(v)} s_3(u) \right\}$$

計算量

- 後行順は $O(|V|)$ 時間で見つけられる
- 各 v に対して，
  $s_1(v), s_2(v), s_3(v), s(v)$ を計算にかかる時間 $= O(|C(v)|)$
- したがって，計算量は

$$O(|V|) + \sum_{v \in V} O(|C(v)|) = O(|V|) + O(|V|) = O(|V|)$$

木における最小支配集合問題：$s_1(v)$ の計算法

再帰式

$$s_1(v) = \min_{u \in C(v)} \left\{ s_2(u) + \sum_{w \in C(v)-\{u\}} s(w) \right\},$$

アルゴリズム

1. $C(v)$ の要素に適当な順番を付け，$C(v) = \{u_1, u_2, \ldots, u_k\}$ とする　($\leftarrow O(|C(v)|)$ 時間)
2. $s_1(v) \leftarrow +\infty$ とする　($\leftarrow O(1)$ 時間)
3. $S \leftarrow \sum_{w \in C(v)} s(w)$ とする　($\leftarrow O(|C(v)|)$ 時間)
4. $i \in \{1, \ldots, k\}$ に対して，次を反復実行　($\leftarrow O(|C(v)|)$ 回)
   - $s_1(v) \leftarrow \min\{s_1(v), S + s_2(u_i) - s(u_i)\}$ とする　($\leftarrow O(1)$ 時間)
5. $s_1(v)$ を出力　($\leftarrow O(1)$ 時間)

計算量：$O(|C(v)|)$ 時間

木における最小支配集合問題：まとめ

アルゴリズム：動的計画法

1. 根 r を任意に定める
2. 木の後行順に従って，
   各頂点 v に対して $s_1(v), s_2(v), s_3(v), s(v)$ を計算
3. $s(r)$ を出力

計算量

$O(|V|)$ 時間

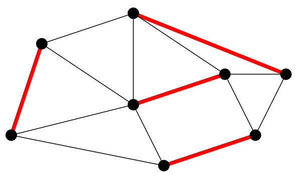
目次

マッチング

$G = (V, E)$ 無向グラフ

マッチングとは？

G のマッチング (matching) とは，
辺部分集合 $M \subseteq V$ で，M のどの 2 辺も端点を共有しないもの



最大マッチング問題

最大マッチング問題とは？

- 入力：無向グラフ $G = (V, E)$
- 出力：G の最大マッチング (の要素数)

これは多項式時間で解けるが，現在最速アルゴリズムの計算量は次の通り

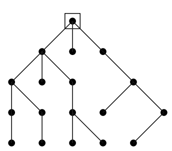
- $O(\sqrt{|V|}|E|)$ 時間 (決定性)　(Micali, Vazirani '80)
- $O(|V|^{\omega})$ 時間 (乱択)　(Mucha, Sankowski '04)
  - $\omega =$ 行列積指数 (matrix multiplication exponent)
    現在最良：$\omega < 2.3728639$　(Le Gall '14)

つまり，多項式時間ながら「遅い」

木における最大マッチング問題

目標

入力が木である場合に限って，最大マッチング問題を効率的に解く



注意

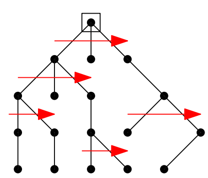
最大独立集合問題	頂点を選ぶ問題	(頂点部分集合を見つける問題)
最大マッチング問題	辺を選ぶ問題	(辺部分集合を見つける問題)

うまく再帰式を作るために，入力の木を「順序木」であると見なす

順序木

順序木 (ordered tree) とは？

根付き木において，各頂点 v の子全体 $C(v)$ に全順序 $\leq_v$ を与えたもの

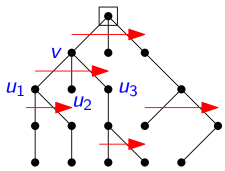


各頂点の子は，全順序に合わせて，左から右に並べて書くことが多い

順序木 (続)

順序木 (ordered tree) とは？

根付き木において，各頂点 v の子全体 $C(v)$ に全順序 $\leq_u$ を与えたもの

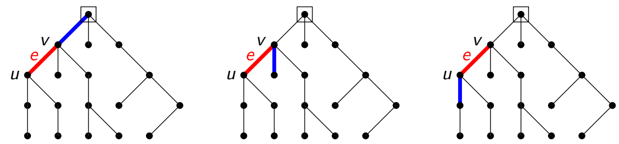


記法：

- 頂点 v の子全体 $C(v) = \{u_1, u_2, u_3\}$ に対して，$u_1 <_u u_2 <_u u_3$
- これを辺にも拡大して，
  $\{v, u_1\} <_u \{v, u_2\} <_u \{v, u_3\}$ と書くこともある

順序木：用語

順序木 $(T, r, \{\leq_v\})$ と
その辺 $e = \{u, v\}$ に対して，v が u の親であるとき，



- e の親辺：v の親と v を結ぶ辺
- e の右妹辺：$\leq_v$ において，e よりも 1 つ大きい辺　(right-sibling)
- e の左子辺：$\leq_u$ において，先頭にある辺　(left-child)

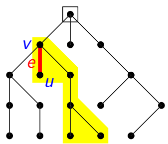
順序木：辺が定める部分木

順序木 $(T, r, \{\leq_v\})$ と
その辺 $e = \{u, v\}$ に対して，v が u の親であるとき，

辺が定める部分木とは？

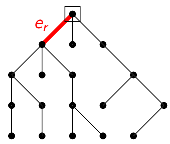
辺 e に対して，e が定める部分木を次のように定義する

- このとき，v の子で，u より大きい (u より右にある) ものをすべて考えて，集合 X とする
- $X \cup \{u\}$ の子孫全体と v が誘導する木を e が定める部分木とする



順序木：辺が定める部分木 — 注意

このとき，唯一の辺 e_r に対して，e_r が定める部分木が T そのものになる
(e_r：根辺)



つまり，任意の辺 e に対して，次を行なえばよい

e が定める部分木において，最大マッチングの要素数の計算

最大独立集合のときと同様に，再帰式を考える

木における最大マッチング問題：アルゴリズム設計方針

アルゴリズム設計方針

各辺 e に対して，e が定める部分木において，次の 2 つを計算する

1. e を持たないマッチングの中で，要素数最大のもの
2. e を持つマッチングの中で，要素数最大のもの

そのどちらかが，e が定める部分木の最大マッチング

記法

e が定める T の部分木を $T(e)$ と書くことにして

$s(e) = T(e)$ のマッチングの最大要素数 ($T(e)$ の最大マッチングの要素数)
$s_1(e) = T(e)$ のマッチングで，e を持たないものの最大要素数
$s_2(e) = T(e)$ のマッチングで，e を持つものの最大要素数

このとき，$s(e) = \max\{s_1(e), s_2(e)\}$

木における最大マッチング問題：根辺を持たないマッチング

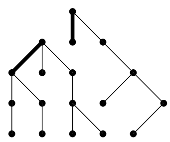
解く部分問題 (1)

e を持たないマッチングの中で，要素数最大のもの

- このとき，

$$s_1(e) = \sum_{f \in LC(e)} s(f) + \sum_{f \in RS(e)} s(f)$$

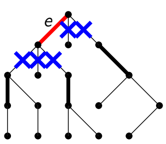
$LC(e)$ は e の左子辺全体の集合，$RS(e)$ は e の右妹辺全体の集合



木における最大マッチング問題：根辺を持つマッチング

解く部分問題 (2)

e を持つマッチングの中で，要素数最大のもの



これはややこしい→ 計算すべきものを再考
(線形時間アルゴリズムができなそう)

木における最大マッチング問題：アルゴリズム設計方針 (改)

記法

e が定める T の部分木を $T(e)$ と書き，
$e = \{u, v\}$ において，v が u の親であるとして，

$s(e) = T(e)$ のマッチングの最大要素数 ($T(e)$ の最大マッチングの要素数)
$s_1(e) = T(e)$ のマッチングで，e を持たないものの最大要素数
$s_2(e) = T(e)$ のマッチングで，e を持つものの最大要素数
$s_3(e) = T(e)$ のマッチングで，e と e の妹をどれも持たないものの最大要素数

このとき，$s(e) = \max\{s_1(e), s_2(e)\}$　$(\because s_1(e) \geq s_3(e))$

木における最大マッチング問題：根辺を持つマッチング (再)

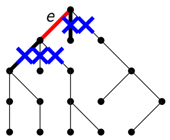
解く部分問題 (2)

e を持つマッチングの中で，要素数最大のもの

- このとき，

$$s_2(e) = \sum_{f \in LC(e)} s_3(f) + \sum_{f \in RS(e)} s_3(f)$$

$LC(e)$ は e の左子辺全体の集合，$RS(e)$ は e の右妹辺全体の集合



木における最大マッチング問題：根辺とその妹を持たないマッチング

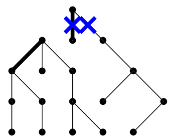
解く部分問題 (3)

e と e の妹をどれも持たないマッチングの中で，要素数最大のもの

- このとき，

$$s_3(e) = \sum_{f \in LC(e)} s(f) + \sum_{f \in RS(e)} s_3(f)$$

$LC(e)$ は e の左子辺全体の集合，$RS(e)$ は e の右妹辺全体の集合



木における最大マッチング問題：ここまでのまとめ

得られた再帰式

$$\begin{aligned}
s(e) &= \max\{s_1(e), s_2(e)\},\\
s_1(e) &= \sum_{f \in LC(e)} s(f) + \sum_{f \in RS(e)} s(f),\\
s_2(e) &= \sum_{f \in LC(e)} s_3(f) + \sum_{f \in RS(e)} s_3(f),\\
s_3(e) &= \sum_{f \in LC(e)} s(f) + \sum_{f \in RS(e)} s_3(f)
\end{aligned}$$

つまり，

木の「下の方」から順番に，$s_1(e), s_2(e), s_3(e), s(e)$ を計算していけばよい

そのためには，順序木の辺集合に対して上手な順序を考える必要あり (演習問題)

木における最大マッチング問題：計算量

得られた再帰式

$$\begin{aligned}
s(e) &= \max\{s_1(e), s_2(e)\},\\
s_1(e) &= \sum_{f \in LC(e)} s(f) + \sum_{f \in RS(e)} s(f),\\
s_2(e) &= \sum_{f \in LC(e)} s_3(f) + \sum_{f \in RS(e)} s_3(f),\\
s_3(e) &= \sum_{f \in LC(e)} s(f) + \sum_{f \in RS(e)} s_3(f)
\end{aligned}$$

計算量解析

- 上手な順序を $O(|V|)$ 時間で見つける (演習問題)
- 各 e に対して，$s_1(e), s_2(e), s_3(e), s(e)$ を計算するために必要な値は高々2つ
- $\therefore$ 計算量 $= O(|V|)$

木における最大マッチング問題：まとめ

アルゴリズム：動的計画法

1. 根 r を任意に定め，順序木と見なす (根辺を e_r とする)
2. 辺集合上のある上手な順序に従って，各辺 e に対して $s_1(e), s_2(e), s_3(e), s(e)$ を計算
3. $s(e_r)$ を出力

計算量

$O(|V|)$ 時間

これは $O(\sqrt{|V|}|E|) = O(|V|^{1.5})$ 時間よりも速い

目次

今日の目標

今日の目標

木に対して，効率的アルゴリズムが設計できるようになる

- 最大独立集合問題
- 最小支配集合問題
- 最大マッチング問題

キーワード：再帰，動的計画法

次回以降

今回の手法を，木分解に対して適用していく

次回

いきなり木分解を扱うのは難しいので，「道分解」を扱う