離散最適化基礎論 第 7 回
木分解を用いたアルゴリズム設計

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2016 年 12 月 9 日

最終更新：2016 年 12 月 16 日　10:35

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	(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：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

今日の目標

今日の目標

木分解を用いた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

キーワード：素敵な木分解，再帰，動的計画法

目次

グラフの木分解：定義

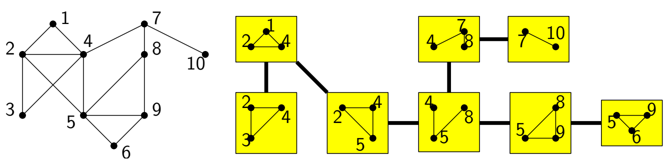
木分解 (tree decomposition) とは？

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

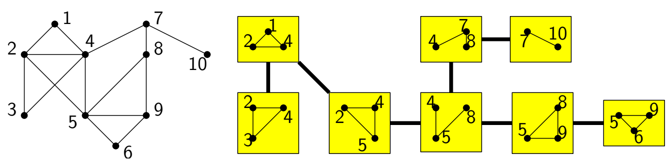
グラフの木幅とは？

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

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

- 無向グラフ G の木幅 (treewidth)

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



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

素敵な木分解 (nice tree decomposition) とは？

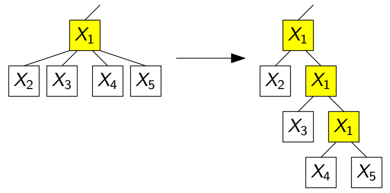
無向グラフ $G=(V,E)$ の木分解 $\mathcal{T}$ が素敵であるとは，
$\mathcal{T}$ の 1 つの節点 X_r を根として $\mathcal{T}$ を根付き木と見なしたときに
次を満たすこと

- $X_r=\emptyset$，かつ，葉である節点 X に対して，$X=\emptyset$
- 各節点の子の数は 2 以下
- 節点 X の子の数が 2 のとき，その子を X', X'' とすると，

$$X = X' = X''$$

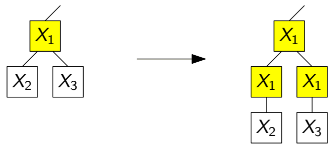
- 節点 X の子の数が 1 のとき，その子を X' とすると，
次のどちらかが成立
  - ある頂点 $v \notin X'$ が存在して，$X = X' \cup \{v\}$
  - ある頂点 $w \in X'$ が存在して，$X = X' - \{w\}$

木分解から，同じ幅の素敵な木分解が (効率よく) 作れる



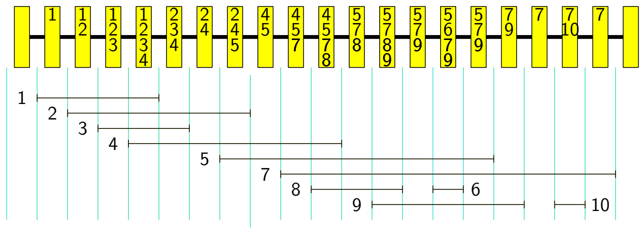
子の数が 3 以上のときは，この操作で子の数を 2 にする

木分解から，同じ幅の素敵な木分解が (効率よく) 作れる

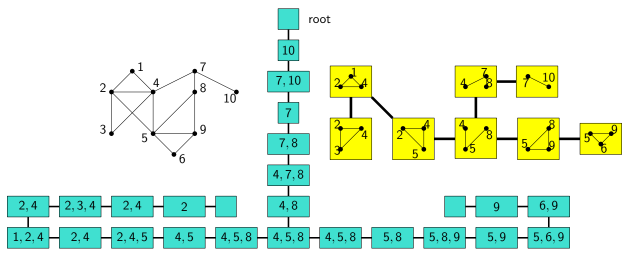


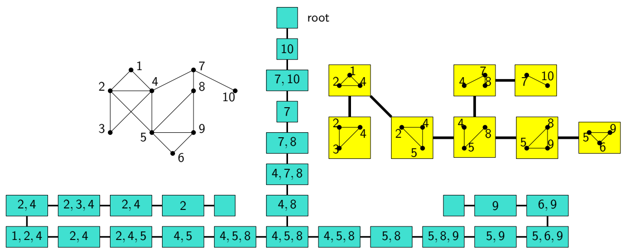
子の数が 2 のとき，親子が同じになるように変形する

木分解から，同じ幅の素敵な木分解が (効率よく) 作れる



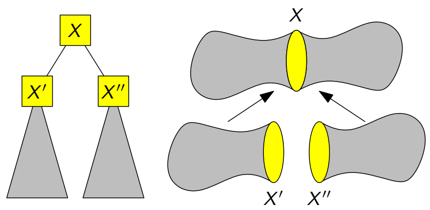
子の数が 1 のところは，素敵な道分解のときと同じように変形する





X の子の数	X の要素数は X の子の要素数より	
2	—	X は結合節点 (join node)
1	大きい	X は導入節点 (introduce node)
1	小さい	X は忘却節点 (forget node)
0	—	X は葉

結合節点 X の子が X', X'' であるとき



木分解を用いたアルゴリズム：基本戦略

入力：無向グラフ G

1. G の素敵な木分解 $\mathcal{T}$ を構成する
2. 木分解 $\mathcal{T}$ 上の動的計画法アルゴリズムを動かす

以下，G の素敵な木分解 $\mathcal{T}$ は与えられるものとする

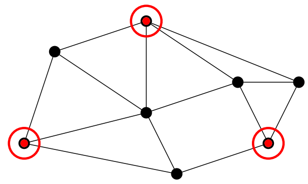
目次

独立集合

$G = (V, E)$ 無向グラフ

独立集合とは？

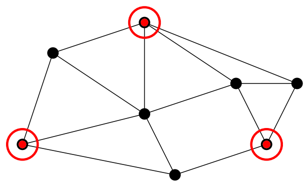
G の独立集合 (independent set) とは，
頂点部分集合 $I \subseteq V$ で，I のどの 2 頂点も隣接しないもの



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)

木分解と最大独立集合問題

目標

素敵な木分解を使って，最大独立集合問題を効率的に解く

設定

- 無向グラフ $G = (V, E)$
- G の 素敵 な木分解 $\mathcal{T}$ (その根を X_r とする)
- $t = \mathtt{tw}(\mathcal{T})$ ($\mathcal{T}$ の幅)

最大独立集合問題：基本的な考え方 (1)

$\mathcal{T}$ の各節点 X_i に対して，

- $\mathcal{T}_i = X_i$ を根とする $\mathcal{T}$ の部分木
- $V_i = \bigcup_{X \in V(\mathcal{T})} X$
- $G_i = G[V_i]$
- $s(i) = G_i$ の最大独立集合の要素数

求めたいもの：$s(r) = G_r$ の最大独立集合の要素数
$\quad = G$ の最大独立集合の要素数

最大独立集合問題：基本的な考え方 (2)

$\mathcal{T}$ の各節点 X_i に対して，

- A_i, B_i が次を満たすとする
$$A_i \cup B_i = X_i, \quad A_i \cap B_i = \emptyset$$
- $s(i; A_i, B_i)$ を次のように定義
$$s(i; A_i, B_i) = \max \left\{ |S_i| \;\middle|\; \begin{array}{l} S_i \subseteq V_i, \\ S_i \text{ は } G_i \text{ の独立集合}, \\ A_i \subseteq S_i, B_i \cap S_i = \emptyset \end{array} \right\}$$
- つまり，$s(i; A_i, B_i)$ は G_i の独立集合の中で，
A_i の頂点をすべて含み，B_i の頂点をどれも含まないものの
最大要素数

最大独立集合問題：導入節点と忘却節点

X_i が導入節点・忘却節点の場合

道分解のときと同じ再帰式が成り立つ

X_i が導入節点の場合
(X_j を X_i の子として，ある $v \notin X_j$ が存在して，$X_i = X_j \cup \{v\}$)

$$s(i; A_i, B_i) = \begin{cases} -\infty & (A_i \text{ に隣接 2 頂点が存在}) \\ s(j; A_i - \{v\}, B_i) + 1 & (A_i \text{ に隣接 2 頂点が非存在,} \\ & \ \text{かつ}, v \in A_i) \\ s(j; A_i, B_i - \{v\}) & (A_i \text{ に隣接 2 頂点が非存在,} \\ & \ \text{かつ}, v \in B_i) \end{cases}$$

X_i が忘却節点の場合
(X_j を X_i の子として，ある $w \in X_j$ が存在して，$X_i = X_j - \{w\}$)

$$s(i; A_i, B_i) = \max\{s(j; A_i \cup \{w\}, B_i), s(j; A_i, B_i \cup \{w\})\}$$

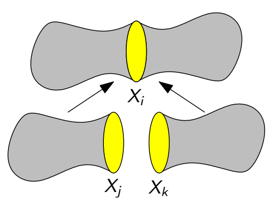
最大独立集合問題：結合節点

X_i が結合節点の場合

木分解に対しては，これを考えなくてはならない
(これさえ考えればよい)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

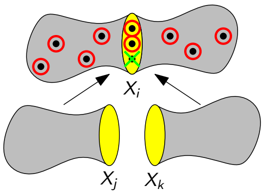
- このとき，$V_i = V_j \cup V_k$
- $V_j - X_i$ と $V_k - X_i$ の頂点を結ぶ辺は存在しない
(第 5 回講義スライド 16 ページ)



最大独立集合問題：結合節点 (続き)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

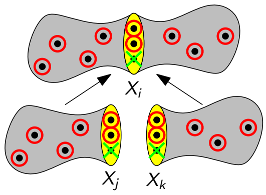
- A_i, B_i が $A_i \cup B_i = X_i$，$A_i \cap B_i = \emptyset$ を満たすとする
- S_j は G_j の独立集合で，A_i の頂点をすべて含み，B_i の頂点をどれも含まないものとする
- このとき，$S_j \cap V_j$ は G_j の独立集合で，A_i の頂点をすべて含み，B_i の頂点をどれも含まない
- 同様に，$S_j \cap V_k$ は G_k の独立集合で，A_i の頂点をすべて含み，B_i の頂点をどれも含まない



最大独立集合問題：結合節点 (続き 2)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

- A_i, B_i が $A_i \cup B_i = X_i$，$A_i \cap B_i = \emptyset$ を満たすとする
- S_j は G_j の独立集合で，A_i の頂点をすべて含み，B_i の頂点をどれも含まないとする
- 同様に，S_k は G_k の独立集合で，A_i の頂点をすべて含み，B_i の頂点をどれも含まないとする
- このとき，$S_j \cup S_k$ は G_i の独立集合で，A_i の頂点をすべて含み，B_i の頂点をどれも含まない



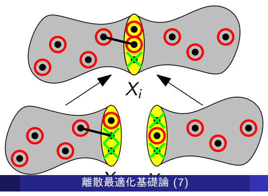
最大独立集合問題：結合節点 (続き 3)

問題？

$S_j \cap X_j \neq A_i$ や $S_k \cap X_k \neq A_i$ の場合は考えなくてよいのか？

回答：考えなくてよい (考えてはいけない)

- $A_j = S_j \cap X_j$，$B_j = X_j - A_j$ とする
- S_j が G_j の独立集合で，A_j を含み，B_j を含まないものの中で，要素数最大のものであるとする
- 任意の $v \in B_j$ を考える
- $S_j \cup \{v\}$ が G_j の独立集合でないならば，$\exists\, w \in S_j : \{v, w\}$ は G_j の辺
- つまり，$v \in S_k$ であると，$S_j \cup S_k$ は G_i の独立集合ではない



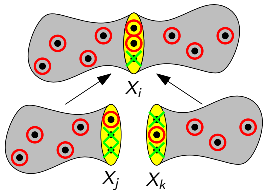
最大独立集合問題：結合節点 (続き 4)

問題？

$S_j \cap X_j \neq A_i$ や $S_k \cap X_k \neq A_i$ の場合は考えなくてよいのか？

回答：考えなくてよい (考えてはいけない)

- $S_j \cup \{v\}$ が G_j の独立集合ならば，$S_j \cup S_k = (S_j \cup \{v\}) \cup S_k$ は G_i の独立集合
- これを繰り返すと，$S_j \cap X_j \neq A_i$ や $S_k \cap X_k \neq A_i$ の場合は考えなくてよいと分かる



最大独立集合問題：結合節点 — 再帰式

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

$$s(i; A_i, B_i) = s(j; A_i, B_i) + s(k; A_i, B_i) - |A_i|$$

これで，アルゴリズムが完成した！

素敵な木分解上の動的計画法アルゴリズムの設計

- 導入節点，忘却節点，結合節点で何をするか，記述すればよい
- 導入節点と忘却節点については，素敵な道分解のときと同じ

最大独立集合問題：計算量

- 素敵な木分解の各節点 X_i において $s(i; A_i, B_i)$ を計算する
- 候補となる (A_i, B_i) の総数 $= 2^{|X_i|} \leq 2^{t+1} = O(2^t)$ (t は木幅)
- 全体の計算量は

$$O(2^t) \cdot (\mathcal{T} \text{ の節点数}) \cdot (\text{再帰式の計算にかかる時間})$$

- 導入節点の場合：1 つの $s(i; A_i, B_i)$ の計算にかかる時間 $\leq O(t)$
- 忘却節点の場合：1 つの $s(i; A_i, B_i)$ の計算にかかる時間 $\leq O(1)$
- 結合節点の場合：1 つの $s(i; A_i, B_i)$ の計算にかかる時間 $\leq O(1)$

$\mathcal{T}$ の節点数は $O(t|V|)$ なので，まとめると，全体の計算量は $O(2^t t^2 |V|)$

まとめ

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

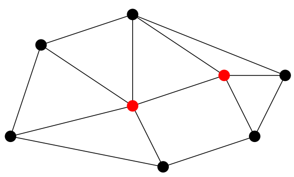
目次

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

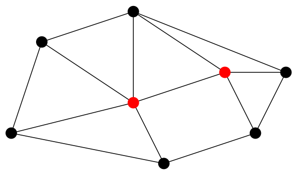


- つまり，$V - D \subseteq N_G(D)$
  $(N_G(D) = \{v \in V \mid \exists\, u \in D\ (\{u, v\} \in E)\})$
- D の頂点 v は，v と v の隣接頂点を支配する，という

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)

木分解と最小支配集合問題

目標

素敵な木分解を使って，最小支配集合問題を効率的に解く

設定

- 無向グラフ $G=(V,E)$
- G の 素敵 な木分解 $\mathcal{T}$ (その根を X_r とする)
- $t = \mathtt{tw}(\mathcal{T})$ ($\mathcal{T}$ の幅)

また，再帰式を導出したい

最小支配集合問題：基本的な考え方 (1)

$\mathcal{T}$ の各節点 X_i に対して，

- $\mathcal{T}_i = X_i$ を根とする $\mathcal{T}$ の部分木
- $V_i = \bigcup_{X \in V(\mathcal{T})} X$
- $G_i = G[V_i]$
- $s(i) = G_i$ の最小支配集合の要素数

求めたいもの：$s(r) = G_r$ の最小支配集合の要素数
$= G$ の最小支配集合の要素数

最小支配集合問題：基本的な考え方 (2)

$\mathcal{T}$ の各節点 X_i に対して，

- A_i, B_i, C_i が次を満たすとする

$$A_i \cup B_i \cup C_i = X_i, \quad A_i, B_i, C_i \text{ は互いに素}$$

- $s(i; A_i, B_i, C_i)$ を次のように定義

$$s(i; A_i, B_i, C_i) = \min \left\{ |S_i| \;\middle|\; \begin{array}{l} S_i \subseteq X_1 \cup \cdots \cup X_i, \\ S_i \text{ は } G_i - C_i \text{ の頂点を支配する}, \\ A_i \subseteq S_i, B_i \cap S_i = \emptyset, \\ S_i \text{ は } C_i \text{ の頂点を支配しない} \end{array} \right\}$$

- つまり，$s(i; A_i, B_i, C_i)$ は $G_i - C_i$ の支配集合の中で，A_i の頂点をすべて含み，B_i の頂点をどれも含まず，C_i を支配しないものの 最小要素数

このとき，$s(i) = \max\{s(i; A_i, B_i, \emptyset) \mid A_i \cup B_i = X_i, A_i \cap B_i = \emptyset\}$

最小支配集合問題：導入節点

X_i が導入節点・忘却節点の場合

道分解のときと同じ再帰式が成り立つ

X_i が導入節点の場合
(X_j を X_i の子として，ある $v \notin X_j$ が存在して，$X_i = X_j \cup \{v\}$)
$s(i; A_i, B_i, C_i)$

$$= \begin{cases} s(j; A_i - \{v\}, B_i - D_i, C_i \cup D_i) + 1 & (v \in A_i) \\ s(j; A_i, B_i - \{v\}, C_i) & (v \in B_i, v \in N_{G_i}(A_i)) \\ \infty & (v \in B_i, v \notin N_{G_i}(A_i)) \\ s(j; A_i, B_i, C_i - \{v\}) & (v \in C_i, v \notin N_{G_i}(A_i)) \\ \infty & (v \in C_i, v \in N_{G_i}(A_i)) \end{cases}$$

ただし，$D_i = \left(N_{G_i}(v) - \bigcup_{w \in A_i - \{v\}} N_{G_i}(w) \right) \cap B_i$

最小支配集合問題：忘却節点

X_i が導入節点・忘却節点の場合

道分解のときと同じ再帰式が成り立つ

X_i が忘却節点の場合
(X_j を X_i の子として，ある $w \in X_j$ が存在して，$X_i = X_j - \{w\}$)

$$s(i; A_i, B_i, C_i) = \min \left\{ \begin{array}{l} s(j; A_i \cup \{w\}, B_i, C_i), \\ s(j; A_i, B_i \cup \{w\}, C_i) \end{array} \right\}$$

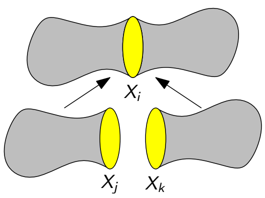
最小支配集合問題：結合節点

X_i が結合節点の場合

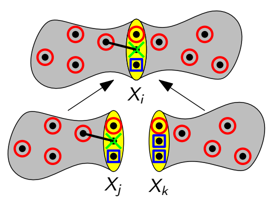
木分解に対しては，これを考えなくてはならない
(これさえ考えればよい)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

- このとき，$V_i = V_j \cup V_k$
- $V_j - X_i$ と $V_k - X_i$ の頂点を結ぶ辺は存在しない
(第 5 回講義スライド 16 ページ)



最小支配集合問題：結合節点 — 考察

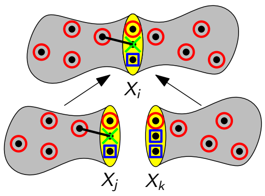


- 3 個組として $(A_i, B_i, C_i) = (A_j, B_j, C_j) = (A_k, B_k, C_k)$ が成り立つとは限らない
- おそらく，$A_i = A_j = A_k$ と $C_i \subseteq C_j, C_k$ は満たされる

最小支配集合問題：結合節点 (続き)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

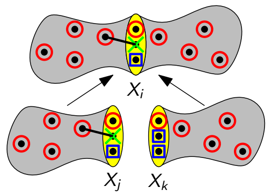
- A_i, B_i, C_i は互いに素で，$A_i \cup B_i \cup C_i = X_i$ を満たすとする
- S_i は $G_i - C_i$ の支配集合で，
A_i の頂点をすべて含み，B_i の頂点をどれも含まず，
C_i の頂点を支配しないとする
- このとき，$S_i \cap V_j$ と $S_i \cap V_k$ は A_i の頂点をすべて含み，
$B_i \cup C_i$ の頂点をどれも含まない



最小支配集合問題：結合節点 (続き 2)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

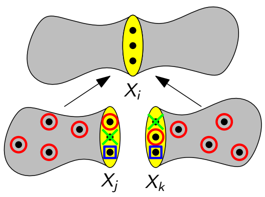
- 互いに素なある $B_j, C_j \subseteq B_i \cup C_i$ が存在して，
$S_i \cap V_j$ は $G_j - C_j$ の支配集合で，B_j の頂点をどれも含まない
- 同様に，互いに素なある $B_k, C_k \subseteq B_i \cup C_i$ が存在して，
$S_i \cap V_k$ は $G_k - C_k$ の支配集合で，B_k の頂点をどれも含まない
- また，$C_j, C_k \supseteq C_i$



最小支配集合問題：結合節点 (続き 3)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

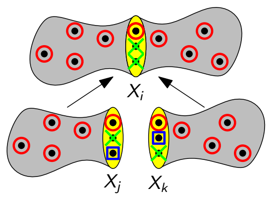
- A_j, B_j, C_j は互いに素で，$A_j \cup B_j \cup C_j = X_i$ を満たすとし，
- S_j は $G_j - C_j$ の支配集合で，
A_j の頂点をすべて含み，B_j の頂点をどれも含まず，
C_j の頂点を支配しないとする
- 同様に，A_k, B_k, C_k を考える



最小支配集合問題：結合節点 (続き 4)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

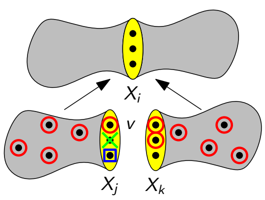
- $A_j = A_k$ のときを考える
- このとき，ある C_i (ただし，$C_i \subseteq C_j, C_i \subseteq C_k$) に対して，
$S_j \cup S_k$ は $G_i - C_i$ の支配集合で，
A_j の頂点をすべて含み，$X_i - (A_j \cup C_i)$ の頂点をどれも含まず，
C_i の頂点を支配しない



最小支配集合問題：結合節点 (続き 4)

$\boxed{\text{問題？}}$：$A_j \neq A_k$ のときは考えなくてもよいのか？

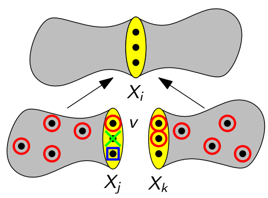
- $A_j \neq A_k$ のときを考えて，$v \in A_j - A_k$ とする
- $v \in B_k$ か $v \in C_k$ が成り立つ
- $v \in B_k$ のとき，$S_k \cup \{v\}$ を考えると，ある $C'_k \subseteq C_k$ が存在して
$S_k \cup \{v\}$ は $G_k - C'_k$ の支配集合で，
$A_k \cup \{v\}$ の頂点をすべて含み，$(B_k - \{v\}) \cup (C_k - C'_k)$ の頂点を
どれも含まず，C'_k の頂点を支配しない
- そして，$S_j \cup S_k = S_j \cup (S_k \cup \{v\})$



最小支配集合問題：結合節点 (続き 5)

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

- $v \in C_k$ のとき，$S_k \cup \{v\}$ を考えると，ある $C'_k \subseteq C_k$ が存在して
$S_k \cup \{v\}$ は $G_k - C'_k$ の支配集合で，
$A_k \cup \{v\}$ の頂点をすべて含み，$B_k \cup (C_k - C'_k)$ の頂点を
どれも含まず，C'_k の頂点を支配しない
- そして，$S_j \cup S_k = S_j \cup (S_k \cup \{v\})$
- つまり，これを繰り返すと，$A_j \neq A_k$ の場合は考えなくてよいこと
が分かる



最小支配集合問題：結合節点 — 再帰式

X_i が結合節点の場合 (X_i の子節点を X_j, X_k とする ($X_i = X_j = X_k$))

$$\begin{aligned} s(i; A_i, B_i, C_i) \ = \ \min\{ & s(j; A_i, B_j, C_j) + s(k; A_i, B_k, C_k) - |A_i| \mid \\ & C_i \subseteq C_j, C_i \subseteq C_k, \\ & B_j = X_j - (A_i \cup C_j), B_k = X_k - (A_i \cup C_k)\} \end{aligned}$$

これで，アルゴリズムが完成した！

素敵な木分解上の動的計画法アルゴリズムの設計

- 導入節点，忘却節点，結合節点で何をするか，記述すればよい
- 導入節点と忘却節点については，素敵な道分解のときと同じ

最小支配集合問題：計算量

- 素敵な木分解の各節点 X_i において $s(i; A_i, B_i, C_i)$ を計算する
- 候補となる (A_i, B_i, C_i) の総数 $= 3^{|X_i|} \leq 3^{t+1} = O(3^t)$ (t は木幅)
- 全体の計算量は

$$O(3^t) \cdot (\mathcal{T} \text{ の節点数}) \cdot (\text{再帰式の計算時間})$$

- 導入節点の場合：1 つの $s(i; A_i, B_i, C_i)$ の計算時間 $\leq O(t)$
- 忘却節点の場合：1 つの $s(i; A_i, B_i, C_i)$ の計算時間 $\leq O(1)$

最小支配集合問題：計算量 (続き)

- 結合節点の場合：
  - C_j の候補の数 $\leq O(2^t)$
  - C_k の候補の数 $\leq O(2^t)$
  - $\therefore$ 1 つの $s(i, A_i; B_i; C_i)$ の計算時間 $\leq O(4^t)$

$\mathcal{T}$ の節点数は $O(t|V|)$ なので，まとめると，全体の計算量は

$$O(3^t) \cdot O(t|V|) \cdot O(4^t) = O(12^t t|V|)$$

まとめ

無向グラフ $G = (V, E)$ の最大支配集合の要素数は，
G の素敵な木分解 $\mathcal{T}$ が与えられていれば，
$O(12^t t|V|)$ 時間で計算できる　　$(t = \mathtt{tw}(\mathcal{T}))$

現在最速のアルゴリズム：$O(3^t t^2|V|)$
(van Rooij, Boalaender, Rossmanith '09)

目次

今日のまとめと次回の予告

今日の目標

木分解を用いた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

キーワード：素敵な木分解，再帰，動的計画法

次回の予告

「木分解と動的計画法」で求めるものに連結性が要求される場合