離散最適化基礎論 第 1 回
離散最適化における木分解の役割

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2016 年 10 月 7 日

最終更新：2016 年 10 月 7 日　07:15

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,

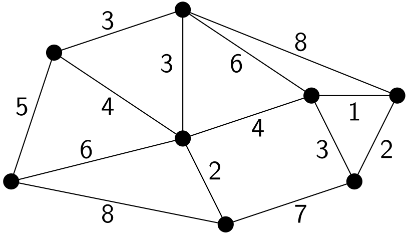
- 木分解とは何か？
- 木分解がなぜ役に立つのか？
- 木分解がどう役に立つのか？

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

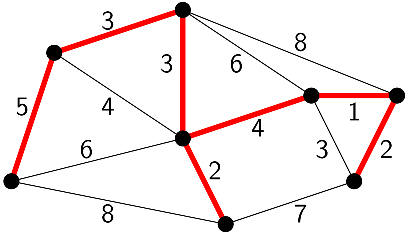
なぜ講義で取り扱う？

- 「離散最適化の神髄」だから

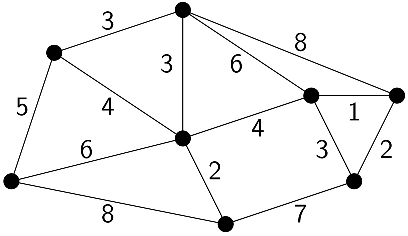
最小全域木問題：すべての頂点間に経路が存在するような
重み和最小のネットワークを作る



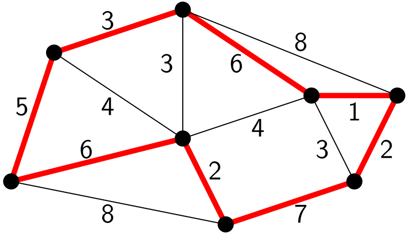
最小全域木問題：すべての頂点間に経路が存在するような
重み和最小のネットワークを作る



巡回セールスマン問題：すべての頂点をちょうど一度ずつ通るような
重み和最小の巡回路を作る



巡回セールスマン問題：すべての頂点をちょうど一度ずつ通るような重み和最小の巡回路を作る



テーマ：離散最適化問題の解きやすさ と 解きにくさ

解きやすい問題

- 最小全域木問題
- 最大マッチング問題
- 最小カット問題
- …

解きにくい問題

- 巡回セールスマン問題
- 最小頂点被覆問題
- 最小彩色問題
- …

「解きやすい」とは

多項式時間解法が存在する

(参考：アルゴリズム論第一・第二)

「解きにくい」とは

NP 困難性が証明されている

(参考：計算理論)

解きやすい問題

- 最小全域木問題
- 最大マッチング問題
- 最小カット問題
- ...

解きにくい問題

- 巡回セールスマン問題
- 最小頂点被覆問題
- 最小彩色問題
- ...

「解きやすい」とは

多項式時間解法が存在する

(参考：アルゴリズム論第一・第二)

「解きにくい」とは

NP 困難性が証明されている

(参考：計算理論)

困ったこと

世の中の「ほとんど」の問題は NP 困難

$\rightsquigarrow$ しかし，いろいろな問題は実際に解けている

理解したいこと

なぜ,「現実問題」は解きやすいのか？

回答

よくわからない

理解したいこと

なぜ,「現実問題」は解きやすいのか？

回答

よくわからない

しかし，部分的な回答はある

部分的な回答

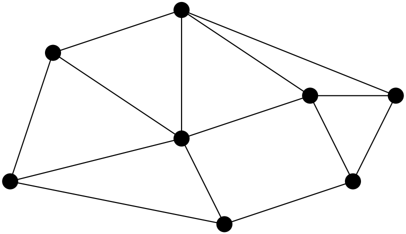
「現実問題」は「特殊な構造」を持っているから

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

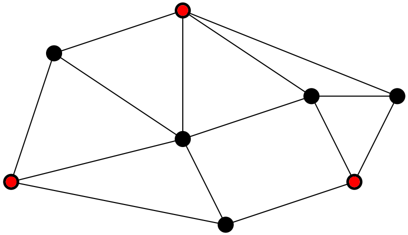
この講義では，その一端に触れたい

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



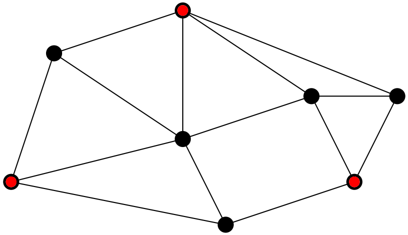
これは NP 困難問題 (解きにくい問題)

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)

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

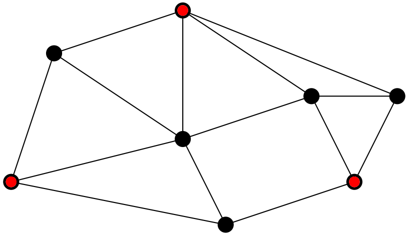


これは NP 困難問題 (解きにくい問題)

事実

グラフが木 (tree) ならば，簡単に解ける (次回)

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)

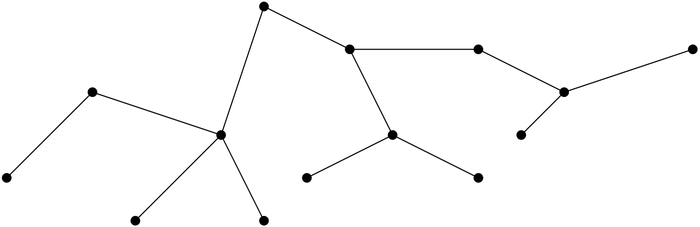
事実

グラフが木 (tree) ならば，簡単に解ける (次回)

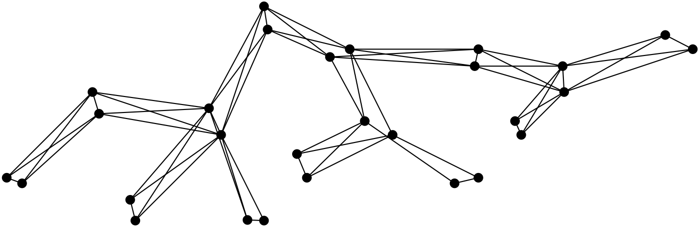
直感？

グラフが木に 近ければ，簡単に解けそう？

これは木

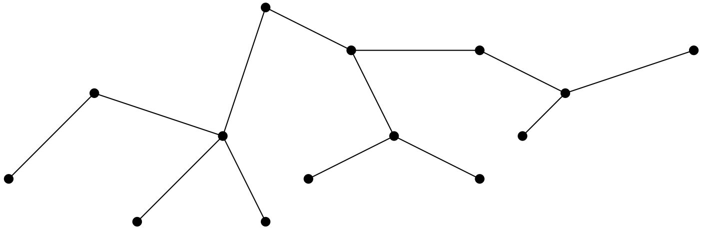


これは木に近い？

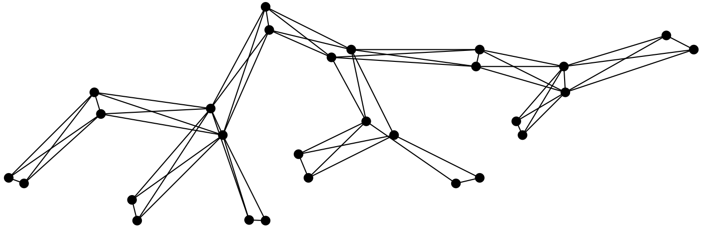


木に近い，とは？

これは木



これは木に近い？ $\rightsquigarrow$ 「木っぽさ」を表す尺度を考える必要あり



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

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

教員

- 岡本 吉央 (おかもと よしお)
- 居室：西 4 号館 2 階 206 号室
- E-mail：okamotoy@uec.ac.jp
- Web：http://dopal.cs.uec.ac.jp/okamotoy/

講義資料

- Web：http://dopal.cs.uec.ac.jp/okamotoy/lect/2016/treewidth/
- 注意：資料の印刷等は各学生が自ら行う
- 講義当日の昼 12 時までに，ここに置かれる
- Twitter (@okamoto7yoshio)：置かれたことを知らせる tweet

http://dopal.cs.uec.ac.jp/okamotoy/lect/2016/treewidth/

- スライド
- 印刷用スライド：8 枚のスライドを 1 ページに収めたもの
- 演習問題

「印刷用スライド」と「演習問題」は各自印刷して持参すると便利

講義 (80 分)

- スライドと板書で進める
- スライドのコピーに重要事項のメモを取る

演習 (10 分)

- 演習問題に取り組む
- 不明な点は教員に質問する

退室 (0 分) ←重要

- コメント (授業の感想，質問など) を紙に書いて提出する (匿名可)
- コメントとそれに対する回答は (匿名で) 講義ページに掲載される

オフィスアワー：金曜 5 限 (岡本居室か CED)

- 質問など
- ただし，いないときもあるので注意 (注意：情報数理工学セミナー)

演習問題の進め方

- 授業のおわり 10 分は演習問題を解く時間
- 残った演習問題は復習・試験対策用
- 注意：「模範解答」のようなものは存在しない

演習問題の種類

- 復習問題：講義で取り上げた内容を反復
- 補足問題：講義で省略した内容を補足
- 追加問題：講義の内容に追加
- 発展問題：少し難しい (かもしれない)

レポートの提出

- 演習問題の答案をレポートとして提出してもよい
- レポートには提出締切がある (各回にて指定)
- レポートは採点されない (成績に勘案されない)
- レポートにはコメントがつけられて，返却される
    - 返却された内容については，再提出ができる (再提出締切は原則なし)
    - 再提出には最初に提出したレポートも添付する

期末試験のみによる

- 出題形式
  - 演習問題と同じ形式の問題を 6 題出題する
  - その中の 3 題以上は演習問題として提示されたものと同一である (ただし,「発展」として提示された演習問題は出題されない)
  - 全問に解答する
- 配点：1 題 20 点満点，計 120 点満点
- 成績において，100 点以上は 100 点で打ち切り
- 時間：90 分 (おそらく)
- 持ち込み：A4 用紙 1 枚分 (裏表自筆書き込み) のみ可

教科書・参考書

教科書

- 指定しない

全般的な参考書

- M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Parameterized Algorithms, 2015, Springer.
- J. Kleinberg, É. Tardos, Algorithm Design, 2006, Pearson. (和訳あり)
- R. Niedermeier, Invitation to Fixed-Parameter Algorithms, 2006, Oxford University Press.
- J. Flum, M. Grohe, Parameterized Complexity Theory, 2006, Springer.

その他，研究論文

- 私語はしない (ただし，演習時間の相談は積極的に OK)
- 携帯電話はマナーモードにする
- この講義と関係のないことを (主に電子機器で) しない
- 音を立てて睡眠しない

約束が守られない場合は退席してもらう場合あり

目次

無向グラフ (undirected graph) とは？

無向グラフとは，順序対 (V, E) で，

- V は集合
- $E \subseteq 2^V$ は V の 要素数 2 の部分集合の集合

であるもののこと

例：

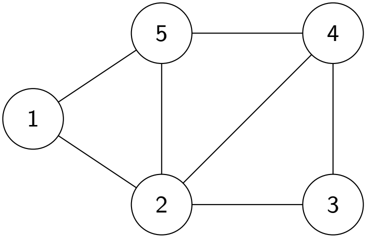
- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$

注意

$\{2, 5\} = \{5, 2\}$ (集合では順序を不問)

無向グラフの図示

- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$

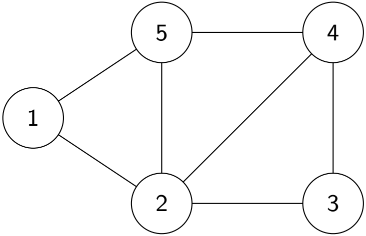


無向グラフ $G = (V, E)$

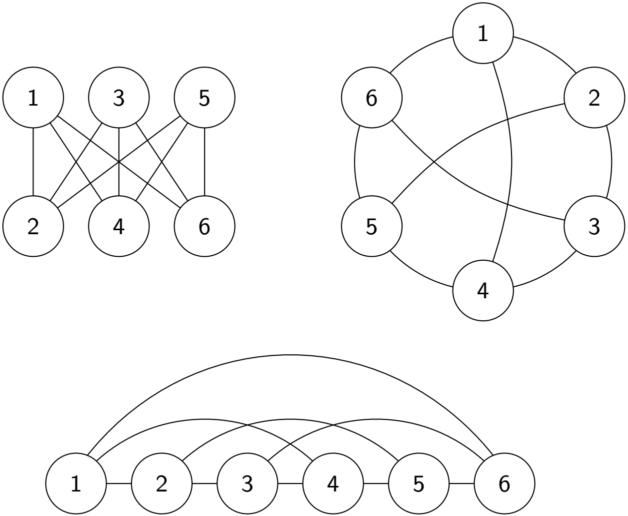
無向グラフの用語

- V の要素を G の頂点と呼ぶ
- E の要素を G の辺と呼ぶ
- V を G の頂点集合と呼ぶ
- E を G の辺集合と呼ぶ
- 辺 $\{u, v\} \in E$ に対して，u, v をその端点と呼ぶ
- 頂点 v が辺 e の端点であるとき，v は e に接続するという
- 頂点 u と v が辺を成すとき，u と v は隣接するという

- $V = \{1, 2, 3, 4, 5\}$
- $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{4, 5\}\}$
- 頂点 2, 3 は辺 $\{2, 3\}$ の端点
- 頂点 2 は辺 $\{2, 3\}$ に接続する
- 頂点 2 と頂点 3 は隣接する



1 つのグラフに対するいろいろな図示



注意

無向グラフにおいて

- 「頂点」の別名：「節点」,「ノード」,「点」 (vertex, node)
- 「辺」の別名：「無向辺」,「枝」,「エッジ」 (edge)

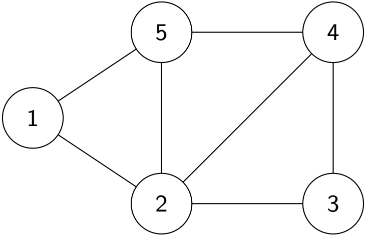
この講義で「グラフ」と言ったら「無向グラフ」を指す

無向グラフ $G = (V, E)$，頂点 $v \in V$

頂点 v の次数 (degree) とは？

頂点 $v \in V$ の次数とは，v に接続する辺の数のこと

$$\deg_G(v) = |\{e \in E \mid \exists\, u \in V\ (e = \{u, v\})\}|$$



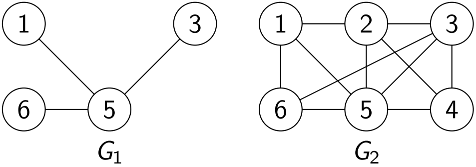
- $\deg_G(1) = 2$
- $\deg_G(2) = 4$
- $\deg_G(3) = 2$
- $\deg_G(4) = 3$
- $\deg_G(5) = 3$

無向グラフ $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$

部分グラフ (subgraph) とは？

G_1 が G_2 の部分グラフであるとは，次を満たすこと

- $V_1 \subseteq V_2$
- $E_1 \subseteq E_2$



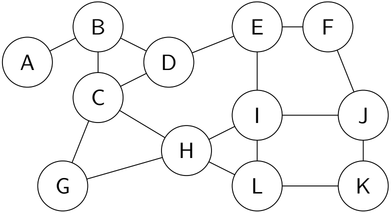
無向グラフ $G = (V, E)$，頂点部分集合 $U \subseteq V$

誘導部分グラフとは？

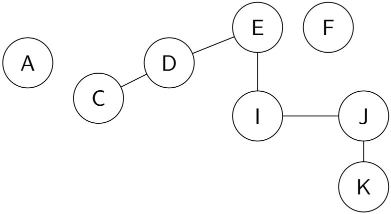
G において U が**誘導する部分グラフ**とは，次で定義されるグラフ

- 頂点集合は U
- 辺集合は $\{\{u, v\} \in E \mid u, v \in U\}$

$G[U]$ と表記する



G



G の誘導部分グラフではない

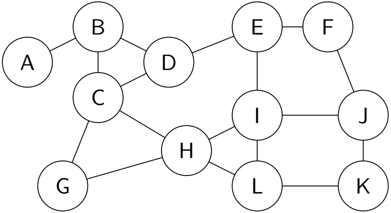
無向グラフ $G = (V, E)$，頂点部分集合 $U \subseteq V$

誘導部分グラフとは？

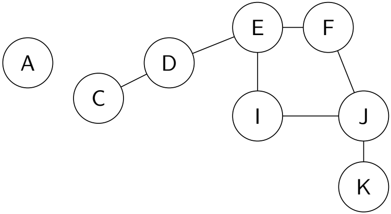
G において U が誘導する部分グラフとは，次で定義されるグラフ

- 頂点集合は U
- 辺集合は $\{\{u, v\} \in E \mid u, v \in U\}$

$G[U]$ と表記する



G



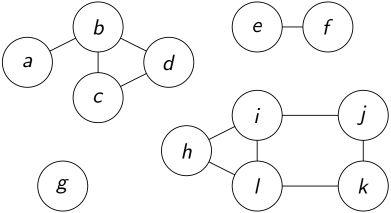
G の誘導部分グラフである

無向グラフ $G = (V, E)$

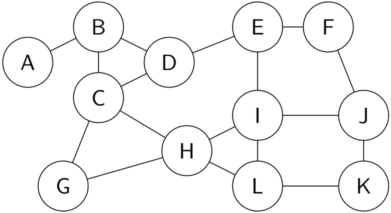
グラフが連結であるとは？

G が連結であるとは，
任意の 2 頂点 $u, v \in V$ に対して，u から v へ至る道が存在すること

連結ではないグラフは非連結と呼ばれる



非連結グラフ



連結グラフ

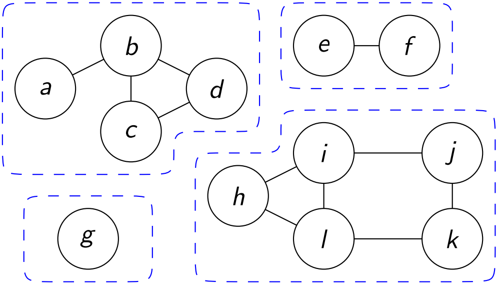
注：「グラフが連結している」とは言わない

無向グラフ $G = (V, E)$

グラフの連結成分とは？

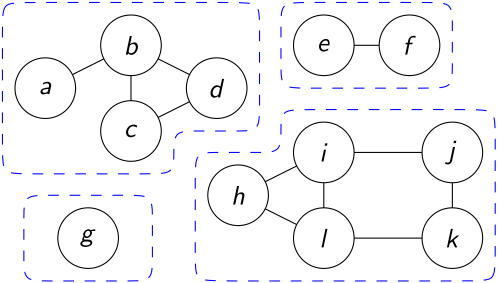
G の連結成分とは，G の極大連結部分グラフのこと

(「極大」とは，グラフの包含関係を半順序であるとみなしたときの「極大」)



連結成分の数 $= 4$

次数 0 の頂点を孤立点と呼ぶ



g は孤立点

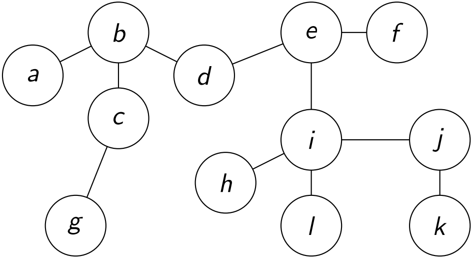
目次

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

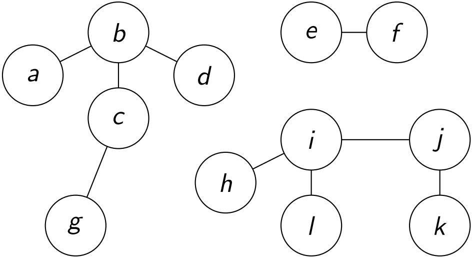


無向グラフ $G = (V, E)$

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

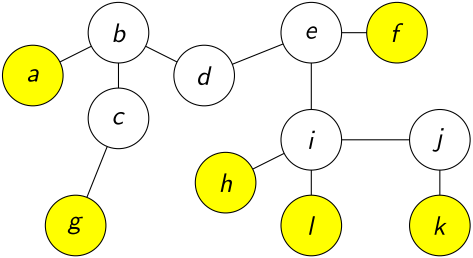


木は森であり，森の各連結成分は木である

木 $G = (V, E)$, $|V| \geq 2$

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

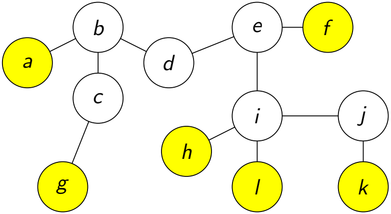


木における次数 1 の頂点を葉 (leaf) と呼ぶ

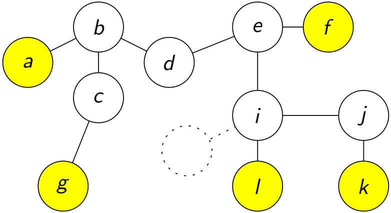
木 $G = (V, E)$，$|V| \geq 2$，その葉 $v \in V$

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

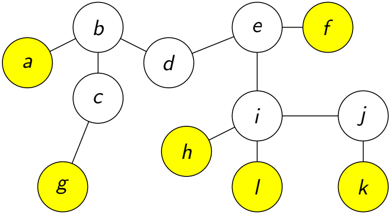


$G - d$

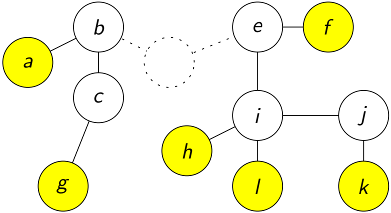
木 $G = (V, E)$，$|V| \geq 2$，葉ではない頂点 $v \in V$

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



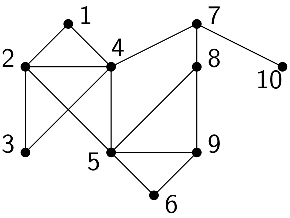
G



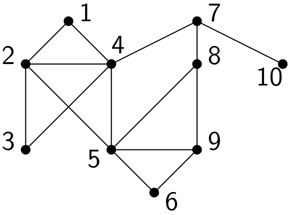
$G - d$

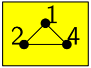
目次

グラフの「木っぽさ」を表現

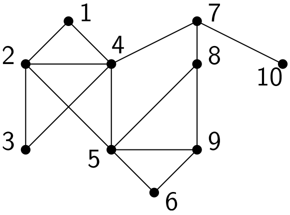


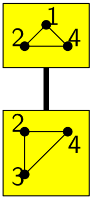
グラフの「木っぽさ」を表現



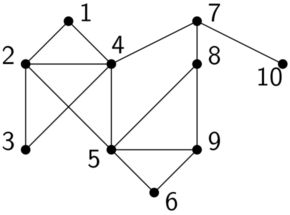


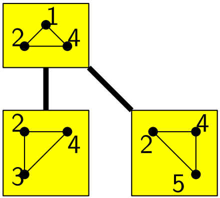
グラフの「木っぽさ」を表現



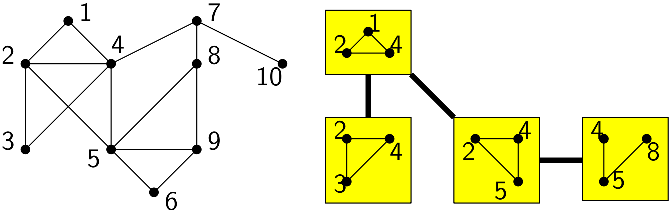


グラフの「木っぽさ」を表現

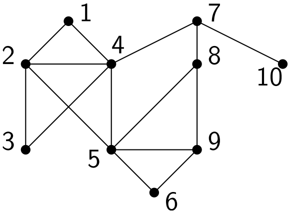


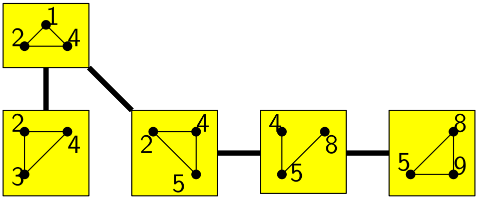


グラフの「木っぽさ」を表現

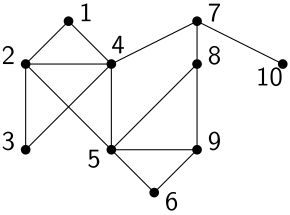


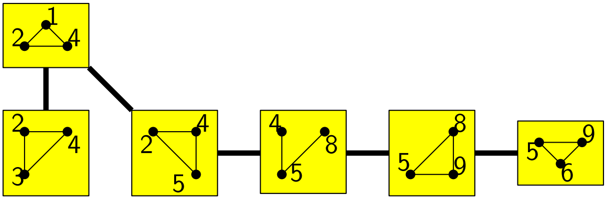
グラフの「木っぽさ」を表現



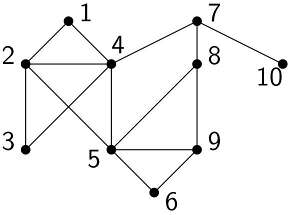


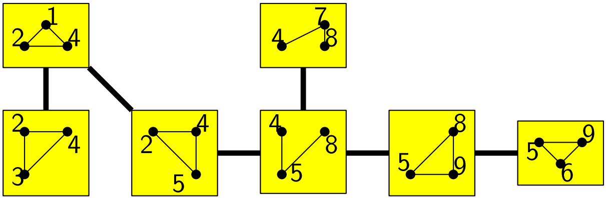
グラフの「木っぽさ」を表現



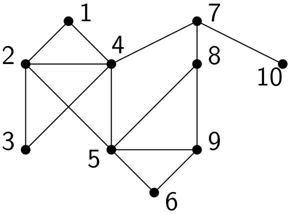


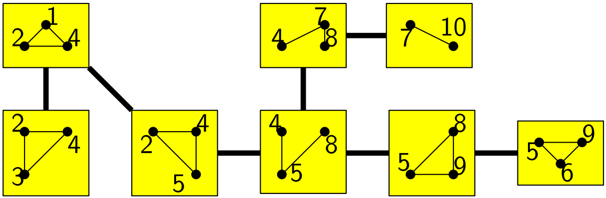
グラフの「木っぽさ」を表現





グラフの「木っぽさ」を表現



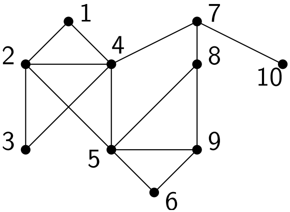


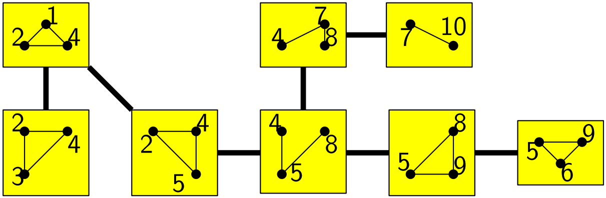
グラフの木分解：定義

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，

- $\mathcal{T}$ の節点はどれも V の部分集合
- 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
- 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



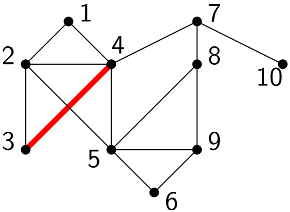


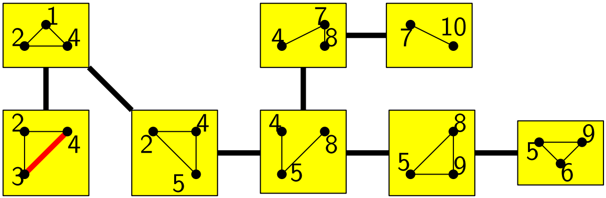
木分解の節点を**バッグ** (bag) と呼ぶことがある

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，

- $\mathcal{T}$ の節点はどれも V の部分集合
- 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
- 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



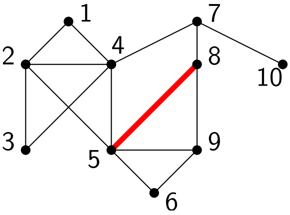


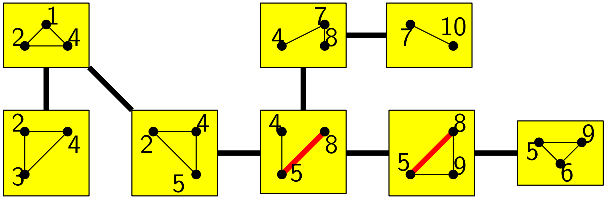
木分解の節点をバッグ (bag) と呼ぶことがある

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,

- $\mathcal{T}$ の節点はどれも V の部分集合
- 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
- 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



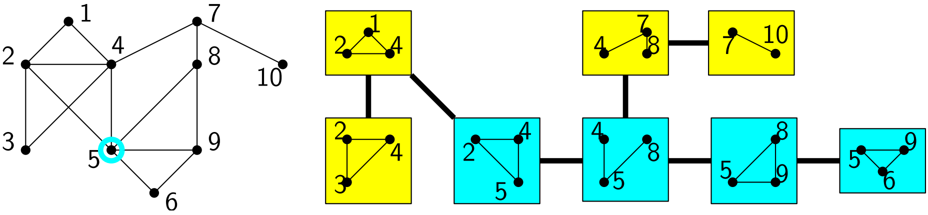


木分解の節点をバッグ (bag) と呼ぶことがある

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，

- $\mathcal{T}$ の節点はどれも V の部分集合
- 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
- 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

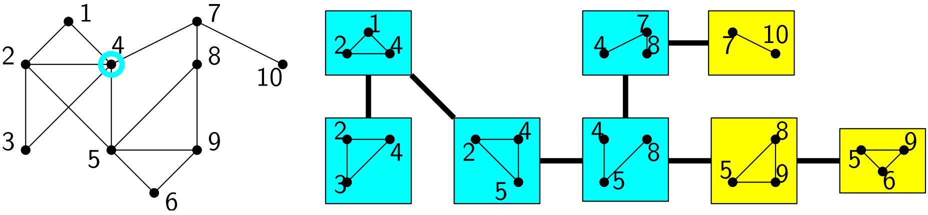


木分解の節点をバッグ (bag) と呼ぶことがある

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,

- $\mathcal{T}$ の節点はどれも V の部分集合
- 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
- 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

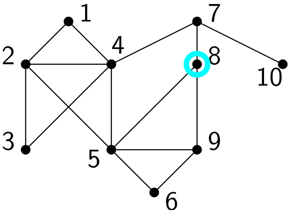


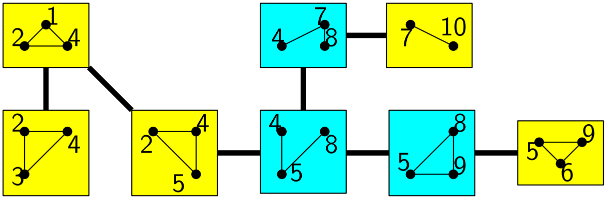
木分解の節点をバッグ (bag) と呼ぶことがある

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，

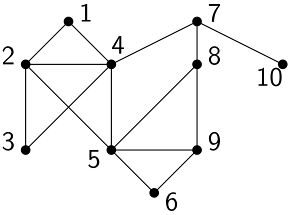
- $\mathcal{T}$ の節点はどれも V の部分集合
- 各辺 $\{u, v\} \in E$ に対して，$u, v \in X$ となる $\mathcal{T}$ の節点 X が存在する
- 各頂点 $v \in V$ に対して，$\mathcal{T}$ の節点で v を含むものは $\mathcal{T}$ の (連結で非空な) 部分木を誘導する

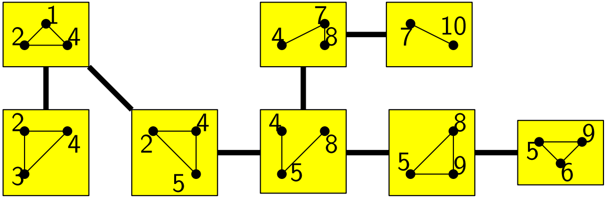




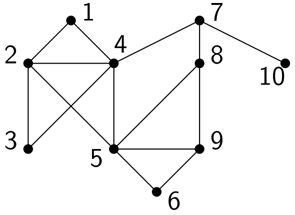
木分解の節点をバッグ (bag) と呼ぶことがある

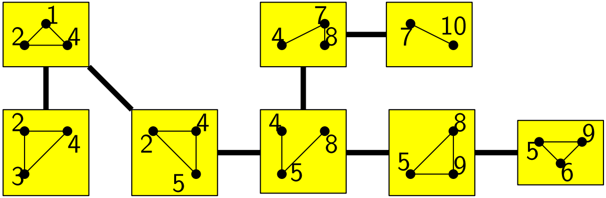
グラフの木分解：例

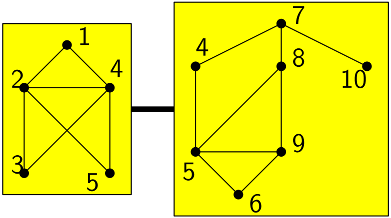




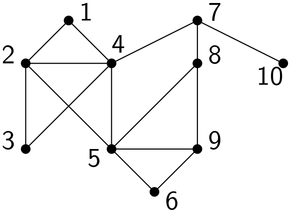
グラフの木分解：例

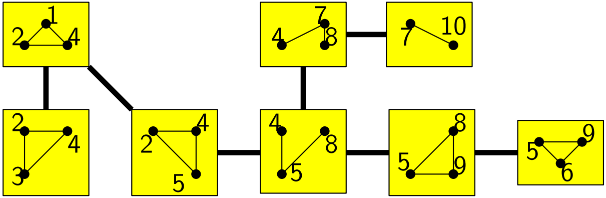


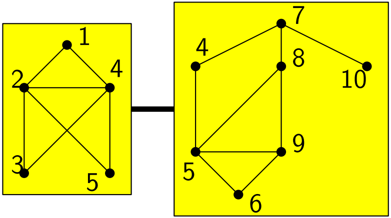


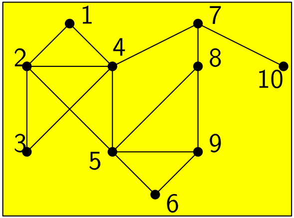


グラフの木分解：例


1
2
3
4
5
6
7
8
9
10


1 2 4
2 3 4
2 4 5
4 7 8
7 10
4 5 8
5 8 9
5 6 9


1 2 3 4 5
4 5 6 7 8 9 10


1
2
3
4
5
6
7
8
9
10

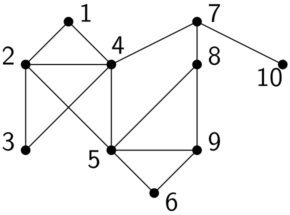
グラフの木幅とは？

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

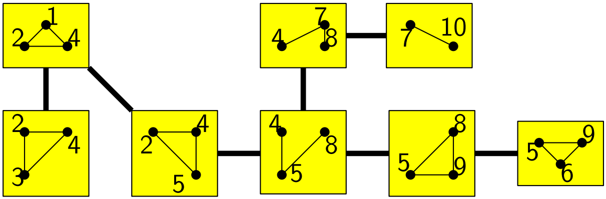
$$\mathtt{tw}(\mathcal{T}) = \max\{|S| - 1 \mid S \text{ は } \mathcal{T} \text{ の節点}\}$$

- 無向グラフ G の木幅 (treewidth)

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

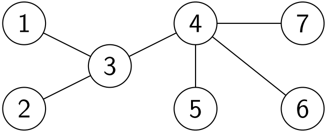


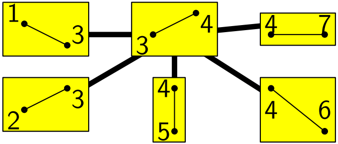
$\mathtt{tw}(G) = 2$



命題：木の木幅

木の木幅は 1 以下 (木は幅が 1 以下の木分解を持つ)

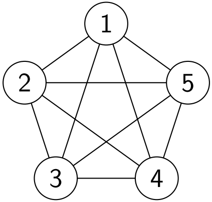


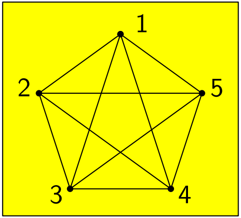


証明は，後の講義で

命題：完全グラフの木幅

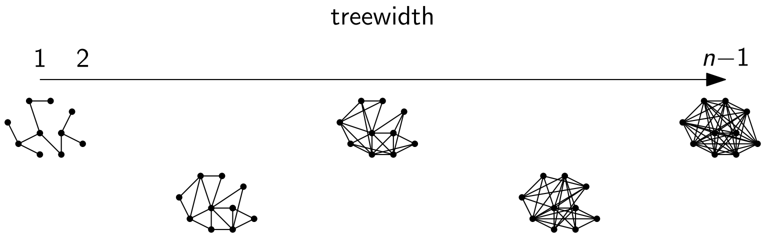
頂点数 n の完全グラフの木幅は $n-1$





証明は，後の講義で

直感：木幅は「木っぽさ」を表す尺度



木は木に近い

完全グラフは木から遠い

なぜ木幅を考えるとうれしいのか？

ご利益 1

木幅が小さいと，効率よく解ける問題がたくさんある

ご利益 2

現実世界から生じるグラフにも，木幅が小さいものがたくさんある

なぜ木幅を考えるとうれしいのか？

ご利益 1

木幅が小さいと，効率よく解ける問題がたくさんある

$\rightsquigarrow$ 効率よいアルゴリズムを設計しないといけない (次回以降)

ご利益 2

現実世界から生じるグラフにも，木幅が小さいものがたくさんある

ご利益 1

木幅が小さいと，効率よく解ける問題がたくさんある

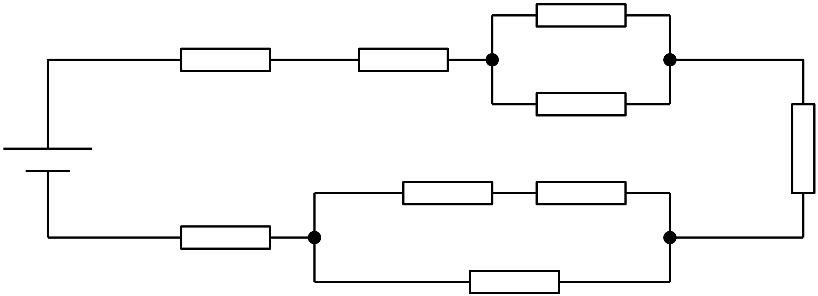
$\rightsquigarrow$ 効率よいアルゴリズムを設計しないといけない (次回以降)

ご利益 2

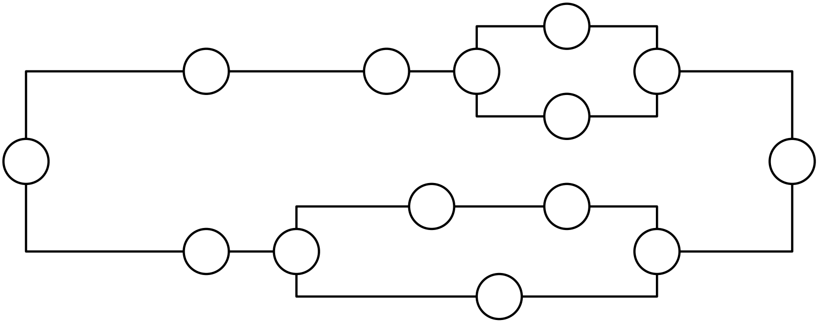
現実世界から生じるグラフにも，木幅が小さいものがたくさんある

$\rightsquigarrow$ 本当にそうか，確認する必要がある

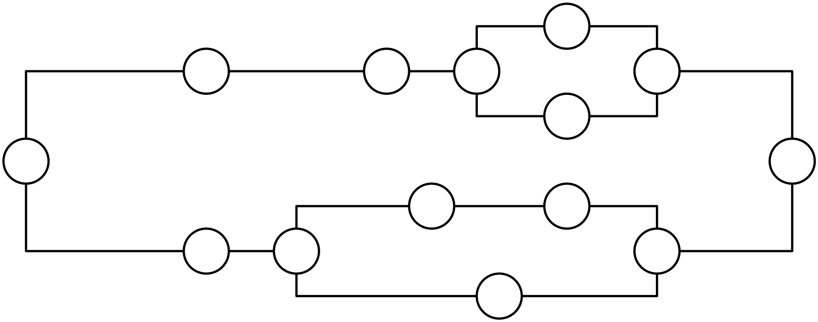
直並列回路



直並列回路：グラフ化



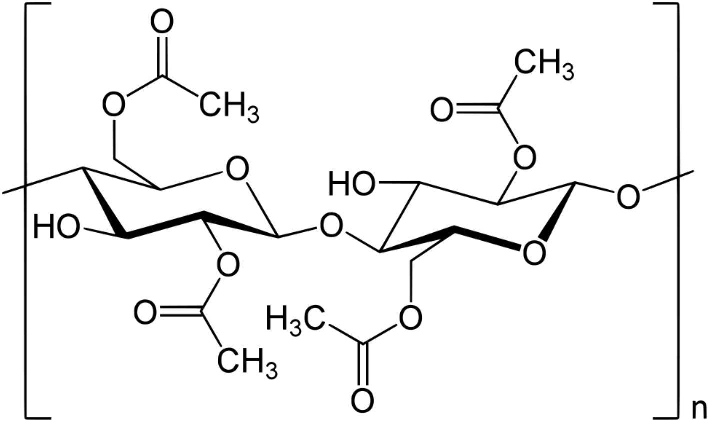
直並列回路：グラフ化



事実 (第 5 回の講義で詳述予定)

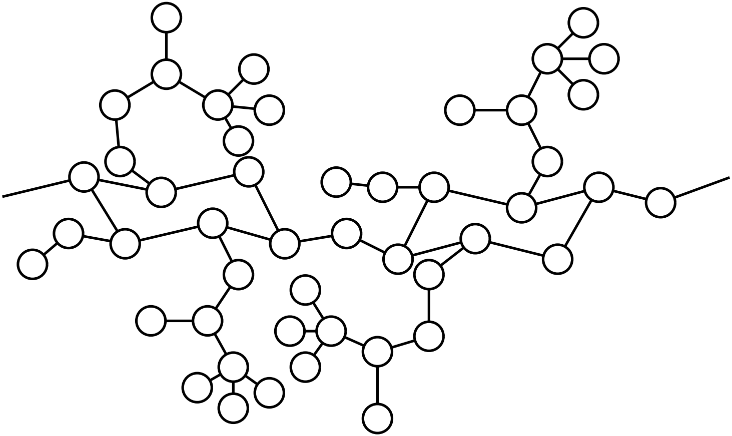
直並列回路から得られるグラフの木幅は必ず 2 以下

例：アセチルセルロース



https://commons.wikimedia.org/wiki/File:Cellulose_acetate.svg

例：アセチルセルロース



Yamaguchi et al. (2003) による調査

LIGAND データベースにおける 9712 個の化合物に対して

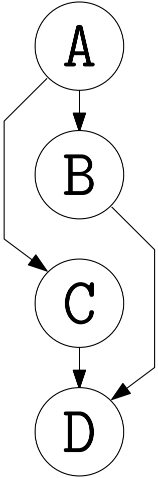
木幅	化合物数	
1	1881	(19.4%)
2	7336	(75.7%)
3	477	(4.9%)
4	1	(0.01%)
5 以上	0	

つまり

生物学で現れるほとんどの化合物から得られるグラフの木幅は 3 以下

A. Yamaguchi, K. F. Aoki, H. Mamitsuka, Graph complexity of chemical compounds in biological pathways, Genome Informatics 14 (2003) 376–377.

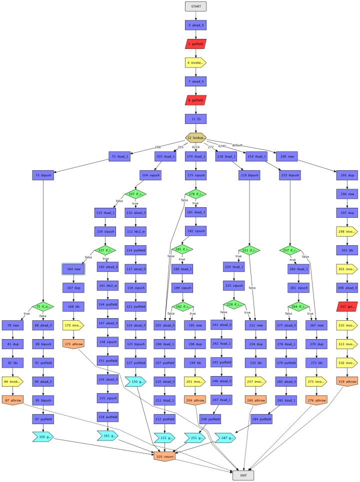
```
0: (A) t0 = read_num
1: (A) if t0 mod 2 == 0
2: (B)   print t0 + " is even."
3: (B)   goto 5
4: (C) print t0 + " is odd."
5: (D) end program
```



プログラム解析，コンパイラ構成に用いられる

例の出典：https://en.wikipedia.org/wiki/Control_flow_graph

プログラムの制御フローグラフ：少し大きな例



http://www.drgarbage.com/img/control-flow-graph-factory/gallery/p11keygenerator-engineinit.png

理論的 (数理的) 結果

- goto のない Algol：木幅 ≤ 2 (Thorup '98)
- goto のない Pascal：木幅 ≤ 2 (Thorup '98)
- goto のない C：木幅 ≤ 7 (Krause, Larisch)
- ラベル付き break/continue のない Java：木幅 ≤ 7 (Gustedt, Oæhle, Telle '02)

- M. Thorup, Structured programs have small tree-width and good register allocation. Information and Computation 142 (1998) 159–181.
- P.K. Krause, L. Larisch, The tree-width of C, submitted.
- J. Gustedt, O. Mæhle, J.A. Telle, The treewidth of Java programs. In Proc. ALENEX 2002, Lecture Notes in Computer Science 2409 (2002) 57–59.

実証的結果：Java の API パッケージ　(Gustedt, Oæhle, Telle '02)

パッケージ名	メソッドの数	平均木幅	割合 木幅 = 2	3	4	5
java.lang	604	2.73	27	73	1	0
java.lang.reflect	50	2.86	14	86	0	0
java.math	96	2.94	7	90	3	0
java.net	279	2.72	31	66	3	1
java.io	620	2.56	47	49	4	0
java.util	990	2.68	32	68	1	0
java.util.jar	93	2.73	28	71	1	0
java.util.zip	157	2.55	45	55	0	0
java.awt	1411	2.66	34	65	1	0
java.awt.event	71	2.74	25	75	0	0
java.awt.geom	527	2.71	30	69	1	0
java.awt.image	623	2.69	30	70	1	0
javax.swing	3400	2.62	39	60	1	0
javax.swing.event	87	2.63	37	63	0	0
javax.swing.tree	379	2.65	35	64	1	0

目次

重要概念

- 木分解：グラフを「木っぽく」表す構造
- 木幅：グラフの「木っぽさ」を表す尺度

なぜ重要か？

- 木幅が小さいと，効率よく解ける問題が多い (←次回以降)
- 木幅の小さいグラフが現実世界には多い

次回予告

まず,「木に対して効率よく解く」方法を考える

残った時間の使い方

- 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. グラフに関する用語
2. 木とその性質
3. 木分解と木幅
4. 木幅が現れる場面
   - 電気回路 (回路網グラフ)
   - 分子生物学 (分子グラフ)
   - コンパイラ構成 (制御フローグラフ)
5. 今日のまとめ と 次回の予告