大規模知識処理特論

第９回

脊戸　和寿

今回の講義内容

NP 完全問題に対するアルゴリズムについて学ぶ

- 最小頂点被覆問題に対する 2 近似アルゴリズム
- NP 完全問題に対する厳密アルゴリズムと FPT
- 頂点被覆問題に対する FPT アルゴリズム

最小頂点被覆問題に対する近似アルゴリズム

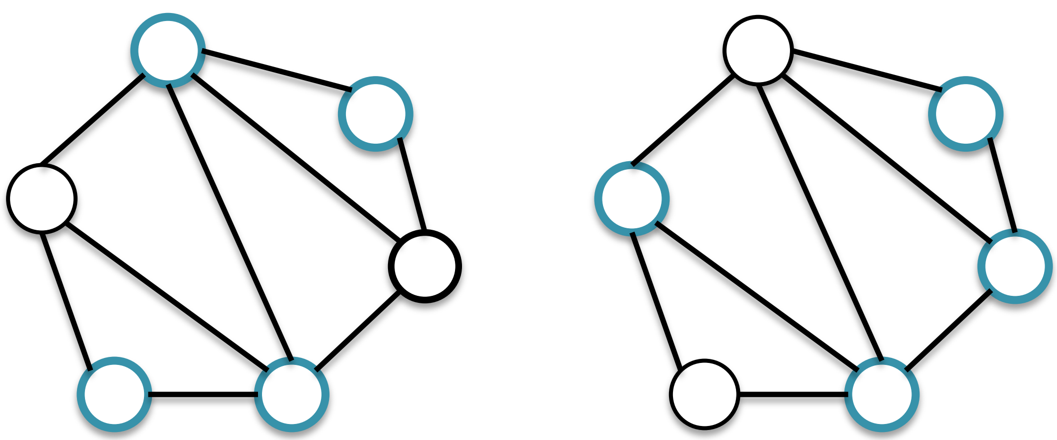
頂点被覆問題（VC）

入力：無向グラフ G と正の整数 k

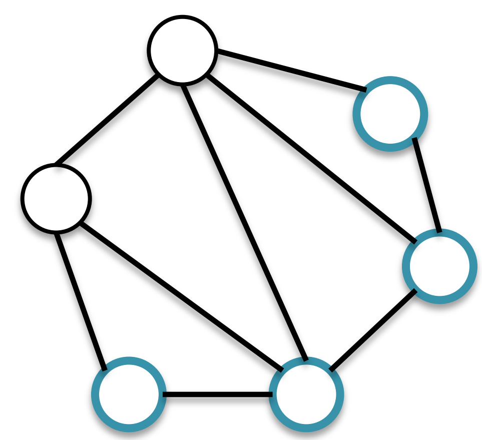
問　：G には大きさ k の頂点被覆が存在するか？

頂点被覆

任意の辺に対して、少なくとも一方の端点が属する頂点集合



k=4 の例

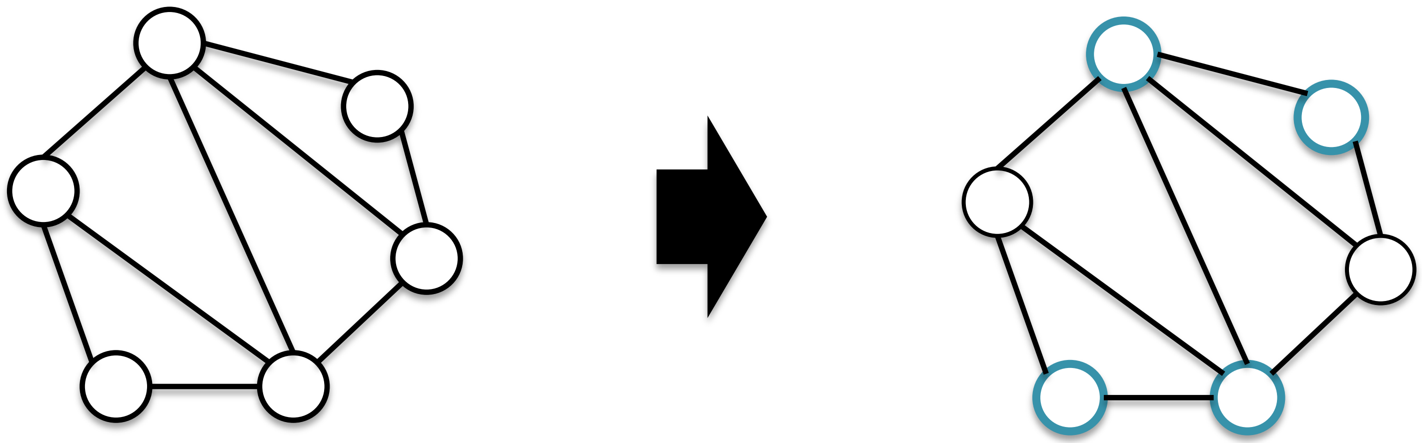


×

最小頂点被覆問題（ MVC ）

入力：無向グラフ G

出力：最小頂点被覆 U



G

MVC の近似度

多項式時間近似アルゴリズム

- 2 近似 [Gavril, Yannakakis]
- $2 - 1/\Theta\left(\sqrt{\log V}\right)$ [Karakostas 04]

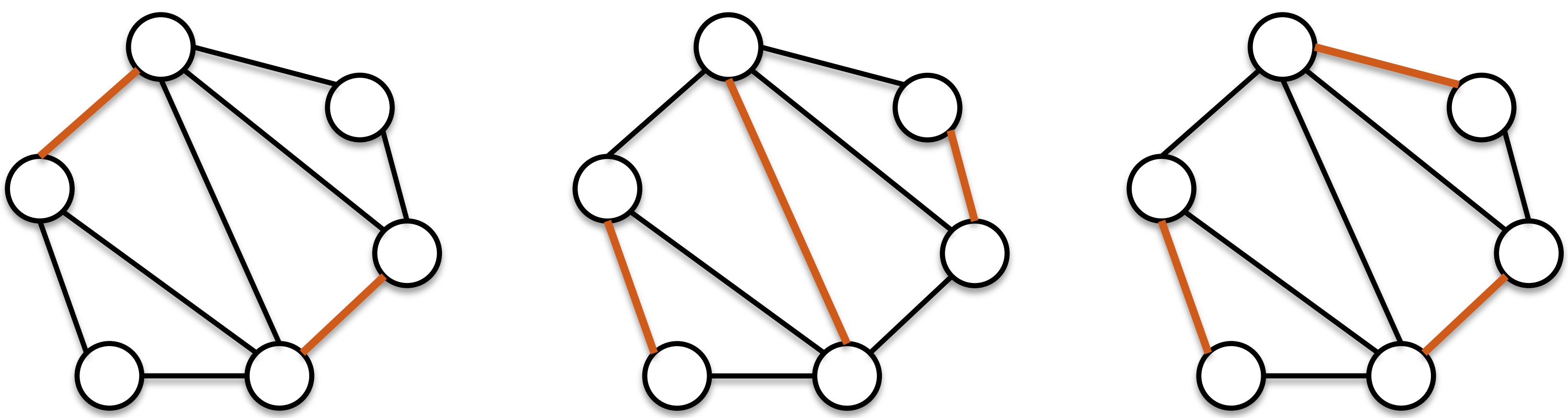
近似困難性

- P=NP でないかぎり，1.3606 近似多項式時間アルゴリズムは存在しない [Dinur and Safra 2005]
- Unique Games Conjecture が成り立つならば，2-ε 近似多項式時間アルゴリズムは存在しない [Knot and Regev 2003]

グラフのマッチング

<u>マッチング</u>

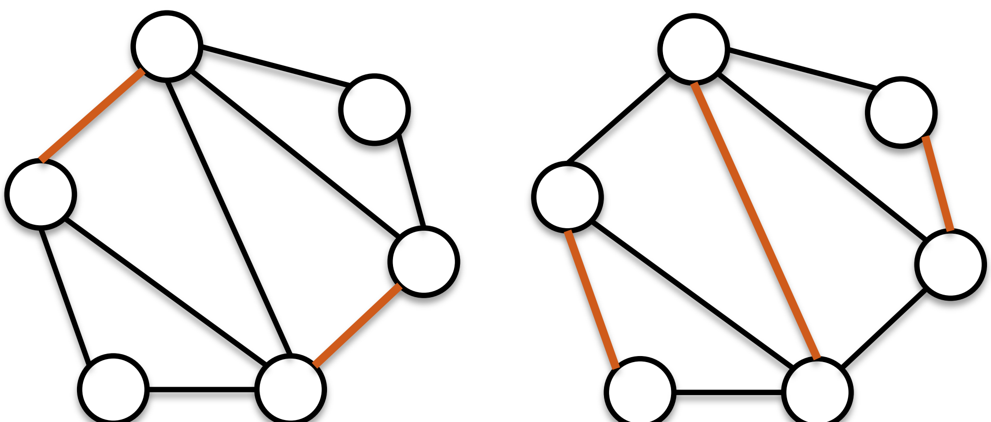
- グラフの辺集合で、集合内の各辺が互いに頂点を共有しないもの。



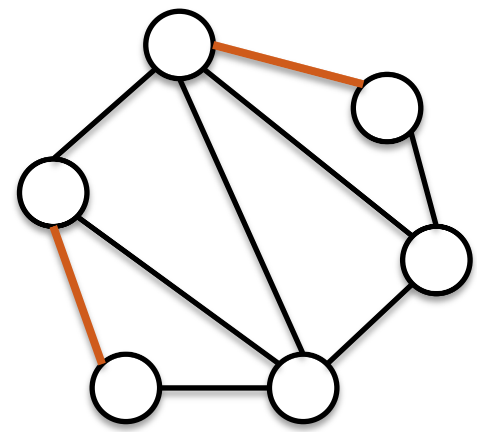
極大マッチング

<u>極大マッチング</u>

- マッチングの中で、これ以上は新しい辺をマッチングに加えることができないもの。



極大マッチング

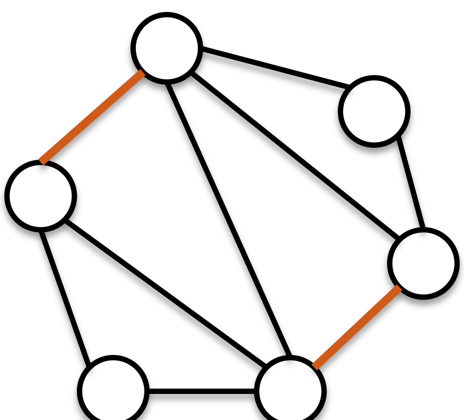


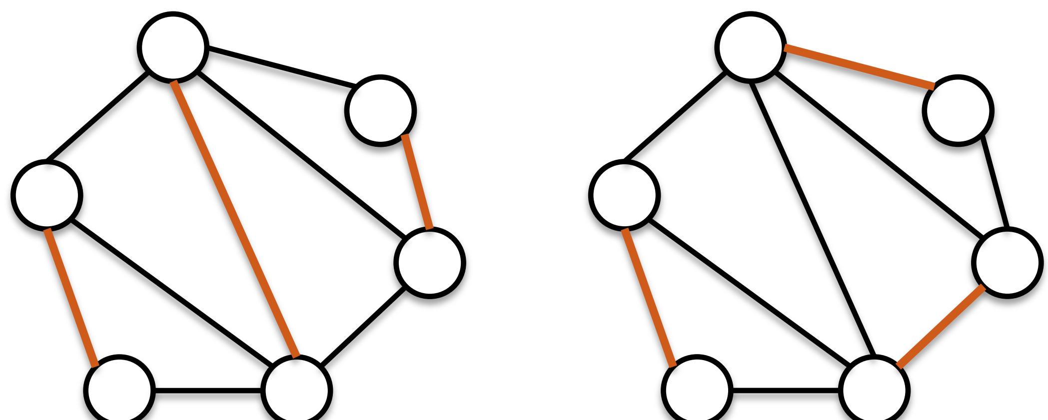
極大マッチングでない

最大マッチング

<u>最大マッチング</u>

➢ 極大マッチングの中で、マッチングに含まれる辺の数が最大のもの。





最大マッチング

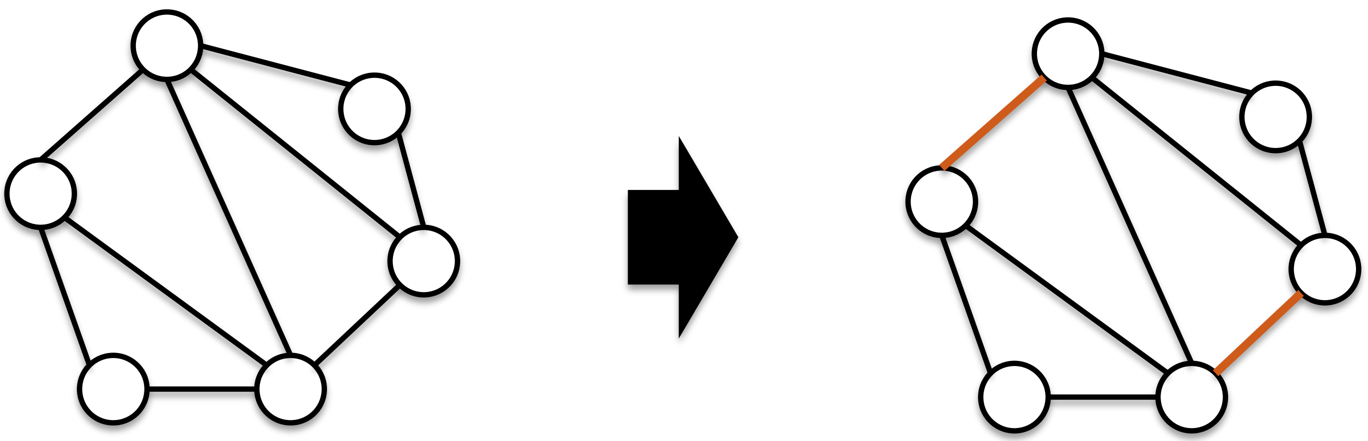
MVC の 2 近似アルゴリズム

極大マッチングを使った 2 近似アルゴリズム

1. グラフ G の極大マッチング M を求める
2. M の各辺の両端の頂点を頂点被覆 U’ として選択する
3. U’ を出力する

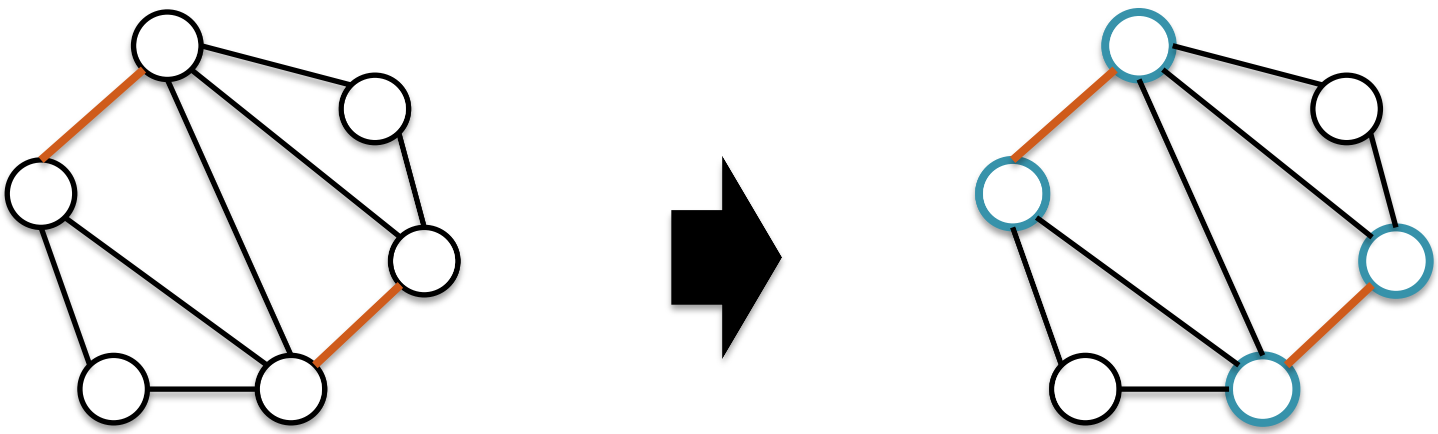
MVC の 2 近似アルゴリズム

1. グラフ G の極大マッチングMを求める
   - 辺を 1 つ 1 つ見ていき、両端点がまだ M に含まれていなければ、M に入れる。そうでなければ、M に入れない。



MVC の 2 近似アルゴリズム

2. M の各辺の両端の頂点を頂点被覆 U’ として選択する
3. U’ を出力する



アルゴリズムの計算時間

アルゴリズム：グラフ G の頂点数 n，辺の数 m

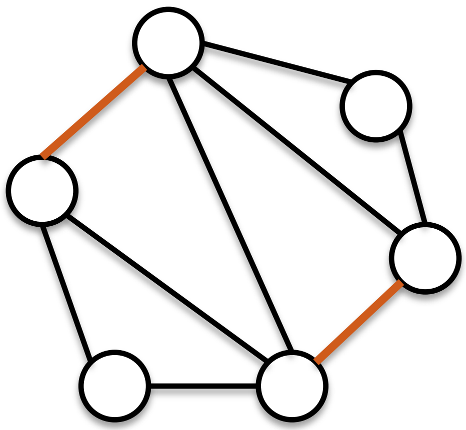
1. グラフ G の極大マッチング M を求める → O(m)
2. M の各辺の両端の頂点を頂点被覆 U’ として選択する → O(n)
3. U’ を出力する → O(n)

計算時間は、O(n+m)

2 近似の証明

最小頂点被覆 U、アルゴリズムの出力 U'、極大マッチング M の間には、次の関係が成り立つ

1. $|U| \leqq |U'|$
2. $|M| \leqq |U|$
3. $|U'| = 2|M|$



2、3より、 $|U'| = 2|M| \leqq 2|U|$ となる

近似率 : $\dfrac{\text{アルゴリズムが出力する解のコスト}}{\text{最適解のコスト}}$ の最悪値

NP 完全問題に対する
厳密アルゴリズムと FPT

NP 完全問題に対する厳密アルゴリズム

やっぱり厳密解が欲しい

- 指数時間はかけたくない
- 解が正確に求まる保証はほしい

頂点被覆問題の単純な厳密アルゴリズム：$O(n^k m) = O(n^{k+2})$

- 全てのサイズ k の頂点集合 ($=O(n^k)$) について，頂点被覆となっているかを確認する（集合1つあたり $O(m)$ ）

k が定数なら多項式時間ではあるが…

- $n=10000$、$k=10$ なら $10000^{12} = 10^{48}$ ！

厳密アルゴリズムを効率化したい.

k が 10 のときくらい、もう少し高速にできないのか？

- もっと言うと，k がある程度小さい間は n が大きくなっても、できるだけ問題を解きたい → $O(n^{0.5k})$ に改良する？？

今は、$O(n^{f(k)})$ の形になっている

⇒ n が増えると、f(k) の影響を大きく受けてしまう

⇒ 計算時間において、n と k が関係していることが影響。

Fixed Parameterized Tractable

FPT（Fixed Parameterized Tractable：固定パラメータ容易）

- 入力サイズを n、あるパラメータを k とすると、
  $O(f(k) \times n^{O(1)})$ 時間アルゴリズムで解ける問題の集合

FPT アルゴリズムは多項式時間アルゴリズム？

- k が n の関数に依存することもあるので、そうではない。
  - ✓ $O(2^k n)$ は FPT だが、k=n のときは $O(n2^n)$

Fixed Parameterized Tractable

FPT アルゴリズムは何が嬉しい？

- 厳密解が得られる
- k と n の関数が独立しているので、k が増えても、n への影響がある程度抑えられる！（n に対しては多項式時間）
  - ✓ $O(2^k n)$ で n=10000、k=10 なら $2^{10} \times 10000 \fallingdotseq 10^7$
    - ※ $O(n^{k+2})$ では約 10^{48}
- k が小さいときには十分高速なアルゴリズム
  - ✓ 入力サイズが大きくても動作する可能性がある

NP 完全問題内での難しさ

NP 完全問題は全て FPT アルゴリズムが存在する？

⇒ 未解決（たぶんない）

NP 完全問題の中でも難しさの階層があるはず.

- FPT アルゴリズムをもつ問題
- FPT アルゴリズムをもたなさそうな問題（ W 階層）
  - W[1] 困難、W[2] 困難、...

頂点被覆問題の FPT

FPT アルゴリズム

- $O(2^k n)$ 時間：単純な Branch and Bound
- $O(k^2 2^k + n^2)$：Kernelization
- $O(1.2738^k + kn)$ [Chen, Kanji and Xia 2006]

困難性

- ETH (Exponential Time Hypothesis)下では，$2^{o(k)} n^{O(1)}$ 時間アルゴリズムは存在しない（cf : $O(2^{\sqrt{k}} n)$ 時間など）
  - ✓ ETH：3SATを解くには、2^{cn} $(c>0)$ 時間必要という仮説
  - ✓ 3SAT の最速決定性アルゴリズムは $O(1.328^n)$ [Liu 2018]

頂点被覆問題に対する FPT アルゴリズム

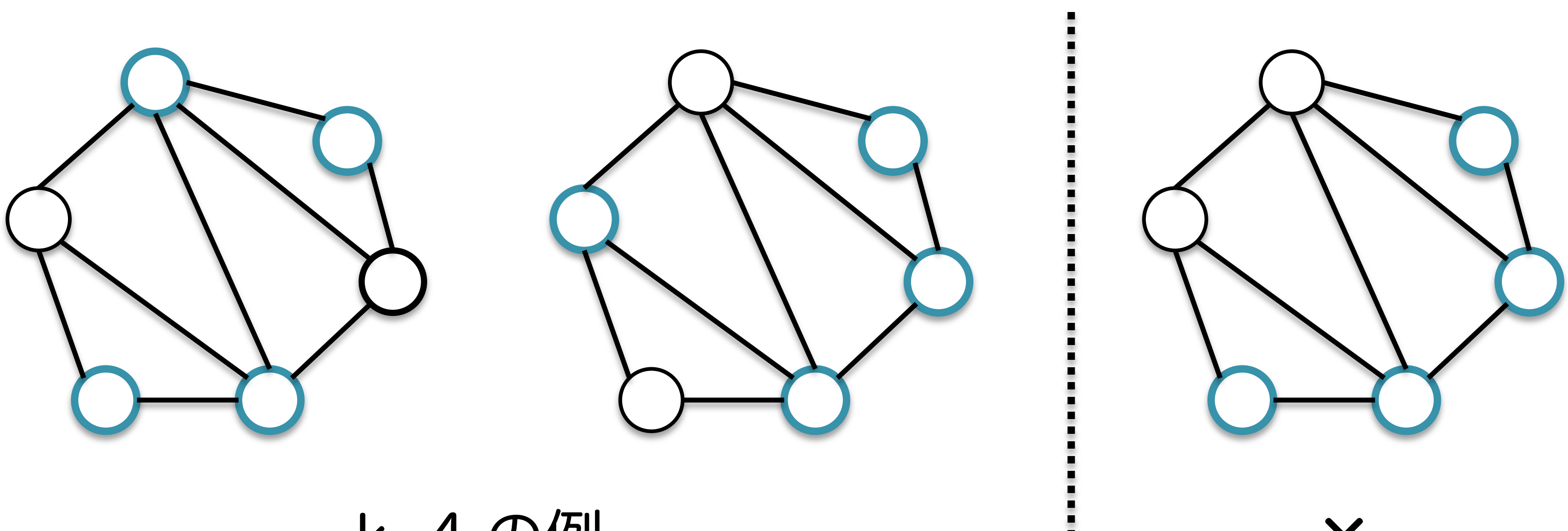
頂点被覆問題（VC）

入力：無向グラフ G と正の整数 k

問　：G には大きさ k の頂点被覆が存在するか？

頂点被覆：

任意の辺に対して、少なくとも一方の端点が属する頂点集合



k=4 の例

×

頂点被覆問題の $O(2^k n)$ アルゴリズム

$O(2^k n)$ 時間アルゴリズム

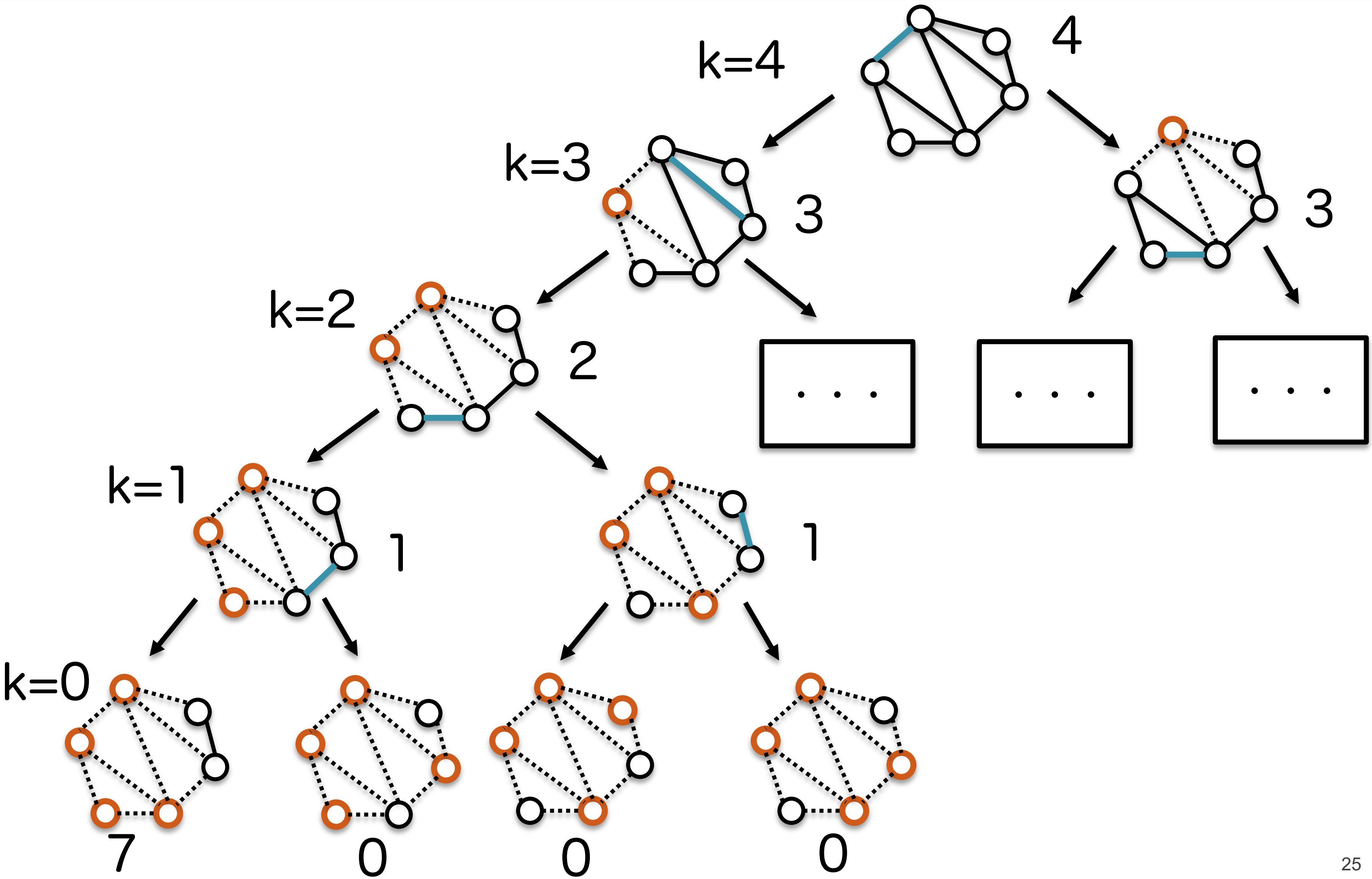
G - v： G から頂点 v につながる辺をすべて取り除いたグラフ

$O(2^k n)$ 時間アルゴリズム BS(G, k)：G の頂点数 n

1. G に辺がない → 0 を返す
2. k=0 → n+1 を返す
3. G の辺 (u, v) を適当に 1 本選ぶ
   ・min(BS(G-u, k-1), BS(G-v, k-1))+1 を返す.

BS(G, k) の値が k 以下であれば Yes、それ以外は No を出力

O($2^k n$) 時間アルゴリズムの挙動



$O(2^k n)$ アルゴリズムの計算時間解析

$T(n, k)$：頂点数 n のグラフ G に対するサイズ k の頂点被覆を求める前述のアルゴリズムの計算時間

3. G の辺 $e = (u, v)$ を適当に 1 本選ぶ

・min(BS(G-u, k-1), BS(G-v, k-1)) + 1 を返す

を考えると、

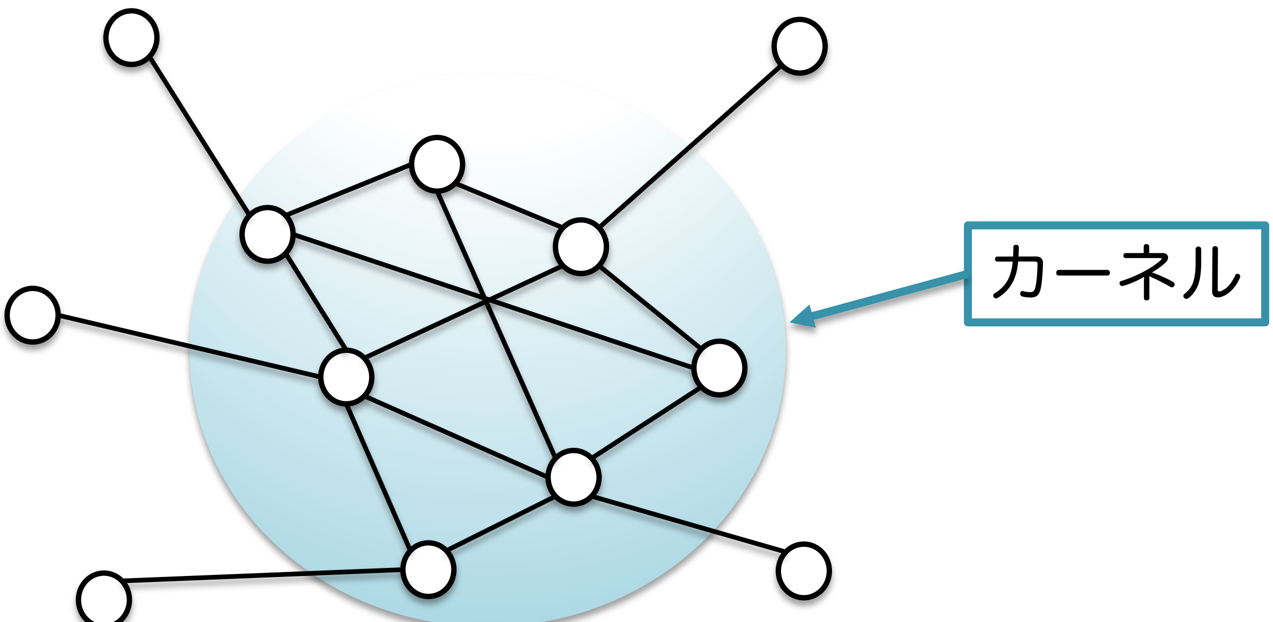
$$T(n, k) = 2T(n, k-1) + O(n)$$ ：辺の削除に $O(n)$

より、$T(n, k) = O(2^k n)$

カーネル（ Kernel ）

ざっくり言うと、「解くために本質的に難しい箇所」

- Kernel を解けば（ほぼ）解になっている



カーネル化（Kernelization）

カーネルに「還元」する.

入力（x, k）：元のインスタンス

出力（x’, k’）：カーネル化後のインスタンス

制約

- $|x'| = f(k)$
- $k' = g(k)$
- $|x| + k$ の多項式時間で還元

FPT アルゴリズムをもつ ⇔ カーネル化可能

[Cai, Chen, Downey, and Fellows 1997]

VC のカーネル化アルゴリズム

$O(k^2 2^k + n^2)$ 時間アルゴリズム：1 がカーネル化

1. 次数が k より大きい頂点 v がある間、次の操作を続ける.

➢ v を頂点被覆 U の集合に加え、その頂点と頂点につながる辺をすべて取り除く

2. 1 を終えた後のグラフを G', U に加えた頂点の数を a とするアルゴリズム BS(G', k-a) を動作させる
3. 2 の出力が k-a 以下であれば Yes、それ以外は No を出力.

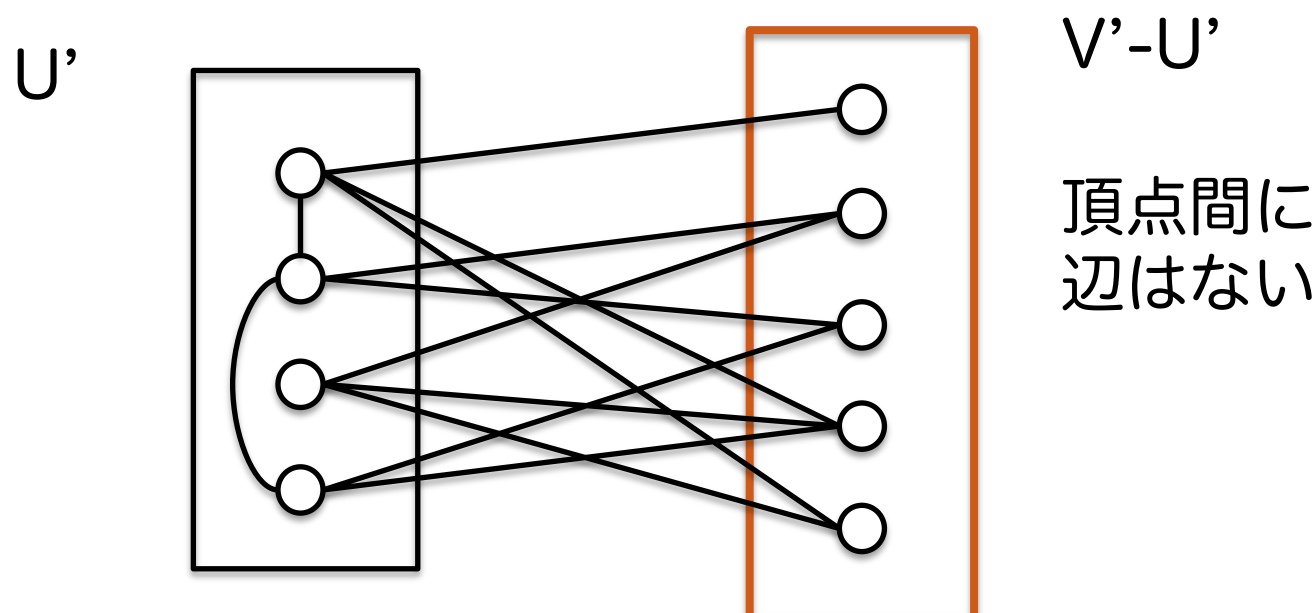
カーネル化の正しさの証明

次数が k より大きい頂点を全て取り除いたグラフを G’ とする.

G’ の頂点集合を V’、頂点被覆を U’ とする

このとき、V’-U’ 内の頂点間には辺はない

もし、あるとすると U’ は頂点被覆になっていない

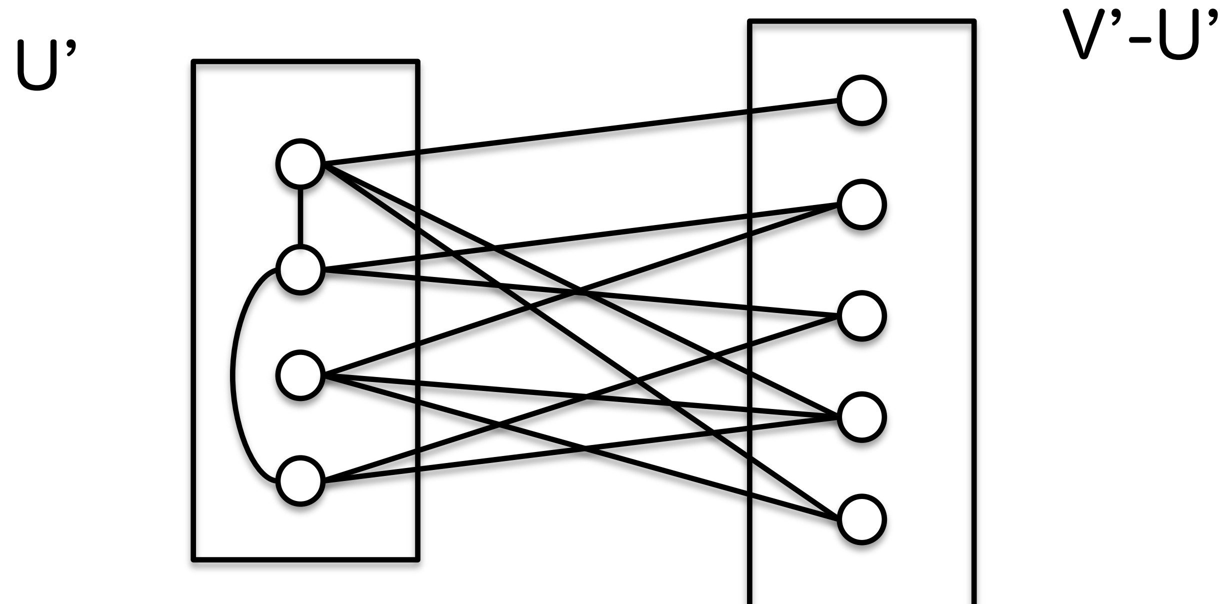


カーネル化の正しさの証明

U’ の各頂点は最大でも V’-U’ 内の k 個の頂点とつながっている

ので、$|V'-U'| \leqq k|U'|$ となる。

よって、$|V'| = |U'|+|V'-U'| \leqq (k+1)|U'| \leqq k(k+1)$



アルゴリズムの計算時間

カーネル化は $O(n^2)$ で終わり、下記を満たす。

- $|V'| \leqq k(k+1)$
- $k' \leqq k$

アルゴリズム BS(n, k) は計算時間が $O(2^k n)$ より、

BS(k(k+1), k)の計算時間は $O(\ 2^k k(k+1)\ ) = O(\ k^2 2^k\ )$となる。

アルゴリズム全体の計算時間は、$O(\ k^2 2^k + n^2\ )$となる。

まとめ

（最小）頂点被覆問題のアルゴリズム

- 2 近似アルゴリズム

NP 完全問題と FPT

- NP 完全問題の FPT アルゴリズム
  - ✓ 解のサイズが小さいときは高速にみつけることが可能.
- カーネルと FPT

他にも色んなアルゴリズム技法があるので、興味のある人は調べてみるとよい。