大規模知識処理特論

第８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

- 厳密アルゴリズムと近似アルゴリズム
- Metric TSP の 2 近似アルゴリズム
- 最小全域木を求める Kruskal アルゴリズム

厳密アルゴリズムと近似アルゴリズム

NP 完全問題

NP 完全問題は次の 2 つを満たす問題.

- クラス NP に属する
- クラス NP に属する任意の問題から多項式時間還元可能

P（決定性多項式時間）では解けないと思われている

⇒ 入力サイズが大きくなると膨大な計算時間を要する可能性

世の中の様々な問題は NP 完全問題に定式化できることが多い

⇒ 諦めずに何らかの形で解く必要がある.

厳密アルゴリズムと近似アルゴリズム

厳密アルゴリズム

- 与えられた問題の**厳密解**を求める.
- 計算時間は**指数時間でもよい.**
- 実世界では，厳密解を要求される状況はある．その解以外では全く意味をなさない状況も多々ある.
- **できる限り高速**に解を求めるアルゴリズムが求められる.

近似アルゴリズム

- 与えられた問題の**近似解**を求める.
- 計算時間は**多項式時間.**
- 実世界では，一定の性能を保証できる解を高速に見つけられることが好ましい状況は多い.
- **できる限り良質な解を求める**アルゴリズムが求められる.

トレードオフの関係

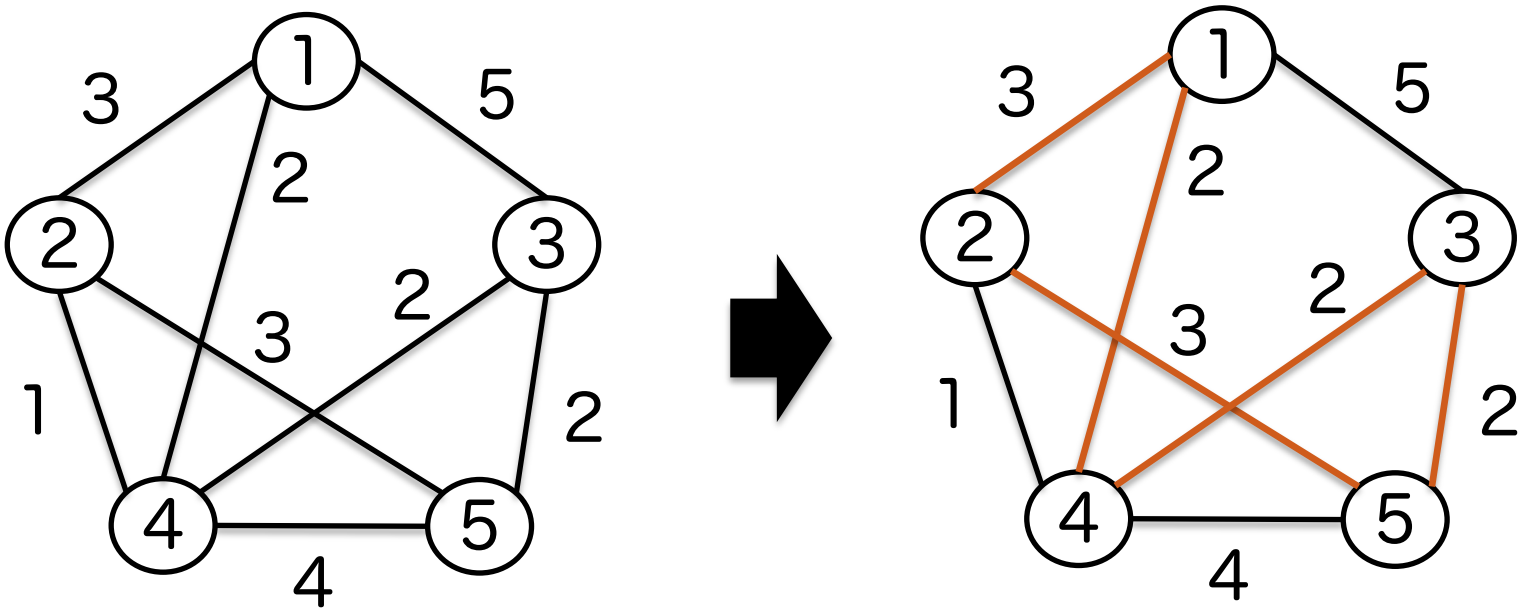
巡回セールスマン問題（ TSP ）

巡回セールスマン問題 (Traveling Salesman Problem)

入力：重み付き無向グラフ G

問　：重みの総和が最小のハミルトン閉路を求めよ.

出力：重みの総和が最小のハミルトン閉路（あれば）



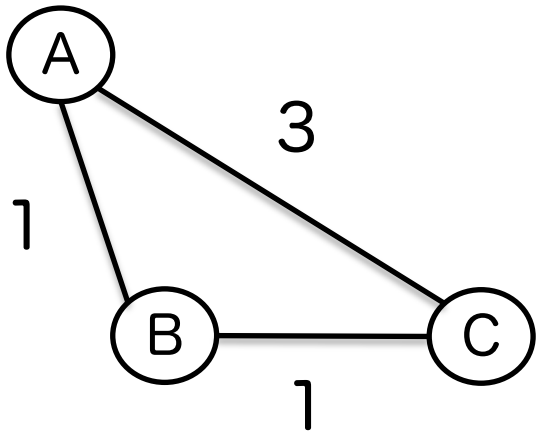
Metric TSP

Metric TSP：TSP に次の制約を加えたもの.

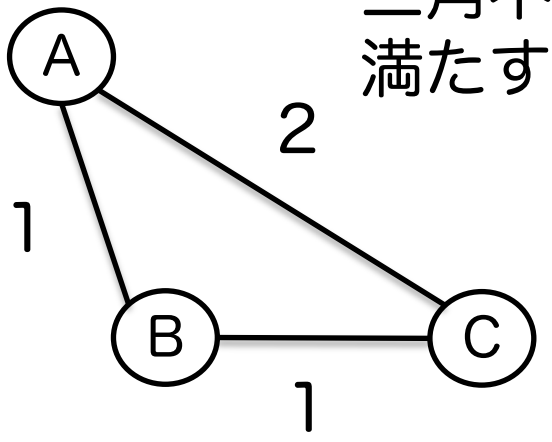
- 入力グラフを完全グラフに限定.
- 2 点 x, y の間の重みを $\mathrm{w}(x, y)$ とするとき，グラフ G の任意の 3 頂点 A, B, C について，下記の三角不等式を満たす.

$$w(A, B) \leq w(A, C) + w(B, C)$$

三角不等式を満たさない



三角不等式を満たす



近似度

Metric TSP は NP 困難（決定問題では NP 完全問題）

なので、最適解を多項式時間で求めることは難しい

⇒ よい近似度を入力サイズの多項式時間で求める

アルゴリズムの能力を「近似度」で比較する.

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

c 近似アルゴリズム ： 近似度 ≦ c を満たすアルゴリズム

Metric TSP の近似度

多項式時間近似アルゴリズム

- 2 近似 [Folklore]
- 1.5 近似 [Cristofides 1976]
- $1.5\text{-}10^{-36}$ 近似 [Karlin, Klein, and Graham 2020]

近似困難性

- P=NP でないかぎり、123/122 近似多項式時間アルゴリズムは存在しない [Karpinski, Lampis, and Schmied 2006]

Metric TSP の厳密アルゴリズム

厳密アルゴリズムは動的計画法（ Dynamic Programming ）による $O(n^2 2^n)$ 時間アルゴリズムより本質的に高速なアルゴリズムは知られていない

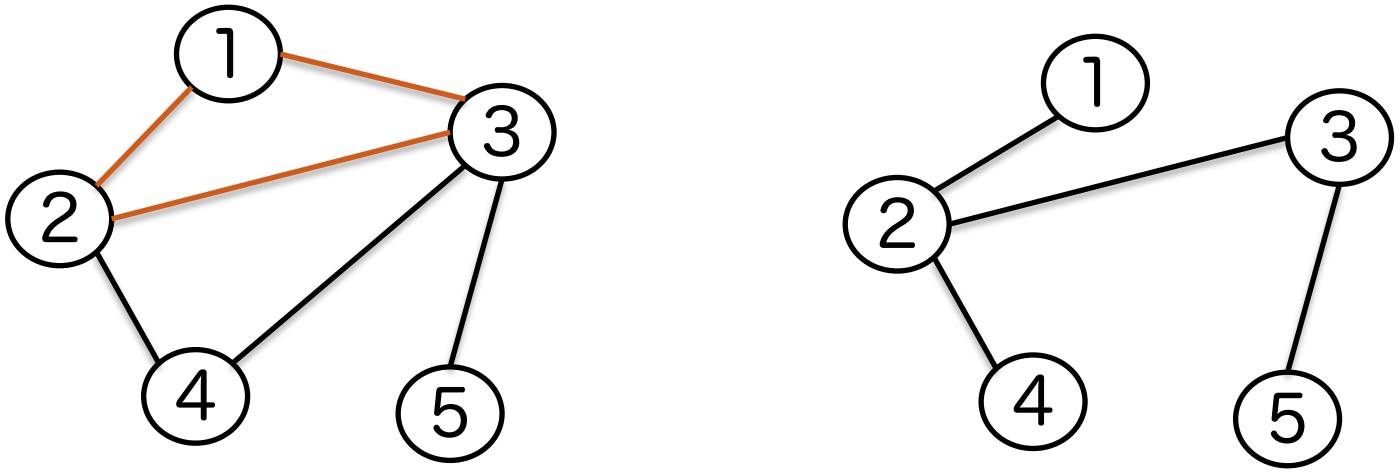
動的計画法での解き方については調べてみてください
日本語の記事も英語の記事もいっぱい出てきます

Metric TSP の 2 近似アルゴリズム

木

閉路（ループ）のないグラフ

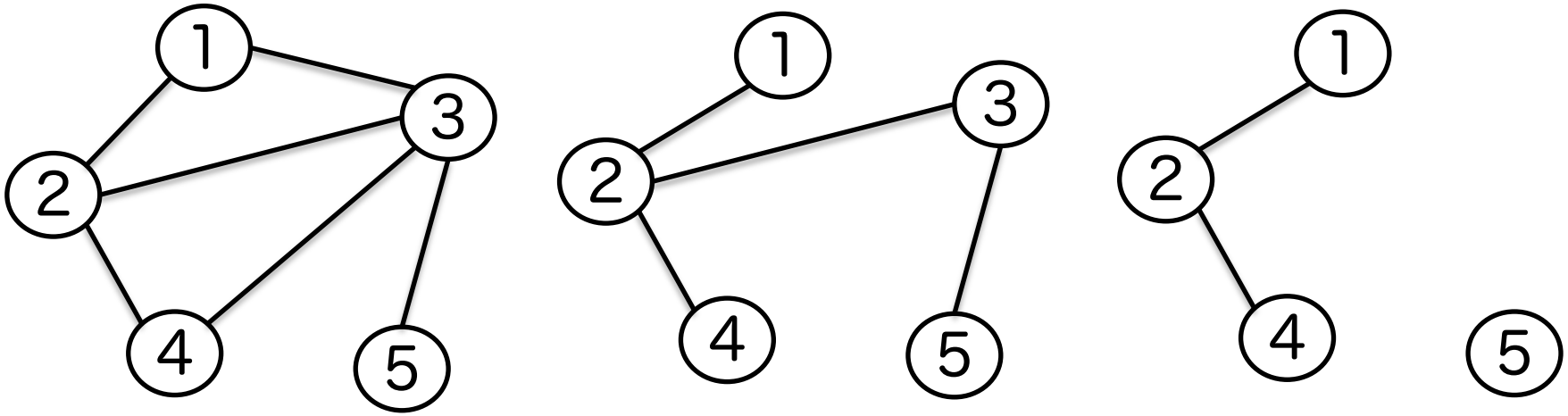
- 左は 1-2-3-1 というループがあるので木ではない
- 右は閉路がないので木



部分グラフ

グラフのうち一部の頂点や辺を利用したグラフ

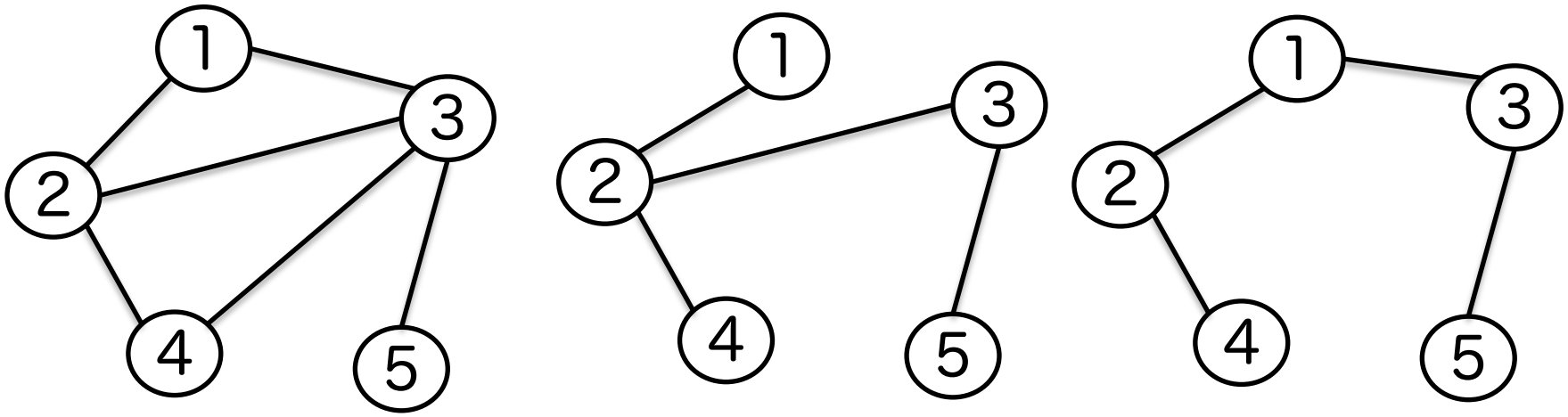
- 右 2 つは一番左の部分グラフ



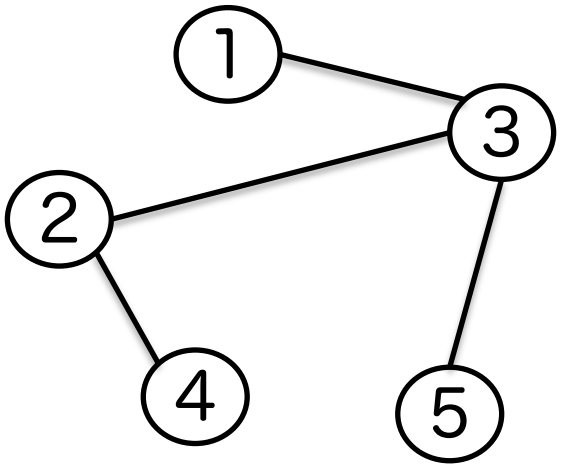
全域木

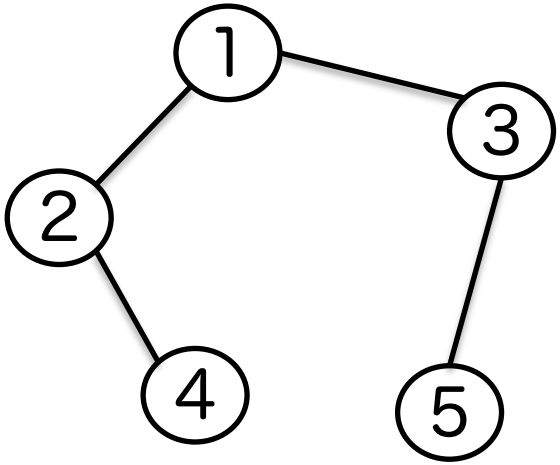
部分グラフのうち全頂点が連結している木

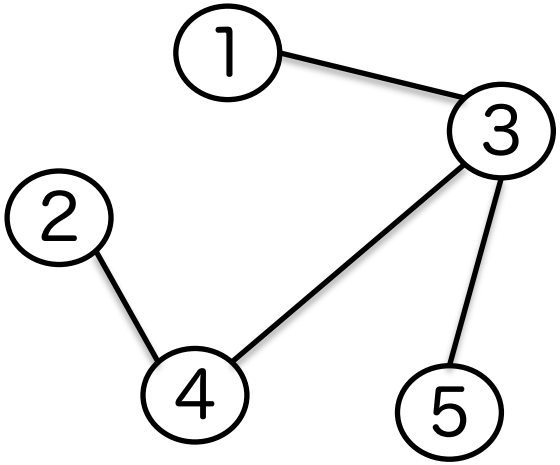
- 右 2 つは一番左の全域木となっている
- 次スライドで一番左のグラフのすべての全域木を列挙

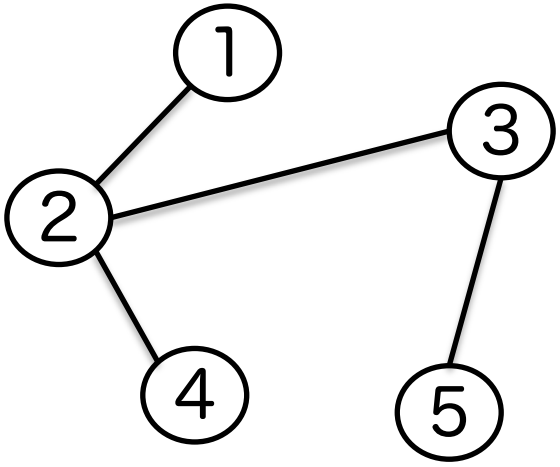


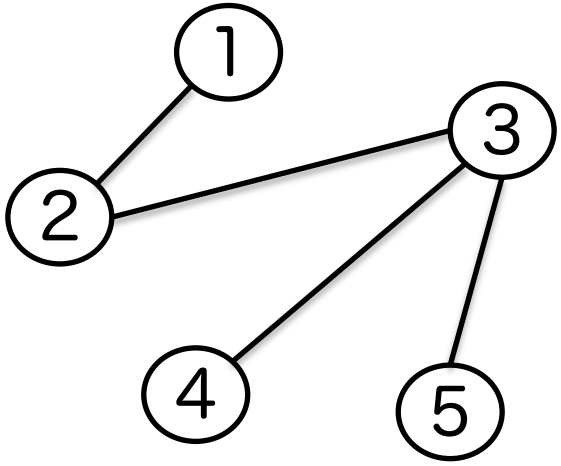
全域木

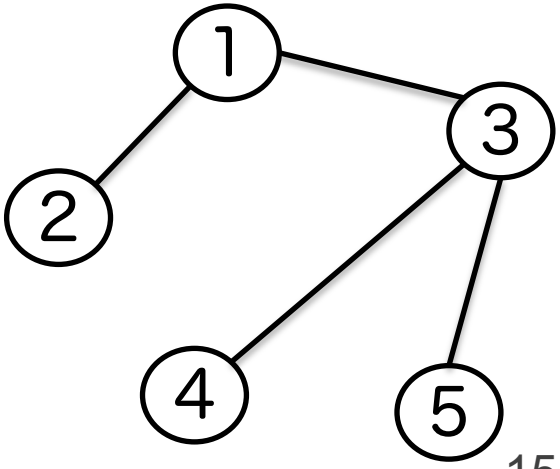








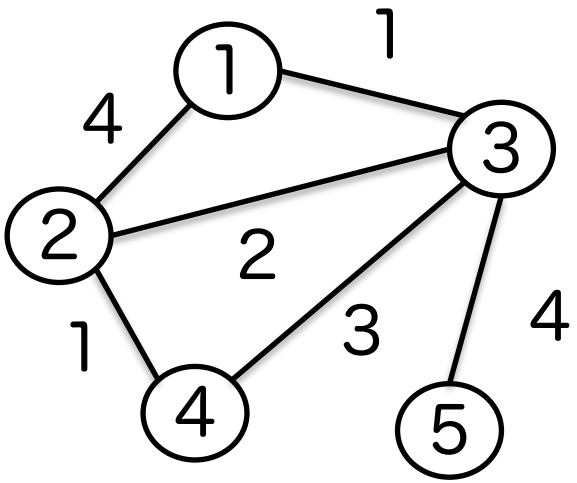


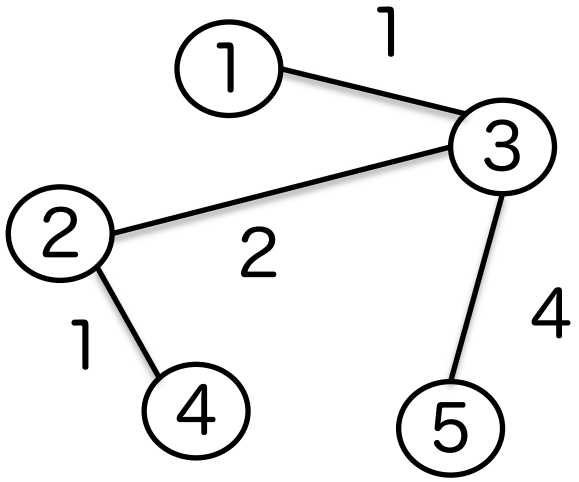


最小全域木

重み付きグラフの全域木の中で辺の重みの総和が最小のもの

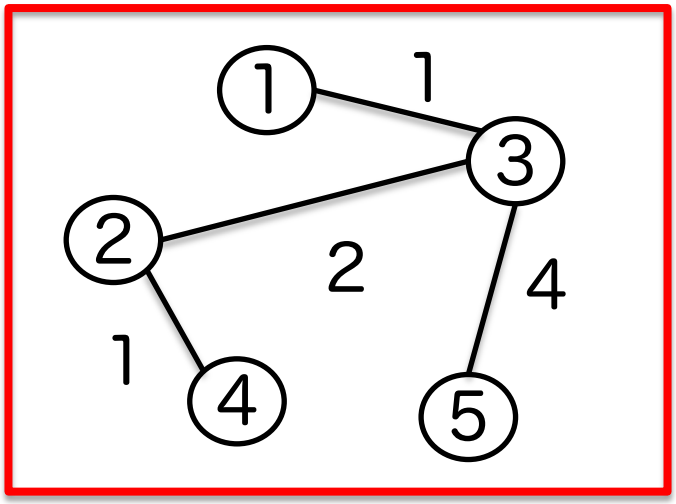
- 左のグラフの最小全域木は右の木

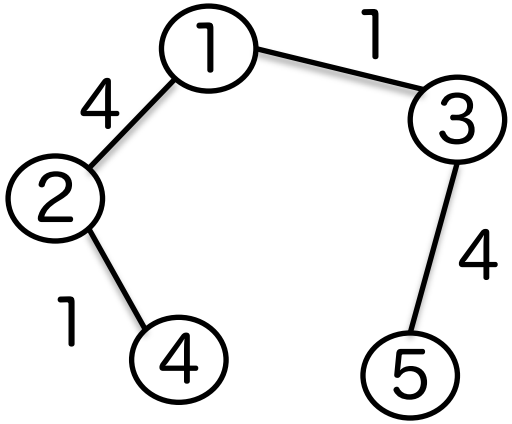


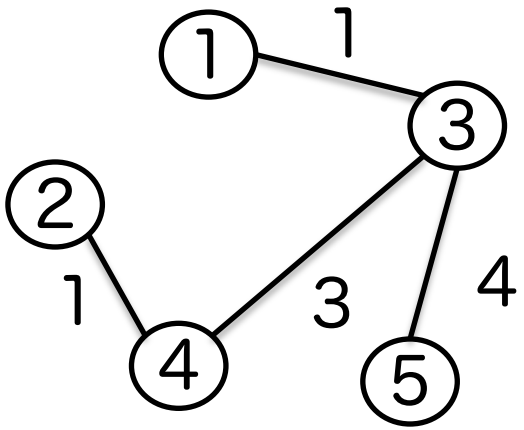


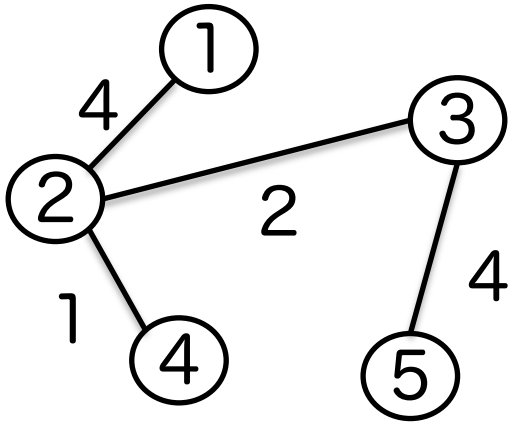
最小全域木

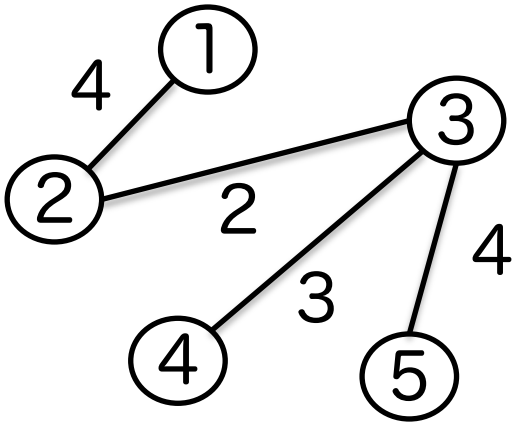
重み付きグラフの全域木の中で辺の重みの総和が最小のもの

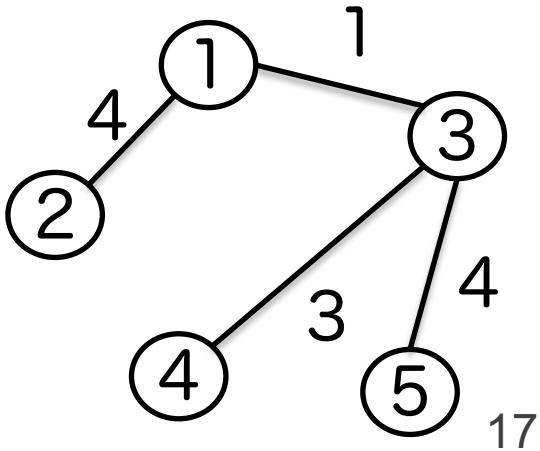












Metric TSP の 2 近似アルゴリズム

最小全域木を使った 2 近似アルゴリズム

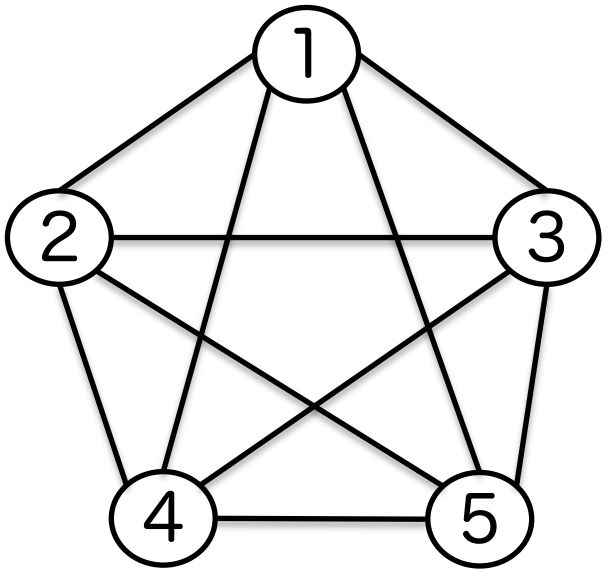
1. グラフ G の最小全域木 T を求める
2. T の各辺を二重にしたグラフ T’ を作成する
3. T’ 上で全ての頂点を通る巡回路 Cを求める

   ※ 同じ頂点を 2 度通ってよい
4. C のうち 2 度目に通る頂点をショートカットした巡回路 C’ を求める
5. C’ と閉路の辺の重みの総和を出力する

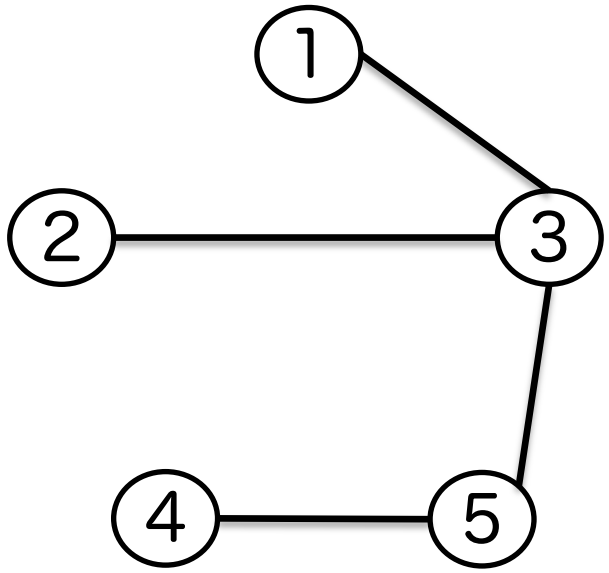
アルゴリズムの挙動

1. 最小全域木 T を求める

   Kruskal 法（後で説明）や Prim 法を用いればよい



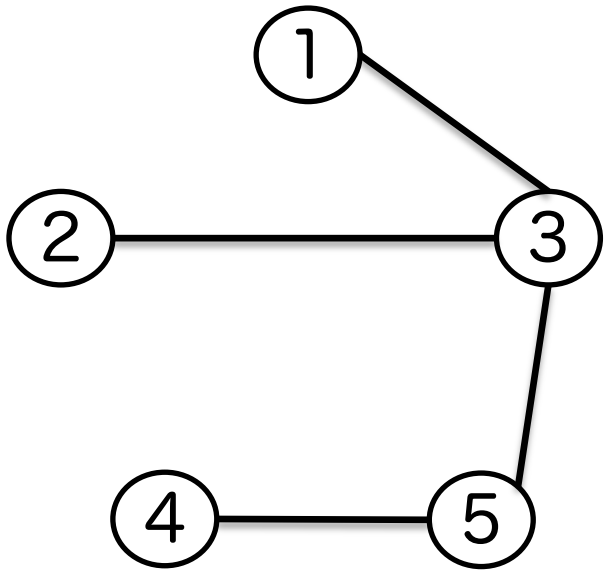
G



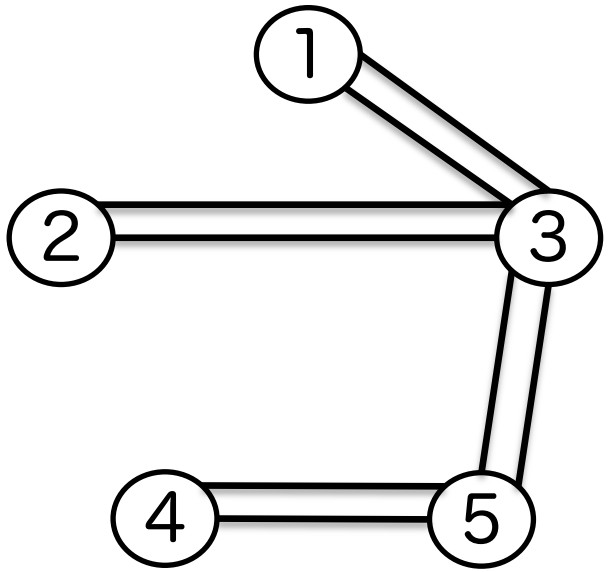
T

アルゴリズムの挙動

2. T の各辺を二重にしたグラフ T’ を作成する



T

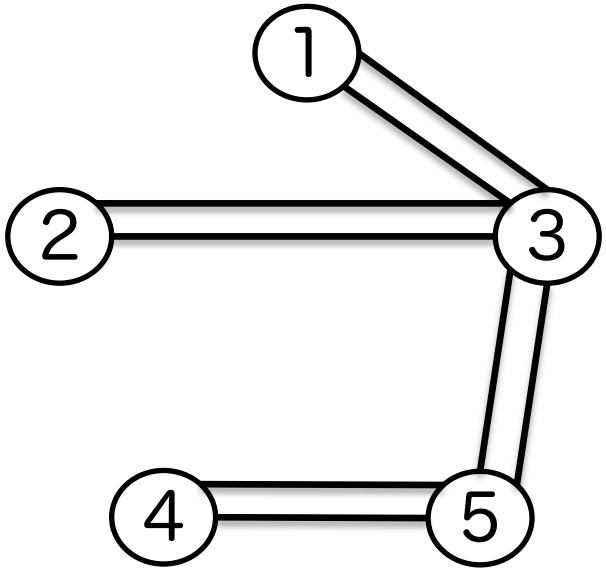


T’

アルゴリズムの挙動

3. T’ 上で全ての頂点を通る巡回路 C を求める

※ 同じ頂点を 2 度通ってよい



T’

1 → 3 → 2 → 3 → 5

→ 4 → 5 → 3 → 1

C

アルゴリズムの挙動

4. C のうち 1 度通ったことのある頂点をショートカットした巡回路 C’ を求める

1 → 3 → 2 → 3 → 5
→ 4 → 5 → 3 → 1

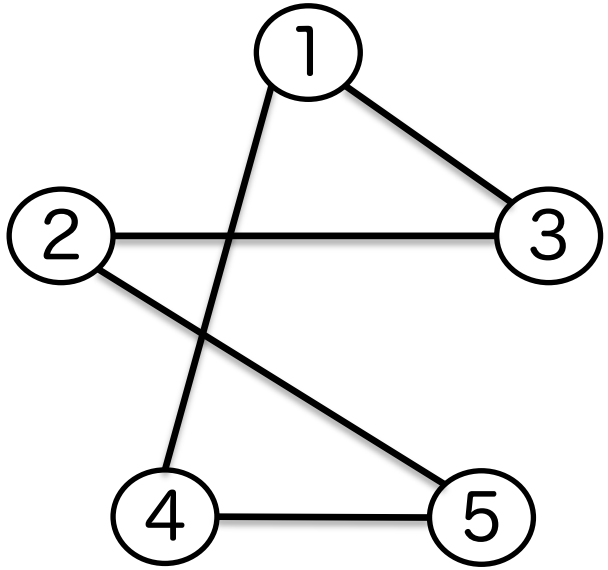
C

1 → 3 → 2 → 5 → 4 → 1

C’

アルゴリズムの挙動

5. C’ とそのコスト（距離の合計）を出力する



1 → 3 → 2 → 5 → 4 → 1

C’

アルゴリズムの計算時間

G の頂点数を n、辺の数を m とする

1. グラフ G の最小全域木 T を求める → $O(m\log_2 n)$（後述）
2. T の各辺を二重にしたグラフ T’ を作成する → $O(n)$
3. T’ 上で全ての頂点を通る巡回路 C を求める → $O(n)$
4. C のうち 2 度目に通る頂点をショートカットした巡回路 C’ を求める → $O(n)$
5. C’ と閉路の辺の重みの総和を出力する → $O(n)$

よって、 $O(m\log_2 n)$

2 近似の証明

最適解の巡回路の辺の重みの総和を OPT

最小全域木の辺の重みの総和を MST

とする。

まず、OPT と MST を比較すると、

$$MST \leqq OPT$$

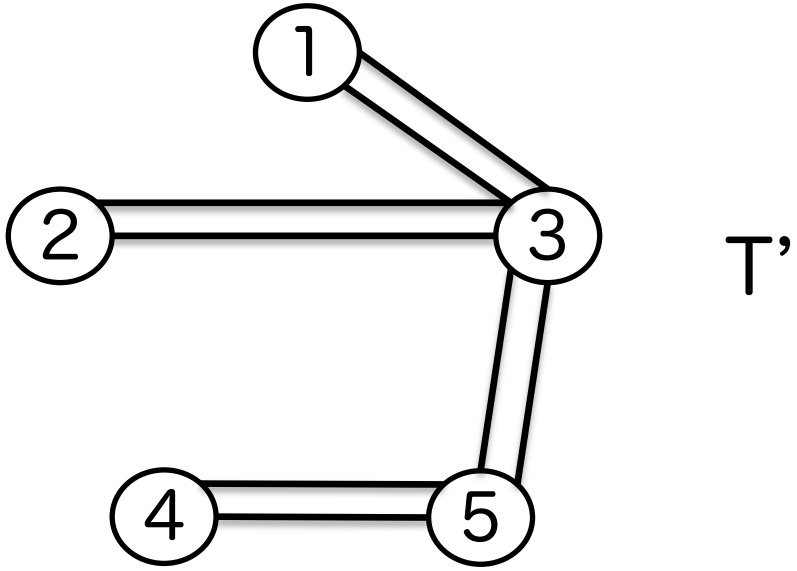
が成り立つ。

2 近似の証明

最小全域木の枝を 2 重化したグラフ T’ 上で得られた閉路 C を考える。

前述の例なら、1 → 3 → 2 → 3 → 5 → 4 → 5 → 3 → 1

この経路のコストは 2×MST となる。



2 近似の証明

次に最終的に得られた巡回路 C’ を考える。

C’ の辺の重みの総和を ALG とすると、

$$ALG \leqq 2 \times MST$$

を満たす。

なぜ？？

2 近似の証明

C : 1 → 3 → 2 → 3 → 5 → 4 → 5 → 3 → 1

C': 1 → 3 → 2 → 5 → 4 → 1

の経路を考えると、

まず、 2 → 3 → 5 を 2 → 5 とショートカットする。

Metric TSP では、任意の 3 頂点間の重みの間に三角不等式が成り立っているので、

$$w(2,5) \leq w(2,3) + w(3,5)$$

が成り立つ。その他のショートカットも同じ議論が成り立つので、

$$\mathrm{ALG} \leqq 2 \times \mathrm{MST}$$

が得られる。

2近似の証明

これまでの議論により、

- MST ≦ OPT
- ALG ≦ 2×MST

が得られたので、

$$ALG \leqq 2 \times MST \leqq 2 \times OPT$$

となることから、前述したアルゴリズムの近似度が 2 であることが示される。

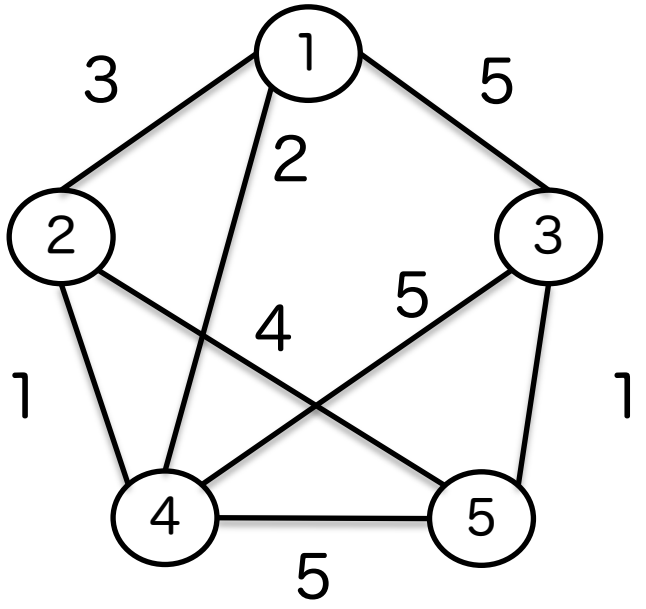
最小全域木を求める

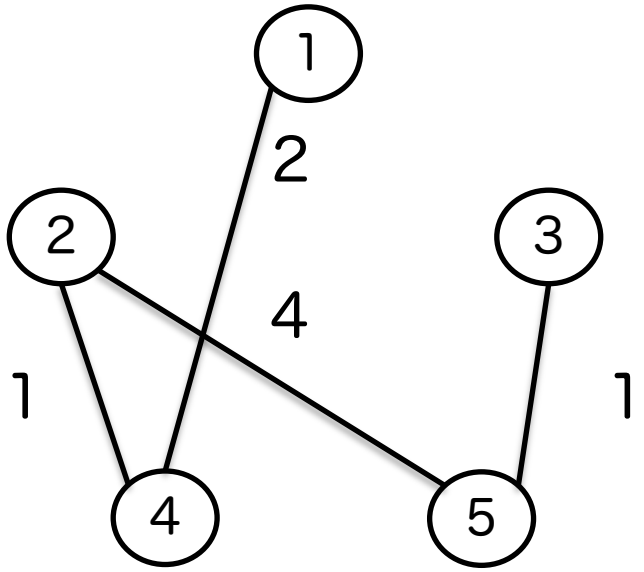
Kruskal アルゴリズム

Kruskal アルゴリズム

左のグラフの最小全域木は右となる。

これを Kruskal アルゴリズムで求める。





Kruskal アルゴリズム

Kruskal のアルゴリズム：n はグラフ G の頂点数

1. 辺が n-1 本採用されるまで下記を繰り返す

   A) 辺の重みが小さい順に辺を選択していく

   - もし、選択した辺がこれまでに採用した辺集合と閉路を作ってしまうなら却下し、次の辺の選択に移る
   - そうでなければ、その辺を採用する

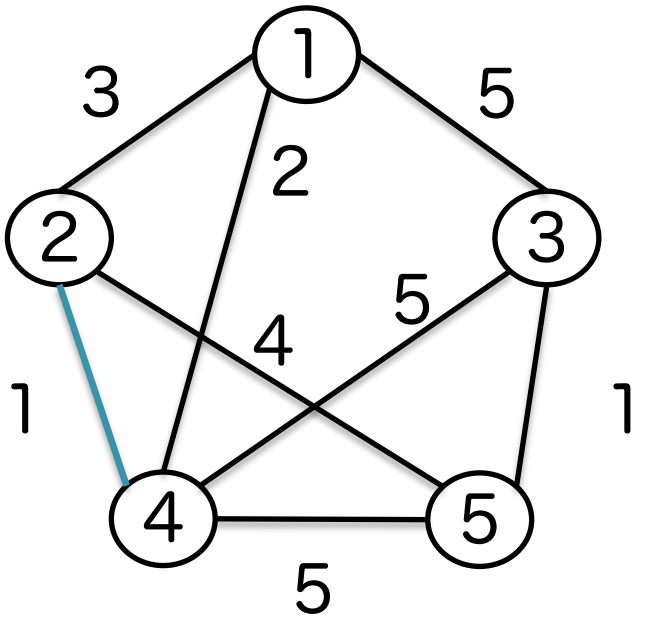
2. 採用した n-1本の辺集合から構成される木 T を出力する

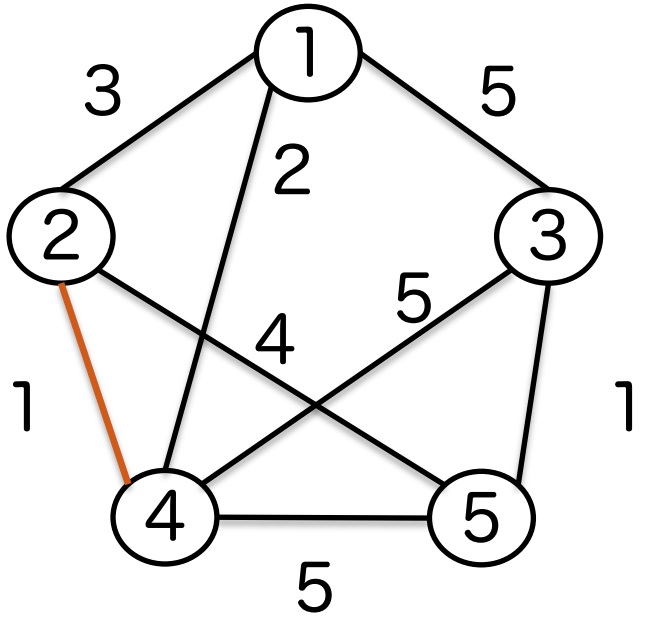
Kruskal アルゴリズムの挙動

辺の重みが小さい順に辺を選択するので,

まずは (2, 4) か (3, 5) を選択する（どちらを選択してもよい）今、(2, 4) を選択することにする。

最初の辺なので、閉路を作らないので採用する。

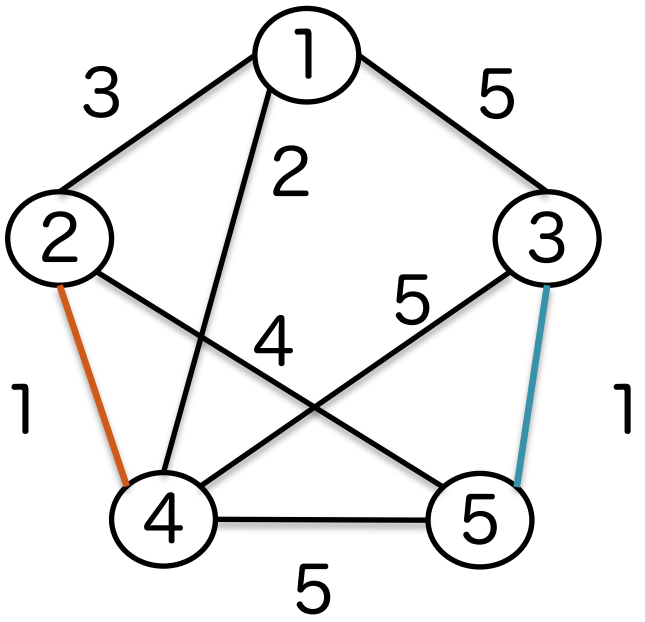


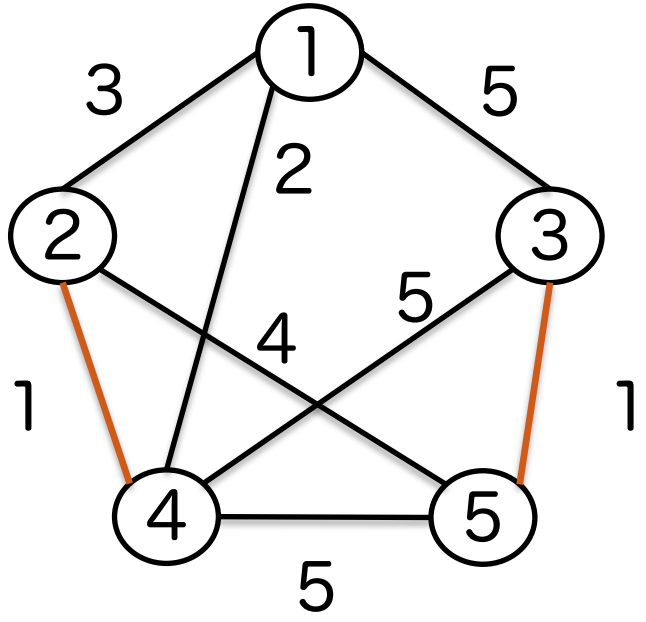


Kruskal アルゴリズムの挙動

次に辺の重みが小さいのは (3, 5) なのでこれを選択する。

2 本目の辺なので、閉路を作らないので採用する。

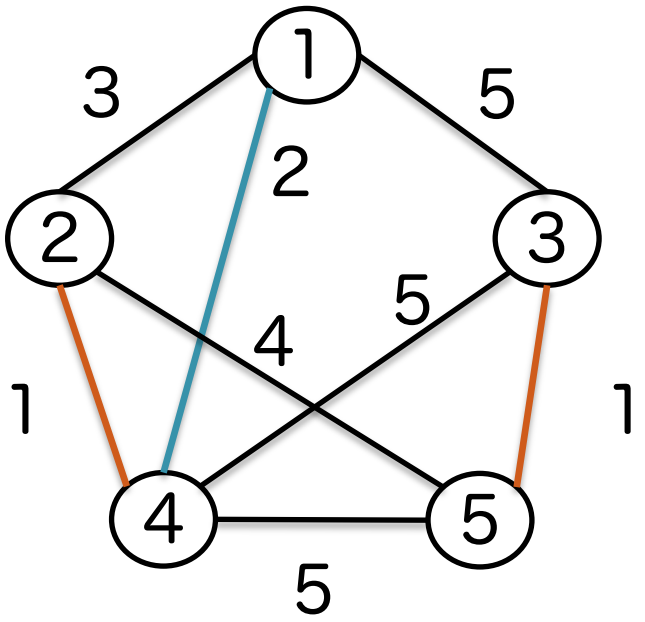


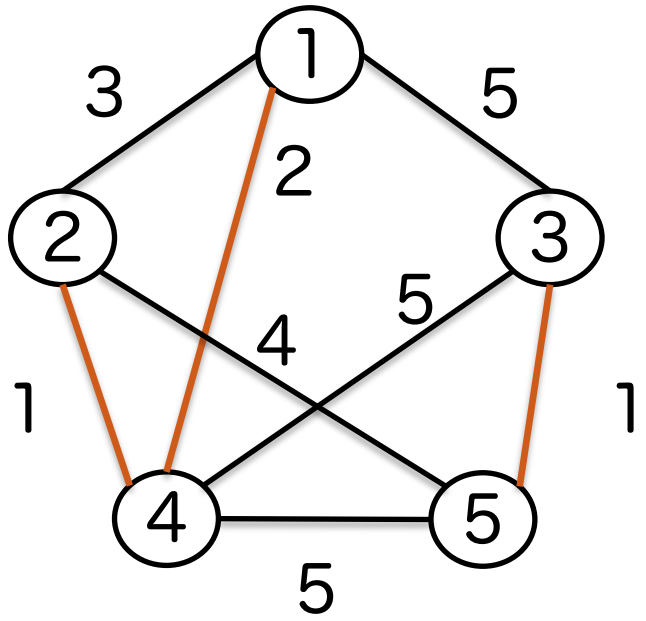


Kruskal アルゴリズムの挙動

次に辺の重みが小さいのは (1, 4) なのでこれを選択する。

これまで採用した辺と閉路を作らないので採用する。

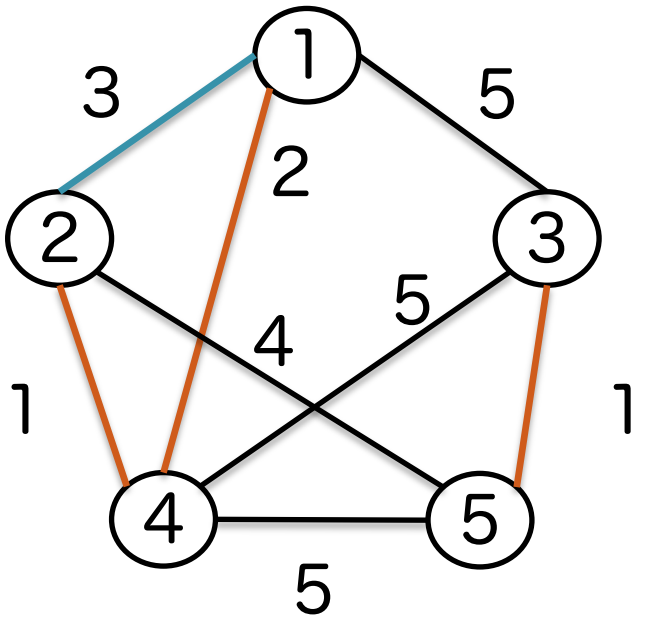


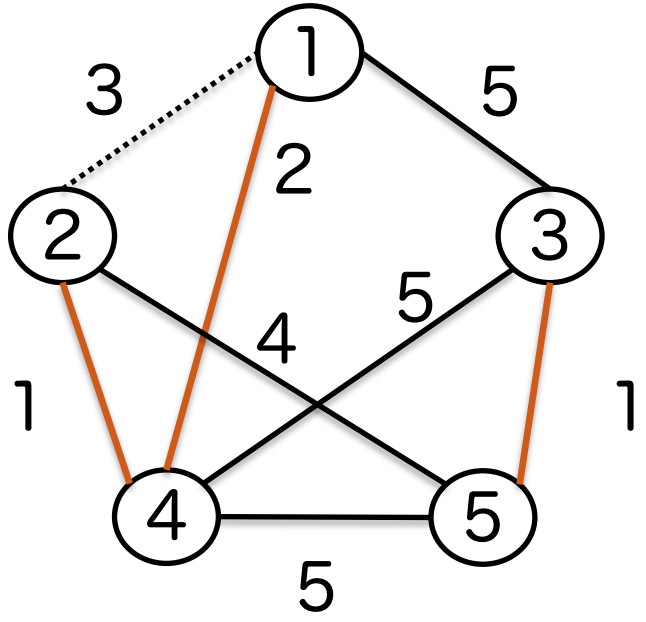


Kruskal アルゴリズムの挙動

次に辺の重みが小さいのは (1, 2) なのでこれを選択する。

しかし、この辺は、これまで採用した辺と閉路を作ってしまうので却下する。

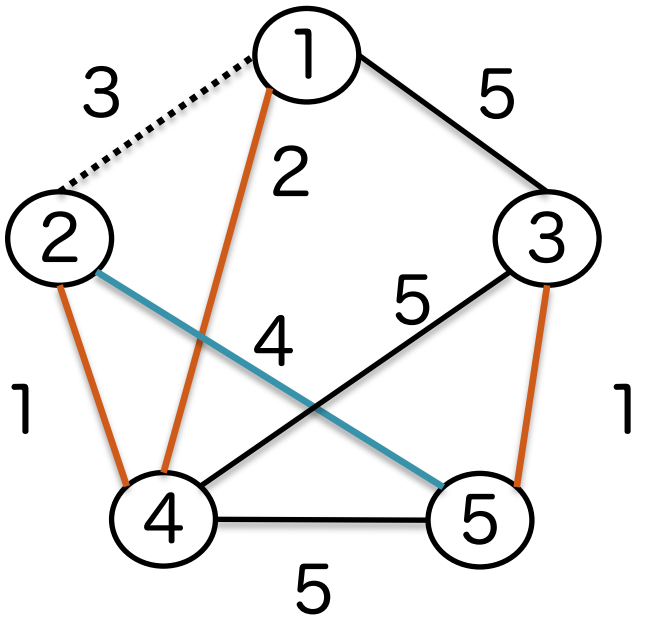


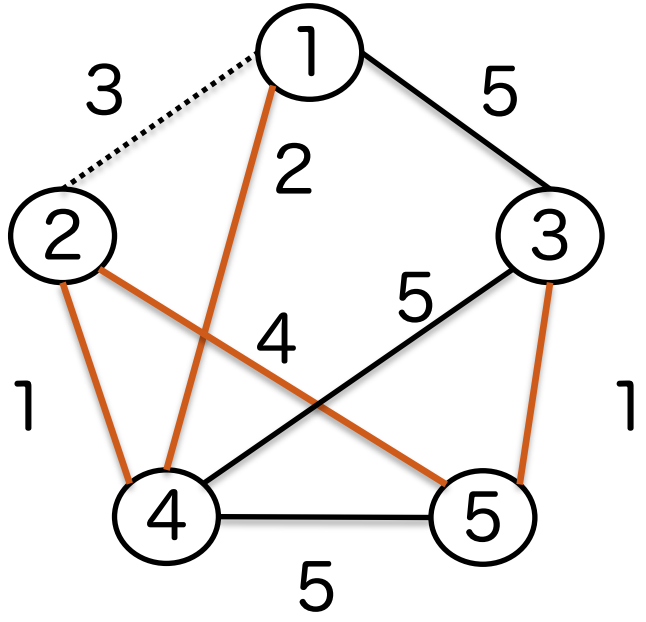


Kruskal アルゴリズムの挙動

次に辺の重みが小さいのは (2, 5) なのでこれを選択する。

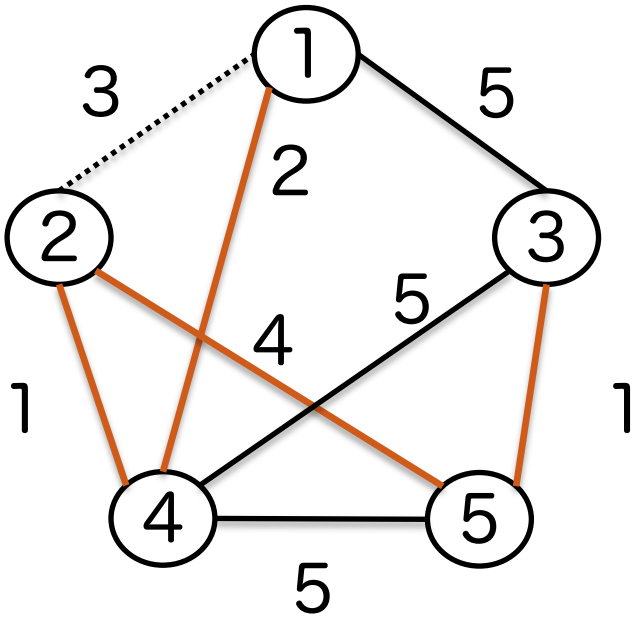
これまで採用した辺と閉路を作らないので採用する。

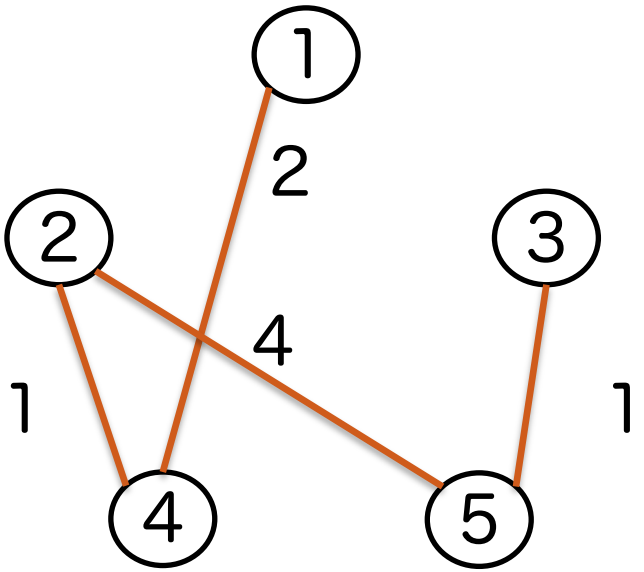




Kruskal アルゴリズムの挙動

辺を 4 本 (= n-1 本) 採用したので、この時点でアルゴリズムは終了し、これまで採用した辺で構成された木を出力する。





Kruskal アルゴリズムの計算時間

Kruskalのアルゴリズム：G の頂点数 n，辺の数 m

1. 辺が n-1 本採用されるまで下記を繰り返す
   - 閉路が構成されるかのチェック → $O(m)+O(n\log_2 n)$
   - 素集合データ構造を利用する
2. 採用した n-1 本から構成される木 T を出力する → $O(n)$

辺のソートが必要：$O(m\log_2 m) = O(m\log_2 n) \leftarrow m \leqq n^2$

よって、$O(m\log_2 n)$

まとめ

NP 完全問題へのアプローチの 1 つである近似アルゴリズムの基礎について学んだ

- Metric TSP の 2 近似アルゴリズム
- 最小全域木を求める Kruskal アルゴリズム

扱うデータの規模は年々巨大になってきている

- $O(n^3)$ 時間のアルゴリズムでも使い物にならない可能性も
- どうやって高速によい解を見つけるかは非常に重要
- 最悪計算時間が保証されているとさらに嬉しい