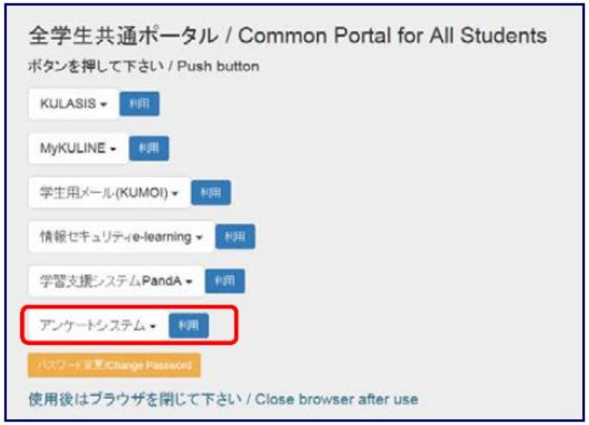
授業アンケートの方法（授業アンケートシステム：KULIQS-クリックス-）

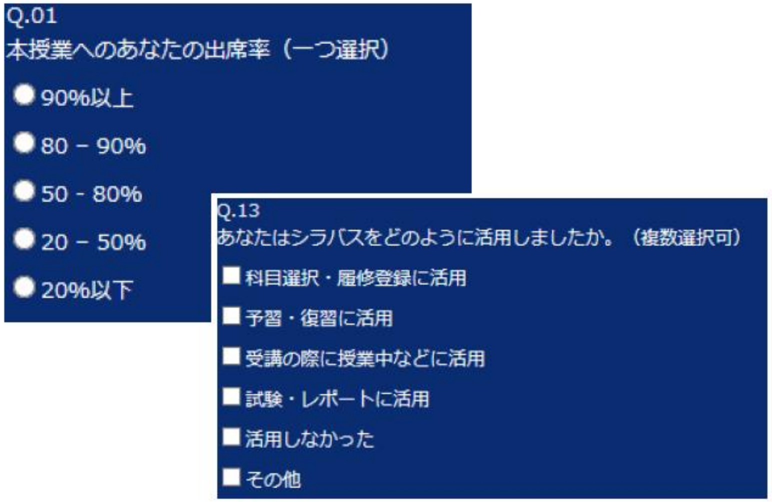
授業アンケートは，スマートフォンを利用したアンケートシステム(KULIQS-クリックス-)で行います。（スマートフォンを利用していない場合は，メディアセンターや自宅のパソコンからご回答ください。）

KULIQSは，「全学生共通ポータル」･「KULASIS TOP」よりログインすることができますので，画面に沿って回答してください。



（アンケートホーム）

（アンケート回答）



Kyoto University

アルゴリズムとデータ構造⑫
～ 近似アルゴリズムとオンラインアルゴリズム ～

鹿島久嗣

Department of Intelligence Science and Technology

近似アルゴリズム

難しい問題への対処法：計算量や性能を犠牲に解を得るための有効な方法

- 多項式時間で最適解を得る保証はないが、有用であることが経験的にわかっている汎用的な方法：
  - 分枝限定法・局所探索：多項式時間ではないが、厳密解を得るための方法
  - 近似アルゴリズム：多項式時間で動くが、厳密解が得られるわけではない

近似アルゴリズム：
理論保証のある近似解を得るアルゴリズム

- 最適解を得ることは保証されないが、
  最適解からの乖離の小ささが保証されるアルゴリズム

- 近似度：任意の入力（問題例）xに対して
  $C_A(x) \leq r\, C_{\mathrm{OPT}}(x) + d$　（最小化問題の場合）

  が成り立つとき、Aの近似度がrであるという（ dは定数）
  - $C_A(x)$：入力xに対するアルゴリズムAのコスト
  - $C_{\mathrm{OPT}}(x)$：厳密解のコスト

- 多くの場合アルゴリズムは単純（貪欲法など）

NP困難問題の例①：
ユークリッド空間での巡回セールスマン問題

- 巡回セールスマン問題（TSP）：最小のコストのハミルトン閉路を求めるNP困難問題
  – 頂点(v_i, v_j)の間に非負のコスト$c(v_i, v_j) \geq 0$がある
- メトリックTSP（ユークリッド空間におけるTSP）：
  – 任意の三点間(v_i, v_j, v_k)に三角不等式が成り立つ：

$$c(v_i, v_j) + c(v_j, v_k) \geq c(v_i, v_k)$$

（直線移動は回り道より常に早い）

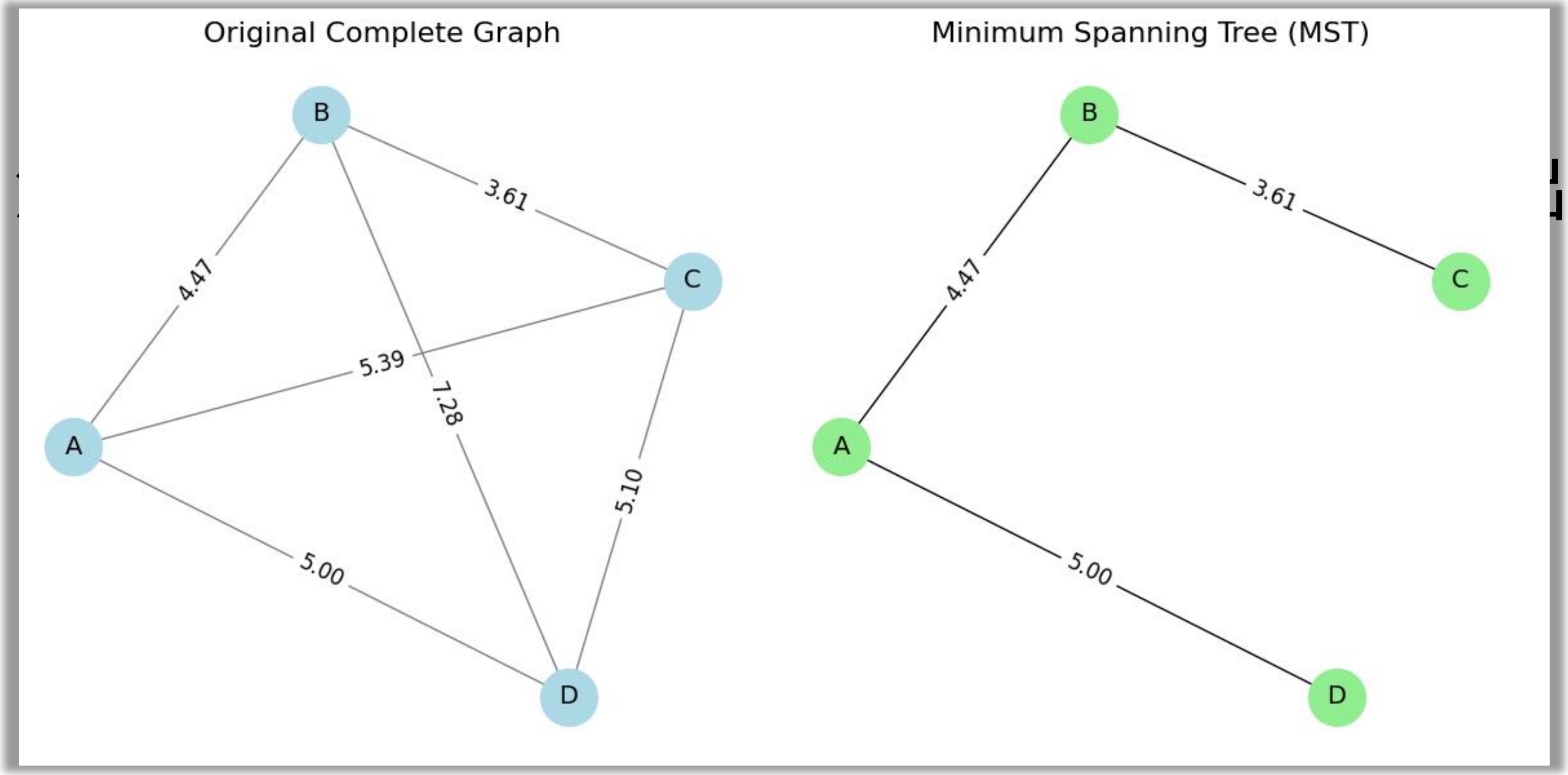
  –グラフは重み付き完全グラフだと思う

メトリックTSPの近似アルゴリズム：最小全域木をもとに経路を構成する

1. 頂点集合に対する最小全域木（MST）を求める
2. MSTを根付き木だと思って、深さ優先探索を行う
3. 深さ優先探索の出力順に訪問し根に戻る閉路を出力

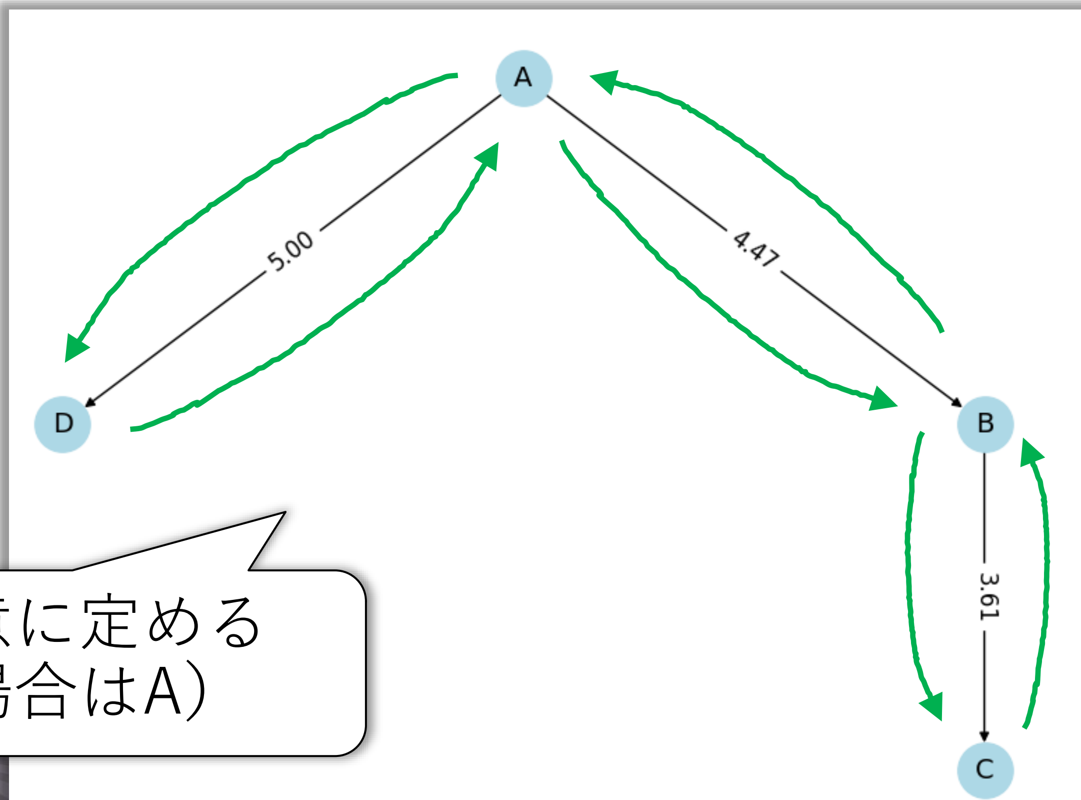
メトリックTSPの近似アルゴリズム：最小全域木をもとに経路を構成する

1. 頂点集合に対する最小全域木（MST）を求める
2.
3. 出力



メトリックTSPの近似アルゴリズム：
最小全域木をもとに経路を構成する

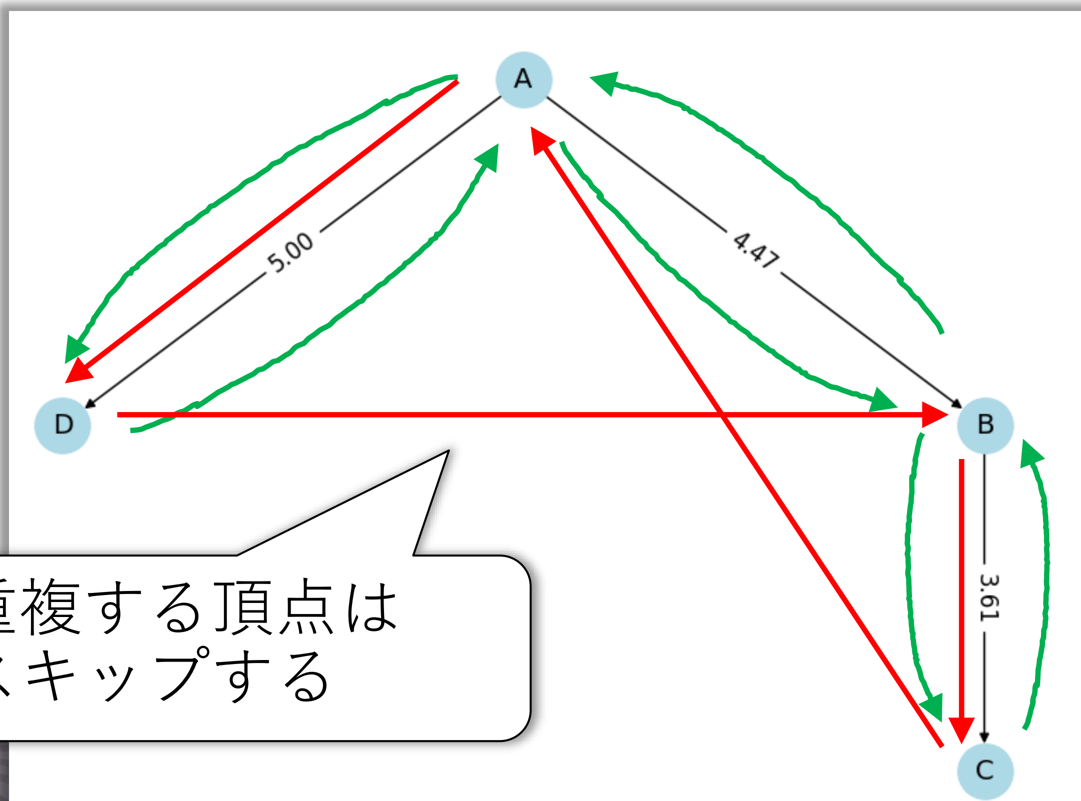
1. 頂点集合に対する最小全域木（MST）を求める
2. MSTを根付き木だと思って、深さ優先探索を行う
3. 深さ優先探索の出力順に訪問し根に戻る閉路を出力



根は任意に定める
（この場合はA）

メトリックTSPの近似アルゴリズム：
最小全域木をもとに経路を構成する

1. 頂点集合に対する最小全域木（MST）を求める
2. MSTを根付き木だと思って、深さ優先探索を行う
3. 深さ優先探索の出力順に訪問し根に戻る閉路を出力



重複する頂点は
スキップする

メトリックTSPの近似アルゴリズムの近似度：近似度は2

- MSTのコストC_{MST}はTSPの最適コストC_{OPT}以下である

$$C_{\mathrm{MST}} \leq C_{\mathrm{OPT}}$$

- 深さ優先探索の経路のコストC_{DFS} は C_{MST}の2倍以内

$$C_{\mathrm{DFS}} \leq 2 \cdot C_{\mathrm{MST}}$$

  – MSTのすべての辺を高々2回通過するため

- 近似アルゴリズムの解のコストC_{TSP}はC_{DFS}以下である

$$C_{\mathrm{TSP}} \leq C_{\mathrm{DFS}}$$

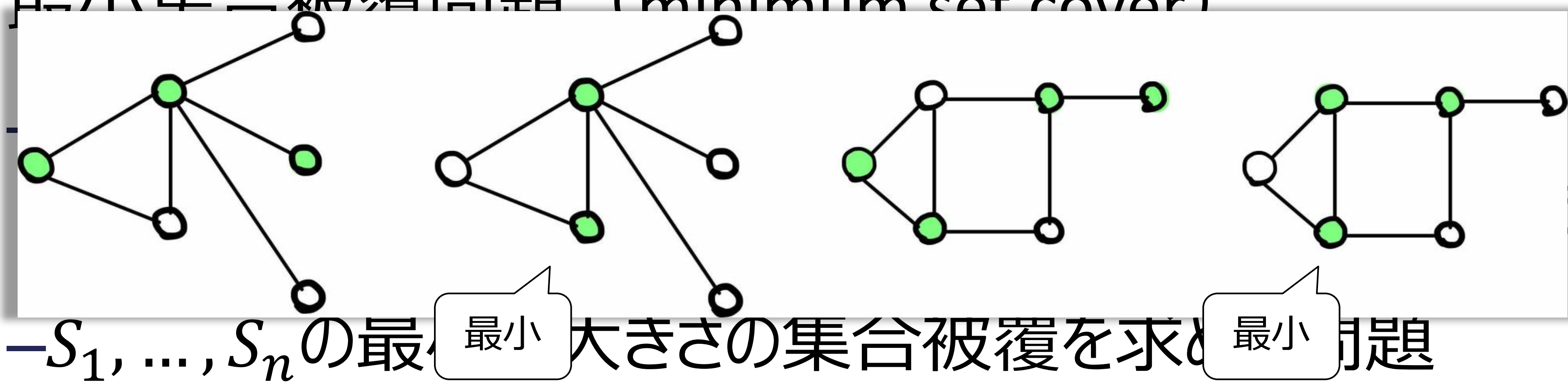
  – 頂点をスキップすると距離は短くなる（三角不等式）

NP困難問題の例②：最小頂点被覆問題、最小集合被覆問題

- 最小頂点被覆問題（minimum vertex cover）
  - 頂点被覆：グラフ$G=(V,E)$において全ての$e\in E$の少なくとも一方が$V'\in V$に含まれているときV'を頂点被覆
  - Gに対する最小頂点数の頂点被覆を求める問題
- 最小集合被覆問題（minimum set cover）：
  - n個の集合$S_1, S_2, \ldots, S_n$があるとき、そのうちk個を用いて$\bigcup_{j=1}^{n} S_j = \bigcup_{j=1}^{k} S_j$とできるとき大きさ$k$の集合被覆という
  - $S_1, \ldots, S_n$の最小の大きさの集合被覆を求める問題

NP困難問題の例：最小頂点被覆問題、最小集合被覆問題

- 最小頂点被覆問題（minimum vertex cover）
  - 頂点被覆：グラフ$G = (V, E)$において全ての$e \in E$の少なくとも一方が$V' \in V$に含まれているときV'を頂点被覆
  - Gに対する最小頂点数の頂点被覆を求める問題
- 最小集合被覆問題（minimum set cover）：



  - $S_1, \dots, S_n$の最 … 大きさの集合被覆を求め … 問題

最小頂点被覆問題の近似アルゴリズム：近似度2のアルゴリズム

- 最小な頂点被覆のサイズの2倍を超えないことが保証されたアルゴリズム：$C_A(x) \leq 2\, C_{\mathrm{OPT}}(x)$
- アルゴリズム：

1. 問題例のグラフGからスタート（$G' = G$とする）
2. 現在のグラフG'から任意の枝eを選び、eの両端の頂点u, vを頂点被覆集合Cに加える
3. u, vに接続している枝をグラフG'から取り除く
4. グラフG'の枝がまだ残っているならステップ2へ

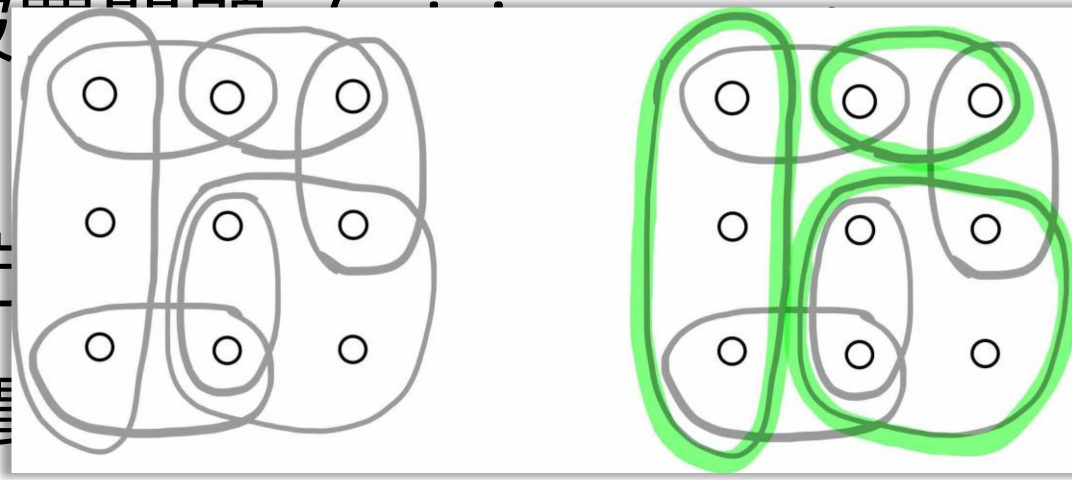
頂点被覆問題の近似アルゴリズム：証明

- ステップ2で選ばれた枝の集合Eを考える
- Eの辺は端点を共有しない（ステップ3より）ので、

  $|C| = 2|E|$ （ Cはこのアルゴリズムで採用した頂点集合）
- 最適な頂点被覆C^*はEの各辺のすくなくとも一方を踏んでいる（頂点被覆の定義より）
- Eの辺は端点を共有しないので、Eをカバーするためには少なくとも$|E|$個の頂点が必要： $|E| \leq |C^*|$
- 上記の2つの式をつなげると、 $|C| = 2|E| \leq 2|C^*|$

NP困難問題の例：最小頂点被覆問題、最小集合被覆問題

- 最小頂点被 … er）
  - 頂点被覆 … $e \in E$の少 なくとも一 … を頂点被覆
  - Gに対する最 … る問題



- 最小集合被覆問題（minimum set cover）：
  - n個の集合$S_1, S_2, \dots, S_n$があるとき、そのうちk個を用いて $\bigcup_{j=1}^{n} S_j = \bigcup_{j=1}^{k} S_j$ とできるとき大きさkの集合被覆という
  - $S_1, \dots, S_n$の最小の大きさの集合被覆を求める問題

集合被覆問題の近似アルゴリズム：貪欲法

- 最小の集合被覆のサイズの$\ln n + 1$倍（nは被覆される要素数）を超えない：$C_A(x) \le (\ln n + 1)\, C_{\mathrm{OPT}}(x)$
  - 注：問題サイズによって近似比が変わる
- アルゴリズム：貪欲法
  - まだ被覆されていない要素をもっとも多く被覆するような集合を次に選ぶ
  - 証明は面倒…（「アルゴリズムイントロダクション」参照）
- 近似アルゴリズムは、アルゴリズムは簡単だが解析は面倒

近似アルゴリズムの性能：
アルゴリズムによって保証される近似比のいろいろ

- 近似アルゴリズムで保証される近似比には、
  何段かのパターンがある：
  - 定数の場合
  - 問題サイズに依存する場合
  - （時間を掛ければ）いくらでも1に近づけられる場合

(F)PTAS：
いくらでも近似比を1に近づけられる近似アルゴリズム

- 多項式時間近似方式
  （PTAS; Polynomial-Time Approximation Scheme）
  - （時間を掛ければ）いくらでも1に近づけられる
    任意の $\epsilon > 0$に対して $C_A(x) \leq (1+\epsilon)\, C_{\mathrm{OPT}}(x)$
    - ただし、ϵが小さくなるほど時間がかかる（$1/\epsilon$に依存）
      - 問題サイズに関しては多項式時間
- さらに、完全多項式時間近似方式（FPTAS；Fully PTAS)
  では、$1/\epsilon$について多項式時間
  - ナップザック問題にはFPTASアルゴリズムがある

オンラインアルゴリズム

オンラインアルゴリズム：各時点で分かっている情報をもとに逐次意思決定を行う

- 通常の問題設定：問題例が与えられてから解をもとめる
- オンライン問題：
  - 問題例が一部分ずつ徐々に与えられる
    - 問題例全体をみれば最適解が求まる
  - これまでに分かっている情報から解の一部を構成する
- 株の取引き：
  - 現時点までの株価をもとに、次に買うか売るかを決める
  - 全期間の株価が分かっていれば最適な売り買いが可能

オンラインアルゴリズムの応用：
現実の多くの問題がオンラインでの意思決定に関わる

- キャッシュ置換：
  - 限られたキャッシュメモリに保持しておくデータを選ぶ
  - 将来のリクエストは事前にはわからない
- ダイナミックプライシング：
  - 商品やサービスの価格を状況に応じてリアルタイムに調整
  - 将来の需要や顧客行動は事前には不明
- オンラインマッチング：
  - 到着するリクエストをリソースに割り当てる（Uberなど）

オンライン問題の例：スキーレンタル問題

- スキー板を買うと10万円、レンタルだと1回1万円かかる
- 今シーズン何回スキーに行くかはあらかじめわからない
  - 0回かもしれないし、∞回行くかもしれない
- 各時点で、スキーに行くか、もう行かないかが観測される
- スキーにいくたびに買うか、レンタルするかを決定する
- どのような戦略をとれば一番得するか？
- 最適解：シーズン中に何回行くか（x）が分かっていれば $x < 10$ なら全部レンタルにして、$n \geq 10$ならば買えばよい

オンラインアルゴリズムの性能評価：競合比

- 最適解：シーズン中に何回行くか（x）が分かっていれば $x < 10$ なら全部レンタルにして、$x \geq 10$ならば買えばよい

- 競合比：任意の（オフライン）問題例xに対して
  $C_A(x) \leq r\, C_{\mathrm{OPT}}(x) + d$ （最小化問題の場合）

  が成り立つとき、Aの競合比がrであるという

  – $C_A(x)$：入力xに対するオンラインアルゴリズムAのコスト

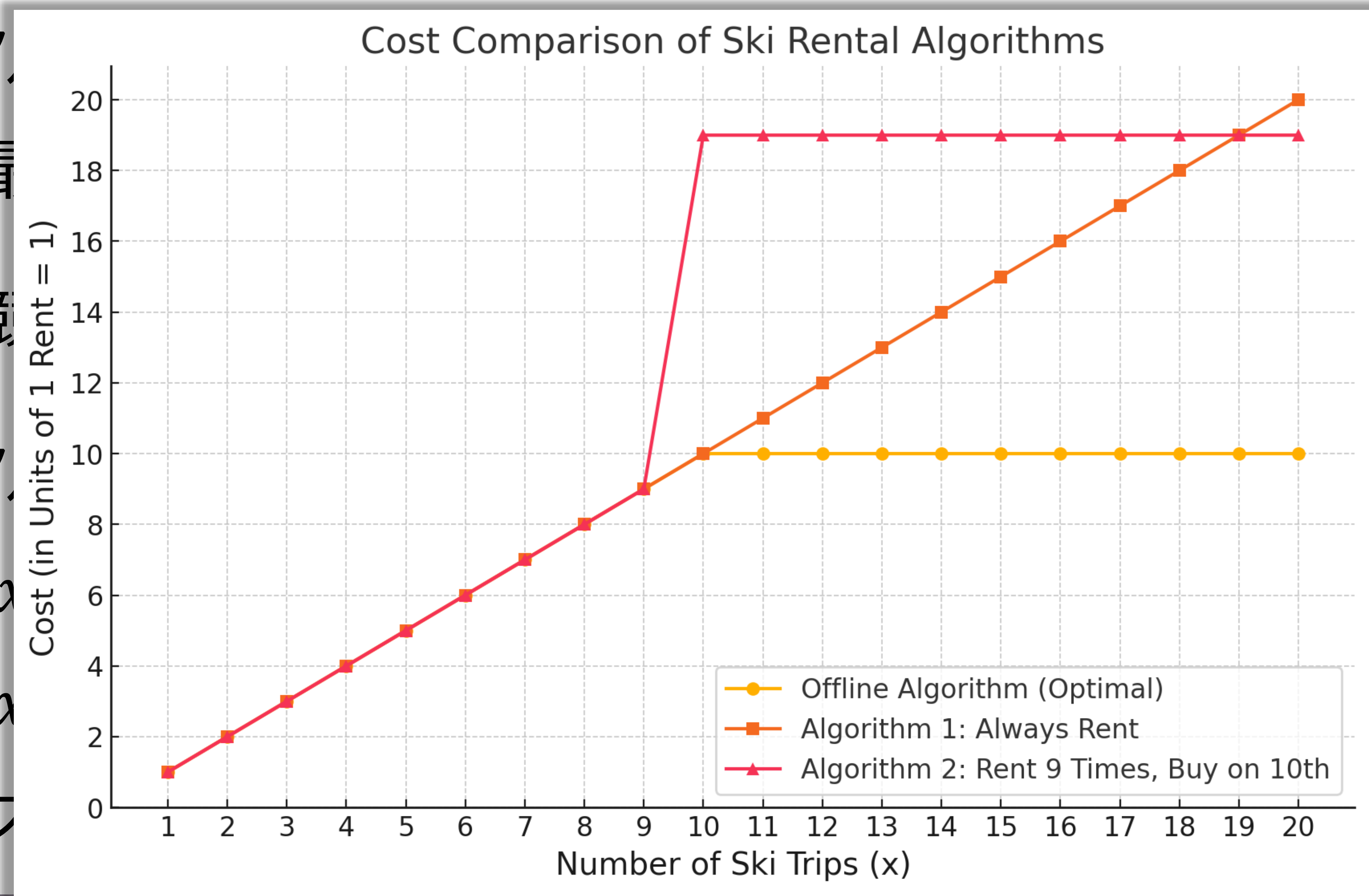
  – $C_{\mathrm{OPT}}(x)$：入力xをあらかじめ知っているときのコスト（オフラインアルゴリズムのコスト）

スキーレンタル問題の競合比：
競合比1.9が実現できる

- アルゴリズム１：つねにレンタル
  - 最終的にx回行ったとすると、レンタル費用はx
  - 競合比は∞ （$\because x \geq 10$で $\frac{c_A(x)}{c_{\mathrm{OPT}}(x)} = \frac{x}{10}$）
- アルゴリズム２：9回目までレンタルし、10回目で買う
  - $x = 9$まではオフラインと同じコスト
  - $x \geq 10$からはオフラインコスト10、オンラインは19
  - オフラインとオンラインの比は最悪でも1.9(= 競合比)

スキーレンタル問題の競合比：競合比1.9が実現できる

- ア
  - 最
  - 競
- ア
  - x
  - x
  - ス



難しい問題への対処法：
最適解は得られる保証はないが、性能保証のある方法

- 近似アルゴリズム： NP困難問題に対して、多項式時間で動くが、厳密解が得られるわけではない
- オンラインアルゴリズム：意思決定時点において、問題のすべての情報が明らかでない
- 理論保証：いずれも、最適解からどれだけ離れているかという性能保証がある
- いずれもアルゴリズムは単純だが、解析は複雑