離散最適化基礎論 (2025 年後学期)

高速指数時間アルゴリズム

第 5 回

動的計画法 (1) : 基礎

岡本 吉央　(電気通信大学)

okamotoy@uec.ac.jp

2025 年 11 月 11 日

最終更新：2025 年 11 月 12 日　09:48

スケジュール (前半)

1. 高速指数時間アルゴリズムの考え方 (10/7)
＊ 休み (体育祭) (10/14)
2. 分枝アルゴリズム：基礎 (10/21)
3. 分枝アルゴリズム：高速化 (10/28)
4. 分枝アルゴリズム：測度統治法 (11/4)
5. **動的計画法：基礎** (11/11)
6. 動的計画法：例 (11/18)

スケジュール (後半)

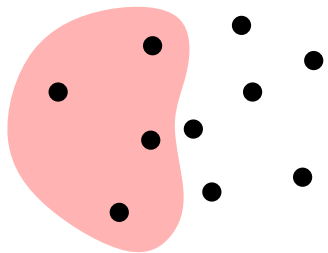
7. 包除原理：原理 (11/25)
* 休み (秋ターム試験) (12/2)
8. 包除原理：例 (12/9)
9. 部分集合たたみ込み：原理 (12/16)
* 休み (出張) (12/23)
* 休み (冬季休業) (12/30)
10. 部分集合たたみ込み：例 (1/6)
11. 指数時間仮説：原理 (1/13)
12. 指数時間仮説：証明 (1/20)
13. 最近の話題 (1/27)
* 休み (修士論文発表会) (2/3)

本日の内容

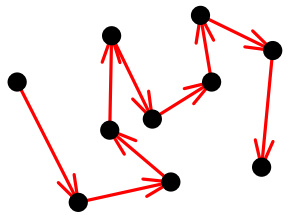
1. **動的計画法と高速指数時間アルゴリズム**
2. 巡回セールスマン問題
3. 最小被覆問題

離散最適化問題の種類

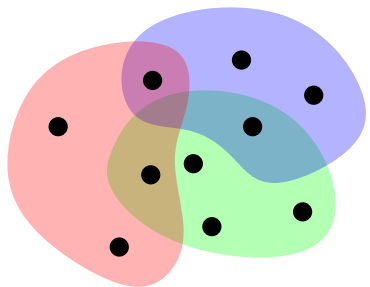
部分集合を見つける



順列を見つける



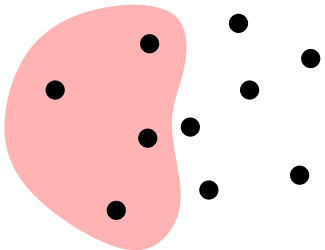
被覆・分割を見つける



離散最適化問題の種類

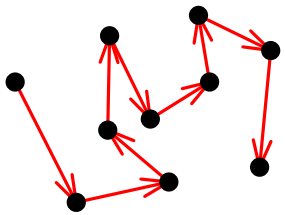
しらみつぶしの計算量

部分集合を見つける



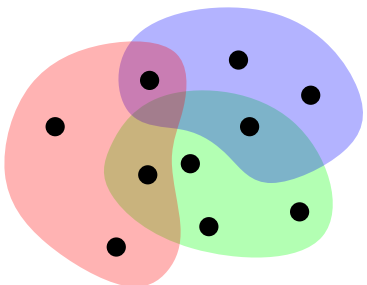
$O^*(2^n)$

順列を見つける



$O^*(n!)$

被覆・分割を見つける



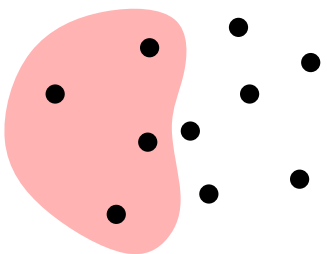
問題設定による

要素数 $= n$

離散最適化問題の種類

しらみつぶしの計算量

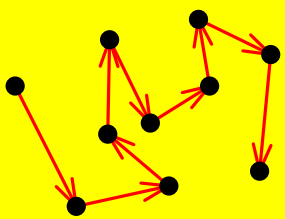
部分集合を見つける



$O^*(2^n)$

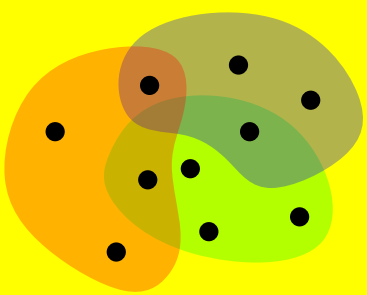
今日の話の主眼

順列を見つける



$O^*(n!)$

被覆・分割を見つける



問題設定による

要素数 $= n$

動的計画法とは

岩波数学辞典第 4 版 (2007) によると

最適化問題の中には，その過程がいくつもの段階から構成される多段決定問題と見なせるものが数多く存在する．**動的計画法** (dynamic programming) は，多段決定問題を体系的に取り扱う研究分野であり，1950 年以降に R. Bellman が発展させた理論・手法である．動的計画法は，離散最適化問題や組合せ問題に対しても，アルゴリズム設計のパラダイムとしてしばしば使われる．

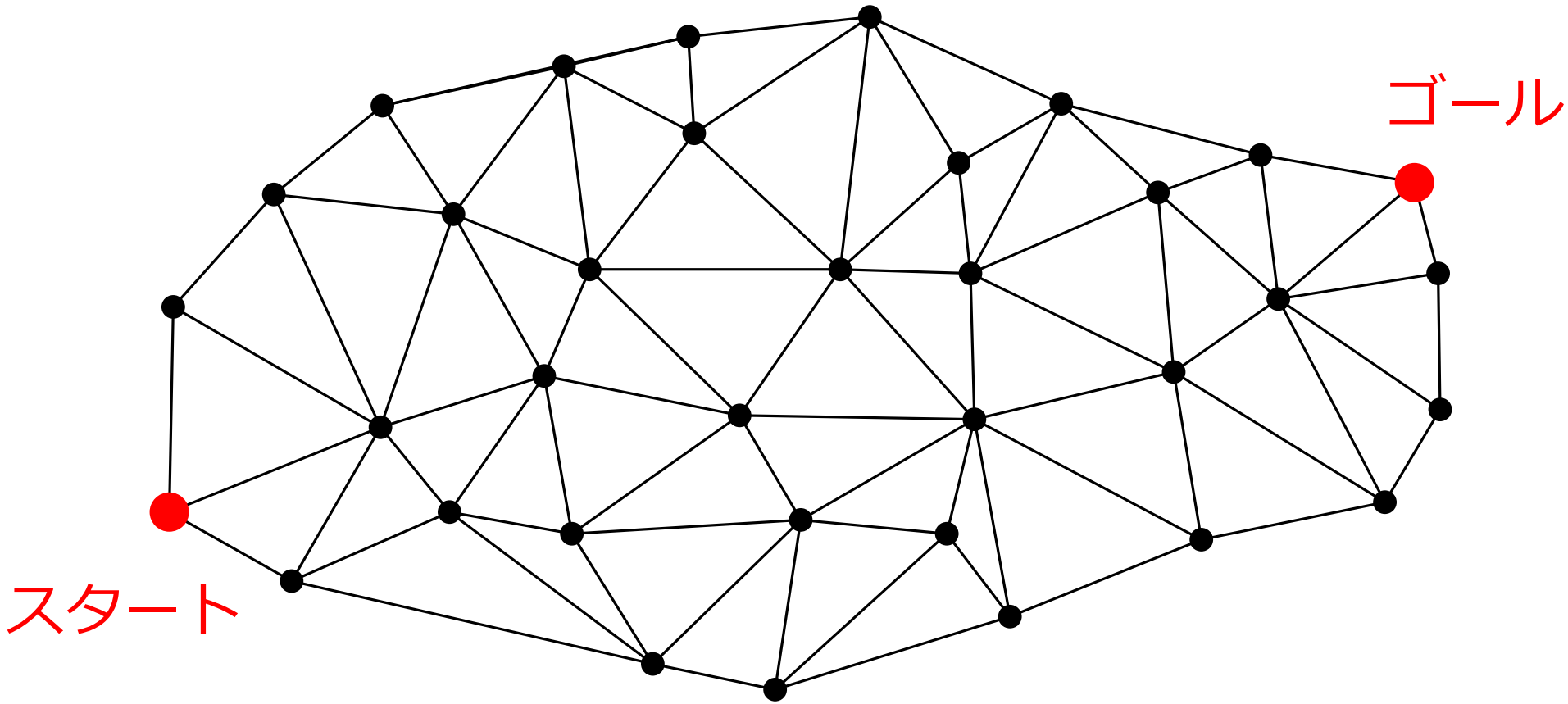
- R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

動的計画法の特徴

応用数理計画ハンドブック (2002) によると

動的計画の特徴は，意思決定が段階ごとになされる点にある．段階は多くの場合，離散化された時間軸を表すことに用いられる．時間軸を考慮したモデルは，しばしば動的 (dynamic) とよばれ，それが “dynamic” programming のよび名の所以である．初期の動的計画は，段階ごとの部分問題の解の情報を「表」にして保存しておいた．もともと “programming” は表による計算法の意味をもち，それが dynamic “programming” のよび名のもう 1 つの所以である．

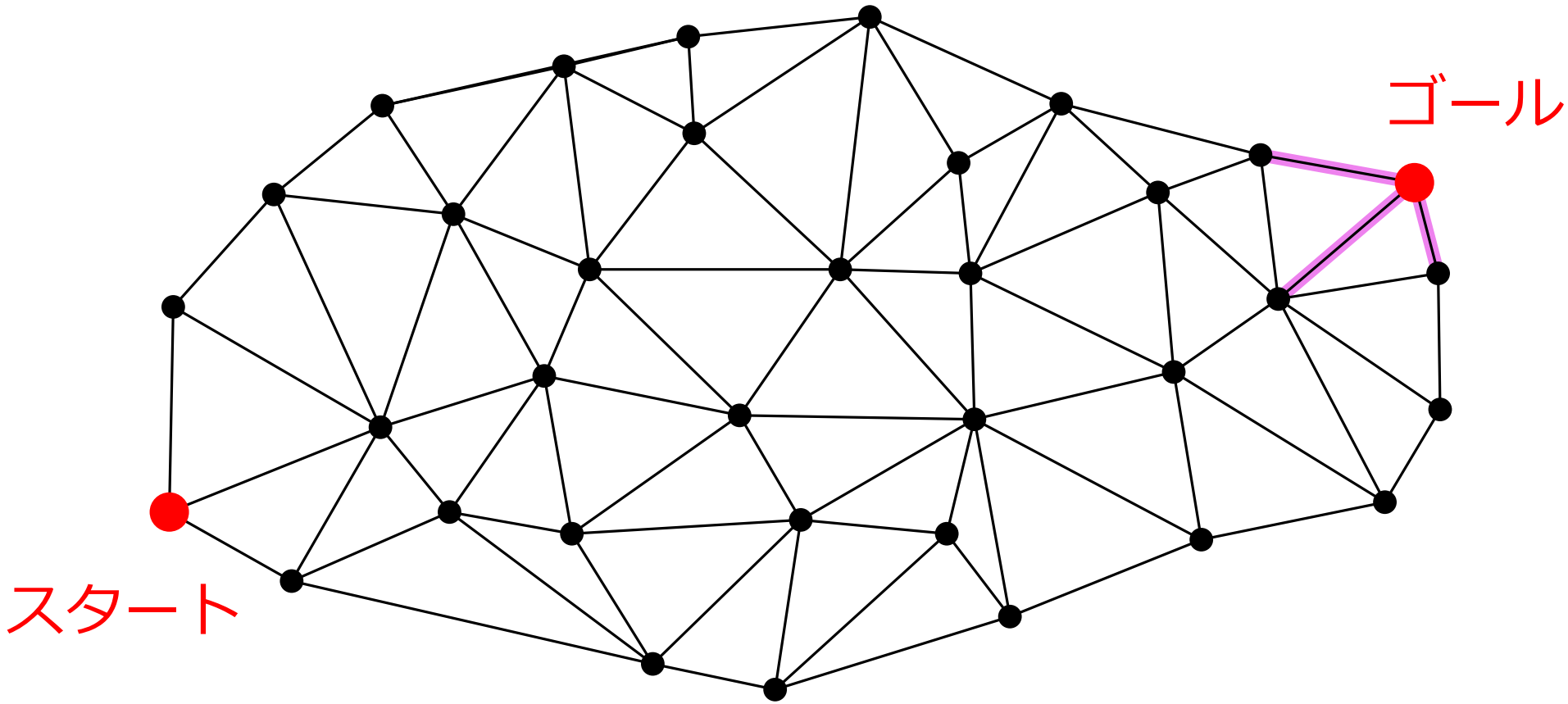
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

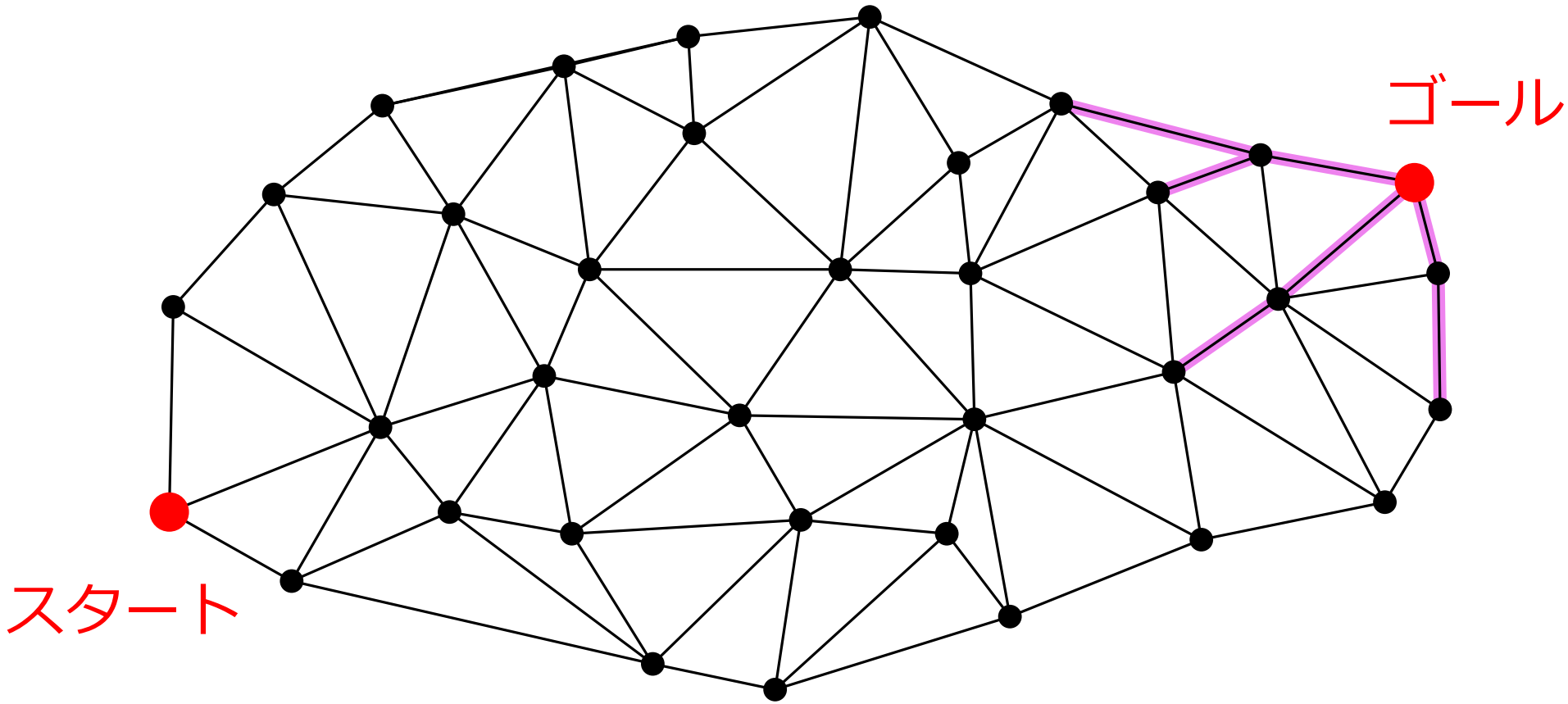
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

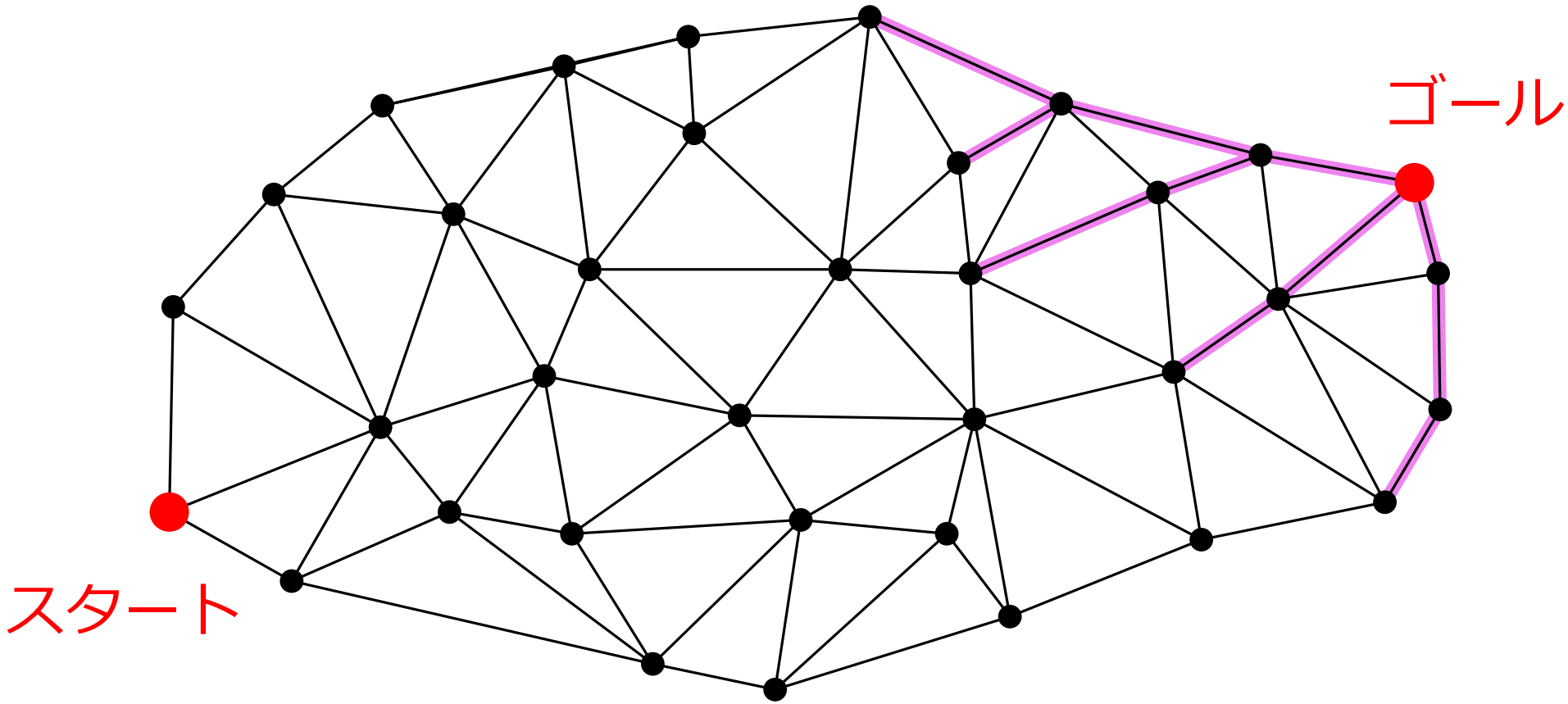
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

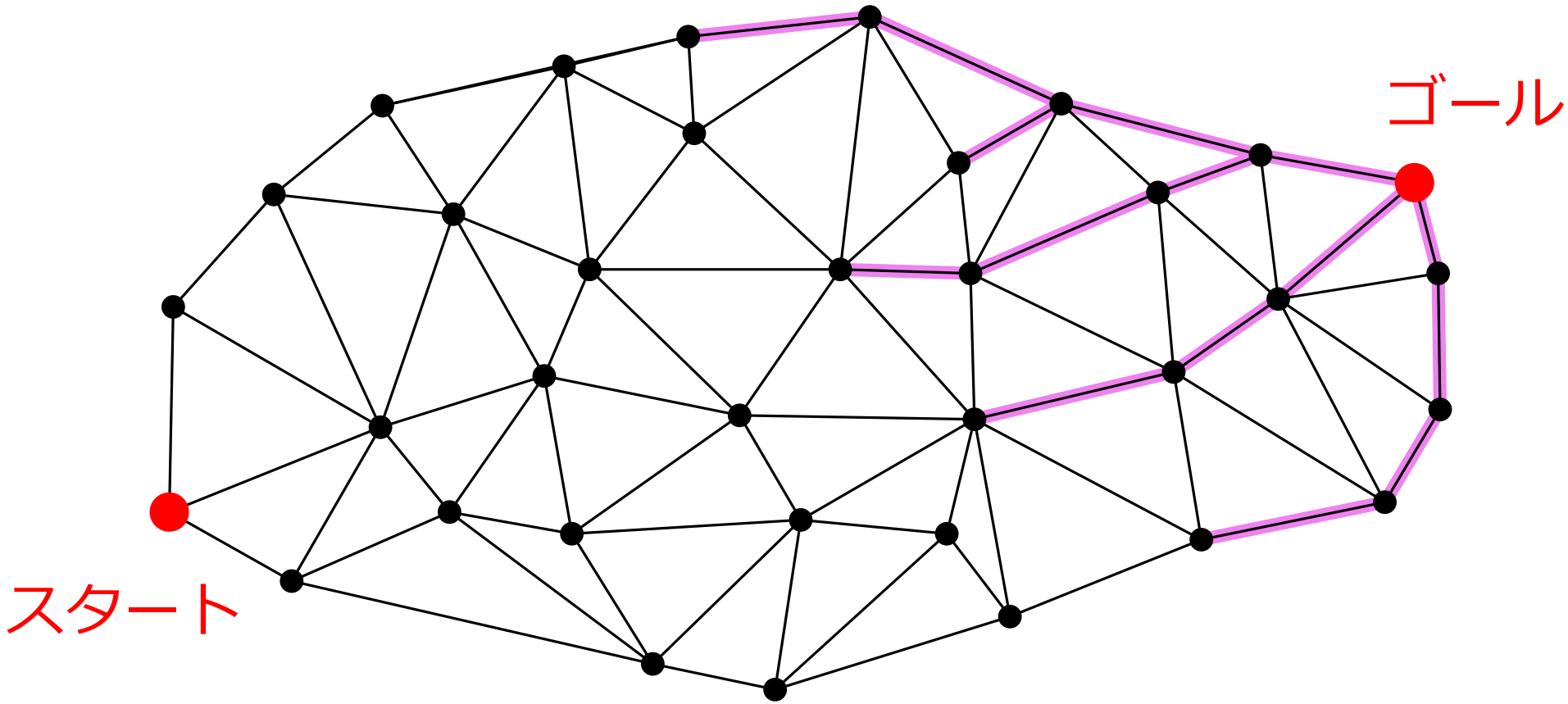
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

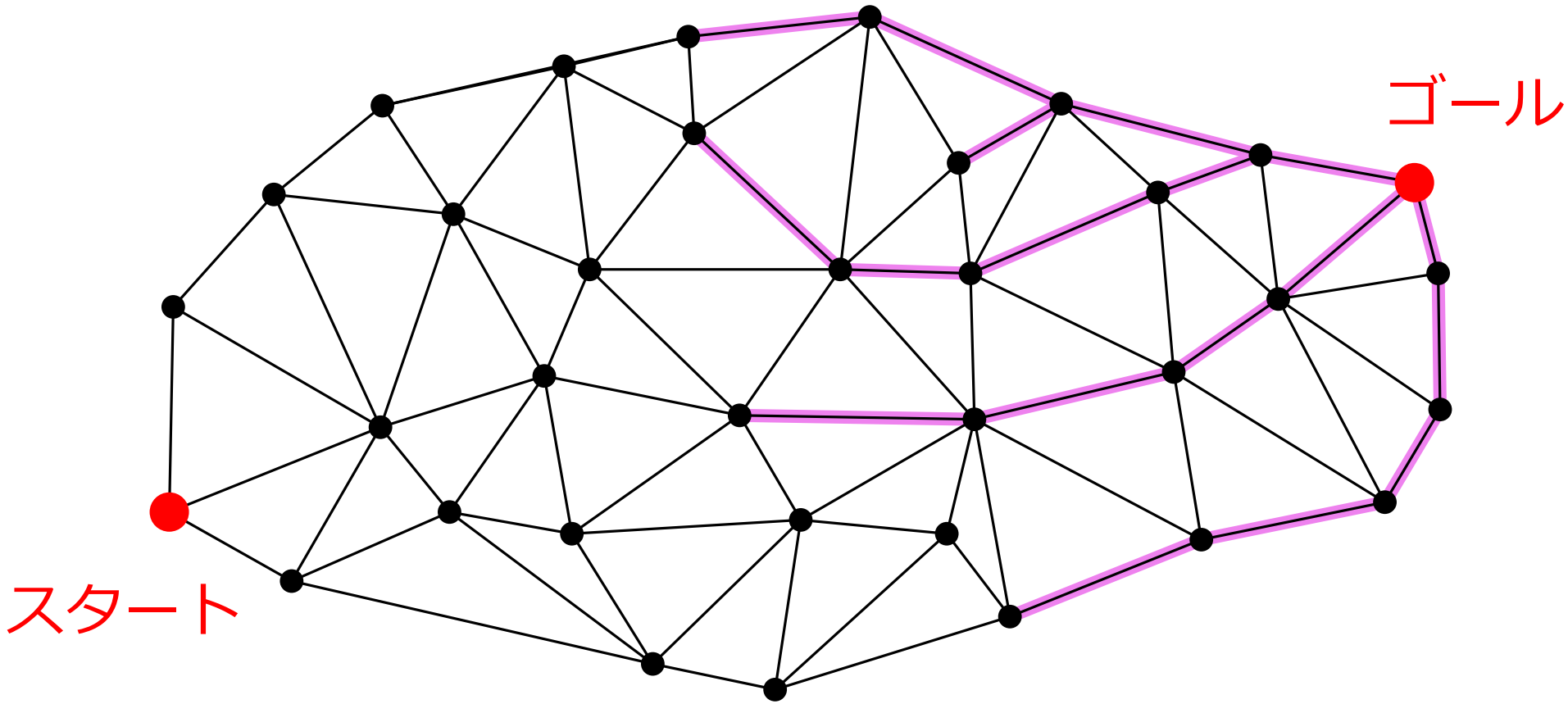
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

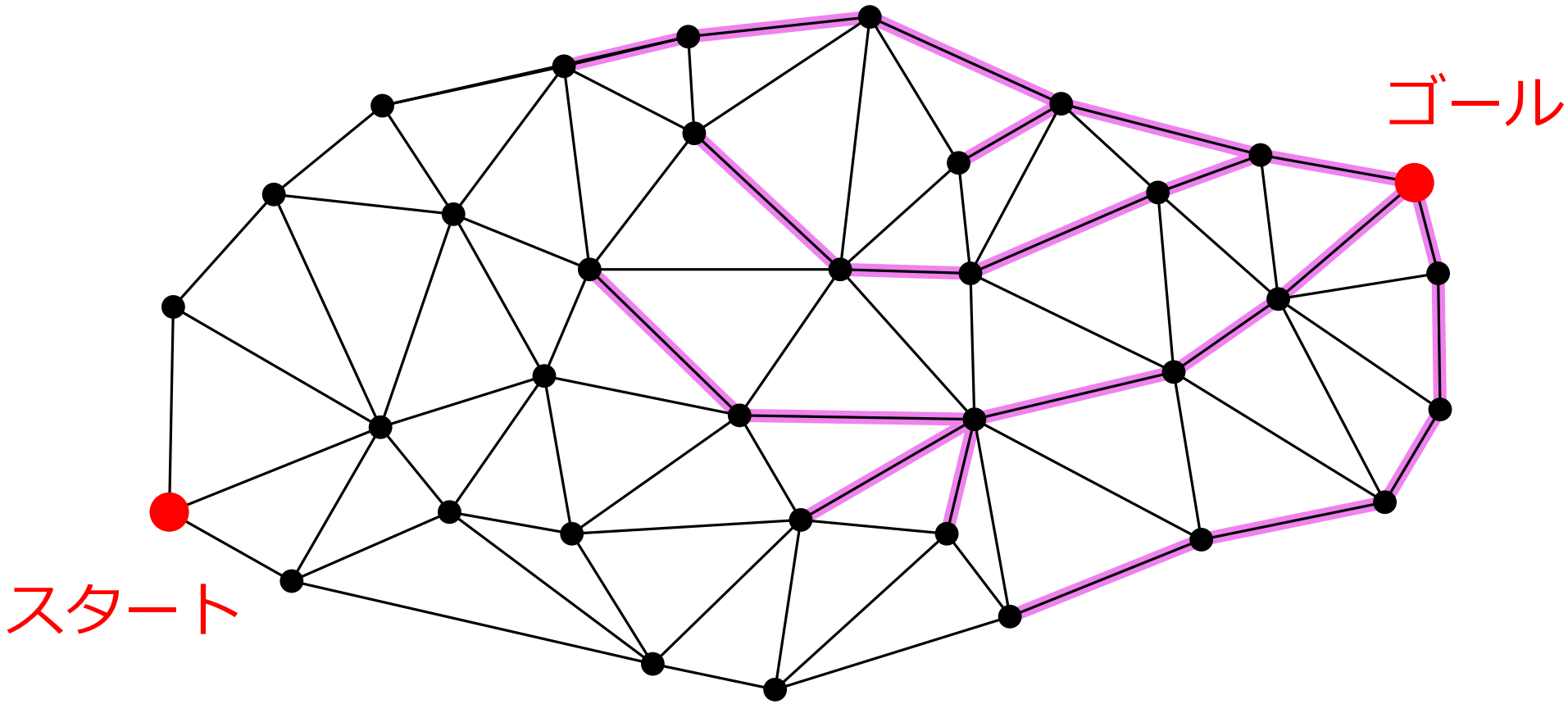
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

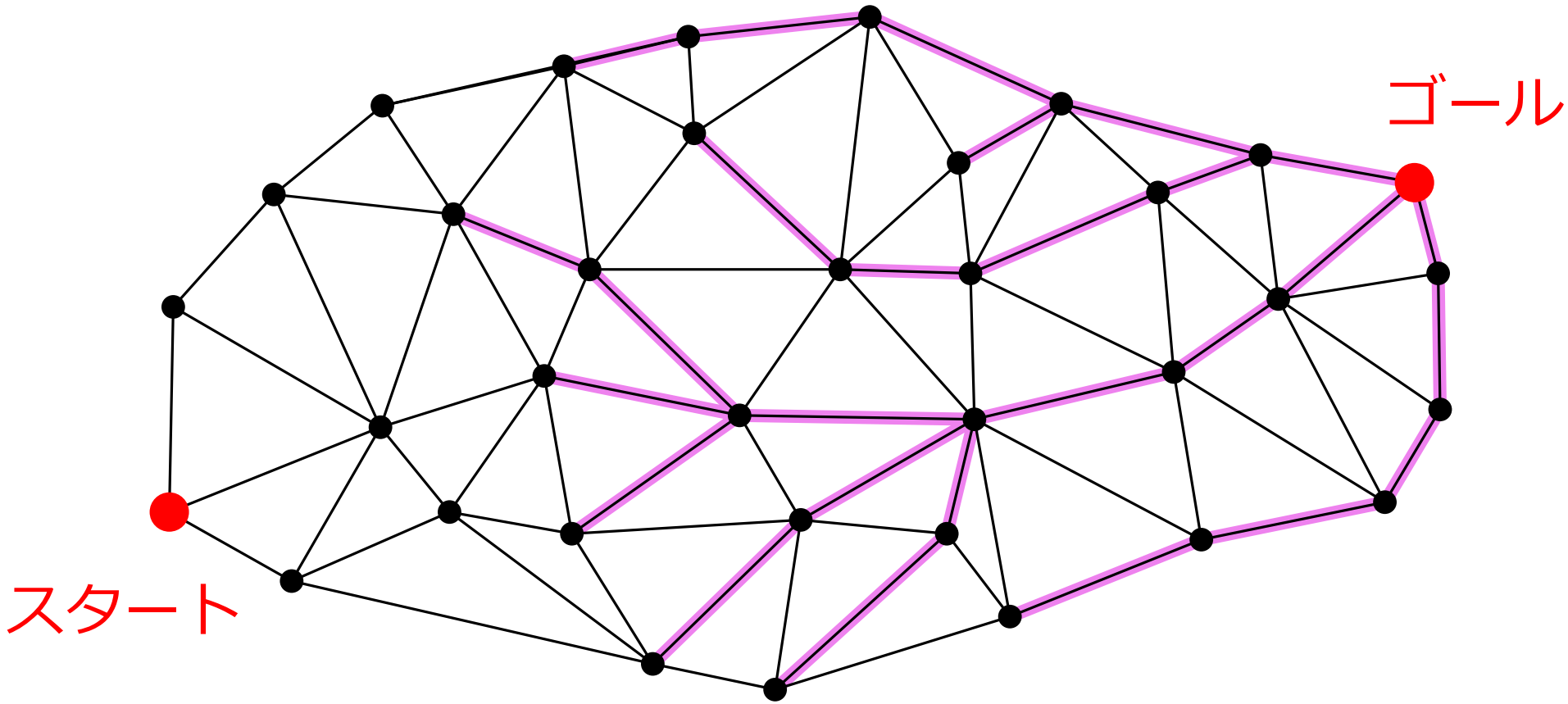
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

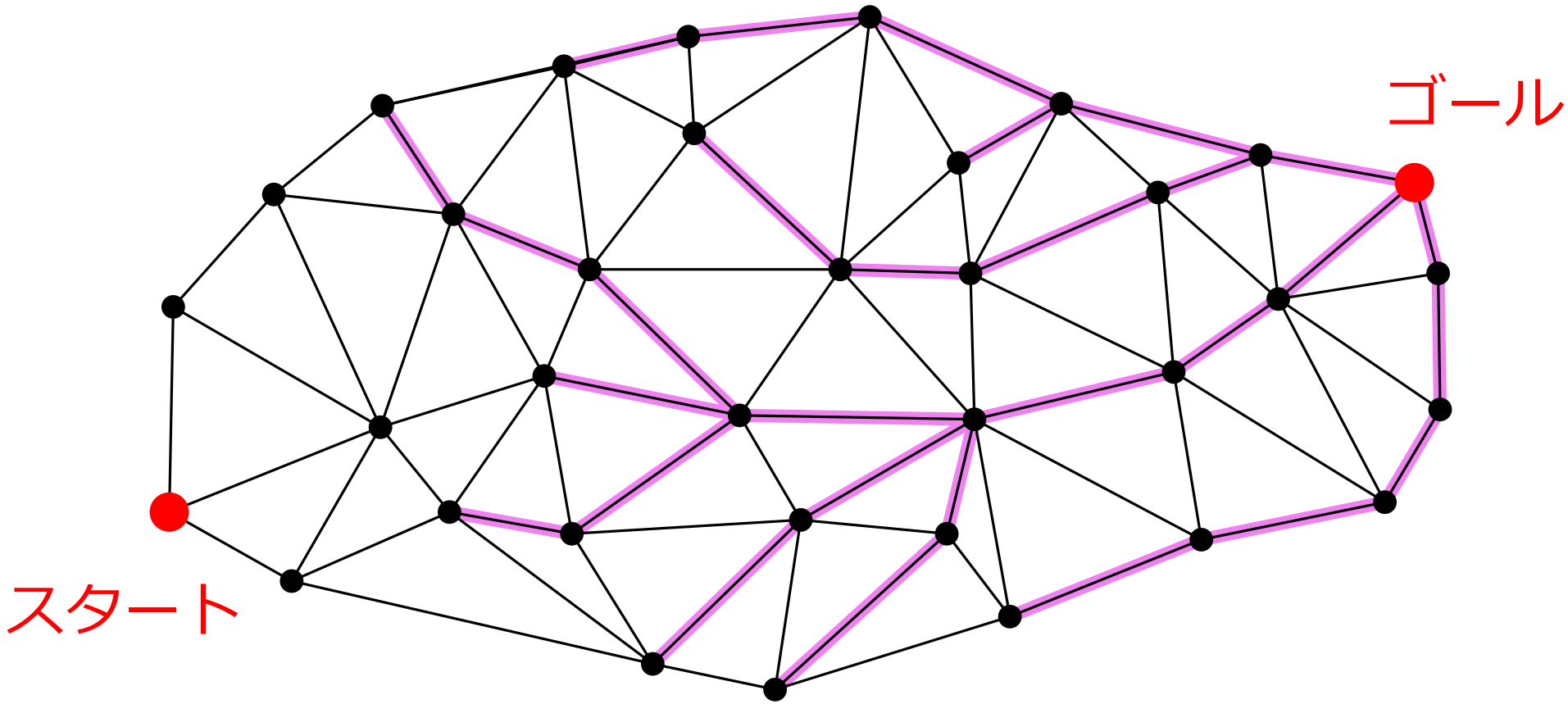
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

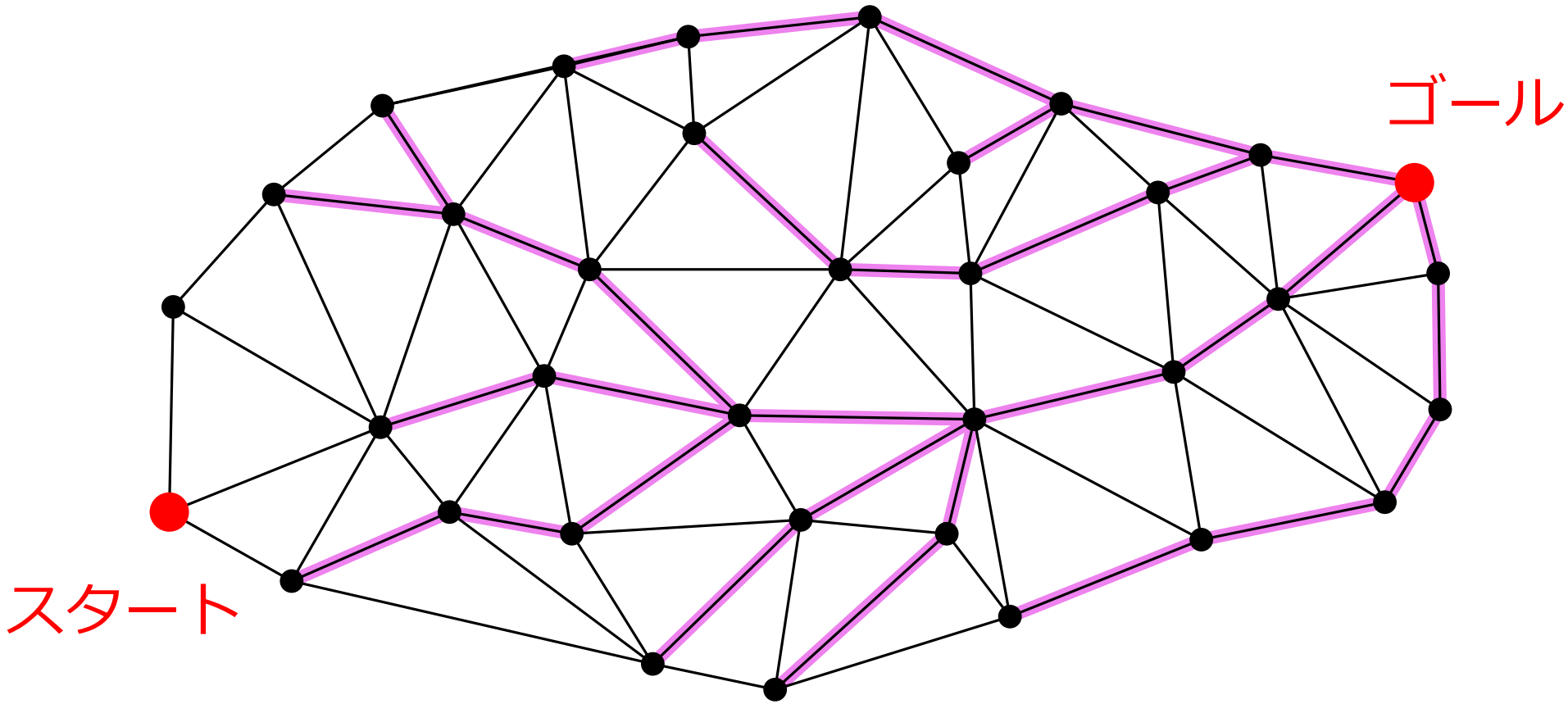
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

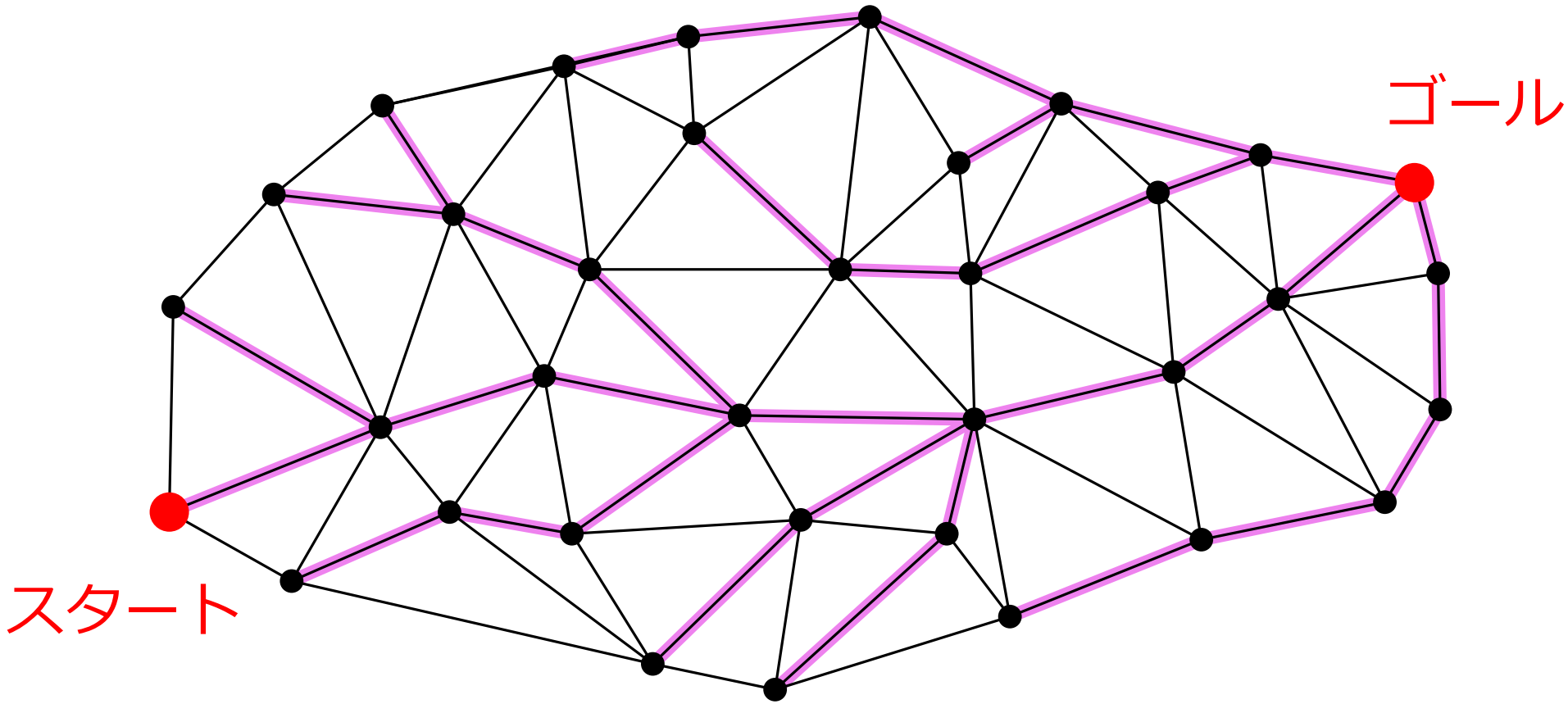
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

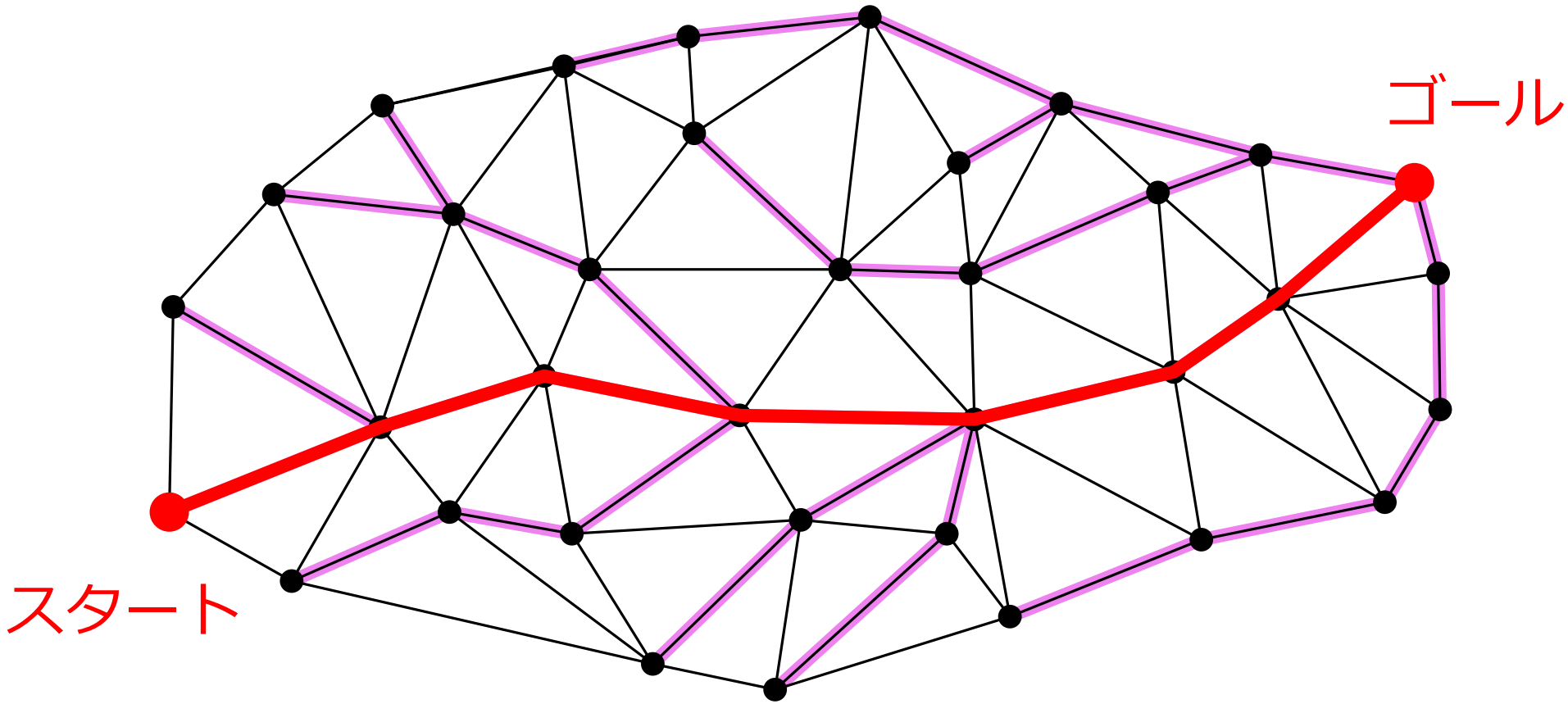
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

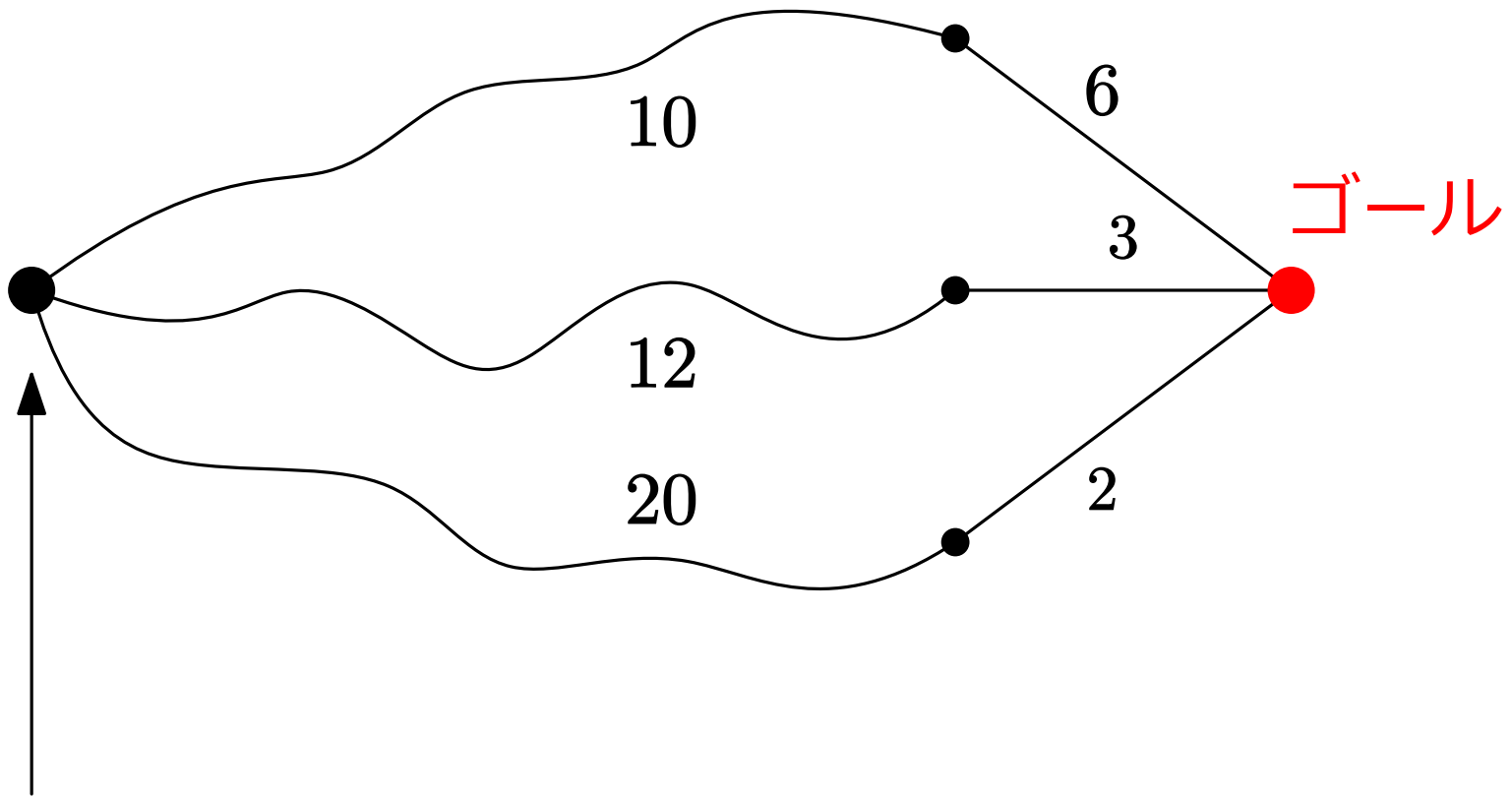
動的計画法の原理：アルゴリズム



最適性の原理 ： 最短経路 の途中の点からの最短経路は
もとの最短経路をたどれば得られる

(principle of optimality)

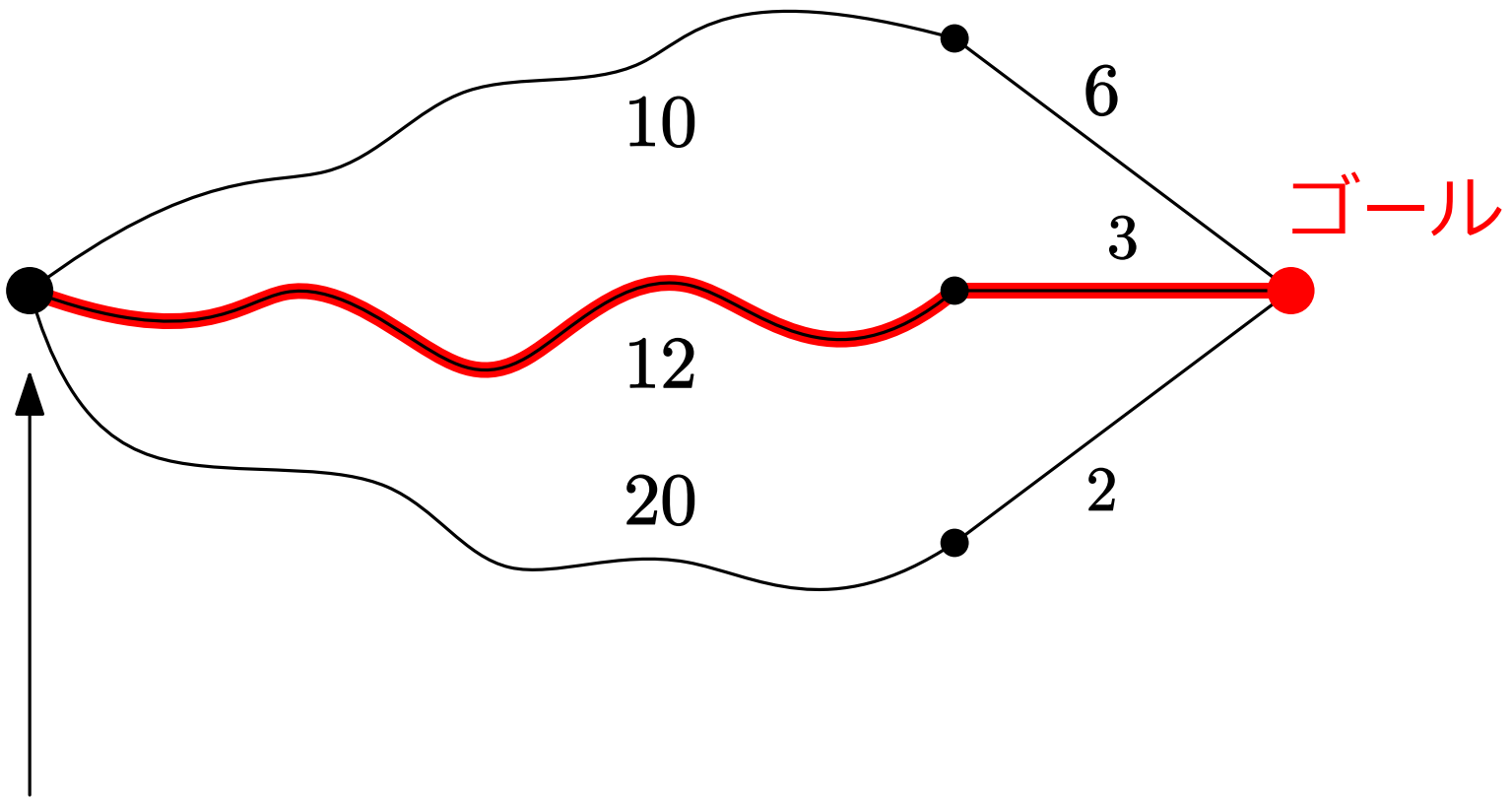
動的計画法の原理：アルゴリズム (続)



ここからゴールへの最短経路の長さ

$= \min\{10+6, 12+3, 20+2\} = \min\{16, 15, 22\} = 15$

動的計画法の原理：アルゴリズム (続)



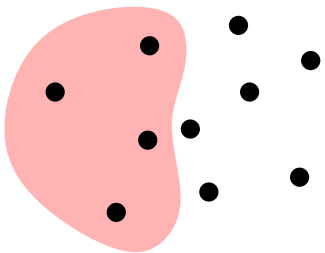
ここからゴールへの最短経路の長さ

$= \min\{10+6, 12+3, 20+2\} = \min\{16, 15, 22\} = 15$

目標

しらみつぶしの計算量

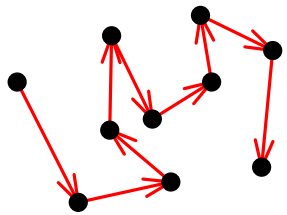
部分集合を見つける



$O^*(2^n)$

動的計画法で

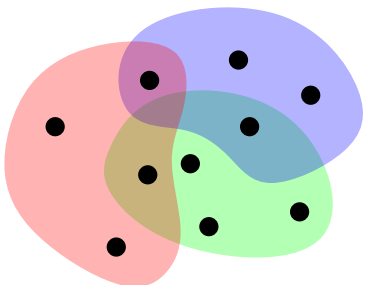
順列を見つける



$O^*(n!)$

$\rightarrow O^*(c^n)$

被覆・分割を見つける



問題設定による

$\rightarrow O^*(c^n)$

要素数 $= n$

c は ある定数

本日の内容

1. 動的計画法と高速指数時間アルゴリズム
2. **巡回セールスマン問題**
3. 最小被覆問題

- R. Bellman, Dynamic programming treatment of the travelling salesman problem, *Journal of the ACM* 9 (1962) pp. 61–63.
- M. Held, R. M. Karp, A dynamic programming approach to sequencing problems, *Journal for the Society for Industrial and Applied Mathematics* 1 (1962) pp. 196–210.

巡回セールスマン問題

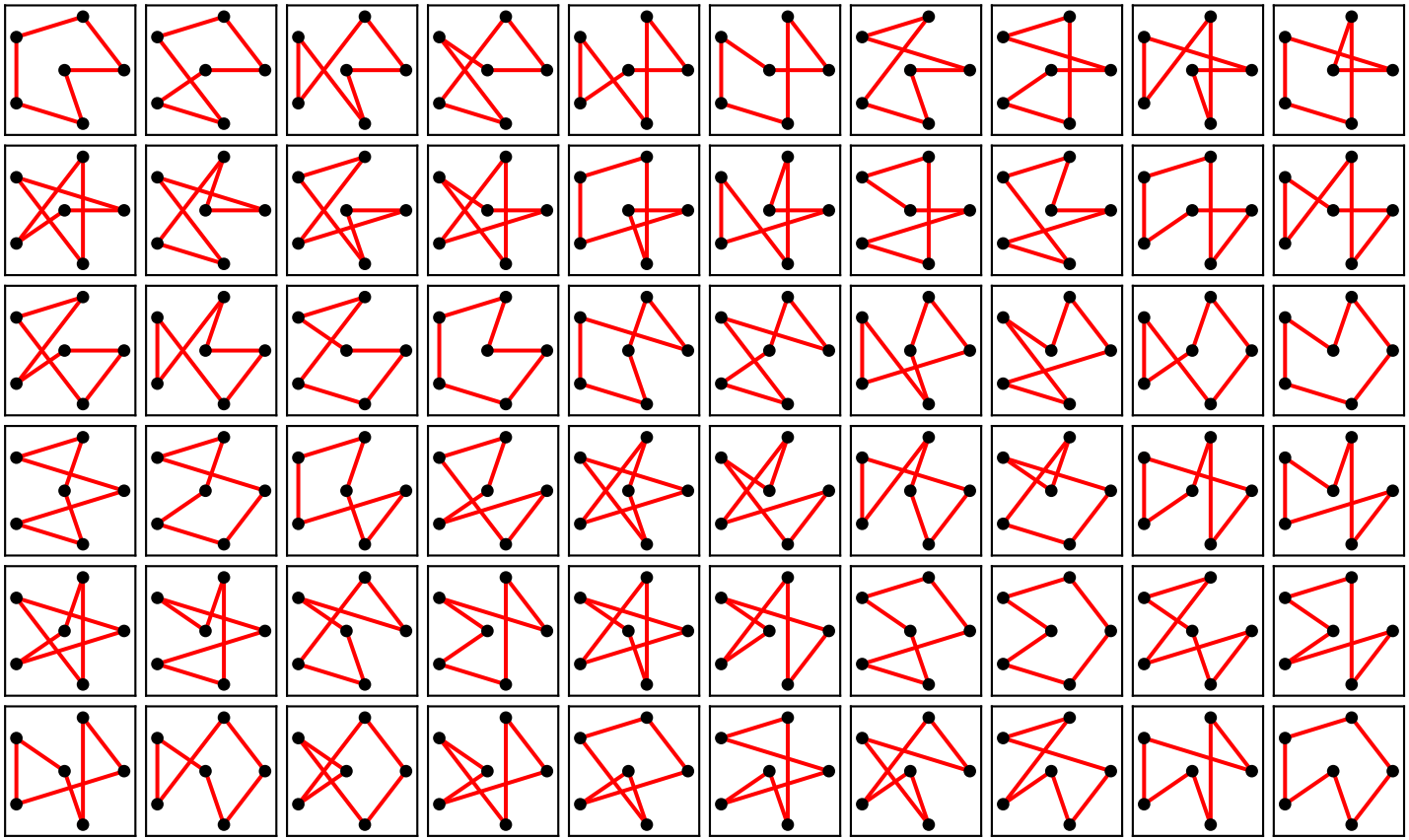
問題：巡回セールスマン問題

入力： 都市の集合 V，都市間の距離行列 $d \in \mathbb{R}^{V \times V}$
出力： V のすべての都市を回って戻る経路の中で，
距離和が最小のもの

入力　　出力

注：巡回セールスマン問題は NP 困難 (Karp '72)

しらみつぶし？

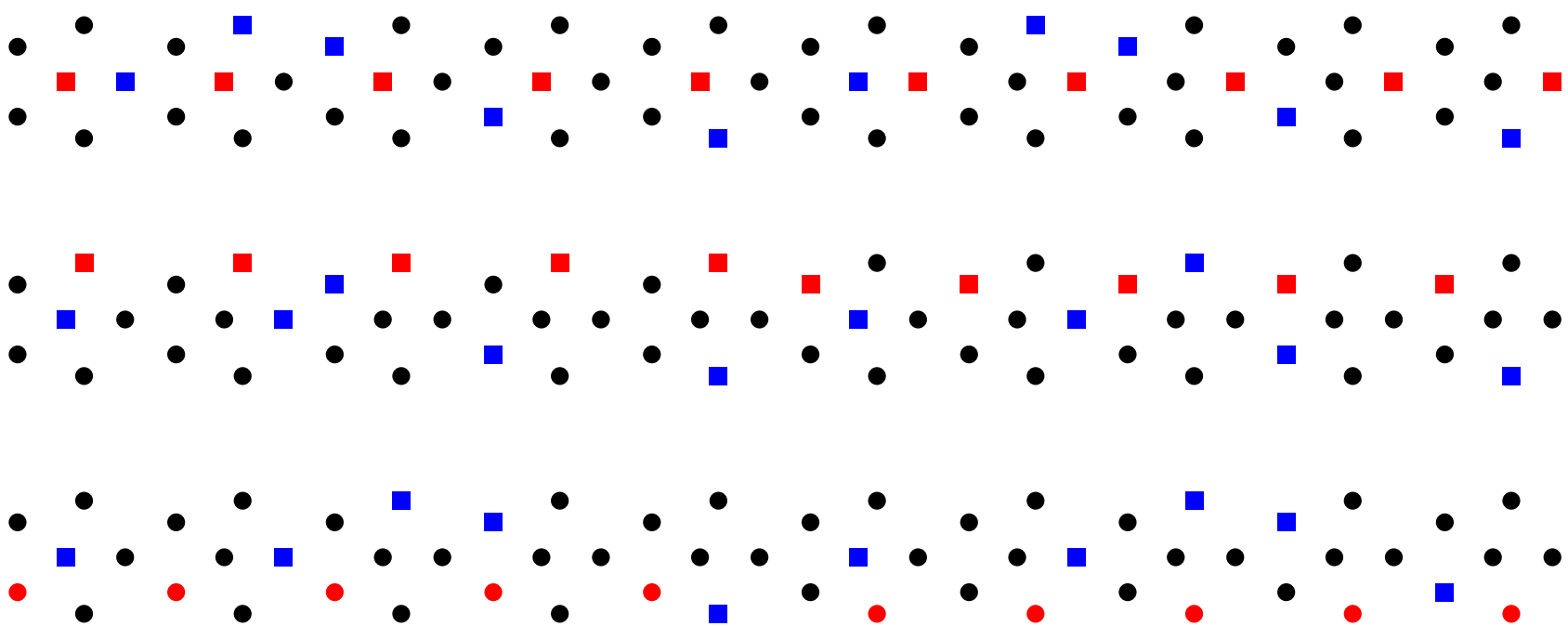


都市数 $n \Rightarrow$ 巡回路数 $= \dfrac{1}{2}(n-1)! = 2^{O(n \log n)}$

補足：スターリングの公式 $n! \approx \sqrt{2\pi n}\left(\dfrac{n}{\mathrm{e}}\right)^n$

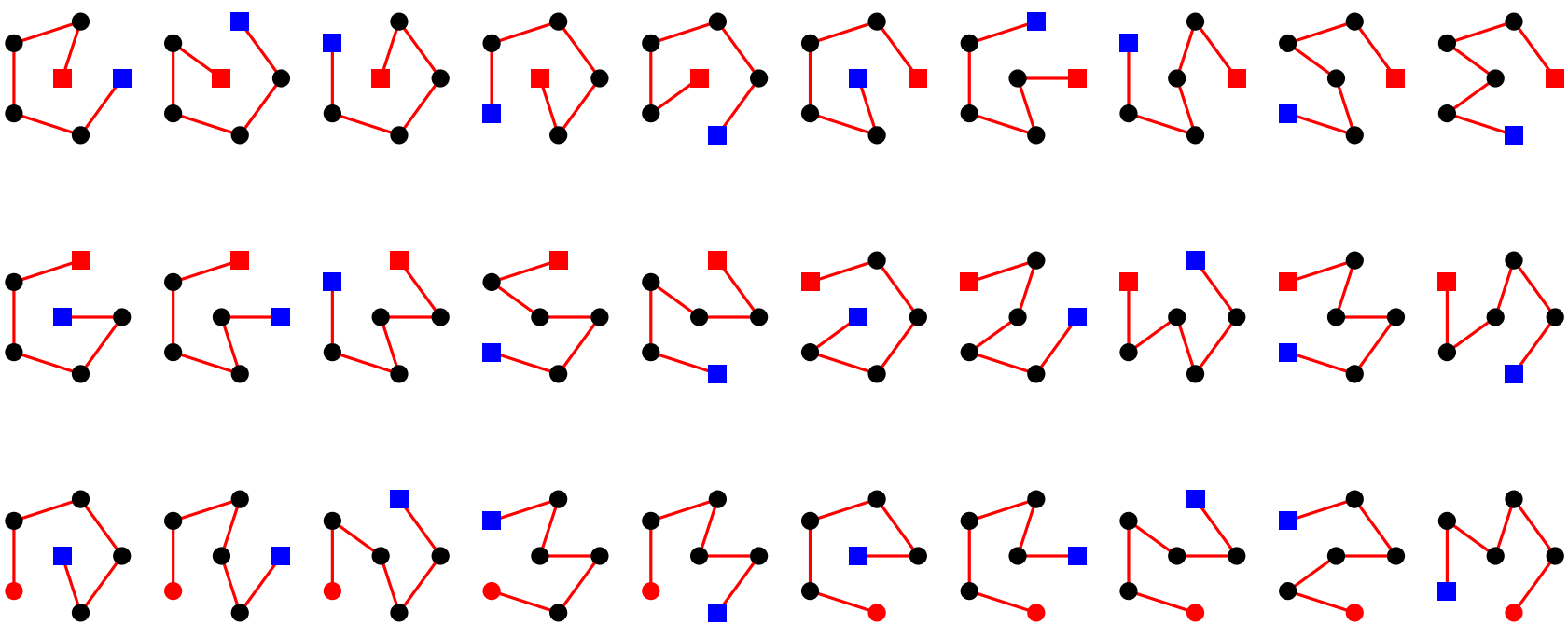
巡回しないことにしてよい

始点と終点を固定して，全頂点を巡る最短経路を考える



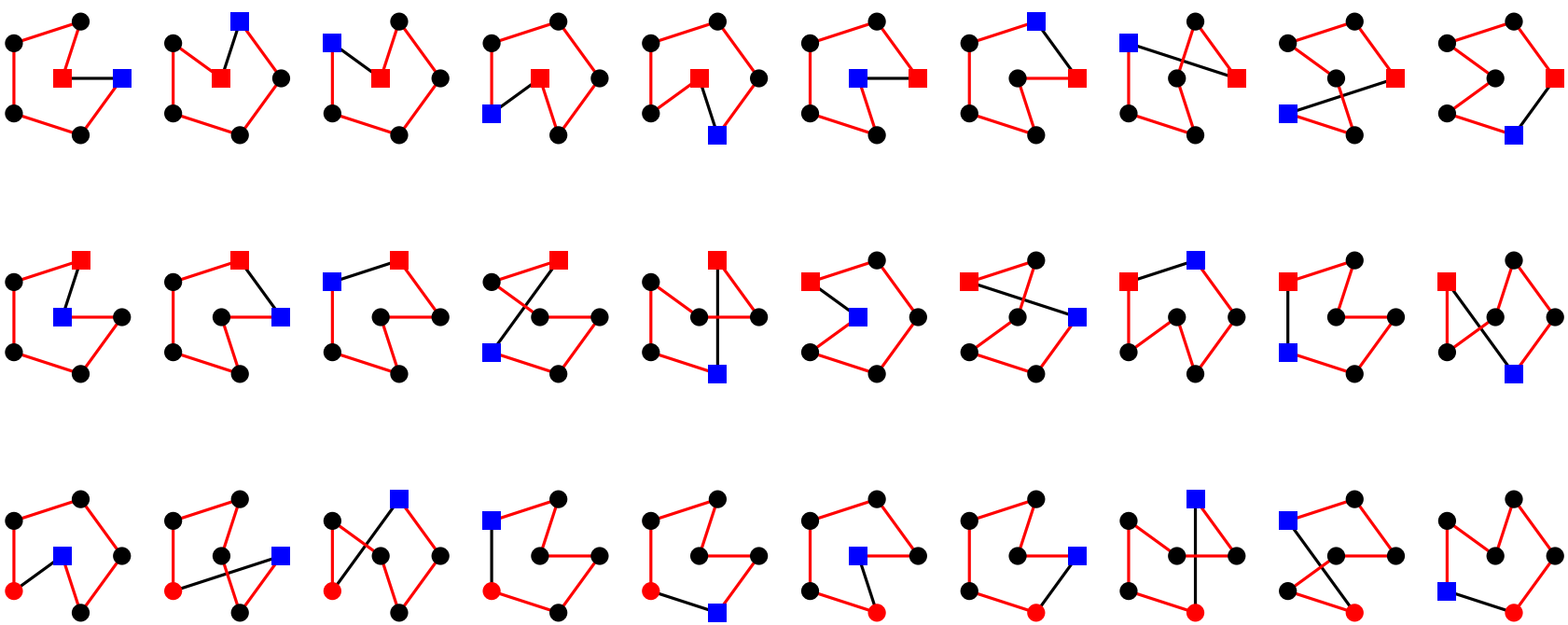
巡回しないことにしてよい

始点と終点を固定して，全頂点を巡る最短経路を考える



巡回しないことにしてよい

始点と終点を固定して，全頂点を巡る最短経路を考える



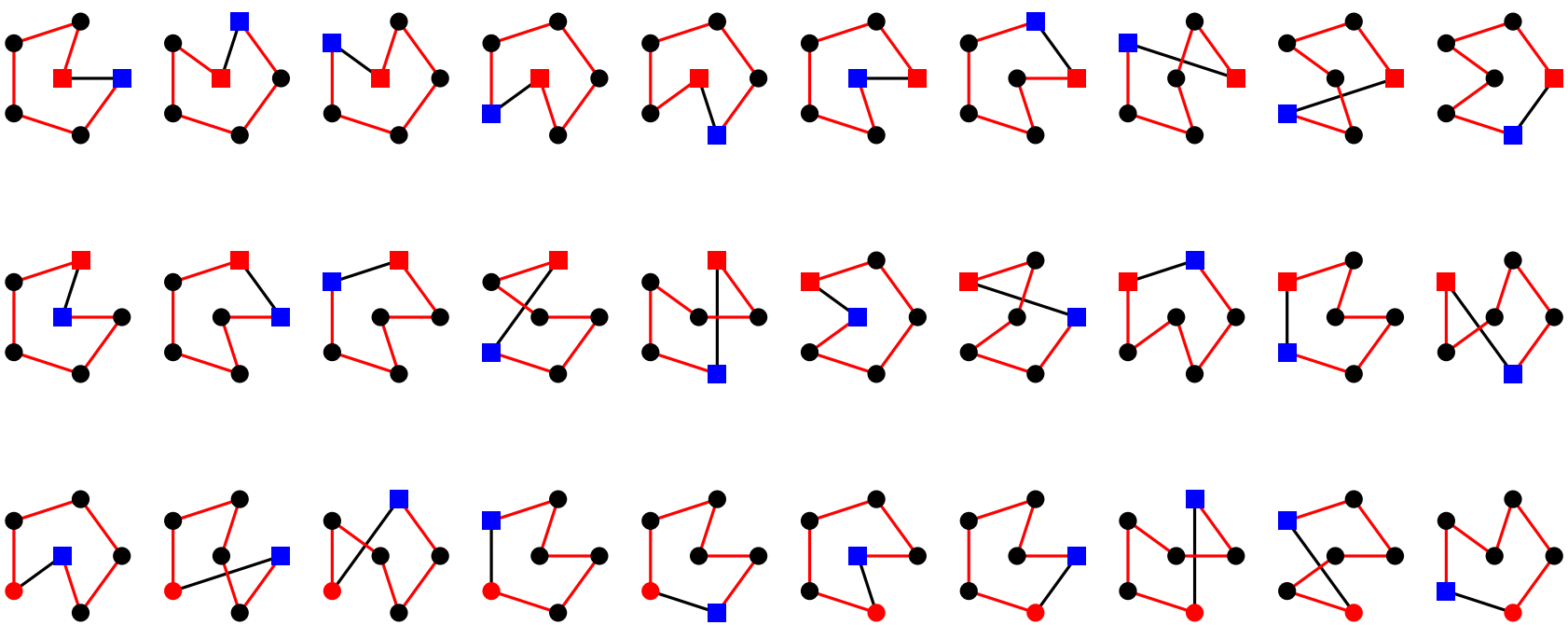
巡回しないことにしてよい

始点と終点を固定して，全頂点を巡る最短経路を考える

それぞれが $O^*(T(n))$ 時間で解ける

$\Rightarrow$ 巡回セールスマン問題が $\underline{O^*(T(n)) \cdot O(n^2)}$ 時間で解ける

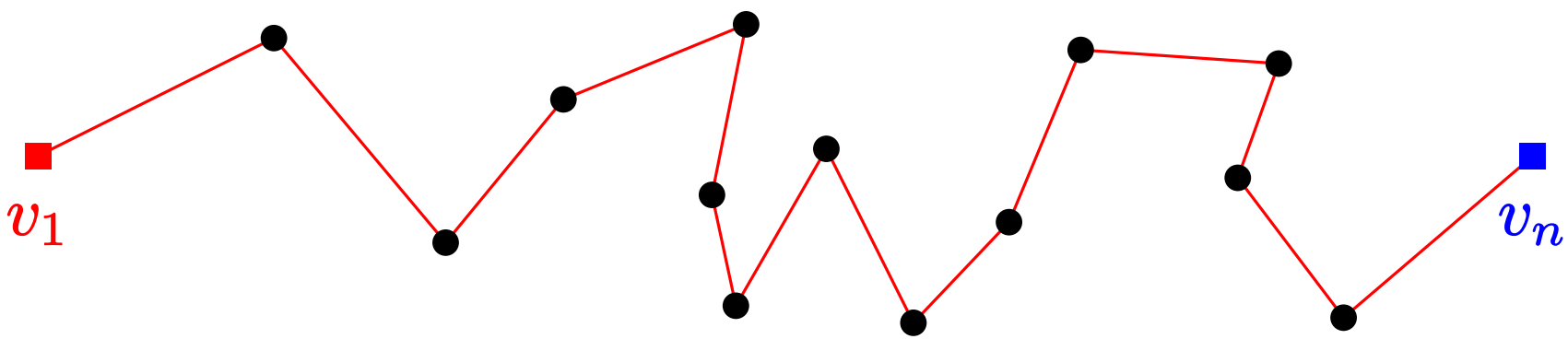
$\underline{= O^*(T(n))}$



動的計画法：考え方

$V = \{v_1, v_2, \ldots, v_n\}$ として，

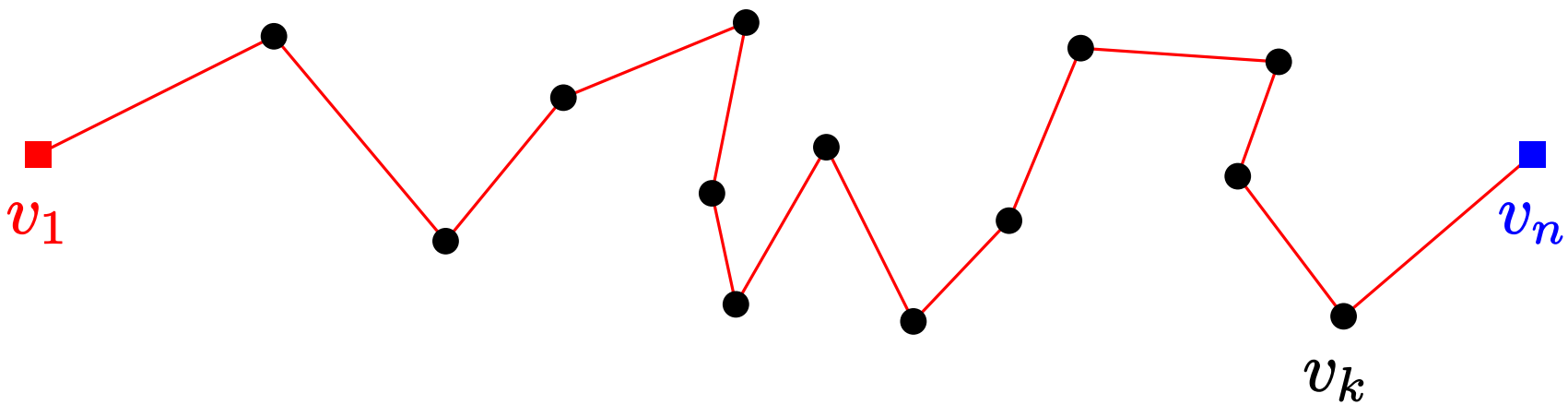
始点が v_1，終点が v_n の最短経路を考える



動的計画法：考え方

$V = \{v_1, v_2, \ldots, v_n\}$ として，

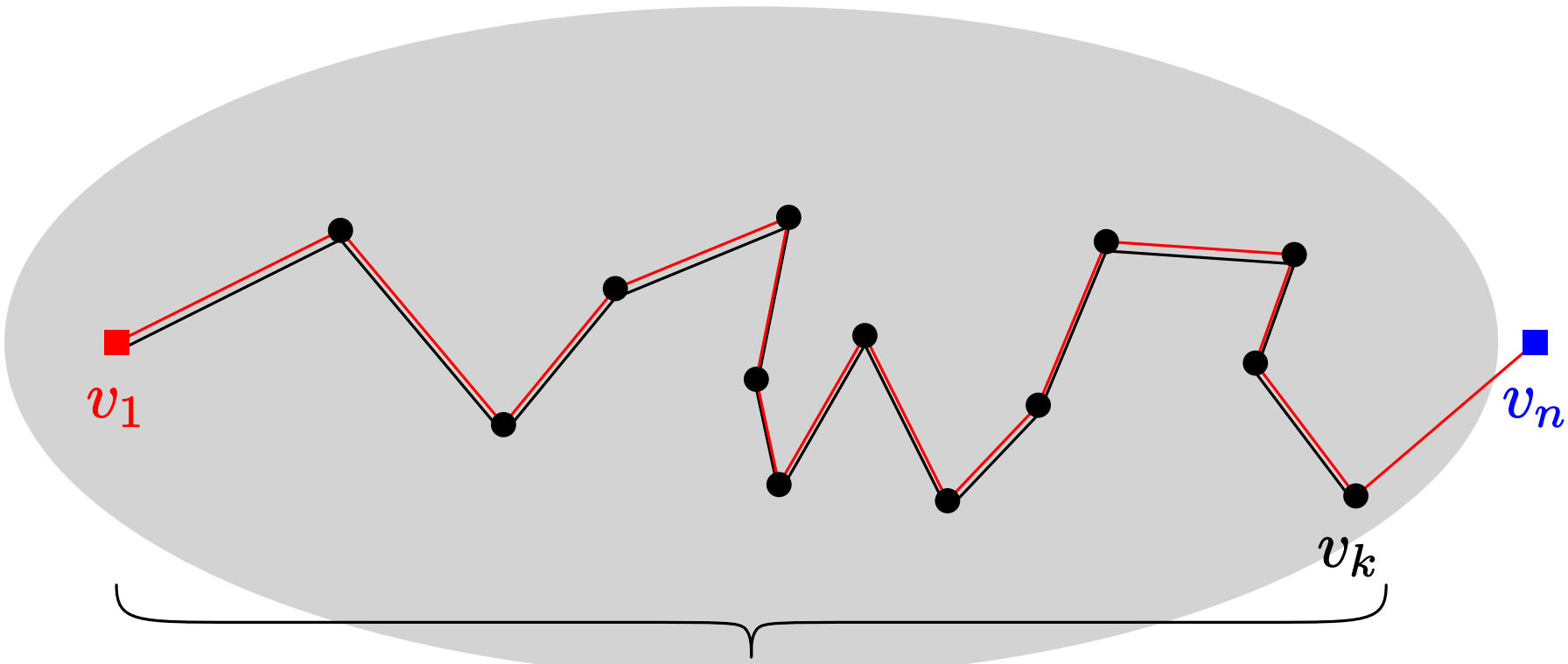
始点が v_1，終点が v_n の最短経路を考える



動的計画法：考え方

$V = \{v_1, v_2, \ldots, v_n\}$ として，

始点が v_1，終点が v_n の最短経路を考える



始点が v_1，終点が v_k の最短経路

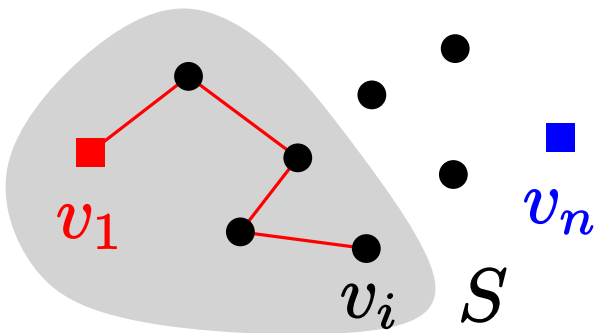
ただし，v_n は通らない (v_n 以外は全部通る)

動的計画法：状態と状態の値

状態 (i, S)　ただし，$i \in \{2, 3, \ldots, n\}, \{v_1, v_i\} \subseteq S \subseteq V$

状態の値 $f(i, S) =$ 始点を v_1，終点を v_i として

S の都市のみをすべて通る最短経路の長さ



動的計画法を考えるときの鍵

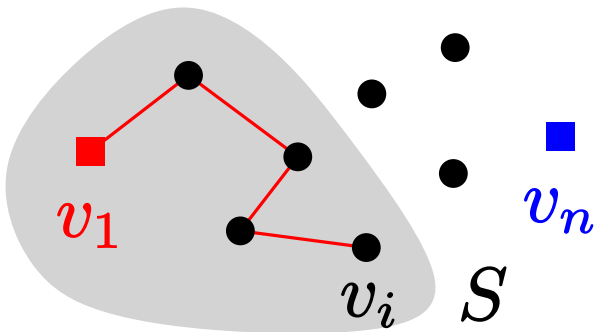
1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：状態と状態の値

状態 (i, S)　ただし，$i \in \{2, 3, \ldots, n\}, \{v_1, v_i\} \subseteq S \subseteq V$

状態の値 $f(i, S) =$ 始点を v_1，終点を v_i として

S の都市のみをすべて通る最短経路の長さ



最終的に出力する値　$f(n, V)$

動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

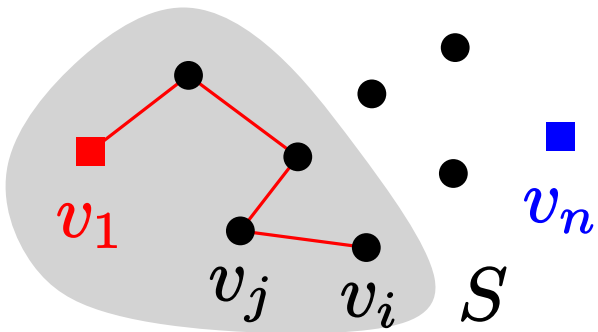
動的計画法：再帰式 (Bellman 方程式)

再帰式

$$f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_j, v_i) \mid v_j \in S - \{v_i\}\}$$

$|S| \geq 3$ のとき

$$f(i, \{v_1, v_i\}) = d(v_1, v_i)$$



動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

Bellman 方程式を探索木で解く

アルゴリズム tsp-td(V, d)

1. $f(n, V)$ を出力

アルゴリズム $f(i, S)$

1. $|S| = 2$ のとき
   $d(v_1, v_i)$ を出力
2. $|S| \geq 3$ のとき，
   すべての $v_j \in S - \{v_i\}$ に対して
   $f(j, S - \{v_i\}) + d(v_i, v_j)$ を計算し
   その中の最小値を出力

Bellman 方程式を探索木で解く

アルゴリズム tsp-td(V, d)

1. $f(n, V)$ を出力

アルゴリズム $f(i, S)$

1. $|S| = 2$ のとき
   $d(v_1, v_i)$ を出力
2. $|S| \geq 3$ のとき，
   すべての $v_j \in S - \{v_i\}$ に対して
   $f(j, S - \{v_i\}) + d(v_i, v_j)$ を計算し
   その中の最小値を出力

計算量：$T(s) \leq (s-1)T(s-1)$　($s = |S|$ とする)

$\rightsquigarrow T(s) \leq (s-1)!$　(しらみつぶしとほぼ同じ)

動的計画法：例

ポイント Bellman 方程式をボトムアップに解く

$$f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_j, v_i) \mid v_j \in S - \{v_i\}\}$$

	v_1	v_2	v_3	v_4
v_1	0	1	2	3
v_2	1	0	2	3
v_3	2	2	0	1
v_4	3	3	1	0

S	$\{v_1, v_2, v_3, v_4\}$
i	2 3 4

$\{v_1, v_2, v_3\}$	$\{v_1, v_2, v_4\}$	$\{v_1, v_3, v_4\}$
2 3	2 4	3 4

$\{v_1, v_2\}$	$\{v_1, v_3\}$	$\{v_1, v_4\}$
2	3	4

動的計画法：例

ポイント Bellman 方程式をボトムアップに解く

$$f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_j, v_i) \mid v_j \in S - \{v_i\}\}$$

	v_1	v_2	v_3	v_4
v_1	0	1	2	3
v_2	1	0	2	3
v_3	2	2	0	1
v_4	3	3	1	0

$\{v_1, v_2, v_3, v_4\}$
2 3 4

$\{v_1, v_2, v_3\}$
2 3

$\{v_1, v_2, v_4\}$
2 4

$\{v_1, v_3, v_4\}$
3 4

$\{v_1, v_2\}$
2
1

$\{v_1, v_3\}$
3
2

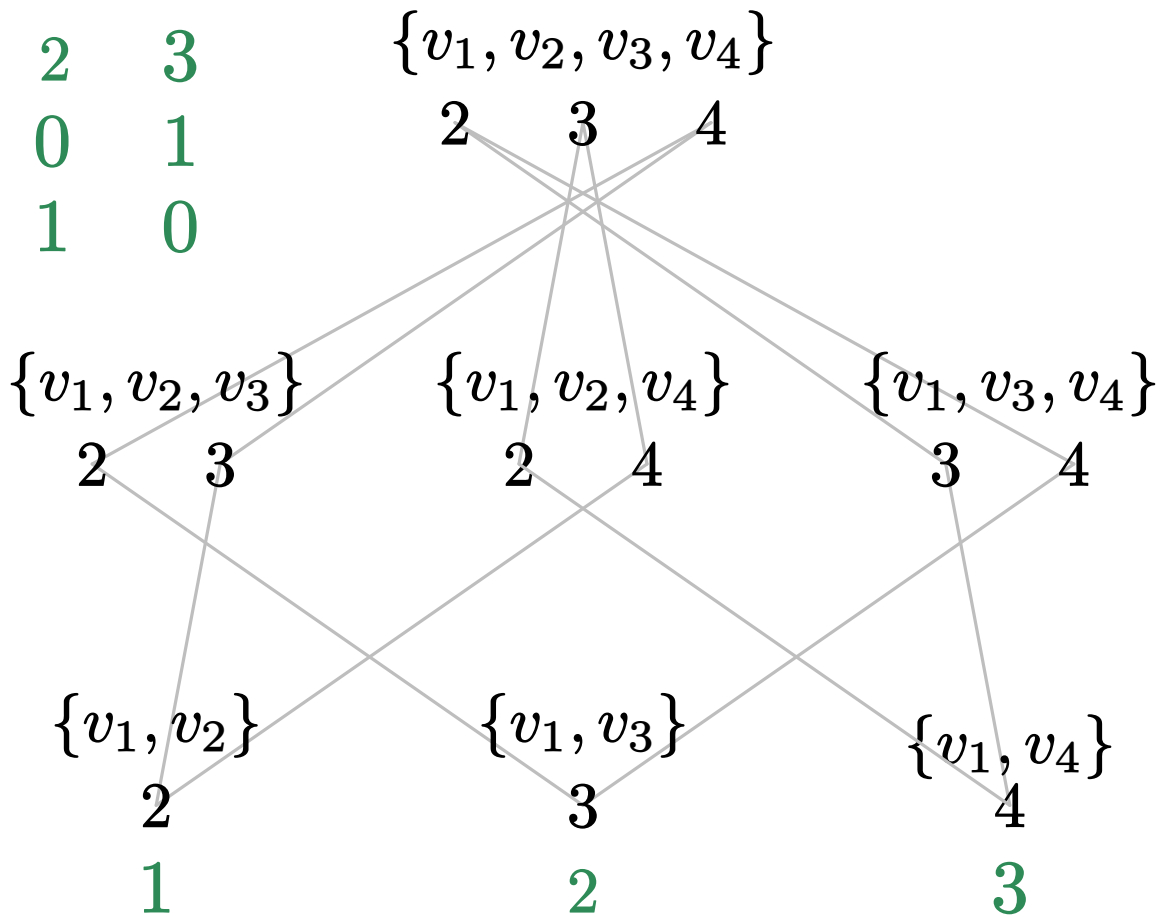
$\{v_1, v_4\}$
4
3

動的計画法：例

ポイント Bellman 方程式をボトムアップに解く

$$f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_j, v_i) \mid v_j \in S - \{v_i\}\}$$

	v_1	v_2	v_3	v_4
v_1	0	1	2	3
v_2	1	0	2	3
v_3	2	2	0	1
v_4	3	3	1	0

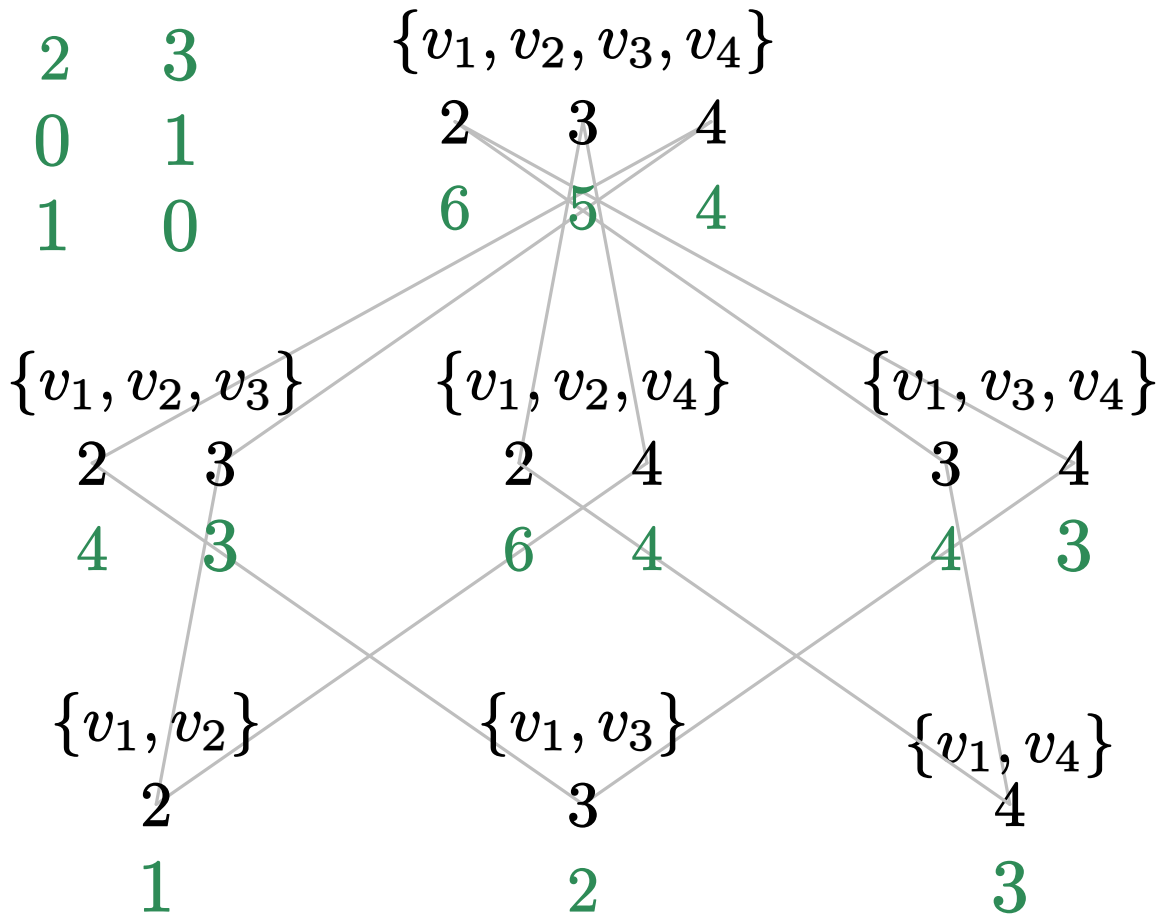


動的計画法：例

ポイント Bellman 方程式をボトムアップに解く

$$f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_j, v_i) \mid v_j \in S - \{v_i\}\}$$

	v_1	v_2	v_3	v_4
v_1	0	1	2	3
v_2	1	0	2	3
v_3	2	2	0	1
v_4	3	3	1	0

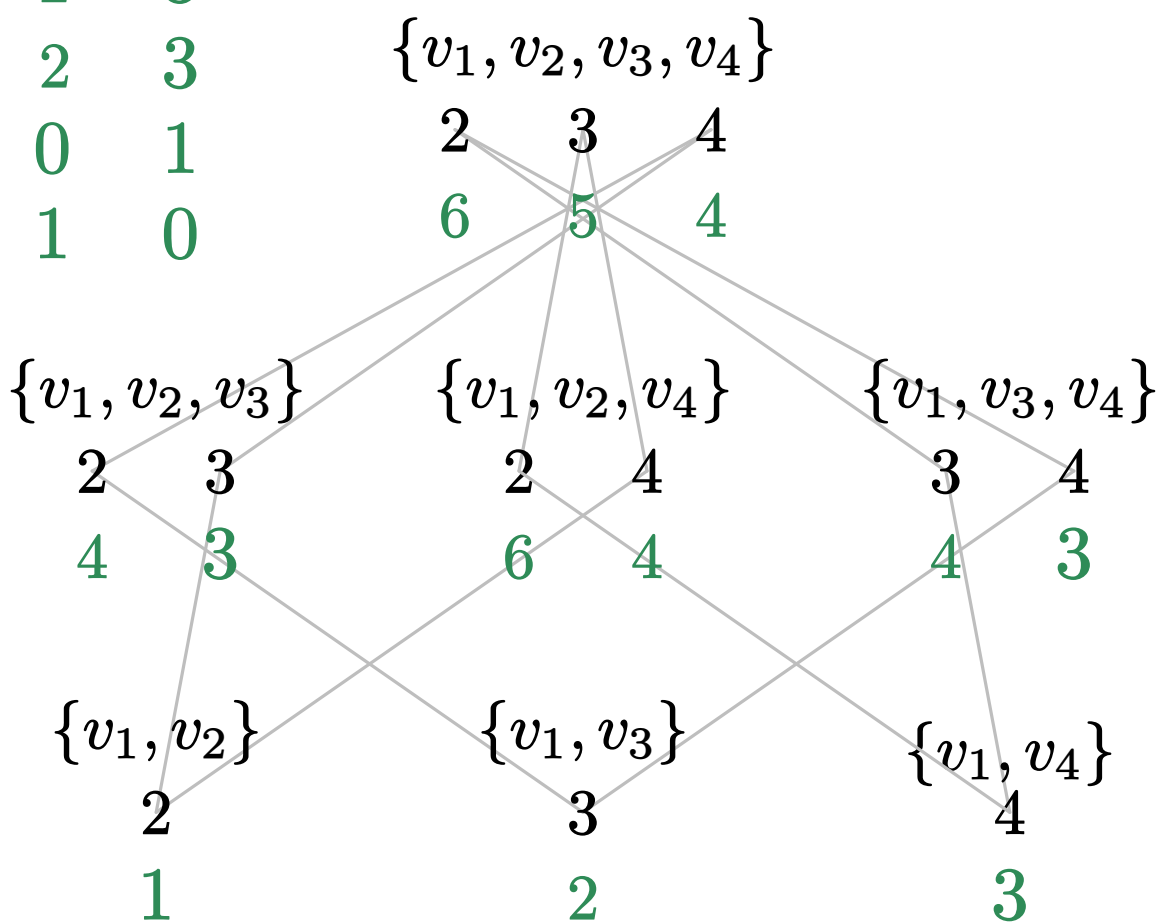


動的計画法：例

ポイント Bellman 方程式をボトムアップに解く

$$f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_j, v_i) \mid v_j \in S - \{v_i\}\}$$

	v_1	v_2	v_3	v_4
v_1	0	1	2	3
v_2	1	0	2	3
v_3	2	2	0	1
v_4	3	3	1	0



状態の総数

$$\binom{4-1}{3} \cdot 3 + \binom{4-1}{2} \cdot 2 + \binom{4-1}{1} \cdot 1$$

動的計画法：アルゴリズム

アルゴリズム tsp-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S)
2. $|S| = 2$ を満たすすべての状態 (i, S) に対して
   $f(i, S) = d(v_1, v_i)$
3. $|S| \geq 3$ を満たすすべての状態 (i, S) に対して
   $|S|$ の小さい方から順に
   $f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_i, v_j) \mid v_j \in S - \{v_i\}\}$
4. $f(n, V)$ を出力

動的計画法：アルゴリズム

アルゴリズム tsp-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S)
2. $|S| = 2$ を満たすすべての状態 (i, S) に対して
   $f(i, S) = d(v_1, v_i)$
3. $|S| \geq 3$ を満たすすべての状態 (i, S) に対して
   $|S|$ の小さい方から順に
   $f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_i, v_j) \mid v_j \in S - \{v_i\}\}$
4. $f(n, V)$ を出力

状態の総数 $= \displaystyle\sum_{k=1}^{n-1} \binom{n-1}{k} \cdot k \leq \sum_{k=0}^{n-1} \binom{n-1}{k} \cdot n = 2^{n-1} n$

状態 (i, S)　ただし，$i \in \{2, 3, \ldots, n\}, \{v_1, v_i\} \subseteq S \subseteq V$

動的計画法：アルゴリズム

アルゴリズム tsp-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S)
2. $|S| = 2$ を満たすすべての状態 (i, S) に対して
   $f(i, S) = d(v_1, v_i)$
3. $|S| \geq 3$ を満たすすべての状態 (i, S) に対して
   $|S|$ の小さい方か
   $f(i, S) = \min\{f($
4. $f(n, V)$ を出力

二項定理：$(a+b)^m = \sum_{k=0}^{m} \binom{m}{k} a^k b^{m-k}$

状態の総数 $= \sum_{k=1}^{n-1} \binom{n-1}{k} \cdot k \leq \sum_{k=0}^{n-1} \binom{n-1}{k} \cdot n = 2^{n-1} n$

状態 (i, S)　ただし，$i \in \{2, 3, \ldots, n\}, \{v_1, v_i\} \subseteq S \subseteq V$

動的計画法：計算量の導出

アルゴリズム tsp-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S)
2. $|S| = 2$ を満たすすべての状態 (i, S) に対して
   $f(i, S) = d(v_1, v_i)$
3. $|S| \geq 3$ を満たすすべての状態 (i, S) に対して
   $|S|$ の小さい方から順に
   $f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_i, v_j) \mid v_j \in S - \{v_i\}\}$
4. $f(n, V)$ を出力

動的計画法：計算量の導出

アルゴリズム tsp-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S) $O^*(2^n)$
2. $|S| = 2$ を満たすすべての状態 (i, S) に対して
   $f(i, S) = d(v_1, v_i)$ $O^*(1)$
3. $|S| \geq 3$ を満たすすべての状態 (i, S) に対して
   $|S|$ の小さい方から順に
   $f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_i, v_j) \mid v_j \in S - \{v_i\}\}$
4. $f(n, V)$ を出力

動的計画法：計算量の導出

アルゴリズム tsp-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S) $O^*(2^n)$
2. $|S| = 2$ を満たすすべての状態 (i, S) に対して
   $f(i, S) = d(v_1, v_i)$ $O^*(1)$
3. $|S| \geq 3$ を満たすすべての状態 (i, S) に対して
   $|S|$ の小さい方から順に
   $f(i, S) = \min\{f(j, S - \{v_i\}) + d(v_i, v_j) \mid v_j \in S - \{v_i\}\}$
4. $f(n, V)$ を出力 $2(|S| - 1)$ 個の値を参照

$$\sum_{k=2}^{n-1} \binom{n-1}{k} \cdot 2k \leq 2^{n-1} \cdot 2n = O^*(2^n)$$

巡回セールスマン問題：まとめ

結論：巡回セールスマン問題 (Bellman ’62; Held, Karp ’62)

巡回セールスマン問題は $O^*(2^n)$ 時間で解ける
(n は都市数)

このように
「すべての部分集合を状態とする」ような動的計画法を
部分集合縦断動的計画法 と呼ぶことがある
(dynamic programming across the subsets)

(Woeginger ’03)

巡回セールスマン問題：補足

	計算時間	領域
しらみつぶし	$O^*((n-1)!)$	$O^*(1)$
Bellman 方程式を トップダウンに解く (探索木)		
Bellman 方程式を ボトムアップに解く (動的計画法)	$O^*(2^n)$	$O^*(2^n)$

(n：都市数)

未解決問題

次のような定数 $c < 2$ が存在するか？

巡回セールスマン問題は $O^*(c^n)$ 時間で解ける

本日の内容

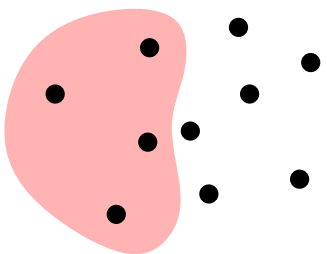
1. 動的計画法と高速指数時間アルゴリズム
2. 巡回セールスマン問題
3. **最小被覆問題**

- F. V. Fomin, D. Kratsch, G. J. Woeginger, Exact (exponential) algorithms for the dominating set problem. Proceedings of WG 2004, *Lecture Notes in Computer Science* 3353 (2004) pp. 245–256.
- M. Cygan, H. Dell, D. Lokshtanov, D. Marx, J. Nederlof, Y. Okamoto, R. Paturi, S. Saurabh, M. Wahlström, On problems as hard as CNF-SAT. ACM Transactions on Algorithms 12 (2016) pp. 41:1–41:24.
- A. Björklund, P. Kaski, The asymptotic rank conjecture and the set cover conjecture are not both true, Proceedings of STOC 2024 (2024) pp. 859–870.

離散最適化問題の種類

しらみつぶしの計算量

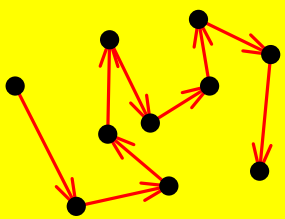
部分集合を見つける



$O^*(2^n)$

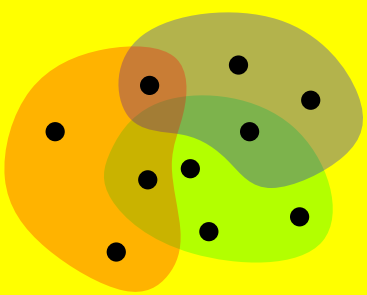
今日の話の主眼

順列を見つける



$O^*(n!)$

被覆・分割を見つける

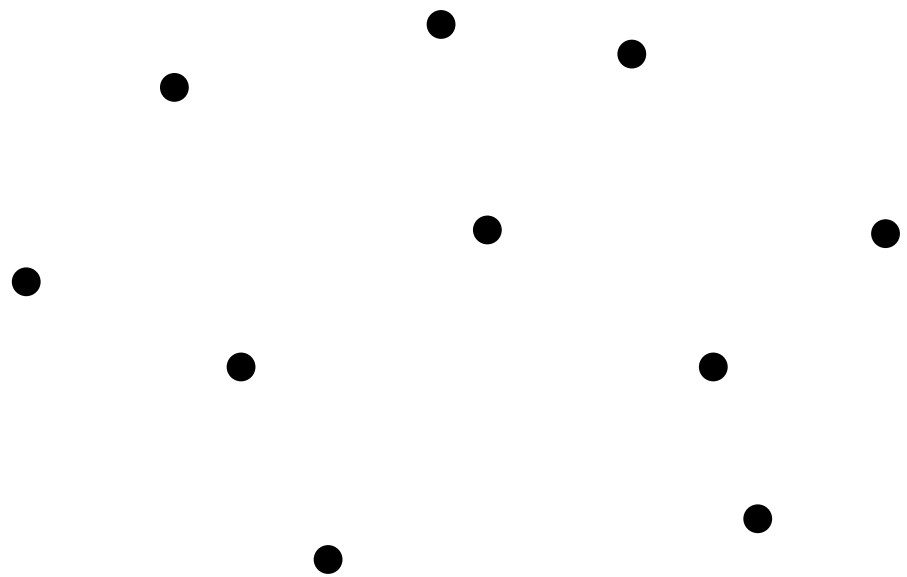


問題設定による

要素数 $= n$

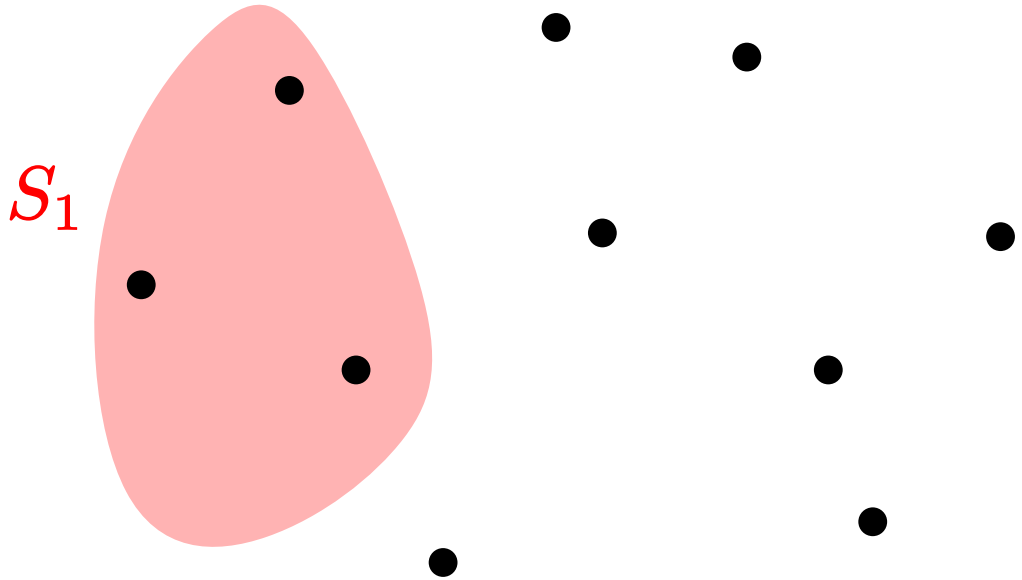
最小被覆問題：例

V：要素数 $n = 10$



最小被覆問題：例

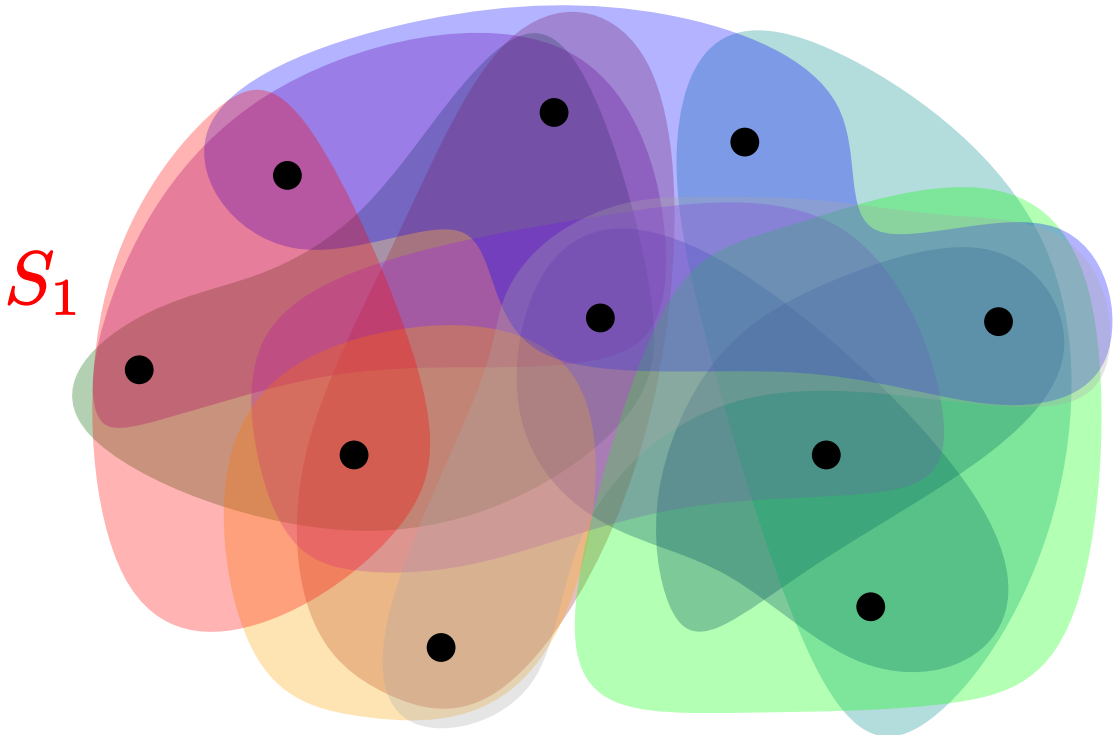
V：要素数 $n = 10$



最小被覆問題：例

V：要素数 $n = 10$

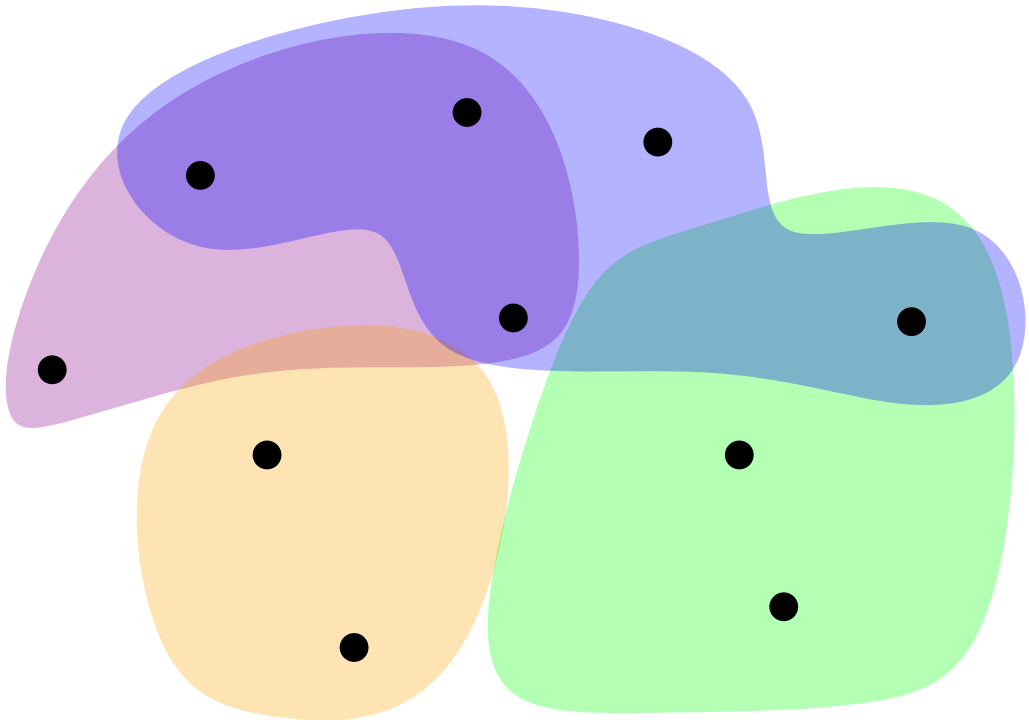
$\mathcal{S}$：要素数 $m = 12$



最小被覆問題：例

V：要素数 $n = 10$

$\mathcal{S}$：要素数 $m = 12$



$\mathcal{S}' \subseteq \mathcal{S}$：要素数 4

最小被覆問題：設定

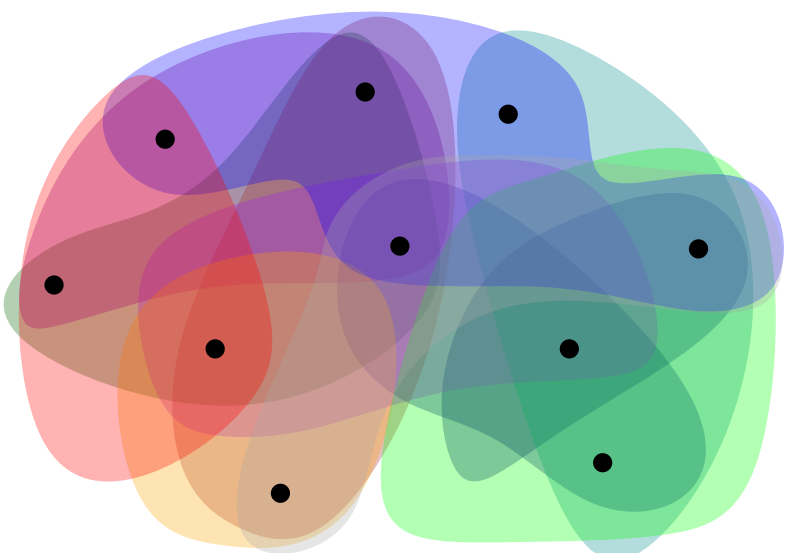
設定

- 有限集合 V
- 集合族 $\mathcal{S} \subseteq 2^V$

定義：被覆 (cover)

V の **被覆** とは，次を満たす $\mathcal{S}' \subseteq \mathcal{S}$

- $\forall v \in V, \exists X \in \mathcal{S}' : v \in X$



$\mathcal{S}'$ は V を **被覆する** ともいう
(**覆う**)

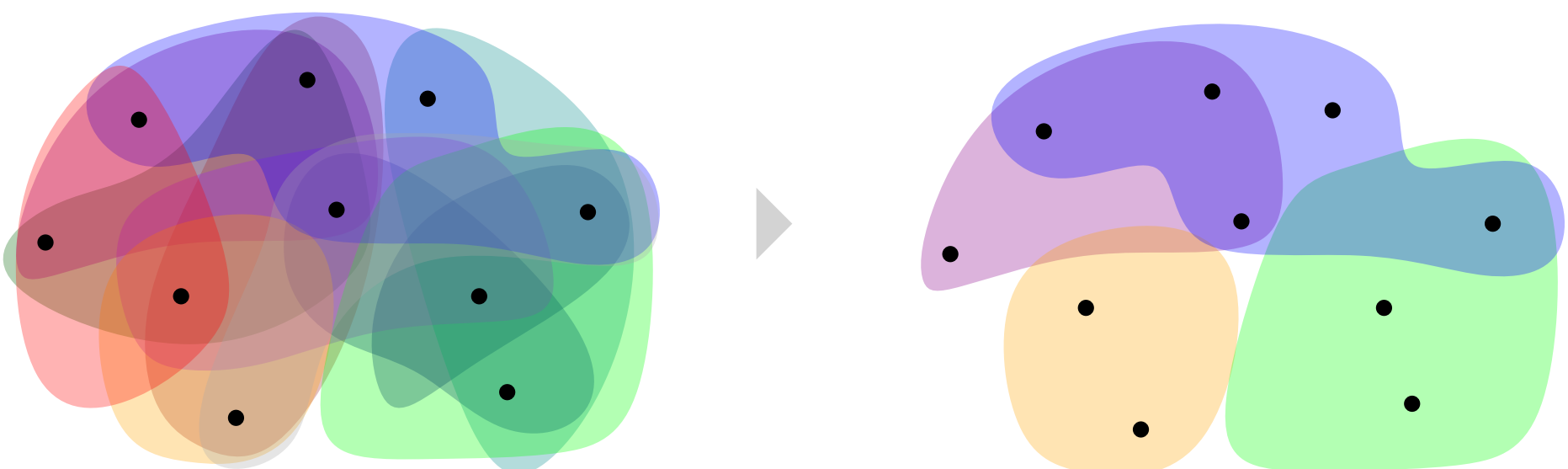
最小被覆問題：定義

定義：最小被覆問題

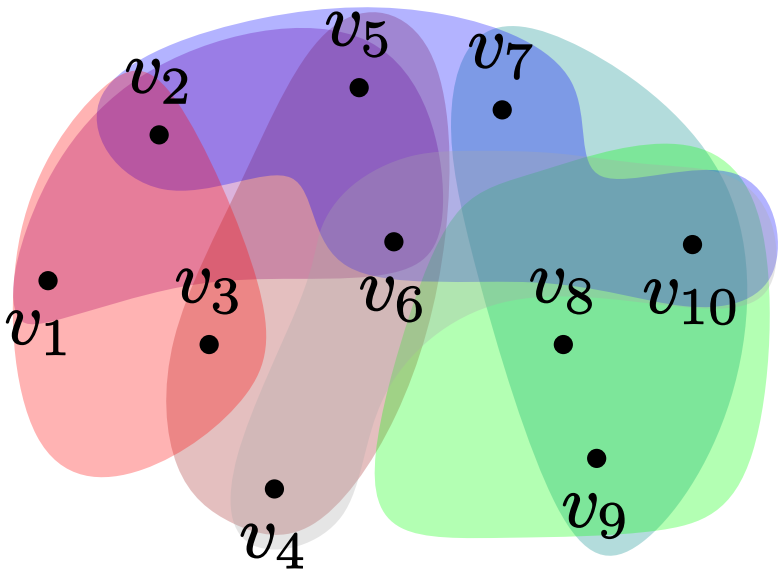
入力： 有限集合 V，集合族 $\mathcal{S} \subseteq 2^V$
出力： V の最小被覆 $\mathcal{S}' \subseteq \mathcal{S}$

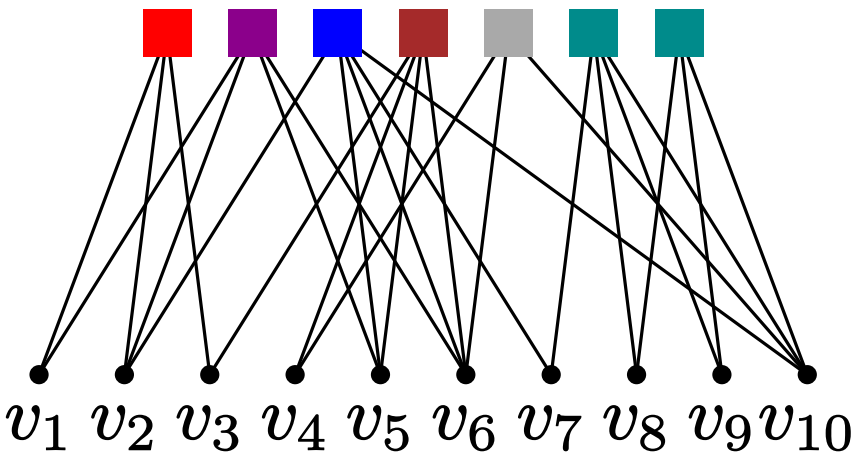
要素数最小の被覆

ここから，$n = |V|, m = |\mathcal{S}|$ とする



グラフで表現する

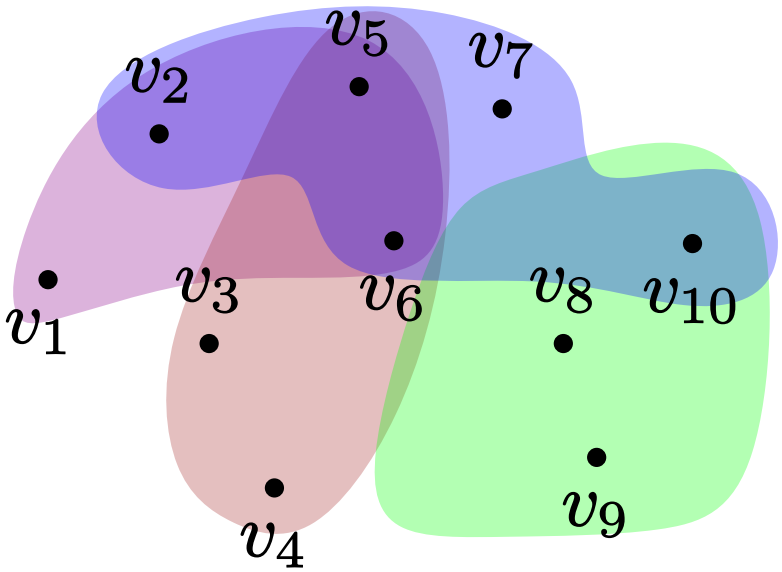


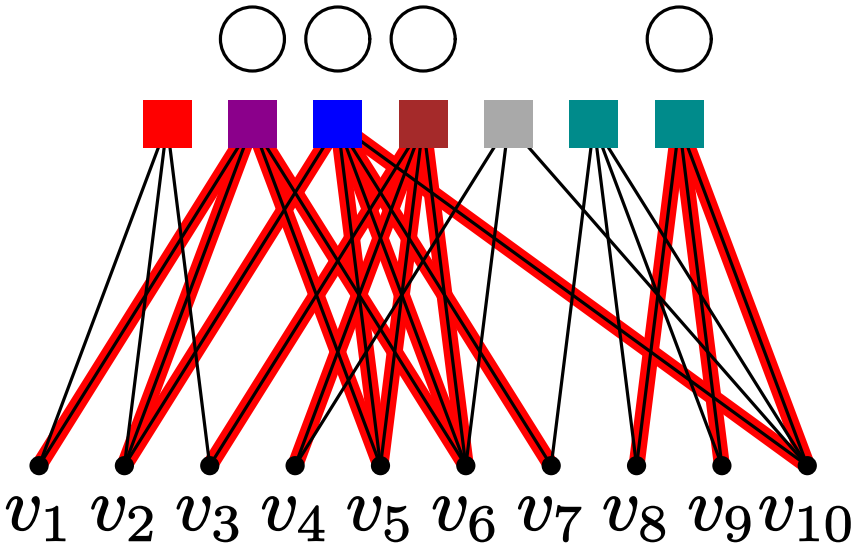


- 有限集合 V
- 集合族 $\mathcal{S} \subseteq 2^V$

- 頂点集合 $\mathcal{S} \cup V$
- 辺 $\{X, v\} \Leftrightarrow X \in \mathcal{S}, v \in X$

グラフで表現する



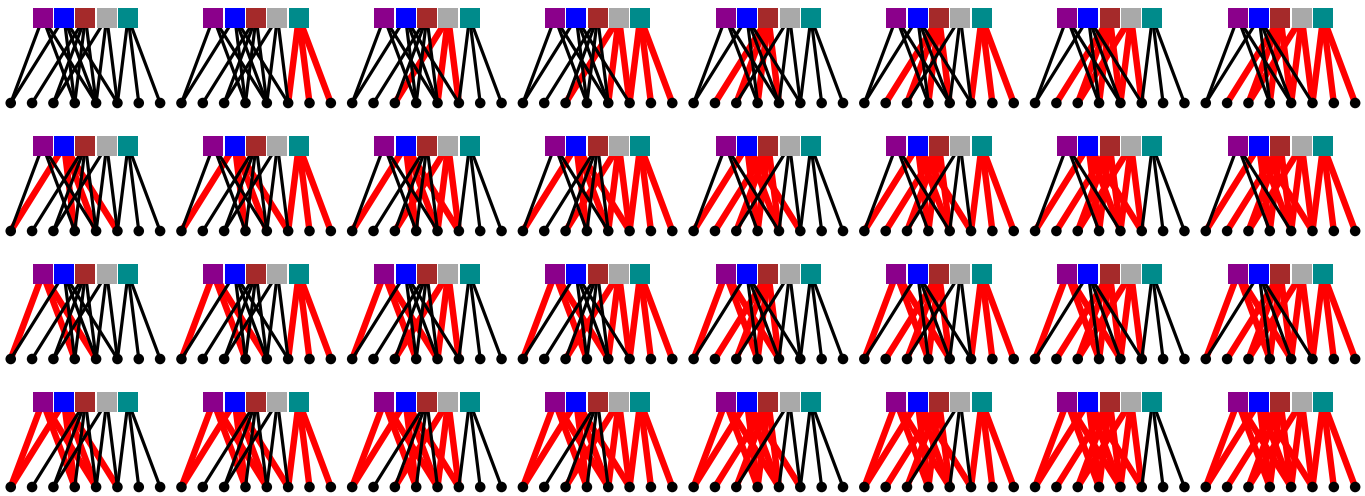


- 有限集合 V
- 集合族 $\mathcal{S} \subseteq 2^V$

- 頂点集合 $\mathcal{S} \cup V$
- 辺 $\{X, v\} \Leftrightarrow X \in \mathcal{S}, v \in X$

しらみつぶし？

すべての $\mathcal{S}' \subseteq \mathcal{S}$ を見ていく



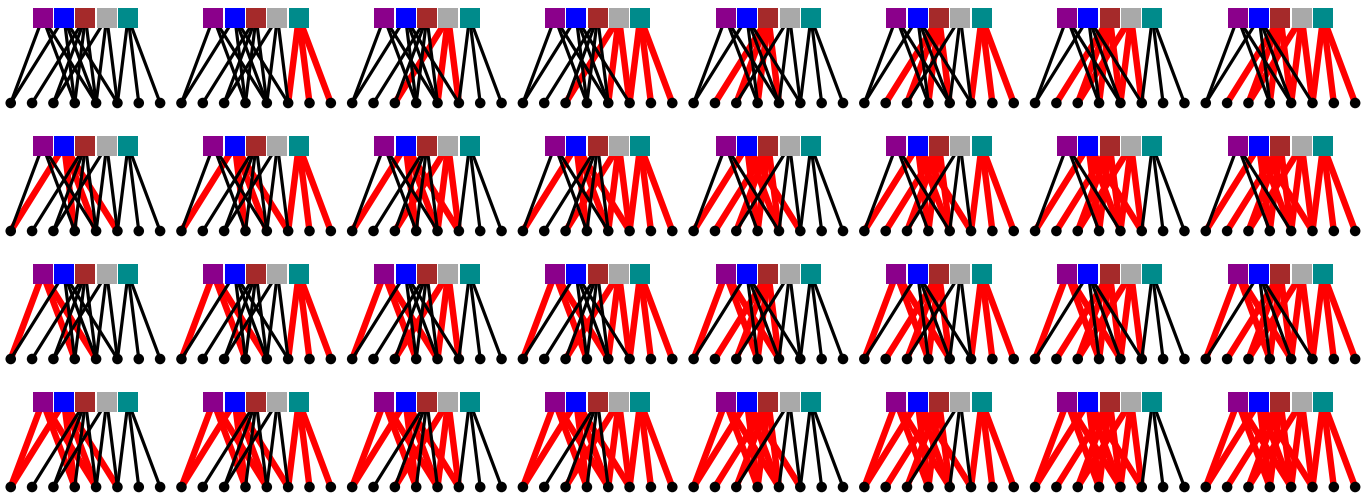
計算量：$O^*(2^m)$　$(m = |\mathcal{S}|)$

注意：m は n に比べてきわめて大きいかもしれない

$(m = 2^n$ かもしれない$)$

しらみつぶし？

すべての $\mathcal{S}' \subseteq \mathcal{S}$ を見ていく



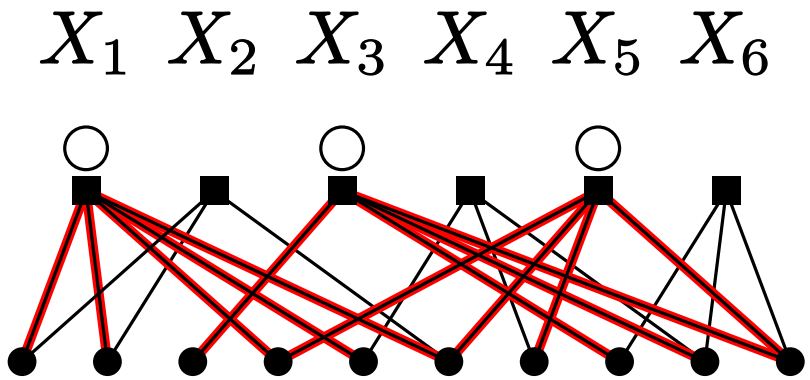
計算量：$O^*(2^m)$　$(m = |\mathcal{S}|)$

注意：m は n に比べてきわめて大きいかもしれない

$(m = 2^n$ かもしれない$)$

目標：$O^*(2^n)$

動的計画法：最適解の構造

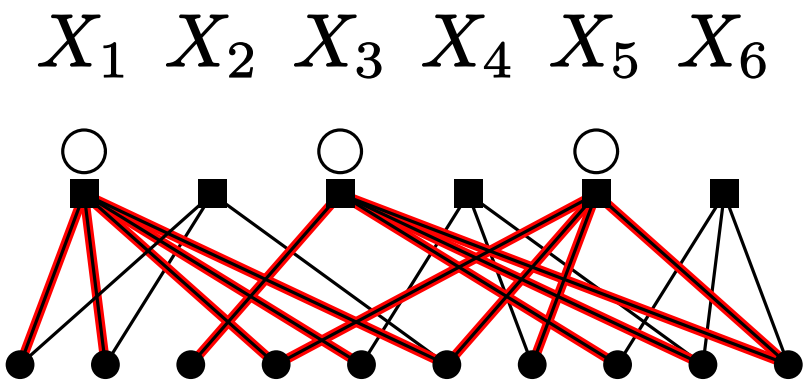


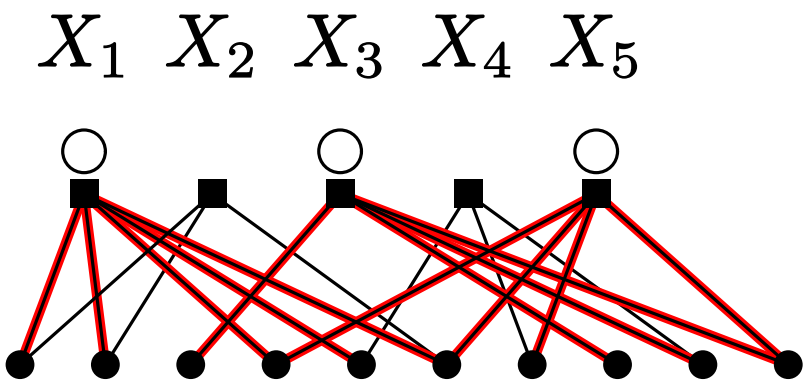
$\{X_1, X_3, X_5\}$ は
V の最小被覆

動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：最適解の構造



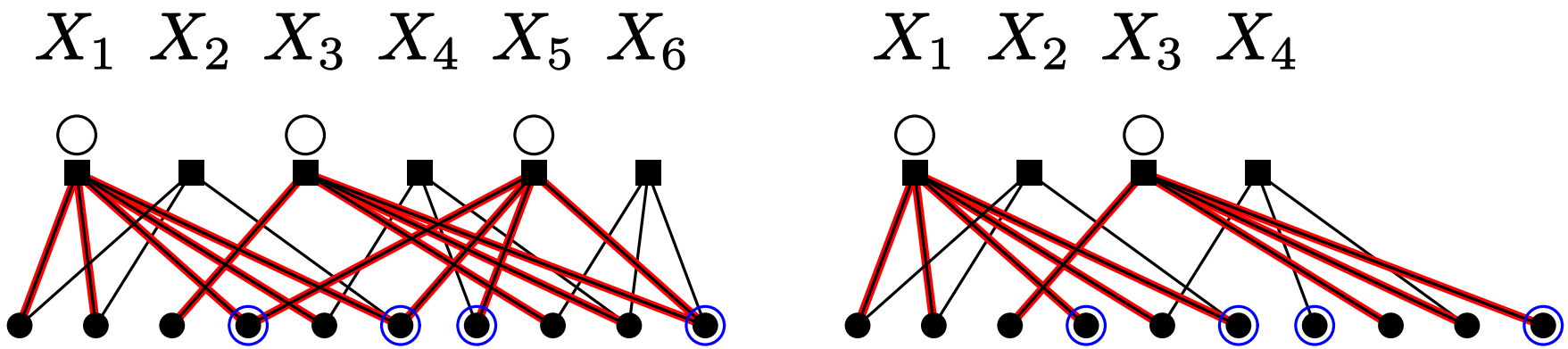


$\{X_1, X_3, X_5\}$ は
V の最小被覆

動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：最適解の構造

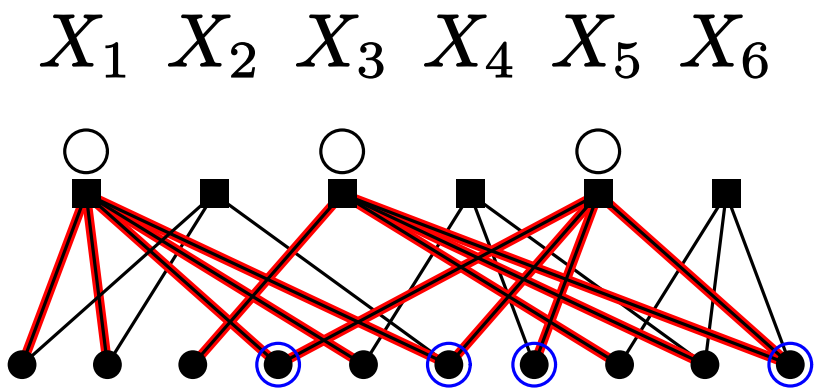


$\{X_1, X_3, X_5\}$ は
V の最小被覆

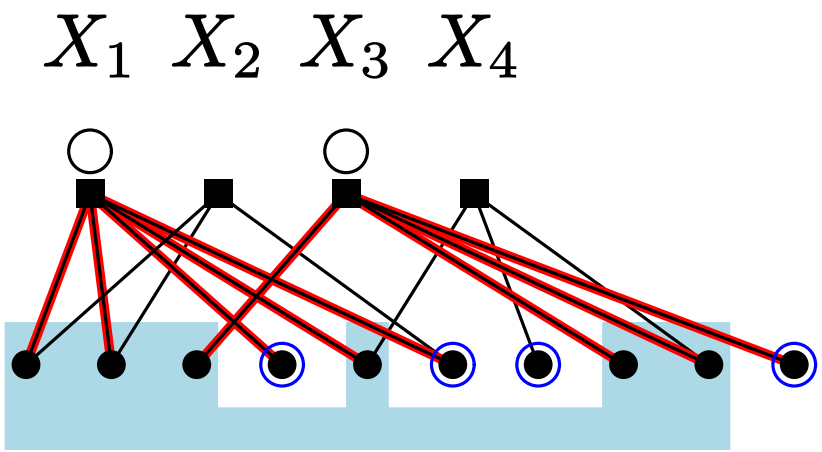
動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：最適解の構造



$\{X_1, X_3, X_5\}$ は
V の最小被覆



$\{X_1, X_3\}$ は
$V - X_5$ の最小被覆

動的計画法を考えるときの鍵

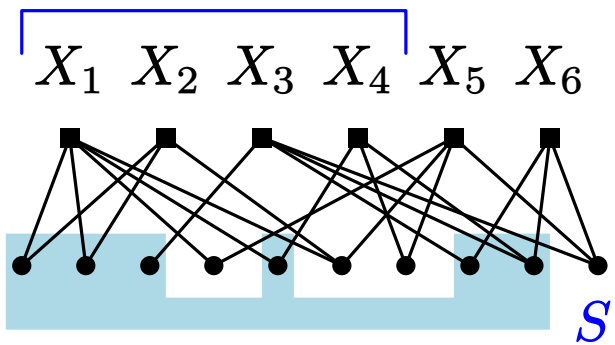
1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：状態と状態の値

状態 (i, S)　ただし，$i \in \{1, 2, \ldots, m\}, S \subseteq V$

状態の値 $f(i, S) = S$ の被覆 $\subseteq \{X_1, \ldots, X_i\}$ の
最小要素数 (被覆が非存在 $\Rightarrow \infty$)

最終的に出力する値　$f(m, V)$



動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：再帰式 (Bellman 方程式)

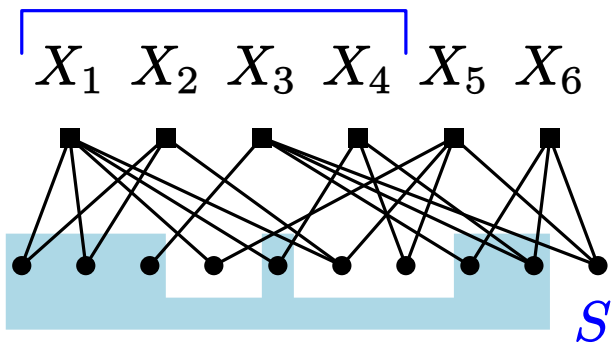
再帰式

$i \geq 2$ のとき

$$f(i, S) = \min\{f(i-1, S), 1 + f(i-1, S - X_i)\}$$

X_i を使わない被覆　　X_i を使う被覆

$$f(1, S) = \begin{cases} 0 & (S = \emptyset) \\ 1 & (\emptyset \neq S \subseteq X_1) \\ \infty & (\text{その他}) \end{cases}$$



動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる

動的計画法：アルゴリズム

アルゴリズム msc-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S)
2. すべての状態 $(1, S)$ に対して
   $f(1, S)$ を前のページのように設定
3. $i = 2, 3, \ldots, m$ の順に，$|S|$ の小さい方から順に
   $f(i, S) = \min\{f(i-1, S), 1 + f(i-1, S - X_i)\}$
4. $f(m, V)$ を出力

動的計画法：アルゴリズム

アルゴリズム msc-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S)
2. すべての状態 $(1, S)$ に対して
   $f(1, S)$ を前のページのように設定
3. $i = 2, 3, \ldots, m$ の順に，$|S|$ の小さい方から順に
   $f(i, S) = \min\{f(i-1, S), 1 + f(i-1, S - X_i)\}$
4. $f(m, V)$ を出力

状態の総数 $= \sum_{i=1}^{m} 2^n = m2^n = O^*(2^n)$

動的計画法：計算量の導出

アルゴリズム msc-dp(V, d)

1. $f(i, S) = \infty$ $\forall$ 状態 (i, S) $O^*(2^n)$
2. すべての状態 $(1, S)$ に対して
   $f(1, S)$ を前のページのように設定 $O^*(2^n)$
3. $i = 2, 3, \ldots, m$ の順に，$|S|$ の小さい方から順に
   $f(i, S) = \min\{f(i-1, S), 1 + f(i-1, S - X_i)\}$ $O^*(2^n)$
4. $f(m, V)$ を出力

結論：最小被覆問題 (Fomin, Kratsch, Woeginger '04)

最小被覆問題は $O^*(2^n)$ 時間で解ける

最小被覆問題：未解決問題と最近の進展

未解決問題

次のような定数 $c < 2$ が存在するか？

最小被覆問題は $O^*(c^n)$ 時間で解ける

最小被覆問題：未解決問題と最近の進展

未解決問題

次のような定数 $c < 2$ が存在するか？

最小被覆問題は $O^*(c^n)$ 時間で解ける

予想：集合被覆予想 (Set Cover Conjecture)

そのような定数 $c < 2$ は存在しない

(Cygan, Dell, Lokshtanov, Marx, Nederlof, Okamoto, Paturi, Saurabh, Wahlström '16)

最小被覆問題：未解決問題と最近の進展

未解決問題

次のような定数 $c < 2$ が存在するか？
最小被覆問題は $O^*(c^n)$ 時間で解ける

予想：集合被覆予想 (Set Cover Conjecture)

そのような定数 $c < 2$ は存在しない

(Cygan, Dell, Lokshtanov, Marx, Nederlof, Okamoto, Paturi, Saurabh, Wahlström '16)

最近の結果 (Björklund, Kaski '24)

多重線形代数に関するある予想 (Strassen '94) が正しい
⇒ 集合被覆予想は正しくない

まとめと次回の予告

今回と次回

動的計画法 (dynamic programming) によるアルゴリズムの設計と解析

今回

- 巡回セールスマン問題
- 最小被覆問題

次回 (予定)

- 最小シュタイナー木問題
- 彩色問題

動的計画法を考えるときの鍵

1. 最適解の持つ **再帰的な構造** を見出す
2. 上の構造から **状態** を適切に定義する
3. 状態の間の **再帰式** を立てる