

# 重み付きグラフ

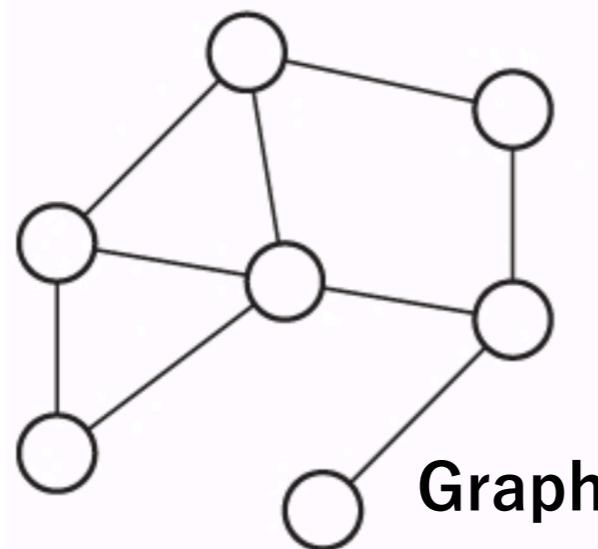
d-hacks B3 mioto

# 概要

- 木
- 全域木
  - 最小全域木
- 最短経路問題
  - 单一始点最短距離
  - 全点対間最短距離

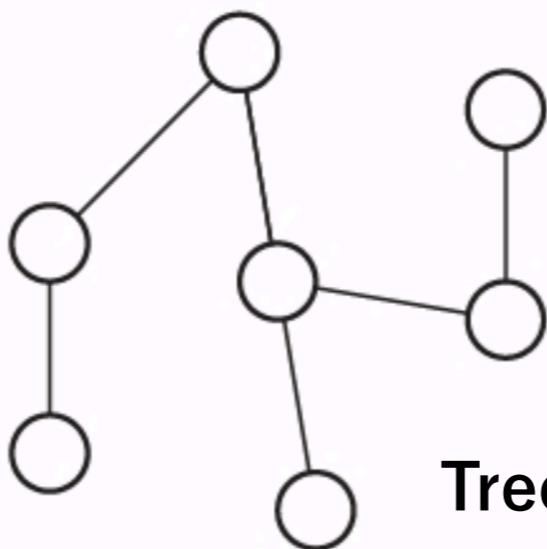
# 木

- tree
- 閉路 (loop) を待たないグラフ
- 一通りの経路



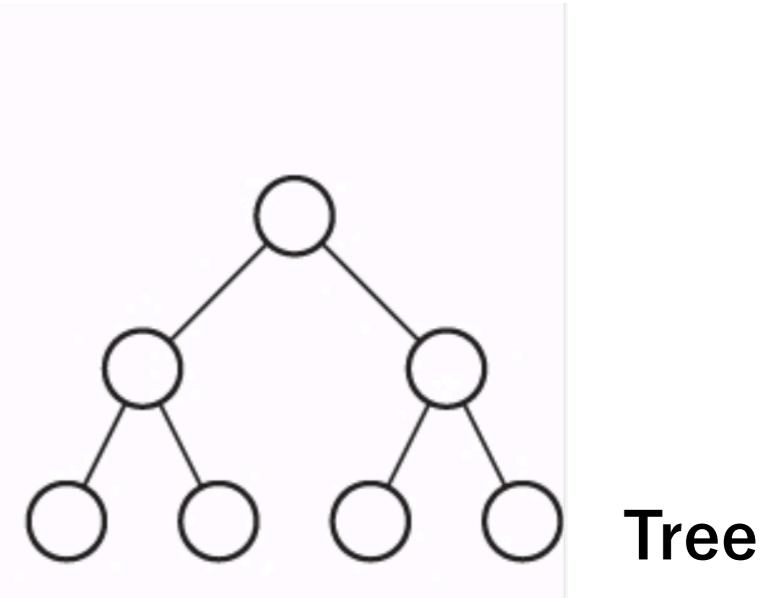
(a)

**Graph**



(b)

**Tree**

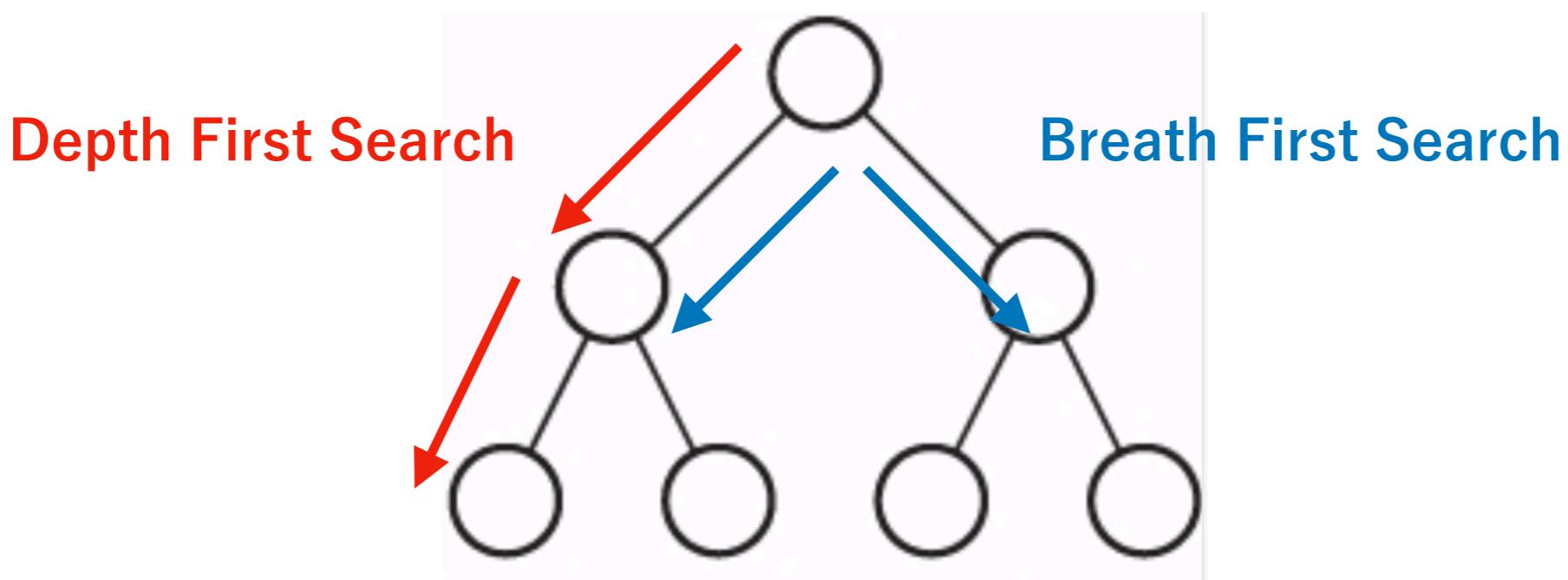


(c)

**Tree**

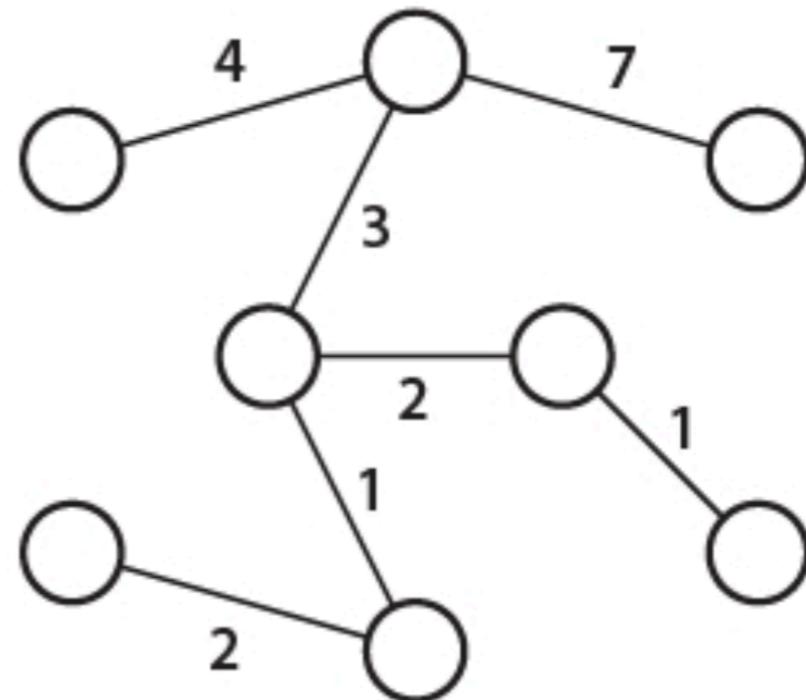
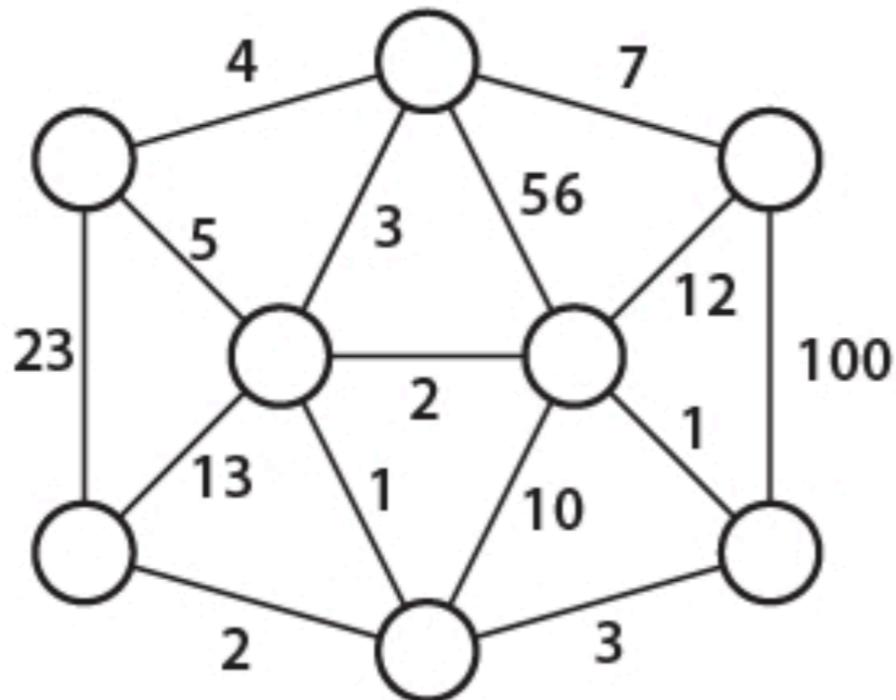
# 全域木

- Spanning Tree
- グラフの全てのノードを含む部分グラフ（木）
- 深さ優先探索（depth first search）
- 幅優先探索（breath first search）



# 最小全域木

- Minimum Spanning Tree
- 辺の重みの総和が最小の全域木



# プリム (Prim) のアルゴリズム

## プリムのアルゴリズム

グラフ  $G=(V, E)$  の頂点全体の集合を  $V$ 、MST に属する頂点の集合を  $T$  とします。

1.  $G$  から任意の頂点  $r$  を選び、それを MST のルートとして、 $T$  に追加します。

2. 次の処理を  $T = V$  になるまで繰り返します。

▶  $T$  に属する頂点と  $V - T$  に属する頂点をつなぐ辺の中で、重みが最小のものである辺  $(p_u, u)$  を選び、それを MST の辺とし、 $u$  を  $T$  に追加します。

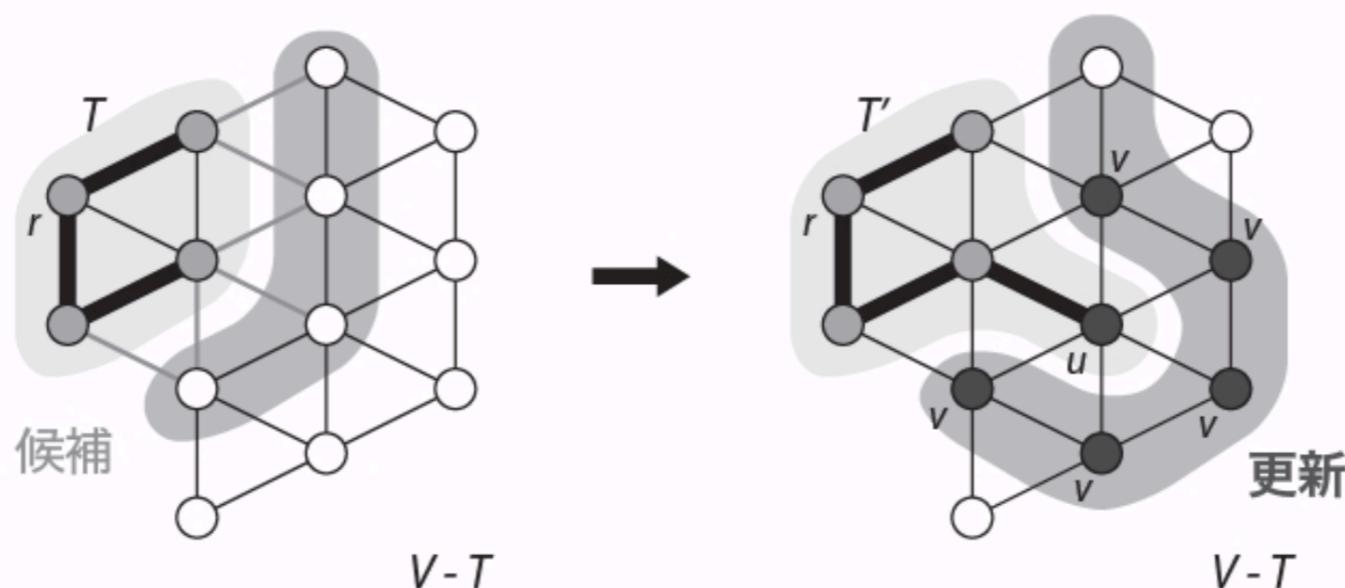
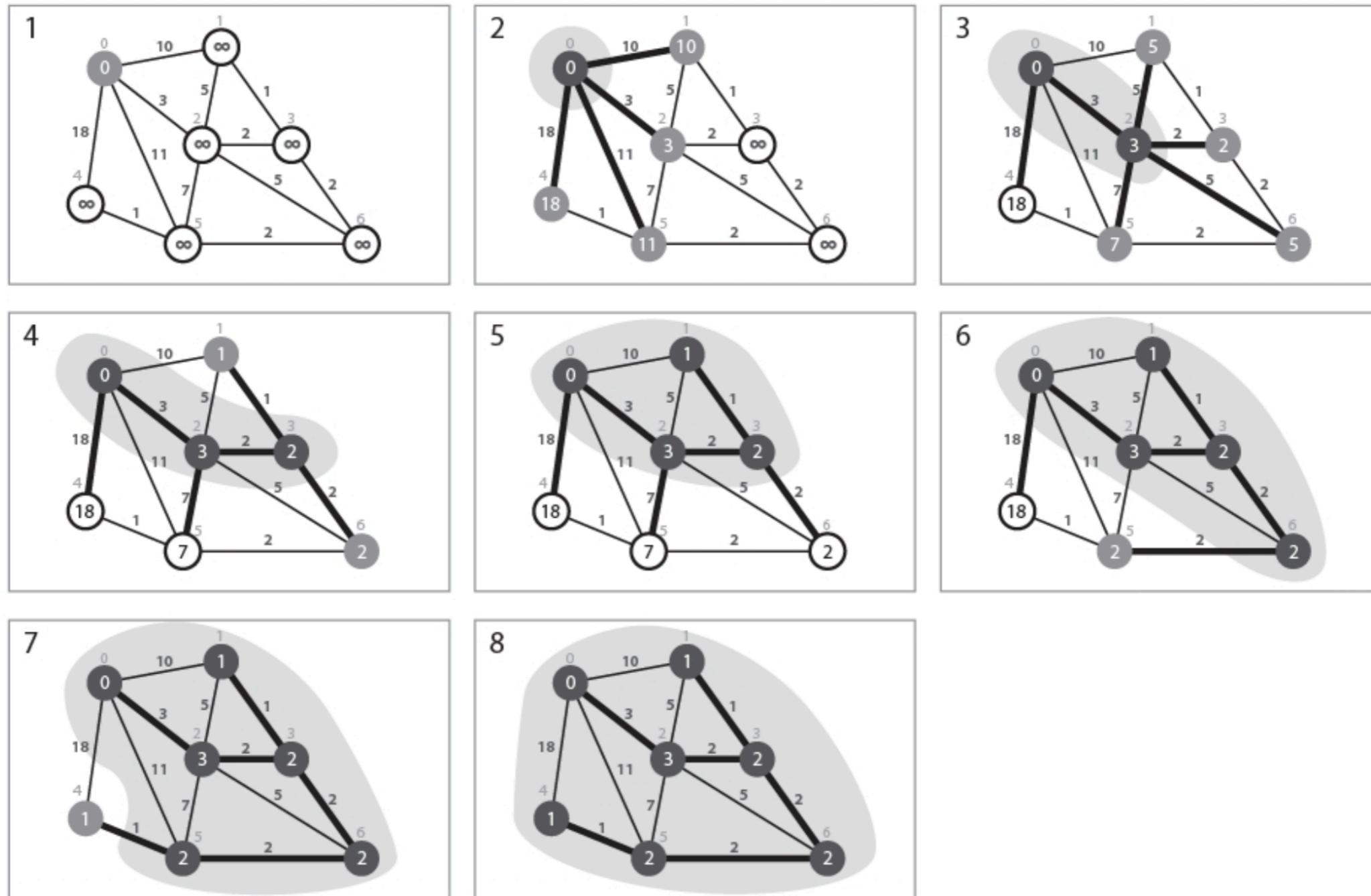


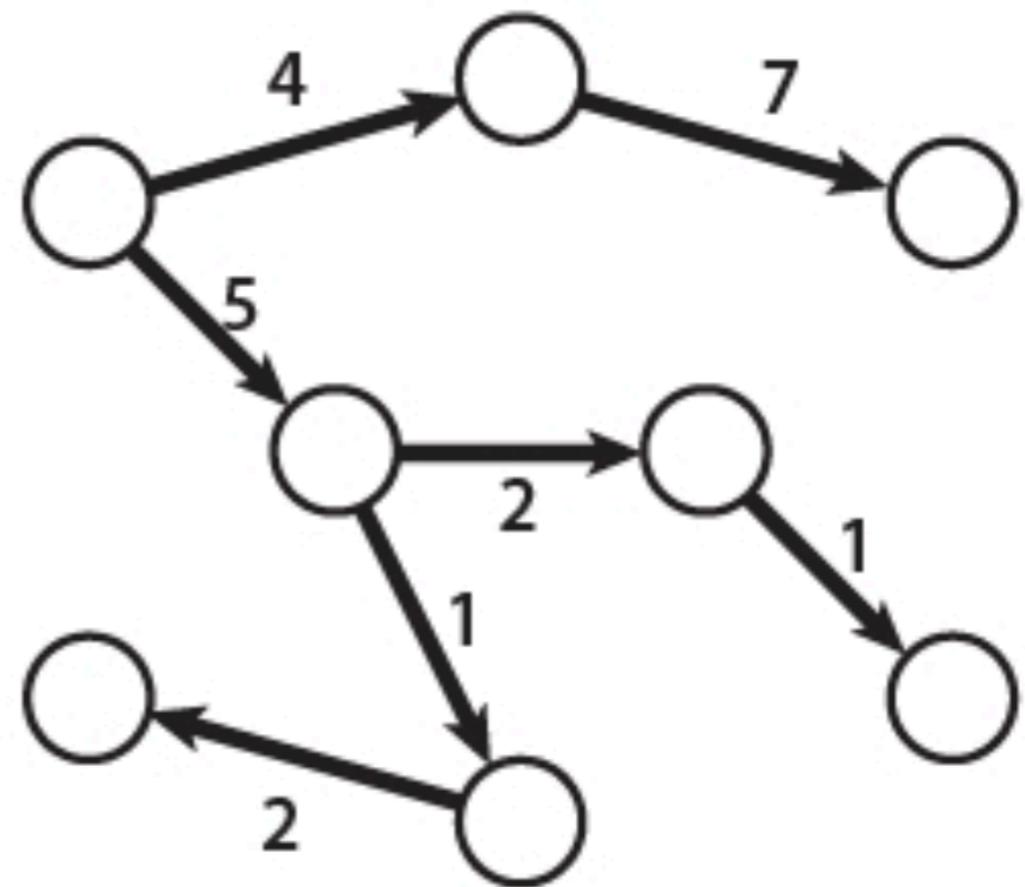
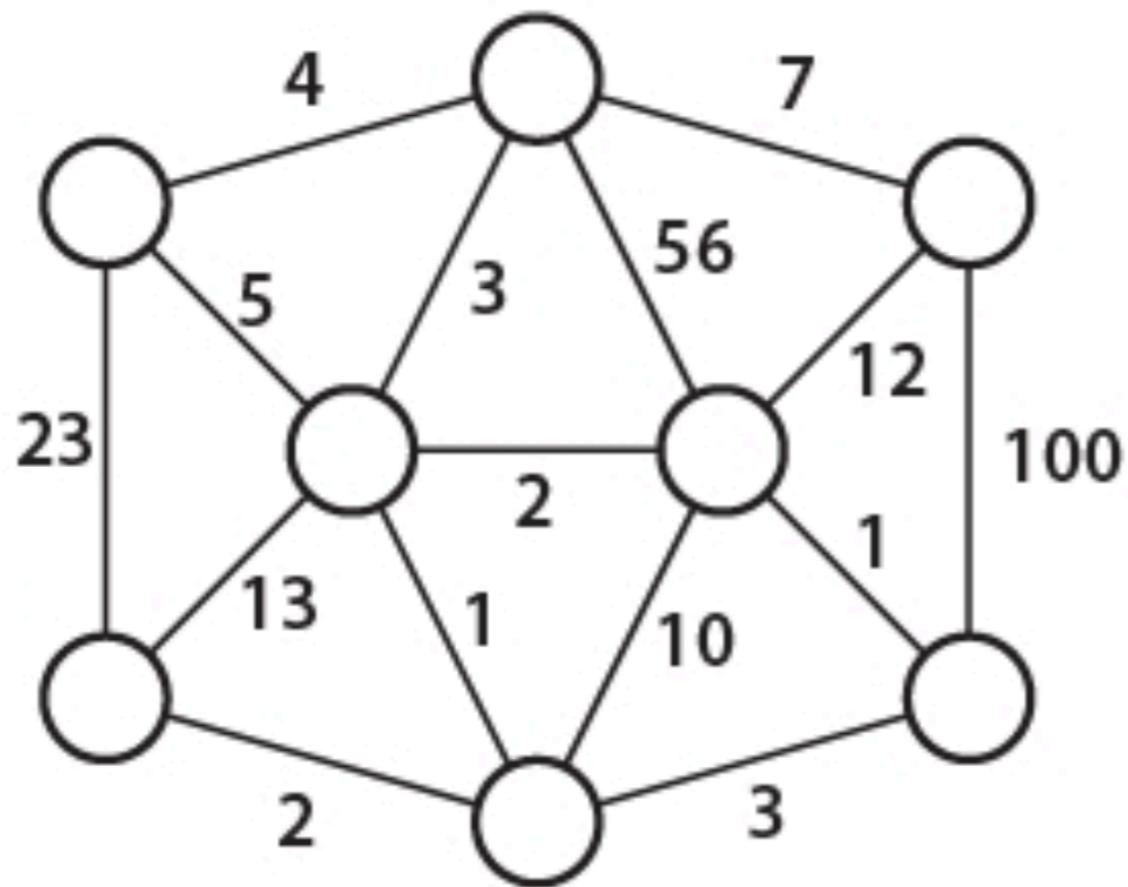
図 13.6: 最小全域木の生成

# プリムのアルゴリズム



# 最短経路問題

- Shortest Path Problem
- 辺の重みの総和が最小の経路



# 最短経路問題の種類

- 単一始点最短経路 (Single Source Shortest Path)
  - 一つの頂点から全ての頂点までの最短経路
- 全点対間最短経路 (All Pairs Shortest Path)
  - 全ての頂点のペア間の最短経路

# ダイクストラのアルゴリズム

- Dijkstra's Algorithm
- 最短経路問題をコストで解決

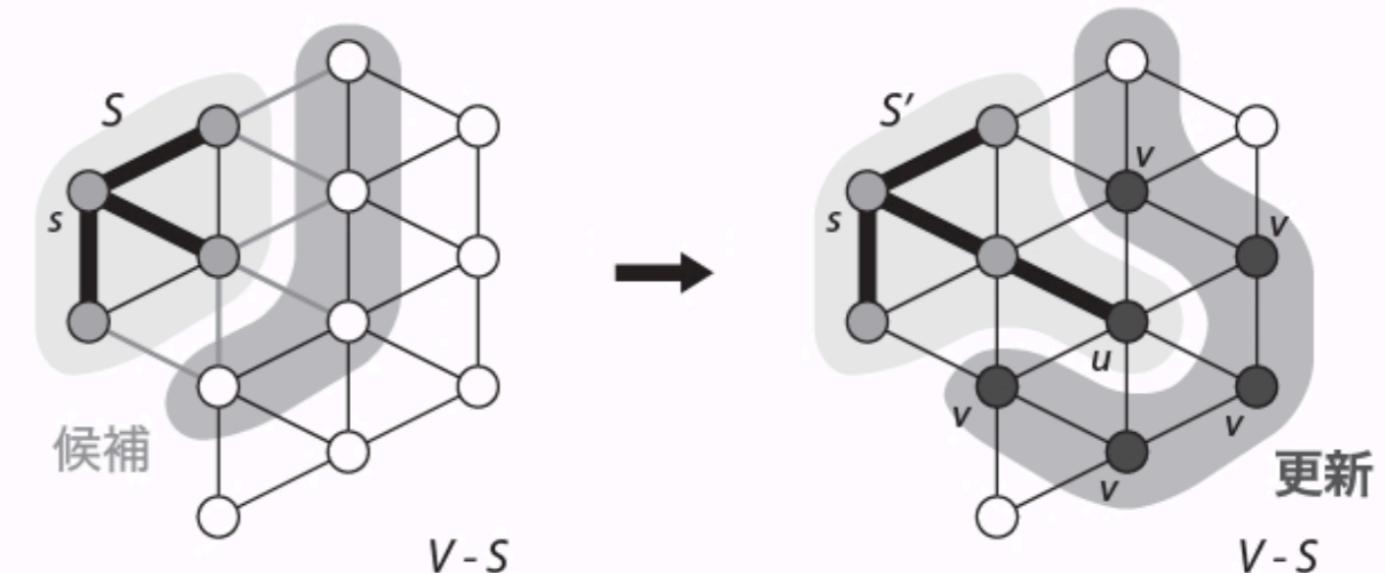


図 13.9: 最短経路木の生成

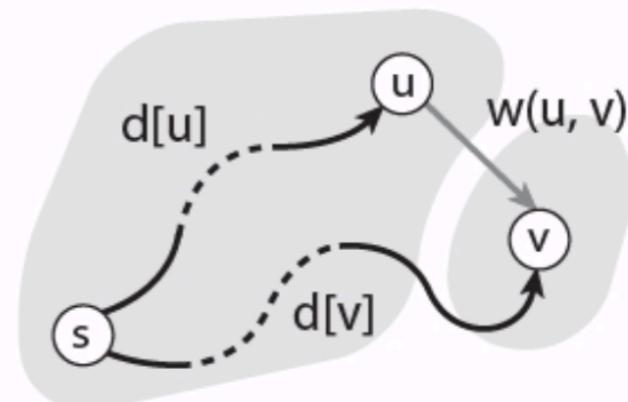


図 13.10: 最短コストの更新

# ダイクストラのアルゴリズム

各頂点 $i$ について、 $S$ 内の頂点のみを経由した $s$ から $i$ への最短経路のコストを $d[i]$ 、最短経路木における $i$ の親を $p[i]$ とします。

1. 初期状態で、 $S$ を空にします。

$s$ に対して  $d[s] = 0$ ,

$s$ 以外の $V$ に属する全ての頂点 $i$ に対して  $d[i] = \infty$   
と初期化します。

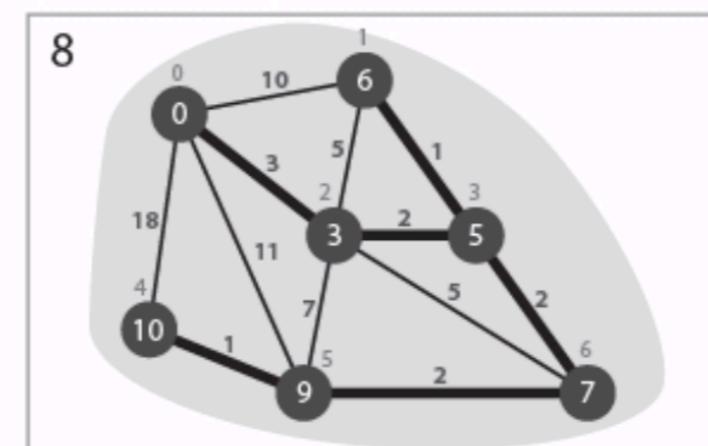
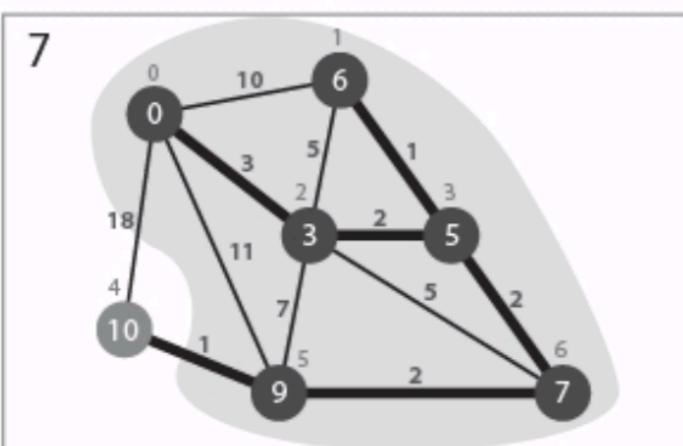
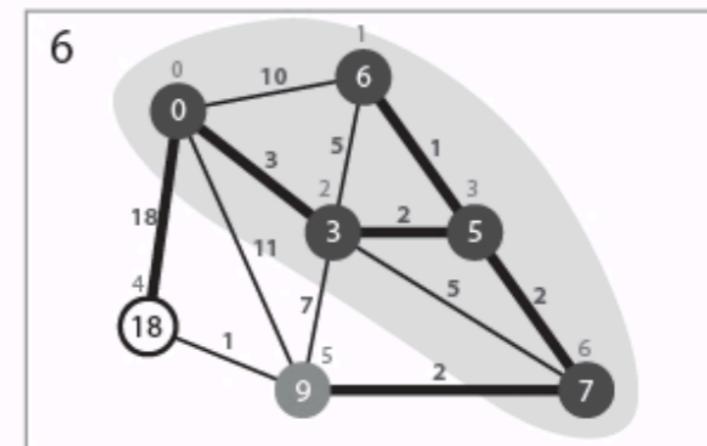
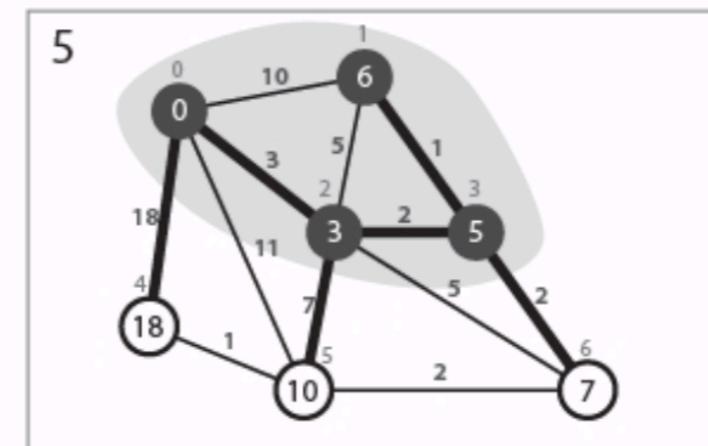
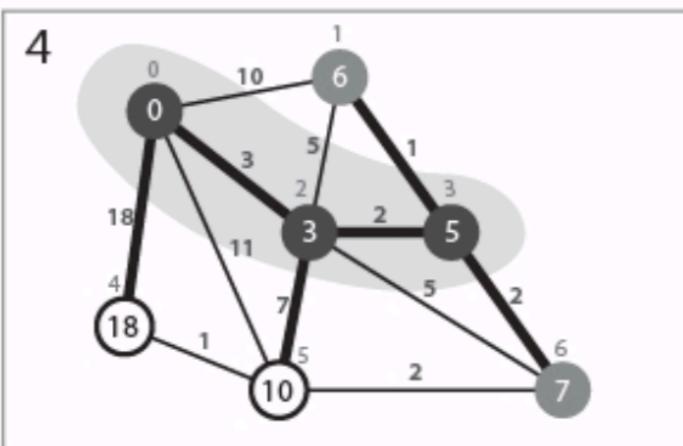
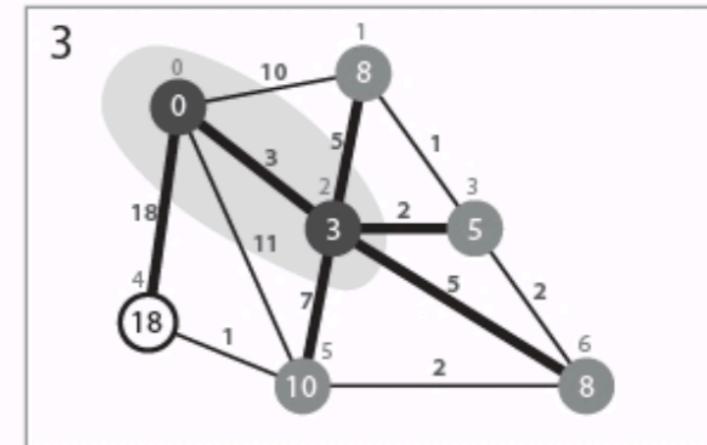
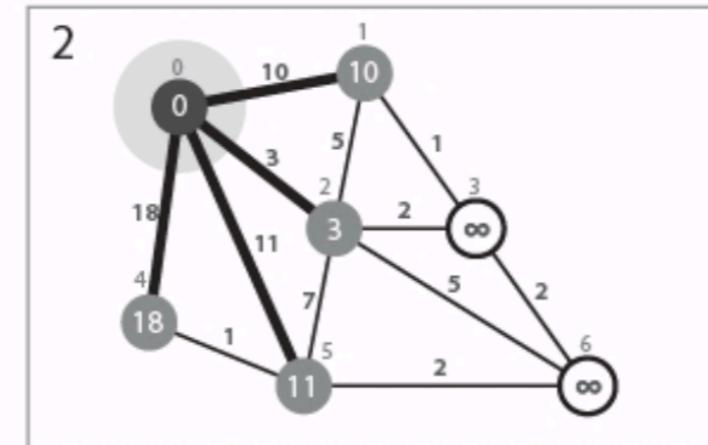
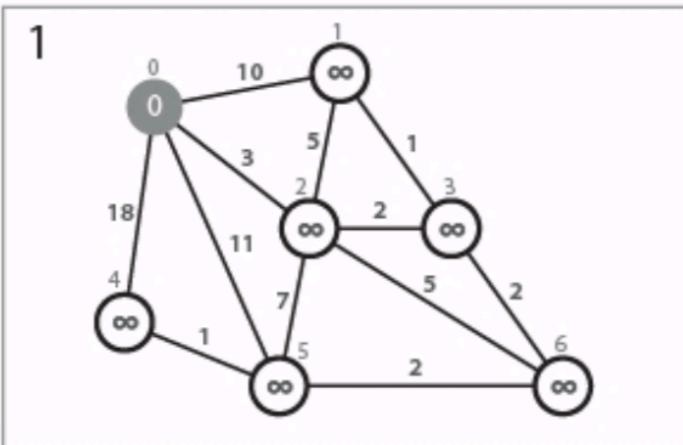
2. 以下の処理を $S = V$ となるまで繰り返します。

▶  $V - S$ の中から、 $d[u]$ が最小である頂点 $u$ を選択します (図13.9)。

▶  $u$ を $S$ に追加すると同時に、 $u$ に隣接しかつ $V - S$ に属する全ての頂点 $v$ に対する値を以下のように更新します(図13.10)。

```
if  $d[u] + w(u, v) < d[v]$ 
   $d[v] = d[u] + w(u, v)$ 
   $p[v] = u$ 
```

# ダイクストラのアルゴリズム



# ダイクストラを高速化

- 一般的には隣接行列を参照
- 隣接重みを保存
- 二分ヒープ（優先度付きキュー）

# ダイクストラ（優先度付き）

各頂点  $i$  について、 $S$ 内の頂点のみを経由した  $s$  から  $i$ への最短経路のコストを  $d[i]$ 、最短経路木における  $i$  の親を  $p[i]$  とします。

1. 初期状態で、 $S$  を空とします。

$s$  に対して  $d[s] = 0$

$s$  以外の  $V$  に属する全ての頂点  $i$  に対して  $d[i] = \infty$

と初期化します。

$d[i]$  をキーとして、 $V$ の頂点を min- ヒープ  $H$  として構築します。

2. 次の処理を  $S = V$  となるまで繰り返します。

▶  $H$  から  $d[u]$  が最小である頂点  $u$  を取り出します。

▶  $u$  を  $S$  に追加すると同時に、 $u$  に隣接しかつ  $V - S$  に属する全ての頂点  $v$  に対する値を以下のように更新します。

if  $d[u] + w(u, v) < d[v]$

$d[v] = d[u] + w(u, v)$

$p[v] = u$

$v$  を起点にヒープ  $H$  を更新する

# 高速化前後

- 一般的なダイクストラアルゴリズム
  - $O(|V|^2)$
- 二分ヒープ
  - $O(|V| \log |V|)$