オペレーションズ・リサーチ III (4)

田中 俊二

shunji.tanaka@okayama-u.ac.jp

スケジュール

No.	内容
1	導入 (組合せ最適化，グラフ・ネットワーク，整数計画問題)
2	計算複雑さの理論
3	グラフ・ネットワーク 1 (グラフの分類，用語，種々の問題)
4	グラフ・ネットワーク 2 (最短経路問題，動的計画法)
5	グラフ・ネットワーク 3 (最小全域木，最大フロー問題)
6	グラフ・ネットワーク 4 (マッチング)
7	整数計画 (緩和問題，分枝限定法，切除平面法)

最短経路問題

最短経路問題 (shortest path problem)

ネットワーク上で最短経路 (辺の重みの和が最小の道) を求める問題

最短経路問題の種類

- 単一点対 (single-pair) 最短経路問題
  2 頂点間の最短経路を求める問題
- 単一始点 (single-source) 最短経路問題
  ある頂点から残りすべての頂点への最短経路を求める問題
- 全点対 (all-pairs) 最短経路問題
  すべての 2 頂点の組に対して最短経路を求める問題

上の最短経路問題はいずれも多項式時間で求解可能

単一始点最短経路問題の解法 (単一点対問題も解ける)

- ベルマン・フォード法 (Bellman-Ford algorithm)
- ダイクストラ法 (Dijkstra's algorithm)

重み付き有向グラフ（ネットワーク) で考える．無向グラフでも同様

最短経路問題

最短経路問題 (shortest path problem)

ネットワーク上で最短経路 (辺の重みの和が最小の道) を求める問題

最短経路問題の種類

- **単一点対 (single-pair) 最短経路問題**
  2 頂点間の最短経路を求める問題
- **単一始点 (single-source) 最短経路問題**
  ある頂点から残りすべての頂点への最短経路を求める問題
- 全点対 (all-pairs) 最短経路問題
  すべての 2 頂点の組に対して最短経路を求める問題

上の最短経路問題はいずれも多項式時間で求解可能

単一始点最短経路問題の解法 (単一点対問題も解ける)

- ベルマン・フォード法 (Bellman-Ford algorithm)
- ダイクストラ法 (Dijkstra's algorithm)

重み付き有向グラフ（ネットワーク) で考える．無向グラフでも同様

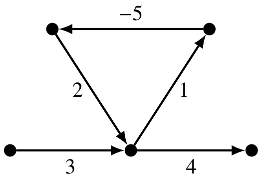
ベルマン・フォード法と負閉路

ベルマン・フォード法

- ダイクストラ法よりも遅いが，負の重みの辺が存在する問題にも適用可能 (ダイクストラ法は適用不可)
- **負閉路**の存在判定にも使用可能

負閉路 (negative cycle)

- 辺の重みの和が負の閉路
- 負閉路を繰り返し通過することで，重み和をいくらでも小さくできてしまう



負閉路

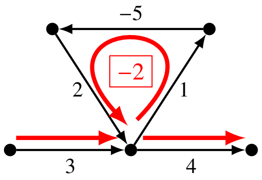
ベルマン・フォード法と負閉路

ベルマン・フォード法

- ダイクストラ法よりも遅いが，負の重みの辺が存在する問題にも適用可能 (ダイクストラ法は適用不可)
- **負閉路**の存在判定にも使用可能

負閉路 (negative cycle)

- 辺の重みの和が負の閉路
- **負閉路を繰り返し通過することで，重み和をいくらでも小さくできてしまう**



負閉路

$$\cdots + 3 + (-2 - 2 - \cdots - 2) + 4 + \ldots$$

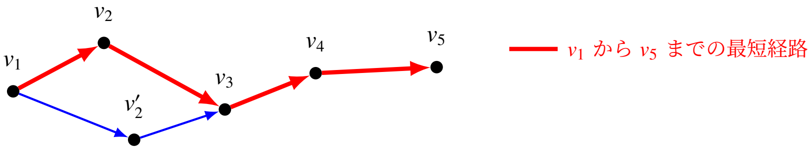
最短経路の最適部分構造

最適部分構造 (optimal substructure)

部分問題の最適解を用いて，もとの問題の最適解を組み立てることができる

最短経路問題の最適部分構造

頂点 v_1 から頂点 v_k までの最短経路 $(v_1, v_2, \ldots, v_k)$ において，部分路 $(v_1, v_2, \ldots, v_i)$ $(1 \leq i \leq K)$ は v_1 から v_i への最短経路



もし v_1 から v_3 までの最短経路が (v_1, v_2', v_3) なら，v_1 から v_5 までの最短経路が (v_1, v_2, v_3) を通ることはありえない．$(v_1, v_2, v_3, v_4, v_5)$ より $(v_1, v_2', v_3, v_4, v_5)$ の方が短くなる

ベルマン・フォード法

準備

- 有向グラフ：$G = (V, E)$　(頂点数 $|V|$，辺数 $|E|$)
- 辺 $e = (u, v)$ の重み (距離)：$w(u, v)$
- 始点：s
- 始点 s から頂点 v までの**最短距離の上界**：$d(v)$

ベルマン・フォード法 (Bellman-Ford algorithm)

1. $d(v)$ の初期化：$d(v) := \begin{cases} 0, & v = s \\ \infty, & \text{それ以外} \end{cases}$
2. 以下を $(|V| - 1)$ 回行う：
   すべての辺 $(u, v) \in E$ について，$d(u) + w(u, v) < d(v)$ ならば $d(v)$ を
   $d(v) := d(u) + w(u, v)$ と更新 (緩和手続き, relaxing)

- 第 k 反復後の $d(v)$ は，通過する辺数が k 以下の経路に限定したときの，始点 s から v までの最短距離
- ただし，辺をチェックする順序によっては，より多くの辺を通過できる
- 負閉路がなければ，最短経路は辺をたかだか $(|V| - 1)$ 本しか通過しない
- 終了時点での $d(v)$ が始点 s から頂点 v までの最短距離

ベルマン・フォード法の擬似コード

擬似コード (pseudocode)

- コンピュータプログラムの簡略表現
- 色々な流儀がある

ベルマン・フォード法の擬似コード

procedure BELLMANFORD(V, E, w, s)
    for all $v \in V$ **do**
        $d(v) \leftarrow \infty$ ▹ $d(v)$ を ∞ で初期化
    $d(s) \leftarrow 0$ ▹ 始点は 0 で初期化
    for $k \leftarrow 1$ **to** $|V| - 1$ **do** ▹ $|V| - 1$ 回反復
        for all $(u, v) \in E$ **do** ▹ すべての辺
            if $d(v) > d(u) + w(u, v)$ **then**
                $d(v) \leftarrow d(u) + w(u, v)$ ▹ 緩和手続き
    for all $(u, v) \in E$ **do** ▹ 負閉路が存在するかを調べる
        if $d(v) > d(u) + w(u, v)$ **then** ▹ $d(v)$ が収束しなかった
            return false ▹ 負閉路あり
    return true ▹ 負閉路なし

計算量は $O((|V| - 1)|E|) = O(|V||E|)$

ベルマン・フォード法の擬似コード (続き)

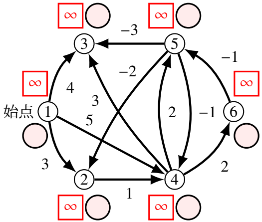
このままでは最短距離しか求まらないので，最短経路が求まるよう修正

ベルマン・フォード法の擬似コード (修正版)

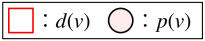
procedure BellmanFord(V, E, w, s)
  for all $v \in V$ **do**
    $d(v) \leftarrow \infty$ ▷ $d(v)$ を ∞ で初期化
    $p(v) \leftarrow$ null ▷ $p(v)$ を null で初期化
  $d(s) \leftarrow 0$ ▷ 始点は 0 で初期化
  for $k \leftarrow 1$ **to** $|V| - 1$ **do** ▷ $(|V| - 1)$ 回反復
    for all $(u, v) \in E$ **do** ▷ すべての辺
      if $d(v) > d(u) + w(u, v)$ **then**
        $d(v) \leftarrow d(u) + w(u, v)$ ▷ 緩和
        $p(v) \leftarrow u$ ▷ v の直前に訪れる頂点を更新
  for all $(u, v) \in E$ **do** ▷ 負閉路が存在するかを調べる
    if $d(v) > d(u) + w(u, v)$ **then** ▷ $d(v)$ が収束しなかった
      return false ▷ 負閉路あり
  return true ▷ 負閉路なし

procedure ShortestPath(p, v) ▷ p を使って v までの最短経路を取り出す
  $L \leftarrow \{v\}$ ▷ 最短経路の初期化
  while $p(v) \neq$ null **do** ▷ $p(v) =$ null となるのは v が始点のときのみ
    $v \leftarrow p(v)$ ▷ v から s 方向へ最短経路を逆に辿る
    $L \leftarrow \{v\} \cup L$ ▷ 最短経路の先頭に追加
  return L

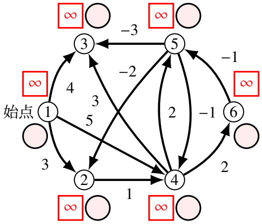
ベルマン・フォード法の例



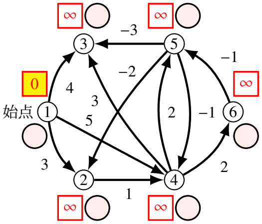
0 回目の反復



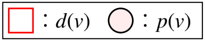
ベルマン・フォード法の例



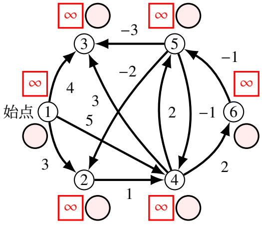
0 回目の反復



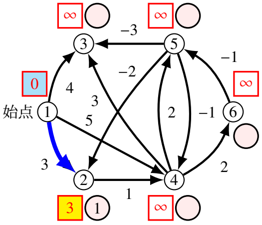
1 回目の反復



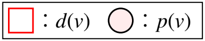
ベルマン・フォード法の例



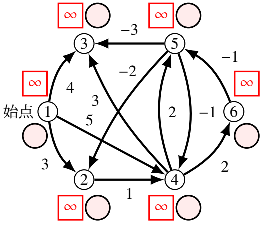
0 回目の反復



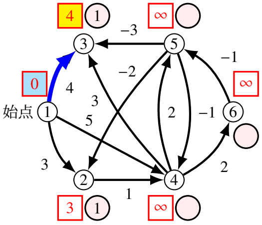
1 回目の反復



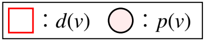
ベルマン・フォード法の例



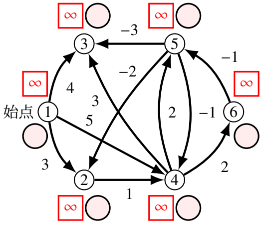
0 回目の反復



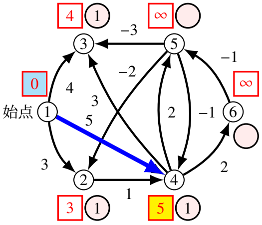
1 回目の反復



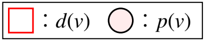
ベルマン・フォード法の例



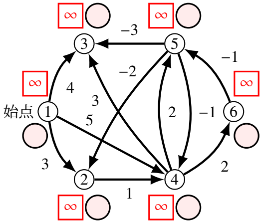
0 回目の反復



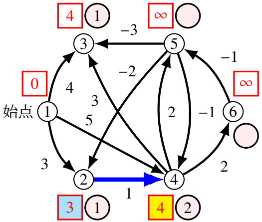
1 回目の反復



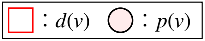
ベルマン・フォード法の例



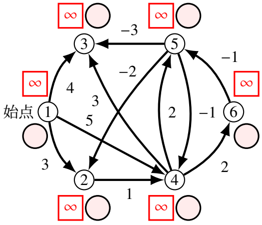
0 回目の反復



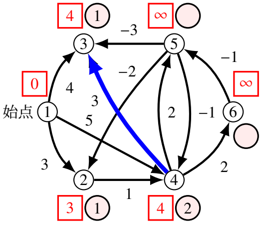
1 回目の反復



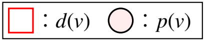
ベルマン・フォード法の例



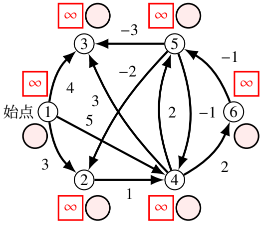
0 回目の反復



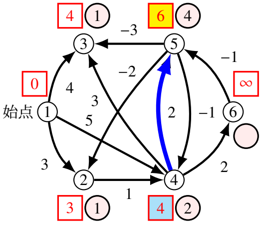
1 回目の反復



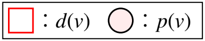
ベルマン・フォード法の例



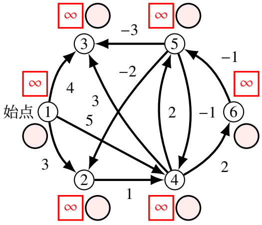
0 回目の反復



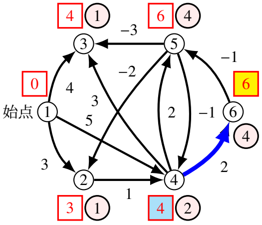
1 回目の反復



ベルマン・フォード法の例



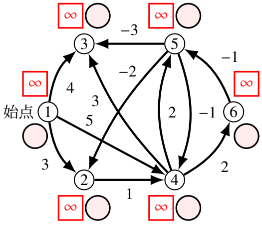
0 回目の反復



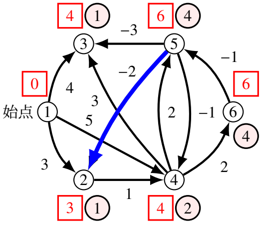
1 回目の反復

□ : $d(v)$ ○ : $p(v)$

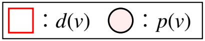
ベルマン・フォード法の例



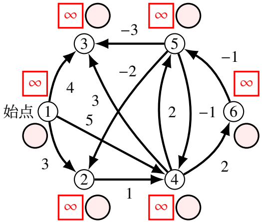
0 回目の反復



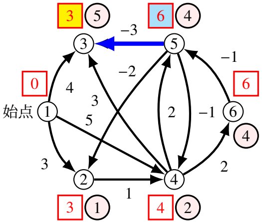
1 回目の反復



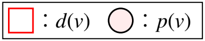
ベルマン・フォード法の例



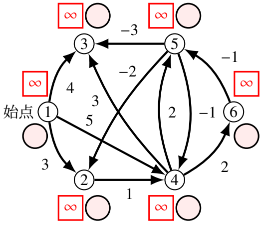
0 回目の反復



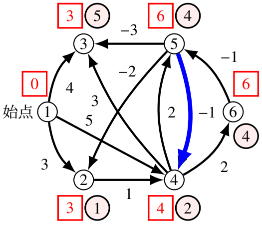
1 回目の反復



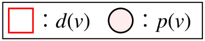
ベルマン・フォード法の例



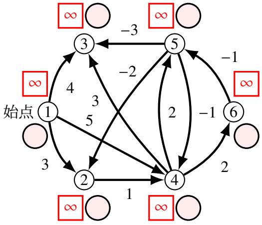
0 回目の反復



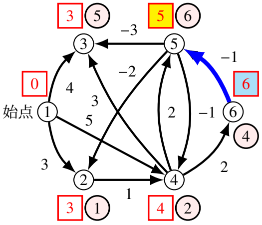
1 回目の反復



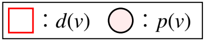
ベルマン・フォード法の例



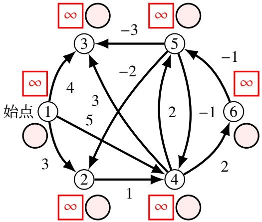
0 回目の反復



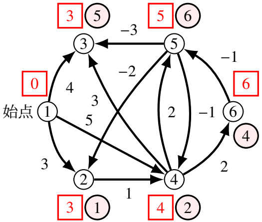
1 回目の反復



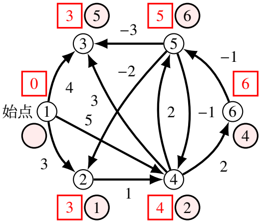
ベルマン・フォード法の例



0 回目の反復



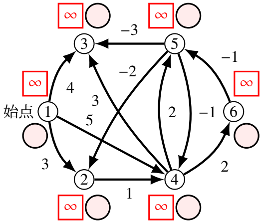
1 回目の反復



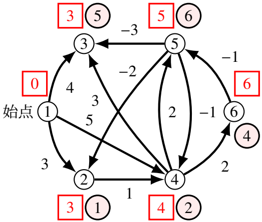
2 回目の反復

□ : $d(v)$ ○ : $p(v)$

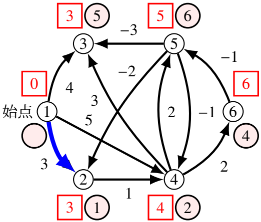
ベルマン・フォード法の例



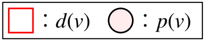
0 回目の反復



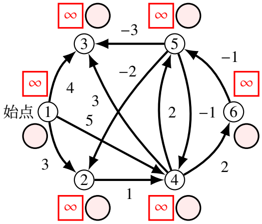
1 回目の反復



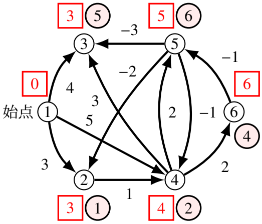
2 回目の反復



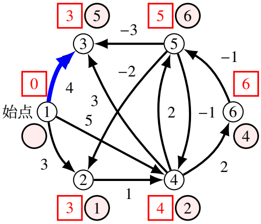
ベルマン・フォード法の例



0 回目の反復



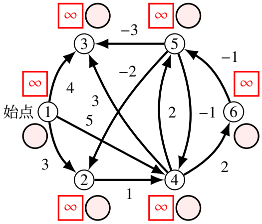
1 回目の反復



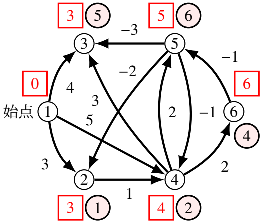
2 回目の反復

□ : $d(v)$ ○ : $p(v)$

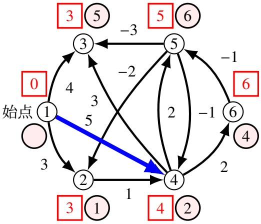
ベルマン・フォード法の例



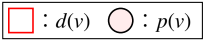
0 回目の反復



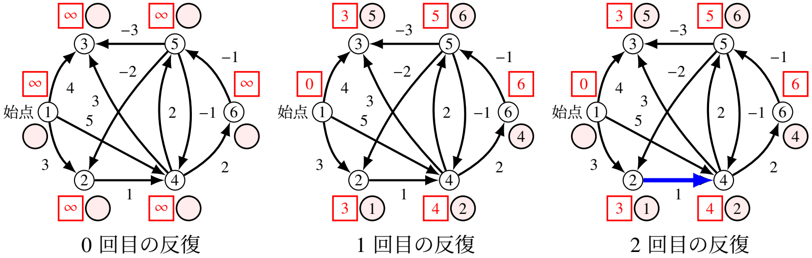
1 回目の反復



2 回目の反復

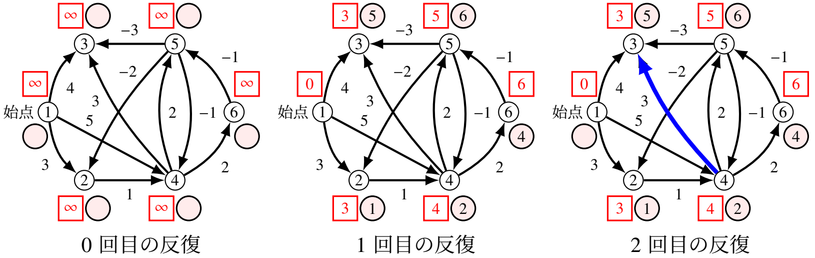


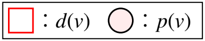
ベルマン・フォード法の例



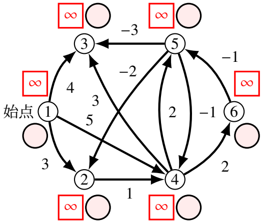
□ : $d(v)$ ○ : $p(v)$

ベルマン・フォード法の例

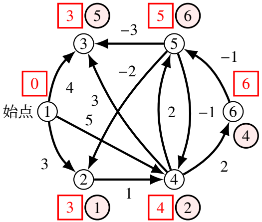




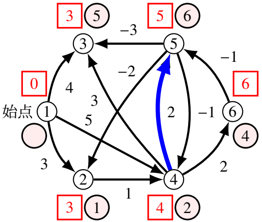
ベルマン・フォード法の例



0 回目の反復



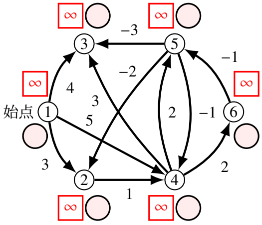
1 回目の反復



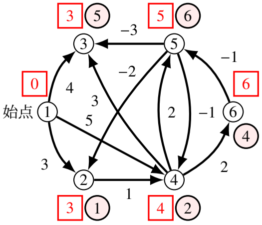
2 回目の反復

□：$d(v)$ ○：$p(v)$

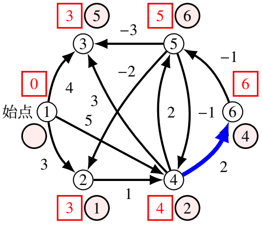
ベルマン・フォード法の例



0 回目の反復



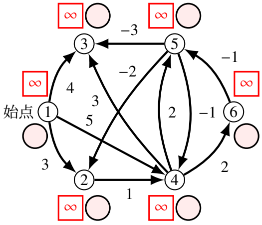
1 回目の反復



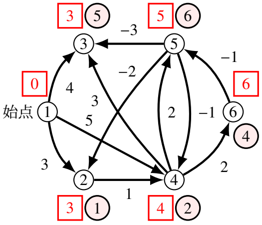
2 回目の反復

□ ：$d(v)$ ○ ：$p(v)$

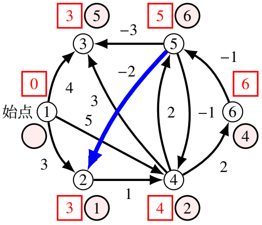
ベルマン・フォード法の例



0 回目の反復



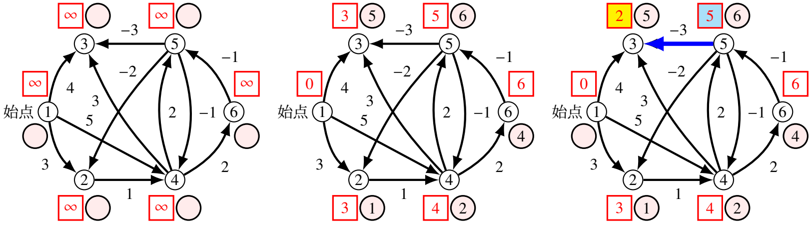
1 回目の反復



2 回目の反復

□ : $d(v)$ ○ : $p(v)$

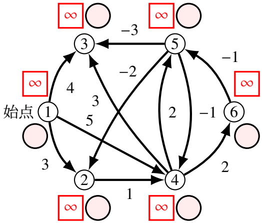
ベルマン・フォード法の例



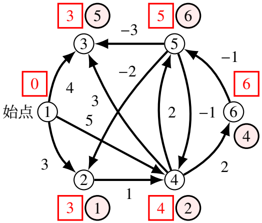
0 回目の反復　　1 回目の反復　　2 回目の反復

□ : $d(v)$　○ : $p(v)$

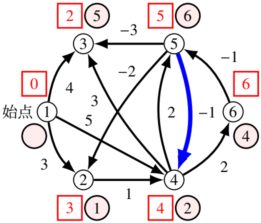
ベルマン・フォード法の例



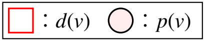
0 回目の反復



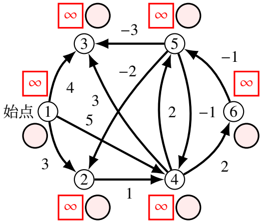
1 回目の反復



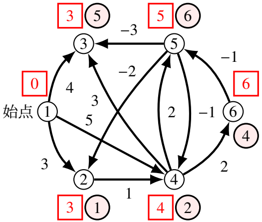
2 回目の反復



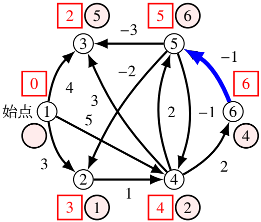
ベルマン・フォード法の例



0 回目の反復



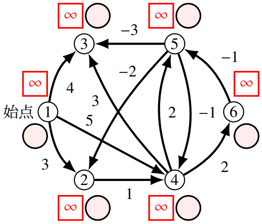
1 回目の反復



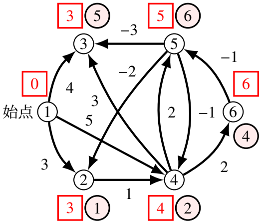
2 回目の反復

□ : $d(v)$ ○ : $p(v)$

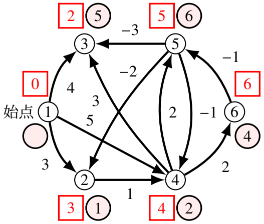
ベルマン・フォード法の例



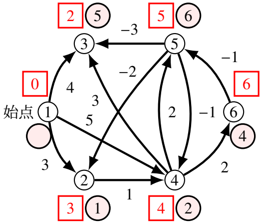
0 回目の反復



1 回目の反復



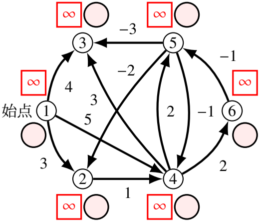
2 回目の反復



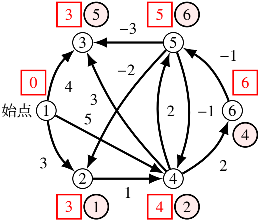
3 回目の反復

□ : $d(v)$ ○ : $p(v)$

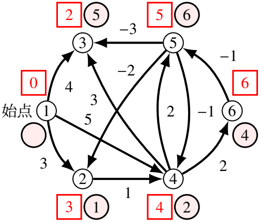
ベルマン・フォード法の例



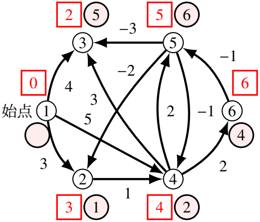
0 回目の反復



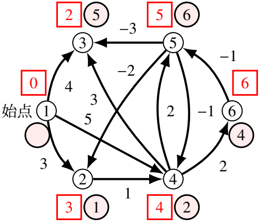
1 回目の反復



2 回目の反復



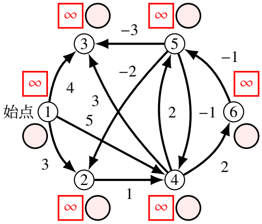
3 回目の反復



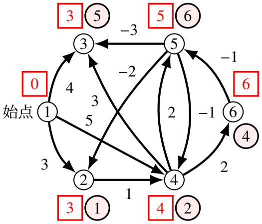
4 回目の反復

□ : $d(v)$ ○ : $p(v)$

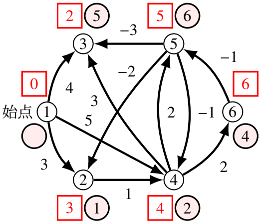
ベルマン・フォード法の例



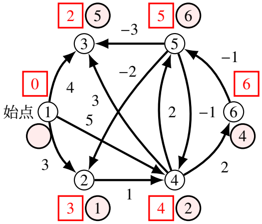
0 回目の反復



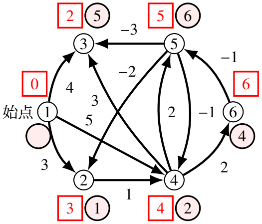
1 回目の反復



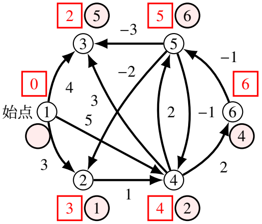
2 回目の反復



3 回目の反復



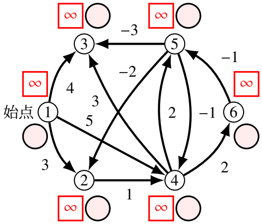
4 回目の反復



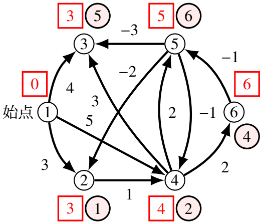
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

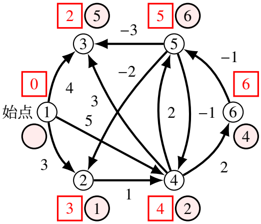
ベルマン・フォード法の例



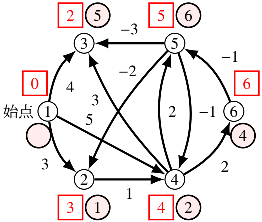
0 回目の反復



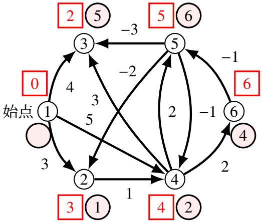
1 回目の反復



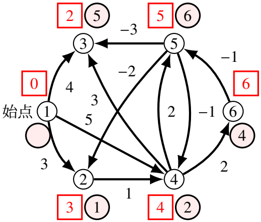
2 回目の反復



3 回目の反復



4 回目の反復

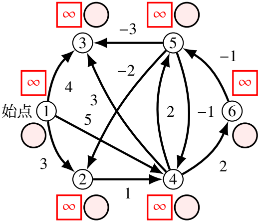


5 回目の反復

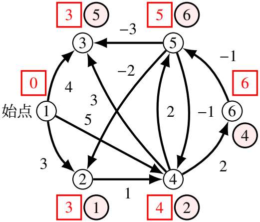
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

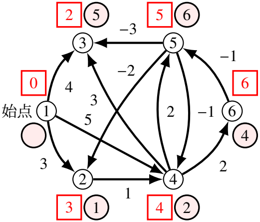
ベルマン・フォード法の例



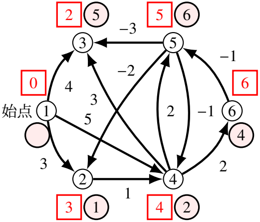
0 回目の反復



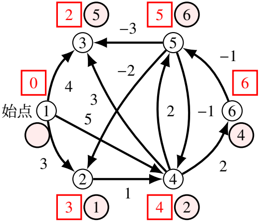
1 回目の反復



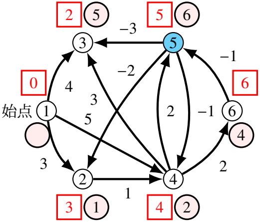
2 回目の反復



3 回目の反復



4 回目の反復



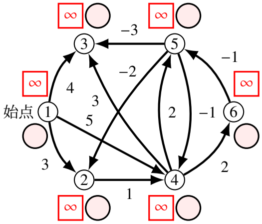
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

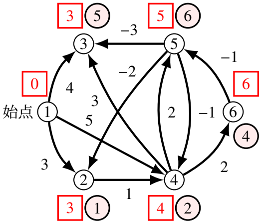
頂点 5 までの最短経路

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

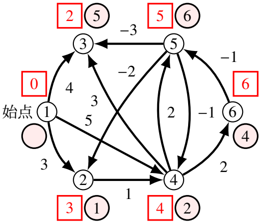
ベルマン・フォード法の例



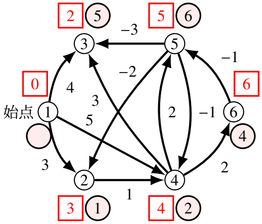
0 回目の反復



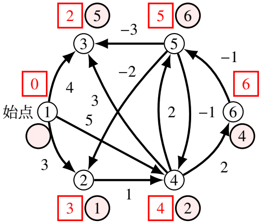
1 回目の反復



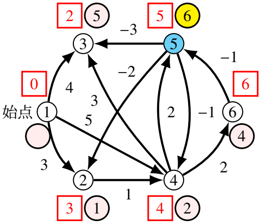
2 回目の反復



3 回目の反復



4 回目の反復



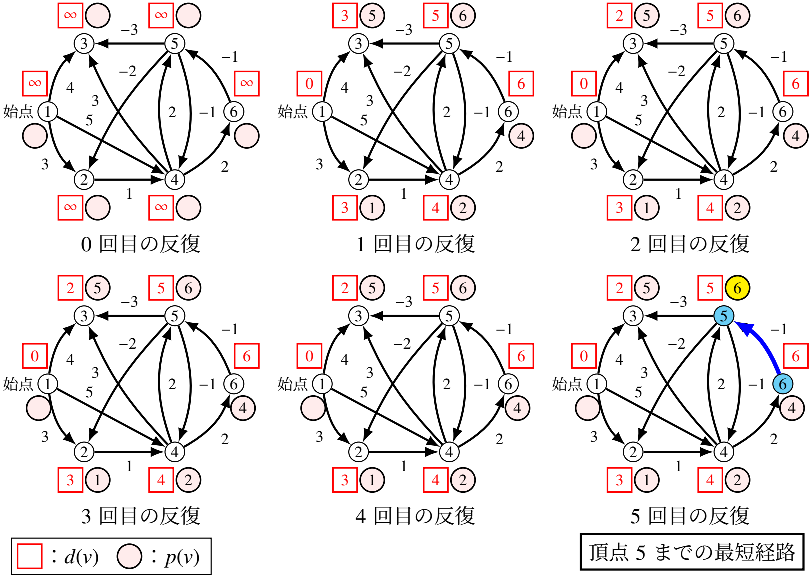
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

頂点 5 までの最短経路

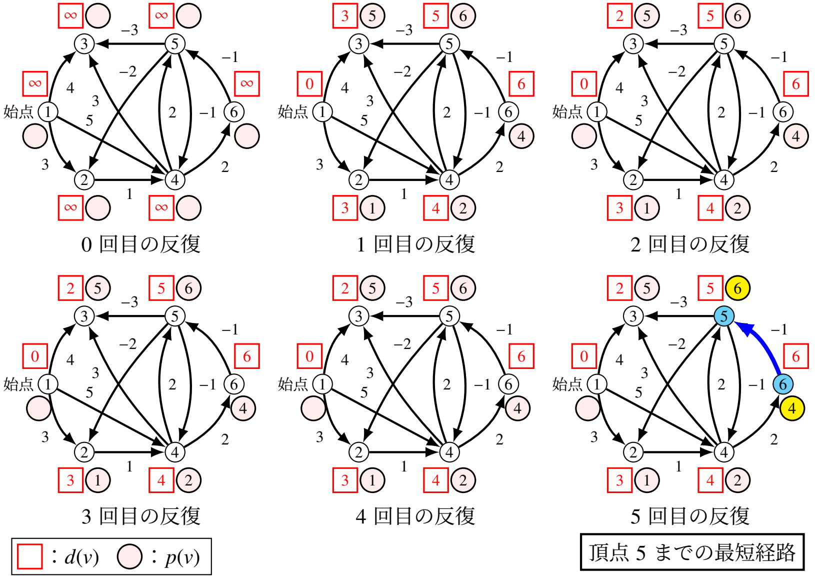
$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

ベルマン・フォード法の例



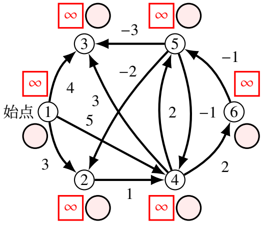
$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

ベルマン・フォード法の例

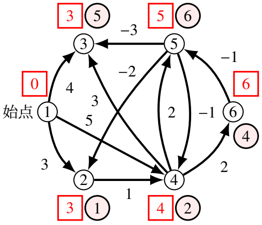


$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

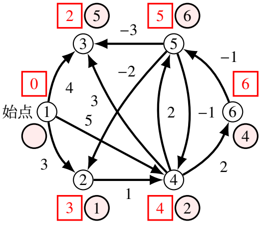
ベルマン・フォード法の例



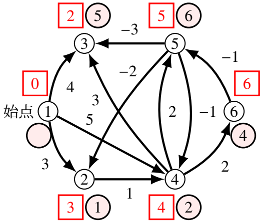
0 回目の反復



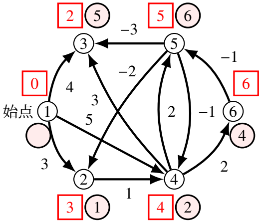
1 回目の反復



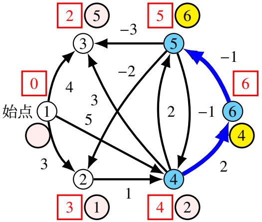
2 回目の反復



3 回目の反復



4 回目の反復



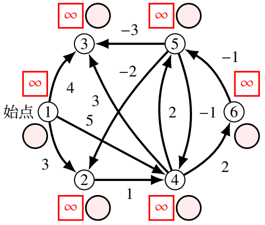
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

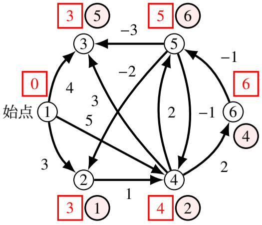
頂点 5 までの最短経路

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

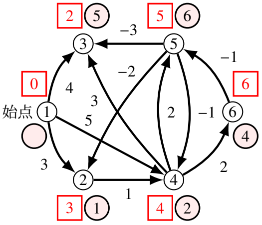
ベルマン・フォード法の例



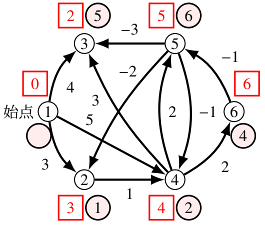
0 回目の反復



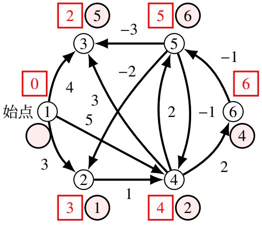
1 回目の反復



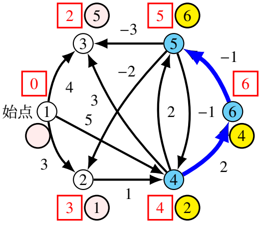
2 回目の反復



3 回目の反復



4 回目の反復



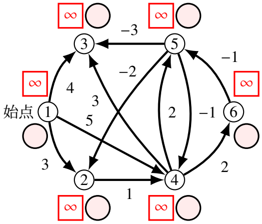
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

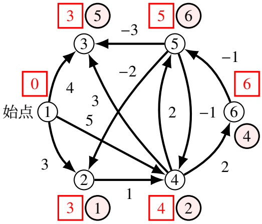
頂点 5 までの最短経路

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

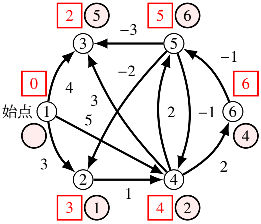
ベルマン・フォード法の例



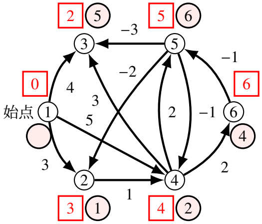
0 回目の反復



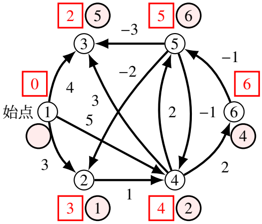
1 回目の反復



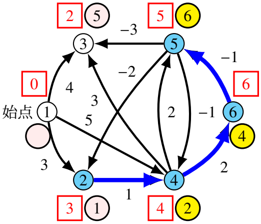
2 回目の反復



3 回目の反復



4 回目の反復



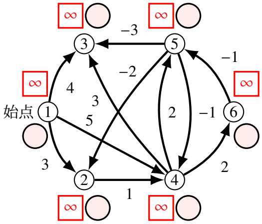
5 回目の反復

□ : $d(v)$　○ : $p(v)$

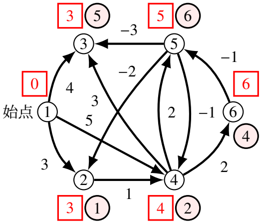
頂点 5 までの最短経路

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

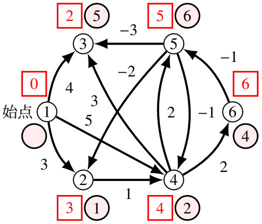
ベルマン・フォード法の例



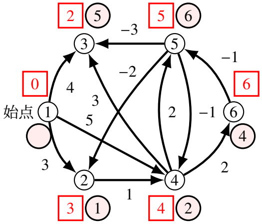
0 回目の反復



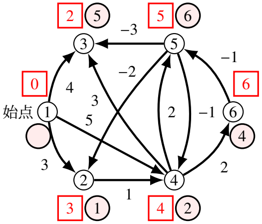
1 回目の反復



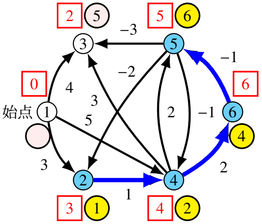
2 回目の反復



3 回目の反復



4 回目の反復



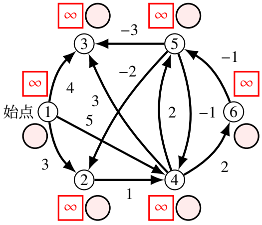
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

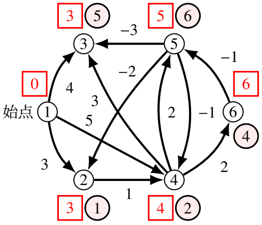
頂点 5 までの最短経路

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

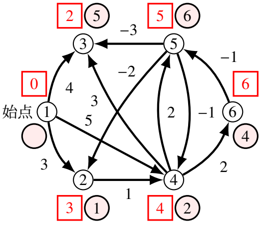
ベルマン・フォード法の例



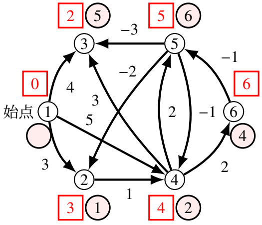
0 回目の反復



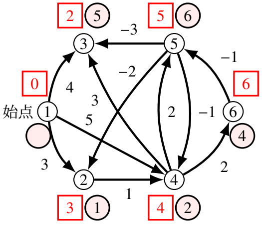
1 回目の反復



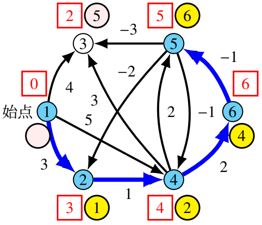
2 回目の反復



3 回目の反復



4 回目の反復



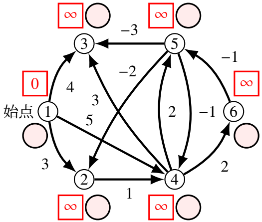
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

頂点 5 までの最短経路

$d(v)$ が変化しなかった 3 回目の反復後に終了してもよい

ベルマン・フォード法の例 (続き)

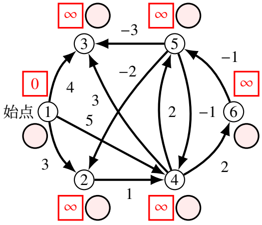


0 回目の反復

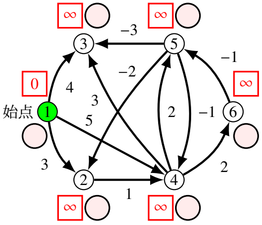
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)



0 回目の反復

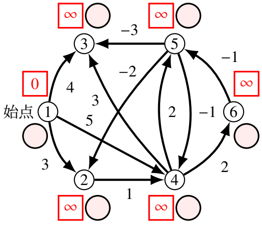


1 回目の反復

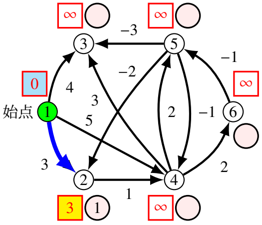
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)



0 回目の反復

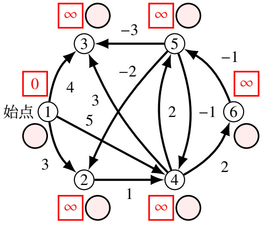


1 回目の反復

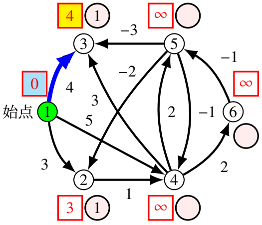
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)



0 回目の反復

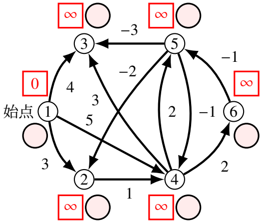


1 回目の反復

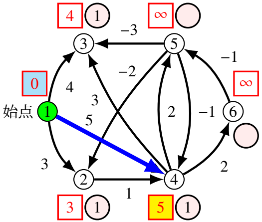
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)



0 回目の反復

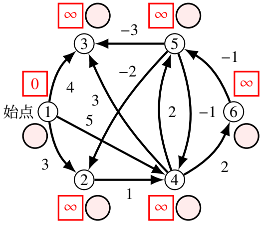


1 回目の反復

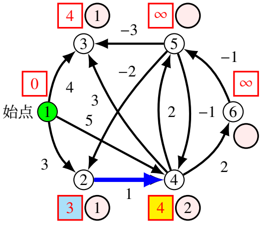
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

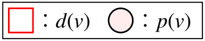
ベルマン・フォード法の例 (続き)



0 回目の反復

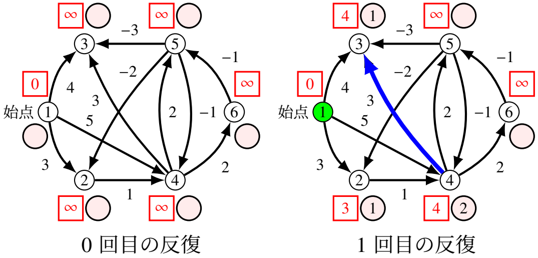


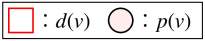
1 回目の反復



$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

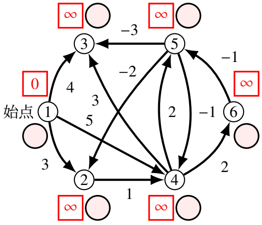
ベルマン・フォード法の例 (続き)

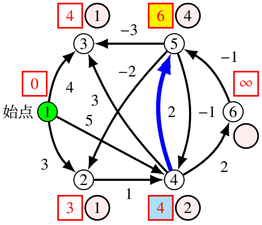




$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)

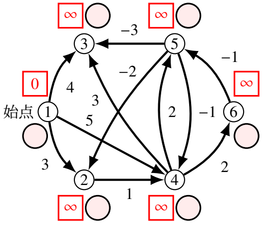




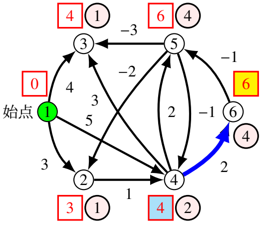
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)



0 回目の反復

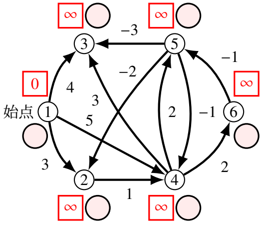


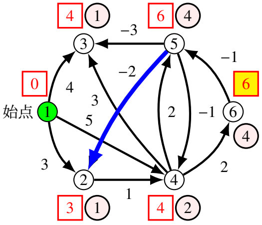
1 回目の反復

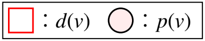
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)

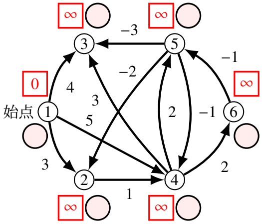




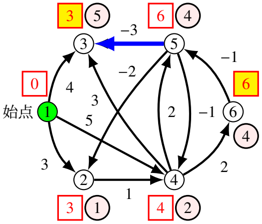


$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)



0 回目の反復

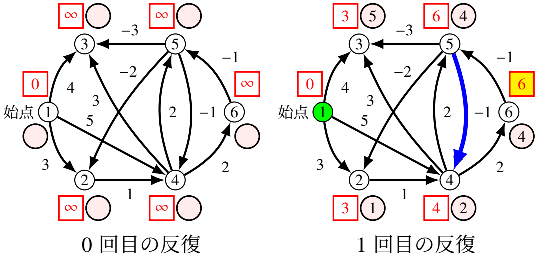


1 回目の反復

□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

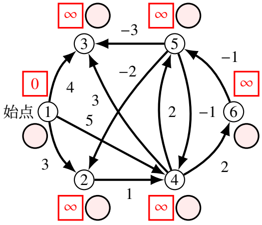
ベルマン・フォード法の例 (続き)



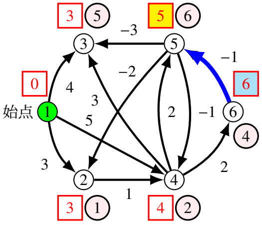
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

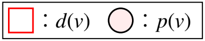
ベルマン・フォード法の例 (続き)



0 回目の反復

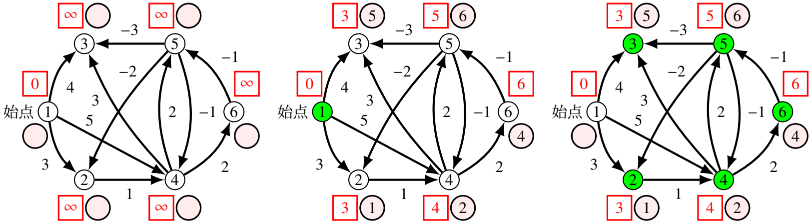


1 回目の反復



$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)

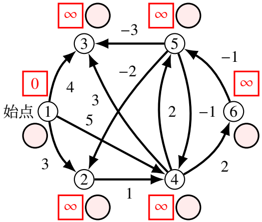


0 回目の反復　　1 回目の反復　　2 回目の反復

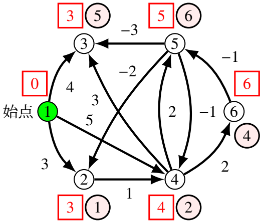
□ : $d(v)$　○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

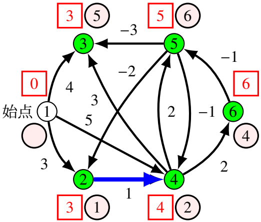
ベルマン・フォード法の例 (続き)



0 回目の反復



1 回目の反復

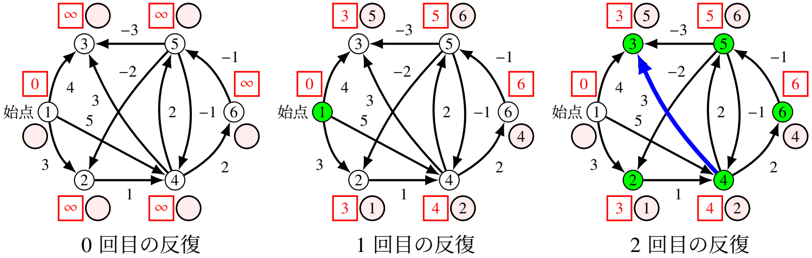


2 回目の反復

□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

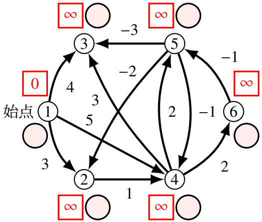
ベルマン・フォード法の例 (続き)



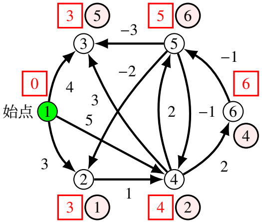
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

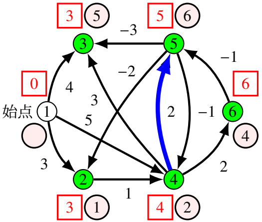
ベルマン・フォード法の例 (続き)



0 回目の反復



1 回目の反復

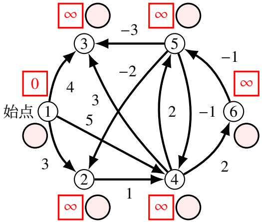


2 回目の反復

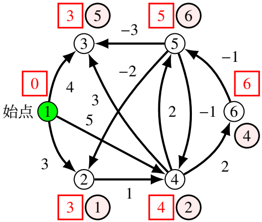
□：$d(v)$ ○：$p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

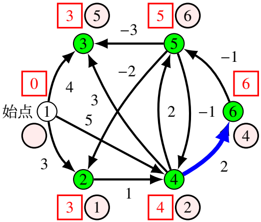
ベルマン・フォード法の例 (続き)



0 回目の反復



1 回目の反復

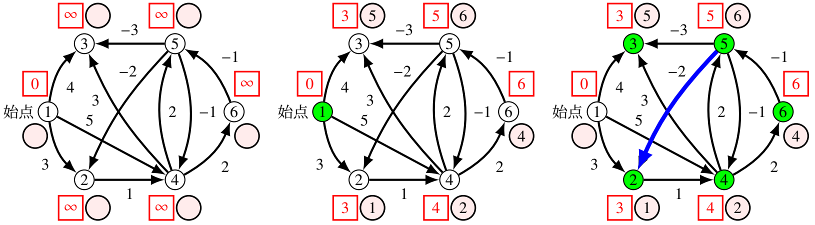


2 回目の反復

□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の例 (続き)

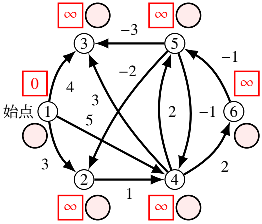


0 回目の反復　　1 回目の反復　　2 回目の反復

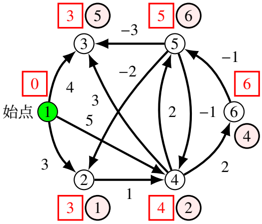
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

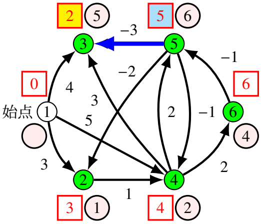
ベルマン・フォード法の例 (続き)



0 回目の反復



1 回目の反復

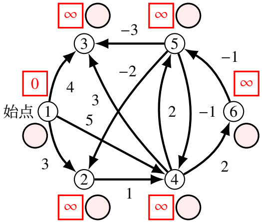


2 回目の反復

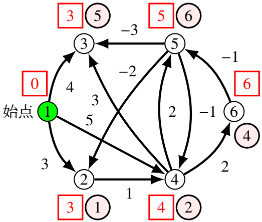
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

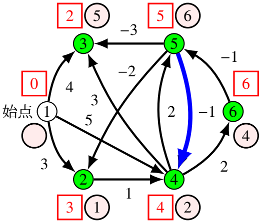
ベルマン・フォード法の例 (続き)



0 回目の反復



1 回目の反復

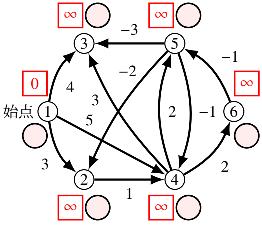


2 回目の反復

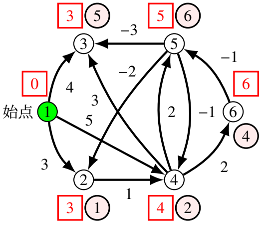
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

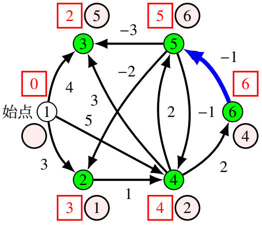
ベルマン・フォード法の例 (続き)



0 回目の反復



1 回目の反復

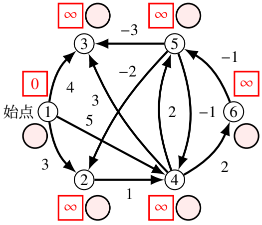


2 回目の反復

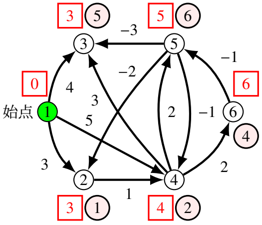
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

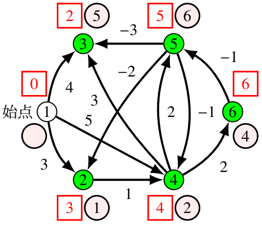
ベルマン・フォード法の例 (続き)



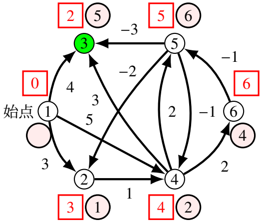
0 回目の反復



1 回目の反復



2 回目の反復



3 回目の反復

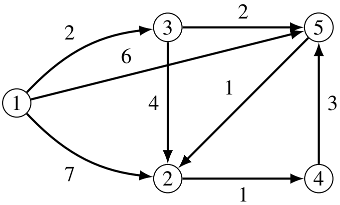
□ : $d(v)$ ○ : $p(v)$

$d(v)$ が変化した頂点 (から出ていく辺) だけ調べれば十分

ベルマン・フォード法の練習問題

練習問題

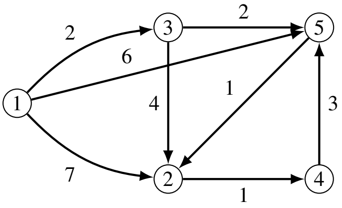
ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める

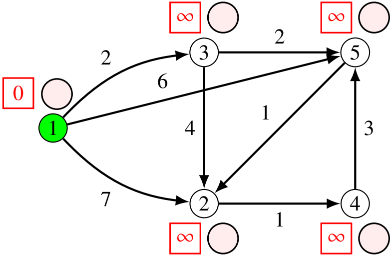


ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



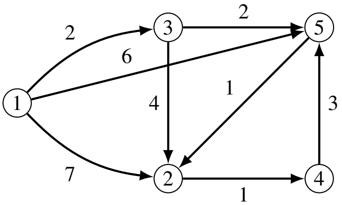


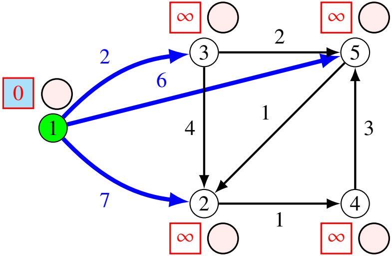
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める

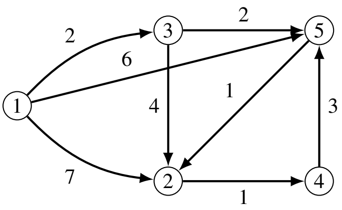


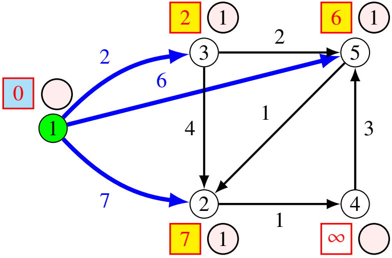


1 回目の反復

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



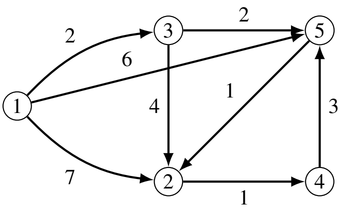


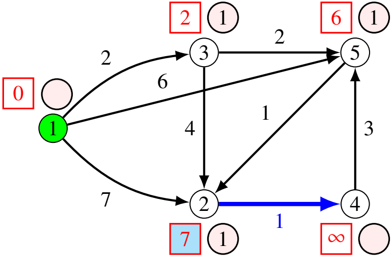
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



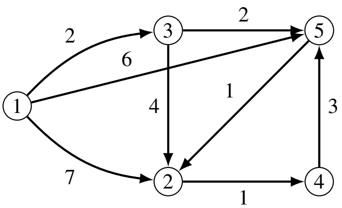


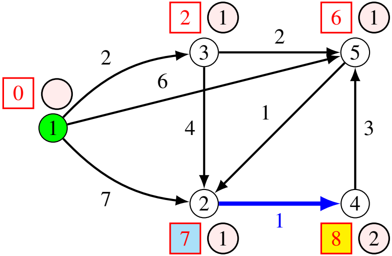
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



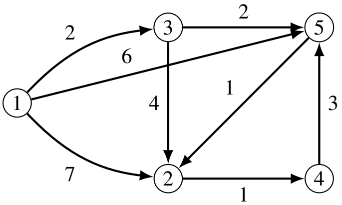


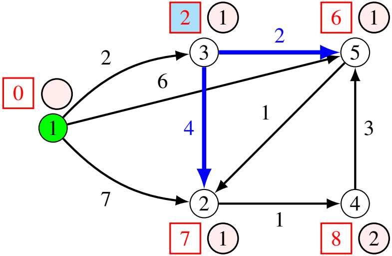
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



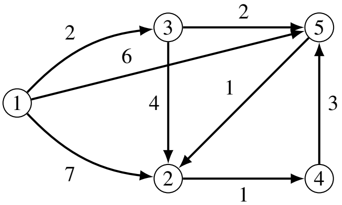


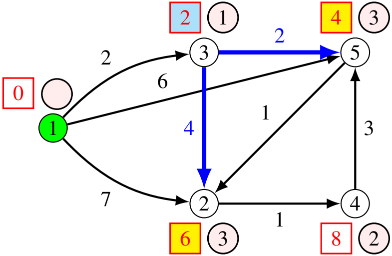
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



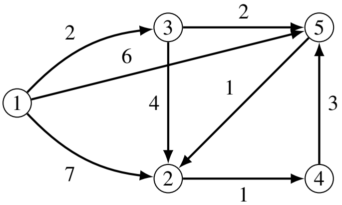


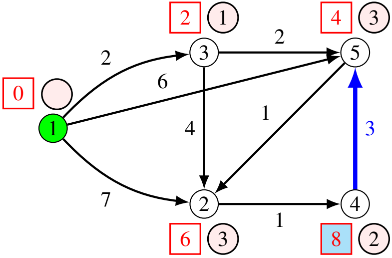
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



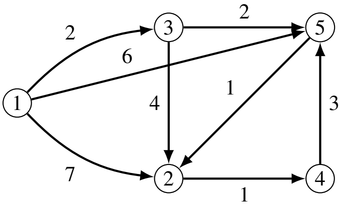


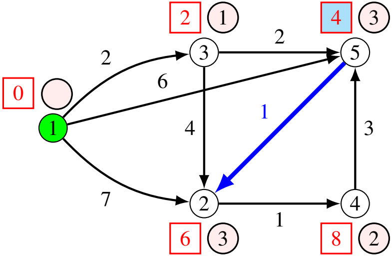
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



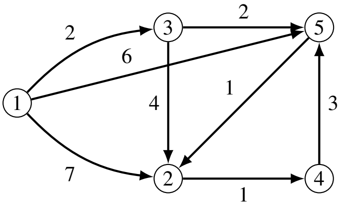


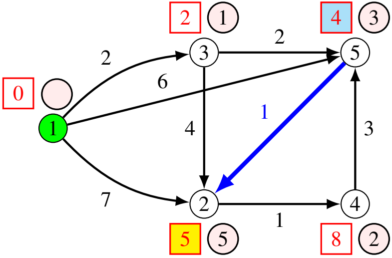
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



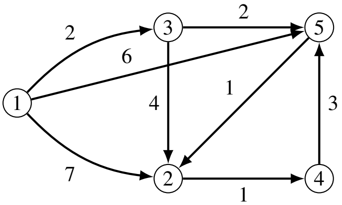


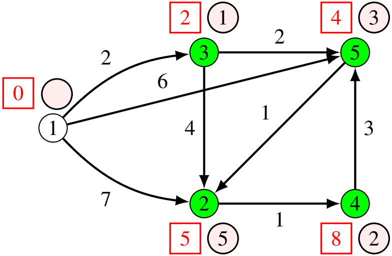
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



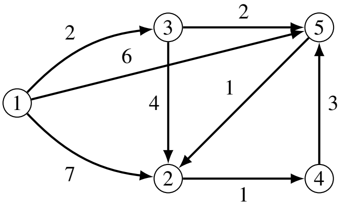


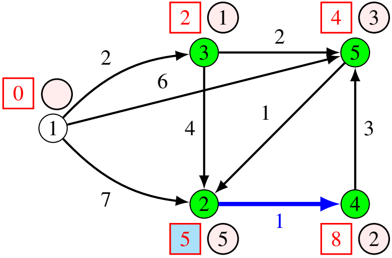
1 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



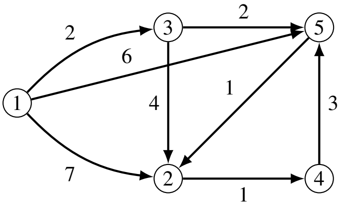


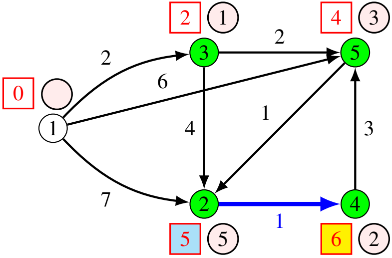
2 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



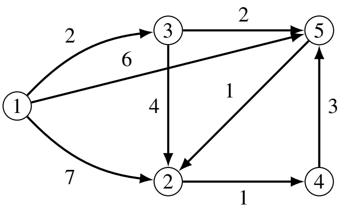


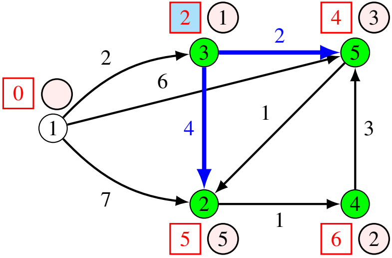
2 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



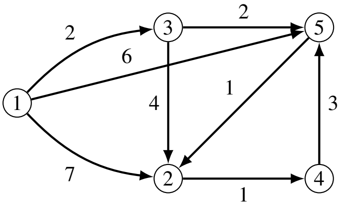


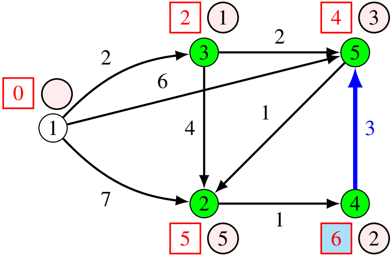
2 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



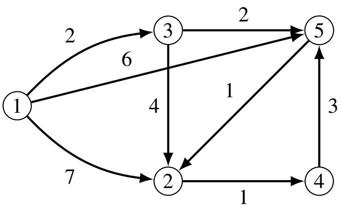


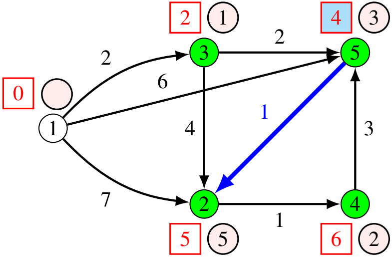
2 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める

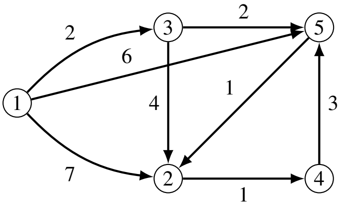


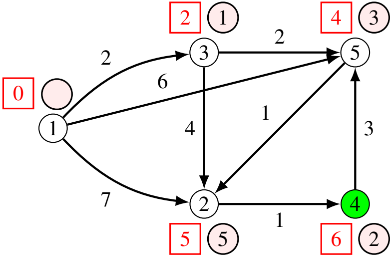


2 回目の反復

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



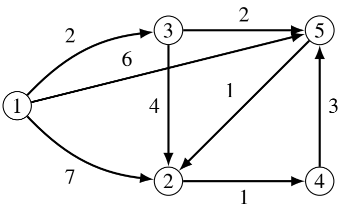


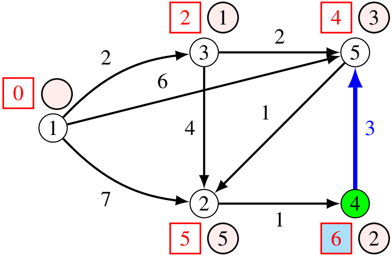
3 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める



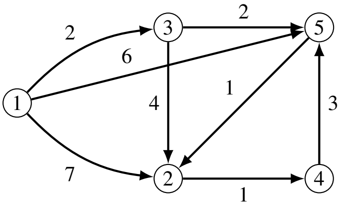


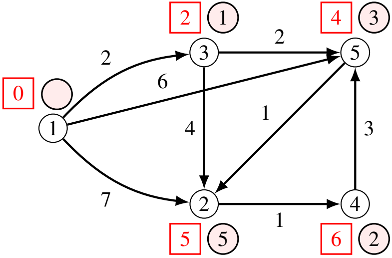
3 回目の反復

ベルマン・フォード法の練習問題

練習問題

ベルマン・フォード法で頂点 1 から他の頂点への最短経路を求める





3 回目の反復

終点	最短距離	最短経路
2	5	$(1,3,5,2)$
3	2	$(1,3)$
4	6	$(1,3,5,2,4)$
5	4	$(1,3,5)$

ダイクストラ法

ダイクストラ法 (Dijkstra's algorithm)

- 計算量 $O(|V|^2)$．さらに工夫すれば $O((|E| + |V|) \log |V|)$ や $O(|E| + |V| \log |V|)$
- とくに辺の数が少ないとき，ベルマン・フォード法より高速
- ただし，重みが負の辺が存在する場合は適用できない

ダイクストラ法の擬似コード

procedure Dijkstra(V, E, w, s)
  for all $v \in V$ **do**
    $d(v) \leftarrow \infty$ ▷ $d(v)$ を ∞ で初期化
  $d(s) \leftarrow 0$ ▷ 始点は 0
  $S \leftarrow \emptyset$ ▷ 最短経路が求まった頂点集合
  $Q \leftarrow \{s\}$ ▷ S に追加する頂点候補の集合
  while $Q \neq \emptyset$ **do**
    $u \leftarrow$ (Q の頂点で d 最小のもの)
    $S \leftarrow S \cup \{u\}$ ▷ S に u を追加
    $Q \leftarrow Q \setminus \{u\}$ ▷ Q から u を削除
    for all $(u, v) \in E$ **do** ▷ u から出る辺
      if $d(v) > d(u) + w(u, v)$ **then**
        $d(v) \leftarrow d(u) + w(u, v)$ ▷ 緩和手続き
        if $v \notin Q$ **then**
          $Q \leftarrow Q \cup \{v\}$ ▷ u から到達可能な v を Q に追加

ダイクストラ法 (続き)

ダイクストラ法の擬似コード

1: **procedure** DIJKSTRA(V, E, w, s)

2: **for all** $v \in V$ **do**

3: $d(v) \leftarrow \infty$ ▹ $d(v)$ を ∞ で初期化

4: $d(s) \leftarrow 0$ ▹ 始点は 0

5: $S \leftarrow \varnothing$ ▹ 最短経路が求まった頂点集合

6: $P \leftarrow \{s\}$ ▹ S に追加する頂点候補の集合

7: **while** $P \neq \varnothing$ **do**

8: $u \leftarrow$ (P の頂点で d 最小のもの)

9: $S \leftarrow S \cup \{u\}$ ▹ S に u を追加

10: $P \leftarrow P \setminus \{u\}$ ▹ P から u を削除

11: **for all** $(u, v) \in E$ **do** ▹ u から出る辺

12: **if** $d(v) > d(u) + w(u, v)$ **then**

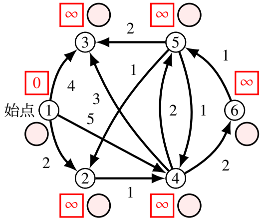
13: $d(v) \leftarrow d(u) + w(u, v)$ ▹ 緩和手続き

14: **if** $v \notin P$ **then**

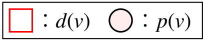
15: $P \leftarrow P \cup \{v\}$ ▹ u から到達可能な v を P に追加

- 始点 s から近い ($d(v)$ が小さい) 頂点から順に最短距離が求まる
- 最終的な $d(v)$ の値が頂点 v までの最短距離 ($d(v) = \infty$ の場合は経路が存在しない)
- 最短経路を求める場合，頂点 v の直前に訪れる頂点を $p(v)$ として，13 行目の次に「$p(v) \leftarrow u$」を追加する
- ある頂点 w までの最短経路だけ求めたい場合，頂点 w が S に追加された時点で終了してもよい

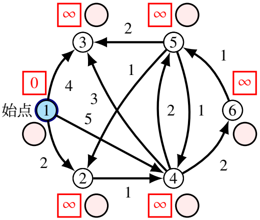
ダイクストラ法の例



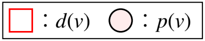
1 回目の反復



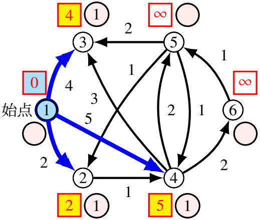
ダイクストラ法の例



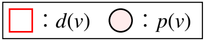
1 回目の反復



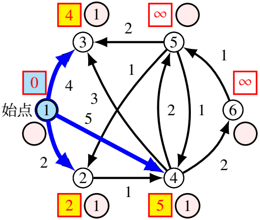
ダイクストラ法の例



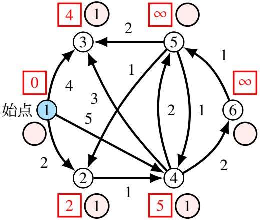
1 回目の反復



ダイクストラ法の例



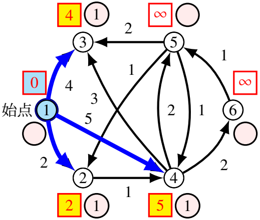
1 回目の反復



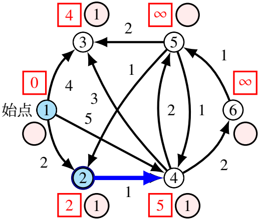
2 回目の反復

□：$d(v)$ ○：$p(v)$

ダイクストラ法の例



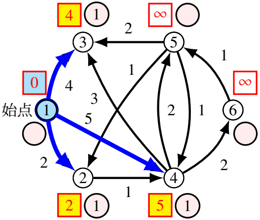
1 回目の反復



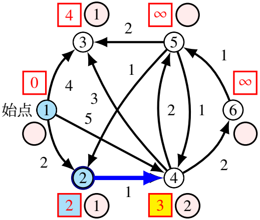
2 回目の反復

□ : $d(v)$ ○ : $p(v)$

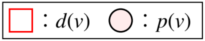
ダイクストラ法の例



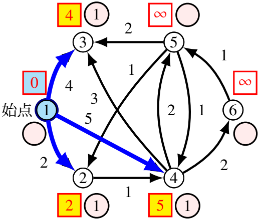
1 回目の反復



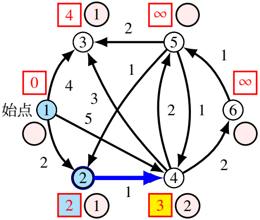
2 回目の反復



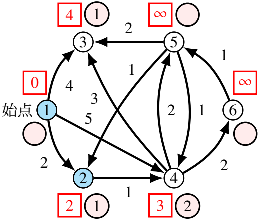
ダイクストラ法の例



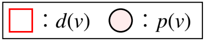
1 回目の反復



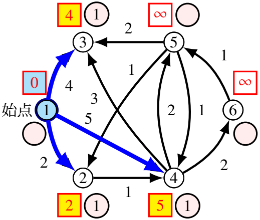
2 回目の反復



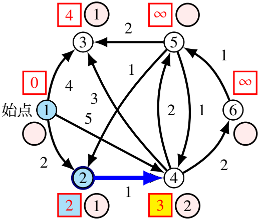
3 回目の反復



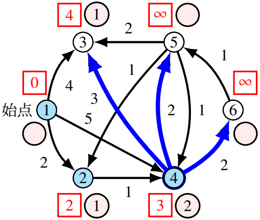
ダイクストラ法の例



1 回目の反復



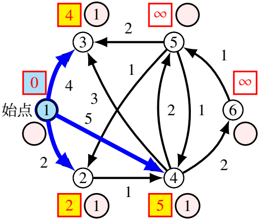
2 回目の反復



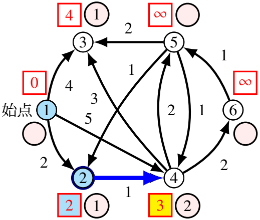
3 回目の反復

□ : $d(v)$ ○ : $p(v)$

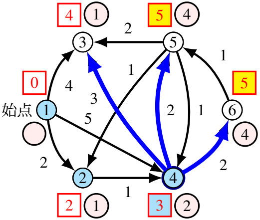
ダイクストラ法の例



1 回目の反復



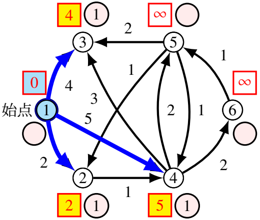
2 回目の反復



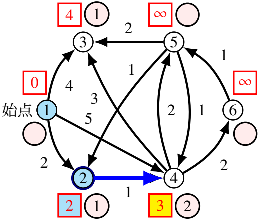
3 回目の反復

□ : $d(v)$　○ : $p(v)$

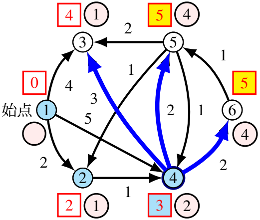
ダイクストラ法の例



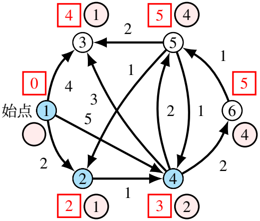
1 回目の反復



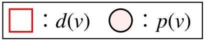
2 回目の反復



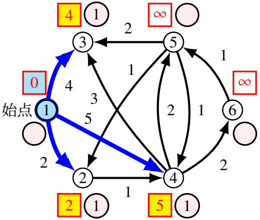
3 回目の反復



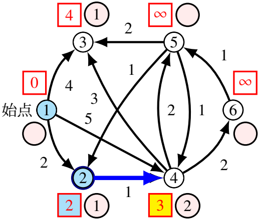
4 回目の反復



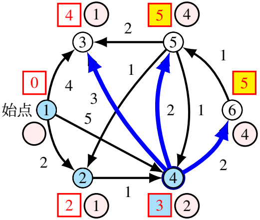
ダイクストラ法の例



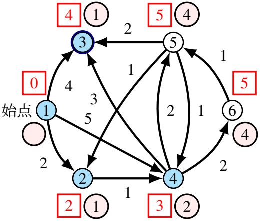
1 回目の反復



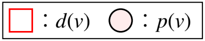
2 回目の反復



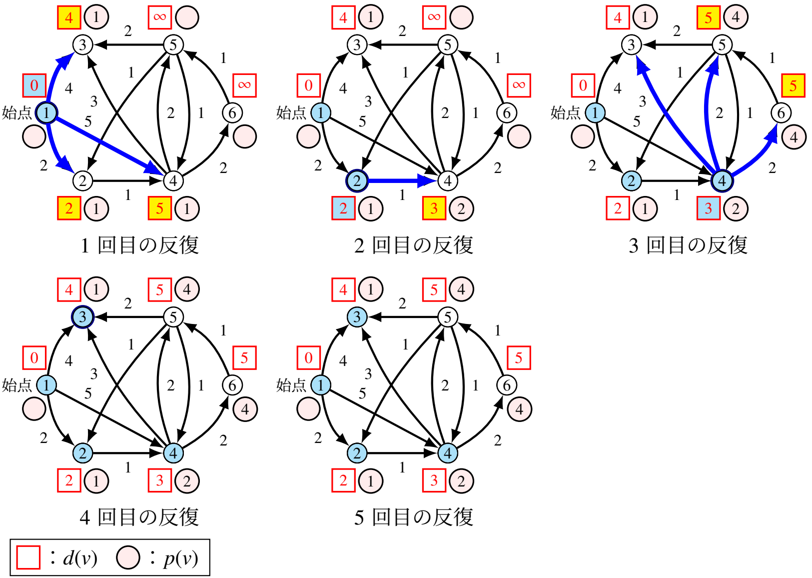
3 回目の反復



4 回目の反復



ダイクストラ法の例



1 回目の反復

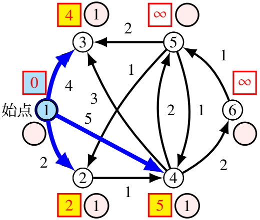
2 回目の反復

3 回目の反復

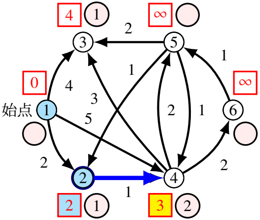
4 回目の反復

5 回目の反復

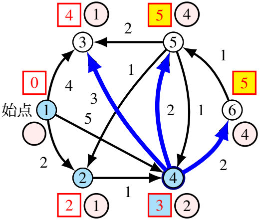
ダイクストラ法の例



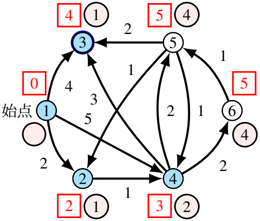
1 回目の反復



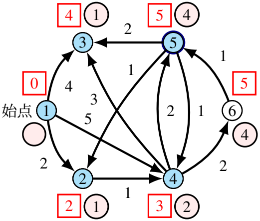
2 回目の反復



3 回目の反復



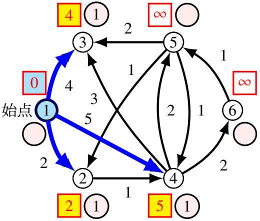
4 回目の反復



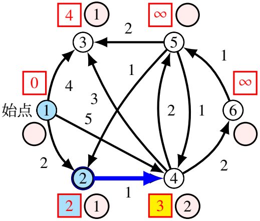
5 回目の反復

□ : $d(v)$ ○ : $p(v)$

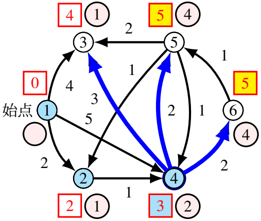
ダイクストラ法の例



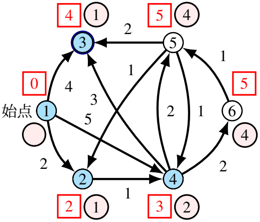
1 回目の反復



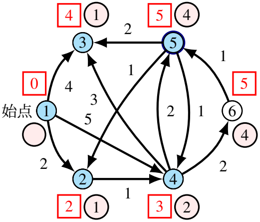
2 回目の反復



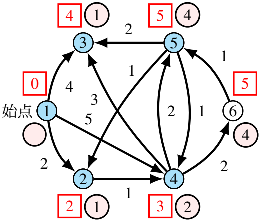
3 回目の反復



4 回目の反復



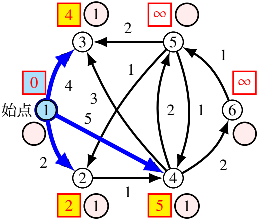
5 回目の反復



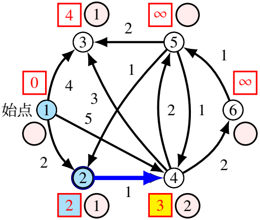
6 回目の反復

□ : $d(v)$ ○ : $p(v)$

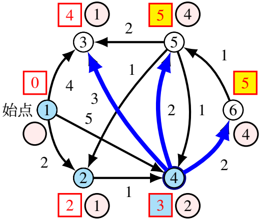
ダイクストラ法の例



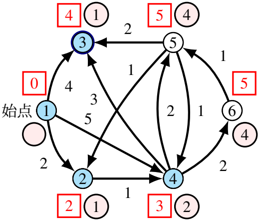
1 回目の反復



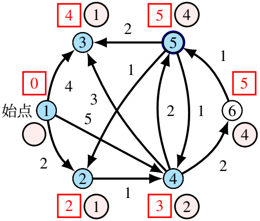
2 回目の反復



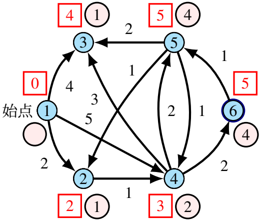
3 回目の反復



4 回目の反復



5 回目の反復



6 回目の反復

□ : $d(v)$ ○ : $p(v)$

ダイクストラ法の正当性

ダイクストラ法終了時の $d(v)$ は，s から v までの最短距離 $\ell(s,v)$ と一致する

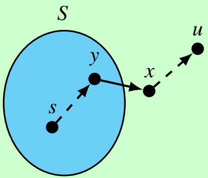
証明 ($|S|$ についての数学的帰納法)

1. $|S| = 1$ のとき，$S = \{s\}$ であり，$d(s) = 0$ は s から s までの最短距離なので成り立つ

2. $|S| = k$ のとき成り立つと仮定して，$|S| = k+1$ のとき成り立つことを示す．d 最小の $u \in V \setminus S$ が S に追加されることから，この u について

$$d(u) = \ell(s,u)$$

を示せばよい．
s から u までの最短経路上で，最初に訪れる S 以外の頂点を $x\ (\in V \setminus S)$，x の直前に訪れる頂点を $y\ (\in S)$ とする．つまり，最短経路は $(s,\ldots,y,x,\ldots,u)$.



(a) $y \in S$ より $d(y) = \ell(s,y)$ だから，s から x までの最短距離は $\ell(s,x) = \ell(s,y) + w(y,x) = d(y) + w(y,x)$.

(b) y を S に追加する際 $d(x)$ を緩和したはずなので，$d(x) = d(y) + w(y,x)$．よって，(a) より $d(x) = \ell(s,x)$

(c) x は u より s に近いので，$\ell(s,x) \leq \ell(s,u)$

(d) d は最短距離の上界なので，$\ell(s,u) \leq d(u)$

(e) u は x より先に S に追加されるので，$d(u) \leq d(x)$

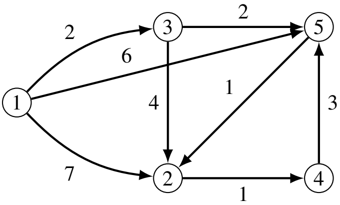
(f) (b)~(d) より，$d(x) = \ell(s,x) \leq \ell(s,u) \leq d(u)$

(g) (e), (f) より，$d(u) = d(x)$ かつ $d(u) = \ell(s,u)$

ダイクストラ法の練習問題

練習問題

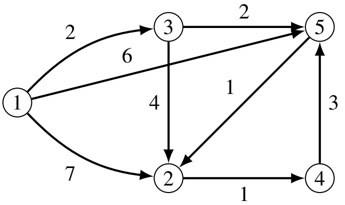
ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

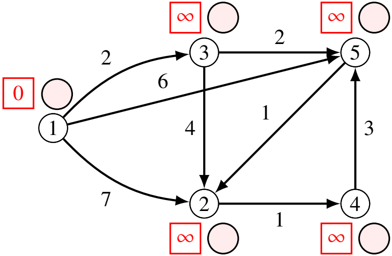


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

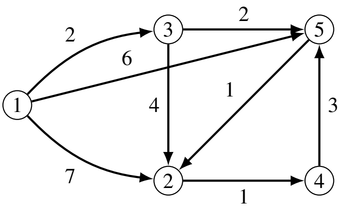


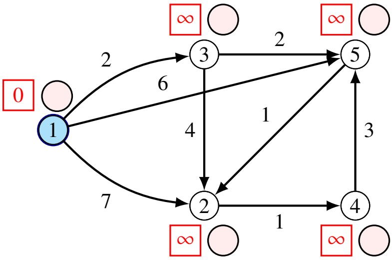


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

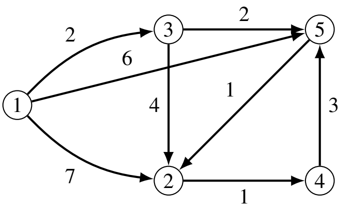


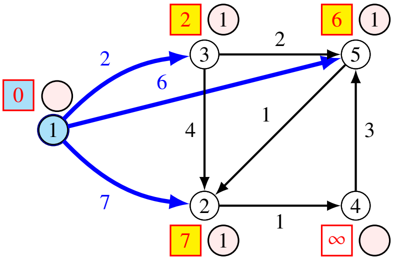


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

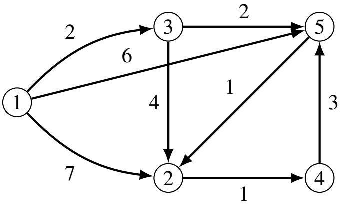


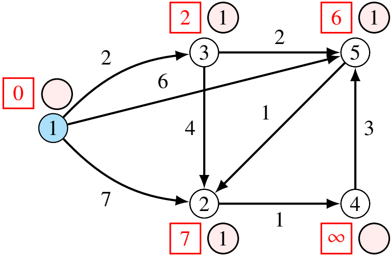


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

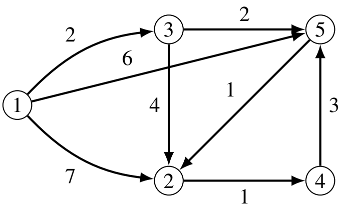


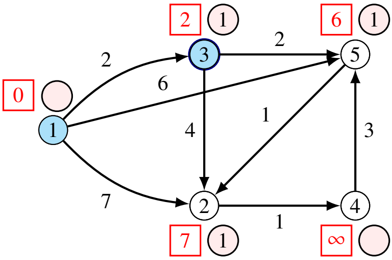


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

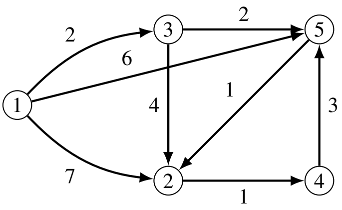


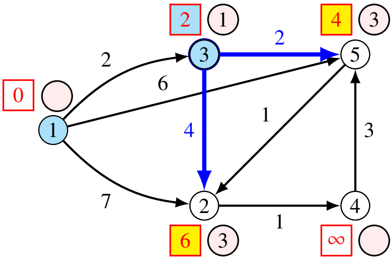


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

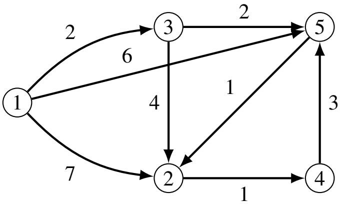


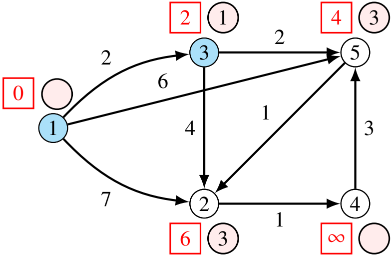


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

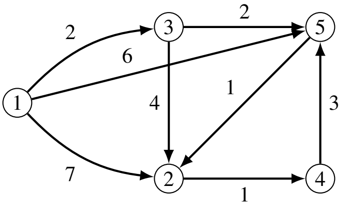


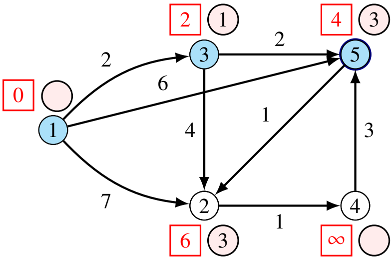


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

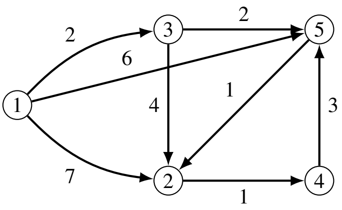


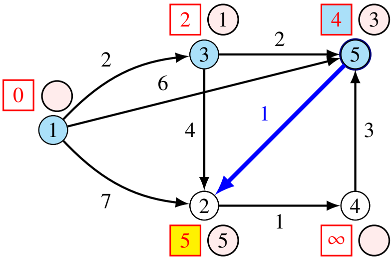


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

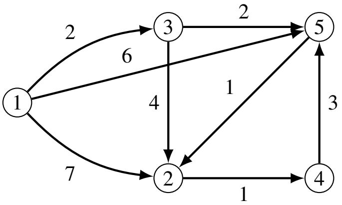


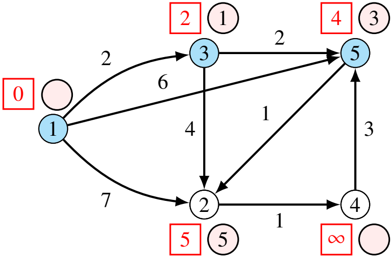


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

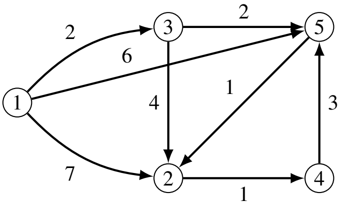


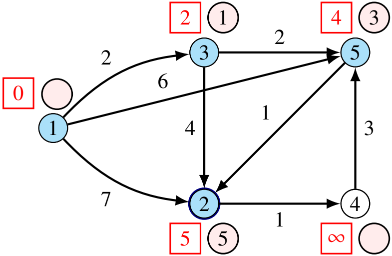


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

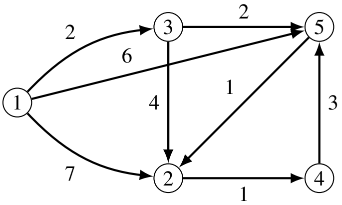


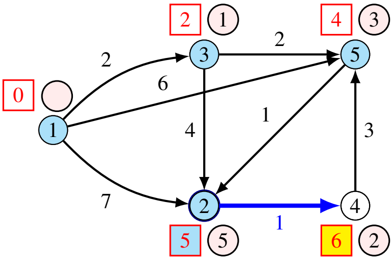


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

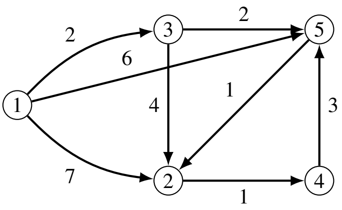


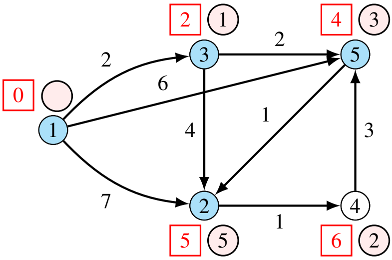


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

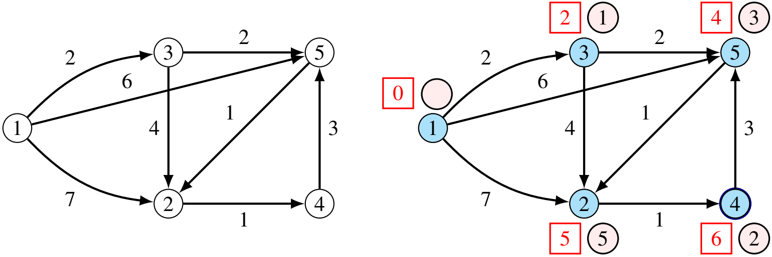




ダイクストラ法の練習問題

練習問題

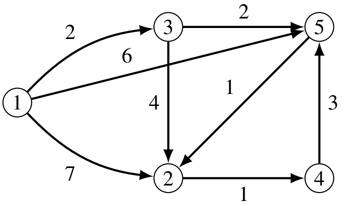
ダイクストラ法で頂点 1 から他の頂点への最短経路を求める

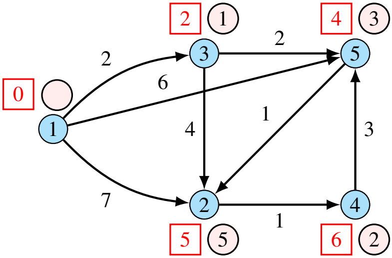


ダイクストラ法の練習問題

練習問題

ダイクストラ法で頂点 1 から他の頂点への最短経路を求める





ベルマン・フォード法と同じ答が求まることを確認

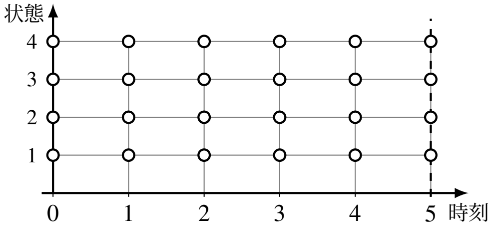
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は,「初期状態と最初の決定がどのようなものであったとしても，残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす.



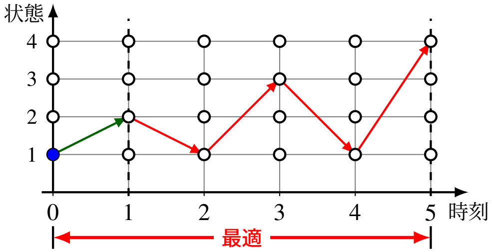
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) **最適な決定**は，「**初期状態**と**最初の決定**がどのようなものであったとしても，残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす．



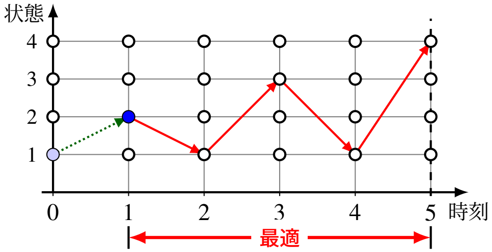
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は，「初期状態と最初の決定がどのようなものであったとしても，**残りの決定**は**最初の決定から生じた状態**に関して**最適**」という性質を満たす．



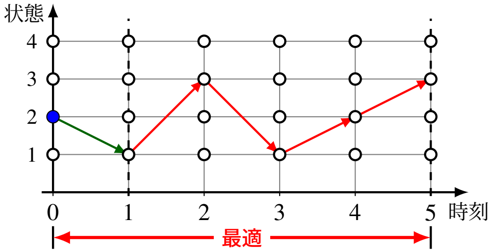
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は，「**初期状態**と**最初の決定**がどのようなものであったとしても，残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす.



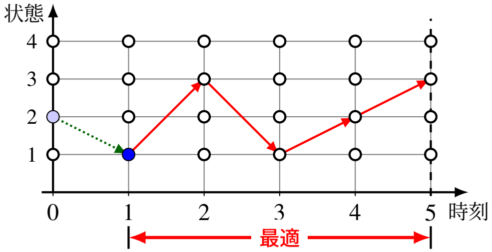
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は,「初期状態と最初の決定がどのようなものであったとしても，**残りの決定**は**最初の決定から生じた状態**に関して**最適**」という性質を満たす.



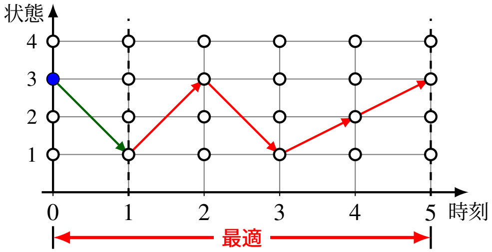
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は,「**初期状態**と**最初の決定**がどのようなものであったとしても，残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす.



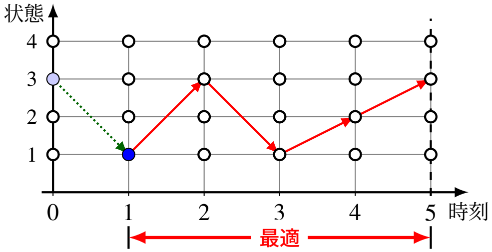
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は，「初期状態と最初の決定がどのようなものであったとしても，**残りの決定**は**最初の決定から生じた状態**に関して**最適**」という性質を満たす．



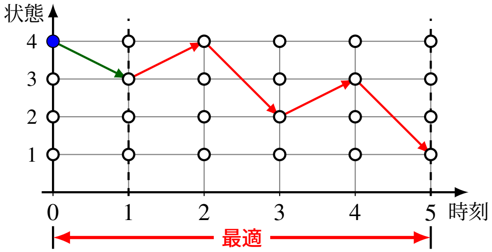
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は，「**初期状態**と**最初の決定**がどのようなものであったとしても，残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす．



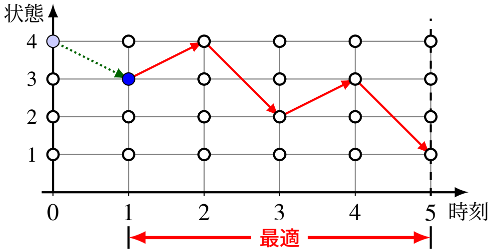
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は,「初期状態と最初の決定がどのようなものであったとしても, **残りの決定**は**最初の決定から生じた状態**に関して**最適**」という性質を満たす.



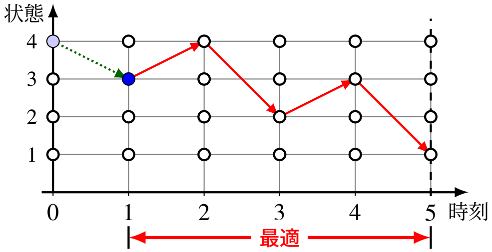
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は，「初期状態と最初の決定がどのようなものであったとしても，残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす．



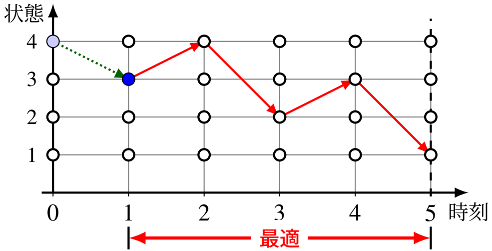
動的計画法と最適性の原理

動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は,「初期状態と最初の決定がどのようなものであったとしても, 残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす.



- 最適性の原理は部分構造最適性の一種
- 時刻 $t = 5$ の状態を始点と見ると, 最短経路問題の最適部分構造を表す

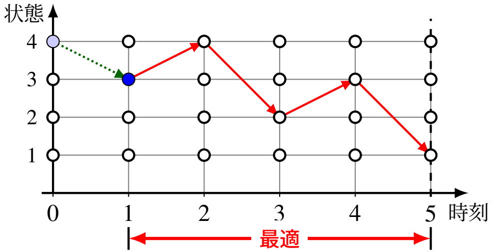
動的計画法と最適性の原理

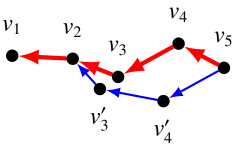
動的計画法 (dynamic programming)

- ベルマン (Richard Bellman) が提案
- **最適性の原理**を満たす種々の問題に適用可能な手法

最適性の原理 (principle of optimality)

(ある期間中の) 最適な決定は,「初期状態と最初の決定がどのようなものであったとしても, 残りの決定は最初の決定から生じた状態に関して最適」という性質を満たす.





v_5 から v_1 までの最短経路 $(v_5, \dots, v_1)$ 上の $(v_5, \dots, v_2)$ は, v_5 から v_2 までの最短経路

- 最適性の原理は部分構造最適性の一種
- 時刻 $t = 5$ の状態を始点と見ると, **最短経路問題の最適部分構造**を表す

動的計画法の例：0-1 ナップサック問題

動的計画法の基本方針

- 最適性の原理を満たす問題の最適解は，部分問題の最適解から構築できる
- 先に部分問題の最適解を求めておくことで，重複する計算を回避

0-1 ナップサック問題

- アイテム i $(1 \leq i \leq n)$ の情報：サイズ a_i および利得 c_i
- ナップサックの容量：C
- サイズの総和が容量 C を超えない範囲で各アイテムを選択し，利得の総和を最大化
- 同じアイテムを重複して選択することはできない

部分問題 $F(i, c)$

- 使用可能なアイテム：$1, \ldots, i$ $(0 \leq i \leq n)$ のみ
- ナップサック容量：c $(0 \leq c \leq C)$ に限定

- アイテム i を選択する
  アイテム $1, \ldots, i-1$ は容量 $c - a_i$ 内で最適に選択．$F(i-1, c-a_i) + c_i$
- アイテム i を選択しない
  アイテム $1, \ldots, i-1$ は容量 c 内で最適に選択．$F(i-1, c)$

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0				
2	0				
3	0				
4	0				
5	0				
6	0				
7	0				
8	0				
9	0				
10	0				

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0 + 8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0			
2	0	0			
3	0	0			
4	0	0			
5	0				
6	0				
7	0				
8	0				
9	0				
10	0				

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0			
2	0	0			
3	0	0			
4	0	0			
5	0				
6	0				
7	0				
8	0				
9	0				
10	0				

$$\begin{aligned}
F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\
&= \max\{F(0,0) + 8, F(0,5)\} \\
&= \max\{0+8, 0\} \\
&= 8
\end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0			
2	0	0			
3	0	0			
4	0	0			
5	0	8			
6	0				
7	0				
8	0				
9	0				
10	0				

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0 + 8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0			
2	0	0			
3	0	0			
4	0	0			
5	0	8			
6	0	8			
7	0	8			
8	0	8			
9	0	8			
10	0	8			

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0 + 8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0		
2	0	0	0		
3	0	0	0		
4	0	0	0		
5	0	8	8		
6	0	8	8		
7	0	8	14		
8	0	8	14		
9	0	8	14		
10	0	8	14		

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	
2	0	0	0	0	
3	0	0	0	0	
4	0	0	0	7	
5	0	8	8	8	
6	0	8	8	8	
7	0	8	14	14	
8	0	8	14	14	
9	0	8	14	15	
10	0	8	14	15	

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	14	14	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	18

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0)+8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	14	14	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	18

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量	アイテム 0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	14	14	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	18

$$\begin{aligned} F(1,5) &= \max\{F(0,5-a_1)+c_1, F(0,5)\} \\ &= \max\{F(0,0)+8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

最適値：$F(4,10) = 18$

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	14	14	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	18

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

最適値：$F(4, 10) = 18$

最適解：

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	14	**14**	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	**18**

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0 + 8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

最適値：$F(4, 10) = 18$

最適解：アイテム 4 選択

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	**14**	**14**	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	**18**

$$\begin{aligned}
F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\
&= \max\{F(0,0) + 8, F(0,5)\} \\
&= \max\{0 + 8, 0\} \\
&= 8
\end{aligned}$$

黄色：アイテムを選択

最適値：$F(4,10) = 18$

最適解：アイテム 4 選択

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	0	**0**	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	**14**	**14**	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	**18**

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1)+c_1, F(0,5)\} \\ &= \max\{F(0,0)+8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

最適値：$F(4,10) = 18$

最適解：アイテム 4 選択，アイテム 2 選択

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i,c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1,c)\}$$

容量 \ アイテム	0	1	2	3	4
0	**0**	**0**	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	**14**	**14**	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	**18**

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0)+8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

最適値：$F(4,10) = 18$

最適解：アイテム 4 選択，アイテム 2 選択

動的計画法の例：0-1 ナップサック問題 (続き)

0-1 ナップサック問題の例題

- アイテム数 $n = 4$
- ナップサック容量 $C = 10$
- 各アイテム i のサイズ a_i，利得 c_i は右表

i	a_i	c_i
1	5	8
2	7	14
3	4	7
4	3	4

$$F(i, c) = \max\{F(i-1, c-a_i) + c_i,\ F(i-1, c)\}$$

容量 \ アイテム	0	1	2	3	4
0	**0**	**0**	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	4
4	0	0	0	7	7
5	0	8	8	8	8
6	0	8	8	8	8
7	0	8	**14**	**14**	14
8	0	8	14	14	14
9	0	8	14	15	15
10	0	8	14	15	**18**

$$\begin{aligned} F(1,5) &= \max\{F(0, 5-a_1) + c_1, F(0,5)\} \\ &= \max\{F(0,0) + 8, F(0,5)\} \\ &= \max\{0+8, 0\} \\ &= 8 \end{aligned}$$

黄色：アイテムを選択

最適値：$F(4, 10) = 18$

最適解：**アイテム 4 選択，アイテム 2 選択**

計算量：$O(nC)$ **(表の要素数)**

動的計画法とダイクストラ法

ダイクストラ法

問題： 頂点 s から他の頂点 (s 含む) までの最短距離を，s から近い順に求める

第 k 反復： 頂点 s から k 番目に近い頂点 v_k までの最短距離を求める

s から $\{v_1, \ldots, v_{k-1}\}$ の頂点だけを通って頂点 $u \in V \setminus \{v_1, \ldots, v_{k-1}\}$ へ向かう最短距離を $\tilde{d}^{(k-1)}(u)$ とすると，$\tilde{d}^{(k-1)}(u)$ は v_j までの最短距離 $\ell(s, v_j)$ と $w(v_j, u)$ の和 (ただし，どの v_j かはわからない) なので，

$$\tilde{d}^{(k-1)}(u) = \min_{1 \le j \le k-1} (\ell(s, v_j) + w(v_j, u)) = \min_{1 \le j \le k-1} (\tilde{d}^{(j-1)}(v_j) + w(v_j, u)) \quad (*)$$

$\tilde{d}^{(k-1)}(u)$ 最小の u について $\tilde{d}^{(k-1)}(u) = \ell(s, u)$ が成り立つので，$v_k := u$ とする．

- $\tilde{d}^{(i)}(u)$ は以下のように変形できる

$$\begin{aligned}
\tilde{d}^{(i)}(u) &= \min_{1 \le j \le i} (\tilde{d}^{(j-1)}(v_j) + w(v_j, u)) \\
&= \min \left\{ \min_{1 \le j \le i-1} (\tilde{d}^{(j-1)}(v_j) + w(v_j, u)), \tilde{d}^{(i-1)}(v_i) + w(v_i, u) \right\} \\
&= \min \left\{ \tilde{d}^{(i-1)}(u), \tilde{d}^{(i-1)}(v_i) + w(v_i, u) \right\}
\end{aligned}$$

- ダイクストラ法の第 k 反復では，$\tilde{d}^{(k-1)}(u)$ から v_k を決定後，$\tilde{d}^{(k)}(u)$ $(u \in V \setminus \{v_1, \ldots, v_k\})$ を計算 (緩和)．いずれも $d(u)$ に格納する

$$\tilde{d}^{(k)}(u) = \min \left\{ \tilde{d}^{(k-1)}(u), \tilde{d}^{(k-1)}(v_k) + w(v_k, u) \right\}$$

$\tilde{d}^{(k)}(u)$ を計算する際，保存した $\tilde{d}^{(k-1)}(u)$ を用いることで重複を回避

最短経路問題

最短経路問題 (shortest path problem)

ネットワーク上で最短路 (辺の重みの和が最小の道) を求める問題

最短経路問題の種類

- 単一点対 (single-pair) 最短経路問題
  2 頂点間の最短路を求める問題
- 単一始点 (single-source) 最短経路問題
  ある頂点から残りすべての頂点への最短路を求める問題
- **全点対 (all-pairs) 最短経路問題**
  すべての 2 頂点の組に対して最短路を求める問題

上の最短経路問題はいずれも多項式時間で求解可能

全点対最短経路問題の解法

- フロイド・ワーシャル法 (Floyd-Warshall algorithm)
- ジョンソン法 (Johnson's algorithm)

フロイド・ワーシャル法

全点対最短経路問題

頂点間の最短距離を与える $|V| \times |V|$ 行列 $D = (d_{ij})$ の要素をすべて求める
(d_{ij}：頂点 i から j への最短距離)

フロイド・ワーシャル法 (Floyd-Warshall algorithm)

- $d^{(k)}(i, j)$：頂点 $\{1, 2, \ldots, k\}$ のみを経由する，頂点 i から頂点 j への最短距離
- $d^{(k)}(i, j)$ を動的計画法により計算 $\Rightarrow d_{ij} = d^{(|V|)}(i, j)$

$d^{(k)}(i, j)$ の計算

- $d^{(k)}(i, j)$ が頂点 k を経由するとき：$d^{(k)}(i, j) = d^{(k-1)}(i, k) + d^{(k-1)}(k, j)$
- $d^{(k)}(i, j)$ が頂点 k を経由しないとき：$d^{(k)}(i, j) = d^{(k-1)}(i, j)$
- どちらが最短距離かわからないので，

$$d^{(k)}(i, j) = \min\left\{d^{(k-1)}(i, j), d^{(k-1)}(i, k) + d^{(k-1)}(k, j)\right\}$$

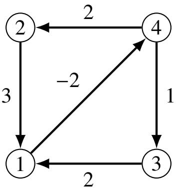
フロイド・ワーシャル法の擬似コード

フロイド・ワーシャル法の擬似コード

procedure FloydWarshall(V, E, w)
  for $i = 1$ **to** $|V|$ **do** ▷ 最短距離の行列を初期化
    $d_{ii}^{(0)} \leftarrow 0$ ▷ 対角成分は 0 にしておく
    for $j = 1$ to $|V|$ **do**
      if $(i, j) \in E$ **then**
        $d_{ij}^{(0)} \leftarrow w(i, j)$ ▷ $(i, j) \in E$ のとき $d_{ij}^{(0)} = w(i, j)$
      else
        $d_{ij}^{(0)} \leftarrow \infty$ ▷ それ以外は $d_{ij}^{(0)} = \infty$
  for $k = 1$ **to** $|V|$ **do**
    for $i = 1$ **to** $|V|$ **do**
      for $j = 1$ **to** $|V|$ **do**
        $d_{ij}^{(k)} \leftarrow \min\left\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right\}$ ▷ 最短距離の行列を更新
  return $\left(d_{ij}^{(|V|)}\right)$

- 計算量は $O(|V|^3)$
- ただし，単純なので比較的高速

フロイド・ワーシャル法の例



$$\begin{array}{c|c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & \infty & \infty & -2 \\ 3 & 0 & \infty & \infty \\ 2 & \infty & 0 & \infty \\ \infty & 2 & 1 & 0 \end{pmatrix} \end{array}$$

(1) $\left(d_{ij}^{(0)}\right)$

$$\begin{array}{c|c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & \infty & \infty & -2 \\ 3 & 0 & \infty & 1 \\ 2 & \infty & 0 & 0 \\ \infty & 2 & 1 & 0 \end{pmatrix} \end{array}$$

(2) $\left(d_{ij}^{(1)}\right)$

$$\begin{array}{c|c} f & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & \infty & \infty & -2 \\ 3 & 0 & \infty & 1 \\ 2 & \infty & 0 & 0 \\ 5 & 2 & 1 & 0 \end{pmatrix} \end{array}$$

(3) $\left(d_{ij}^{(2)}\right)$

$$\begin{array}{c|c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & \infty & \infty & -2 \\ 3 & 0 & \infty & 1 \\ 2 & \infty & 0 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix} \end{array}$$

(4) $\left(d_{ij}^{(3)}\right)$

$$\begin{array}{c|c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 0 & -1 & -2 \\ 3 & 0 & 2 & 1 \\ 2 & 2 & 0 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix} \end{array}$$

(5) $\left(d_{ij}^{(4)}\right)$

$$d^{(k)}(i, j) = \min\left\{d^{(k-1)}(i, j), d^{(k-1)}(i, k) + d^{(k-1)}(k, j)\right\}$$

ジョンソン法

基本的な考え方

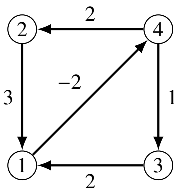
- 各頂点について単一始点最短経路問題を解き，行列 $D = (d_{ij})$ を求める
- 計算量は $|V| \times$ (単一始点最短経路問題の計算量)
- 重みが非負の場合，より高速なダイクストラ法が使える
- 重みが負の場合 (ただし負閉路はなし)，ベルマン・フォード法が必要
  $\Rightarrow$ 非負の重みに変形してダイクストラ法を適用

ジョンソン法 (Johnson's algorithm)

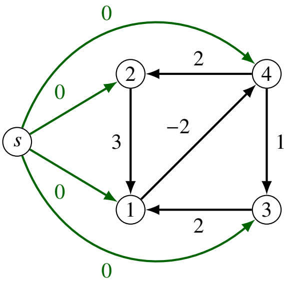
1. $G = (V, E)$ に頂点 s，辺 (s, v) $(v \in V)$ を追加して $G' = (V', E')$ を生成
   $\Rightarrow V' = V \cup \{s\}$，$E' = E \cup \{(s, v) \mid v \in V\}$．重みは $w(s, v) = 0$ $(v \in V)$ とする
2. G' において，s から V への最短経路をベルマン・フォード法で求める †
   $\Rightarrow s$ から v への最短距離を $h(v)$ とする
3. 辺の重みを $w'(u, v) = w(u, v) + h(u) - h(v)$ に変更したグラフ G において，各頂点から残りの頂点までの最短経路をダイクストラ法を $|V|$ 回適用して求める
   $\Rightarrow$ 頂点 i, j 間の最短距離を d'_{ij} とすると，$d_{ij} = d'_{ij} - h(i) + h(j)$

† ここで負閉路も検出可能

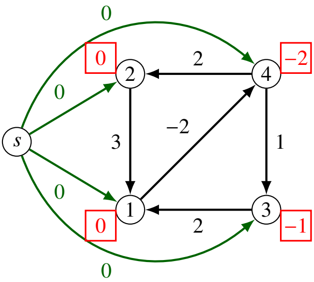
ジョンソン法の例



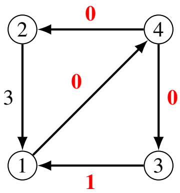
(1) もとのグラフ



(2) s を追加



(3) s からの最短距離 $h(v)$ を求める



(4) 重みを変更

$$
\begin{array}{c c}
 & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 0 & 0 & 0 \\ 3 & 0 & 3 & 3 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}
\end{array}
$$

(5) 各頂点を始点としてダイクストラ法を適用

$$
\begin{array}{c c}
 & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 0 & -1 & -2 \\ 3 & 0 & 2 & 1 \\ 2 & 2 & 0 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix}
\end{array}
$$

(6) もとの重みに戻す
第 i 行から $h(i)$ を引き,
第 j 列に $h(j)$ を足す

ジョンソン法の正当性

$$w'(u,v) = w(u,v) + h(u) - h(v)$$

$w'(u,v)$ の非負性

- 頂点 s から頂点 v までの最短距離は $h(v)$
- 頂点 u を経由する場合の，頂点 s から頂点 v までの最短距離は $h(u) + w(u,v)$

条件つきの方が距離が伸びるので，$h(v) \leq h(u) + w(u,v)$．よって，$w(u,v) + h(u) - h(v) \geq 0$

重み $w(u,v)$ のもとでの最短経路 $\Leftrightarrow$ 重み $w'(u,v)$ のもとでの最短経路

頂点 v_1 から頂点 v_k までの経路 $(v_1, \ldots, v_k)$ の長さ

- 重みが $w(u,v)$ のとき：$\displaystyle\sum_{i=1}^{k-1} w(v_i, v_{i+1})$
- 重みが $w'(u,v)$ のとき：

$$\begin{aligned}\sum_{i=1}^{k-1} w'(v_i, v_{i+1}) &= \sum_{i=1}^{k-1} \{w(v_i, v_{i+1}) + h(v_i) - h(v_{i+1})\} \\ &= \sum_{i=1}^{k-1} w(v_i, v_{i+1}) + h(v_1) - h(v_k)\end{aligned}$$

$(h(v_1) - h(v_k))$ は始点 v_1，終点 v_k が決まれば定数なので，最短経路は影響を受けない