アルゴリズムとデータ構造II

講義5：

最短経路問題

https://elms.u-aizu.ac.jp

仮定：

- しましょう G（V、E、W）で加重グラフになる
  非負 エッジ距離（またはコスト）。
- エッジ用（u、v）、（の距離u、v）で示されます
  d（u、v）。
- パスの距離 P、で示される d（P）、それは
  パス内のエッジの距離の合計。
- 2つのノードの場合 u そして v に G、からの最短経路 u
  に v パスです P そのような
  d（P）$=\min\{d$（Q）$/Q$ からのパスです u に $v\}$。

最短経路上のプロパティ

- しましょう $u \to v \to w$からの最短経路である uに w。
- その後、 $u \to v$からの最短経路です uに vそして
  $v \to w$からの最短経路です vに w。

最短経路問題

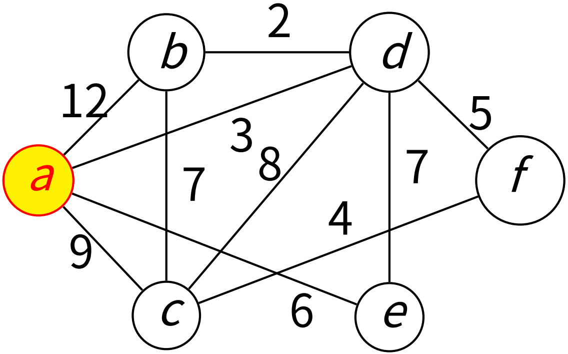
- 単一ソースの最短経路問題 そして すべてのペア 最短経路問題 最も重要な最短経路問題です。

- 単一ソースの最短経路問題 の問題です 特定の頂点からの最短経路を見つける s、と呼ばれる ソース、他のすべての頂点へ G（接続されたグラフ）。
- すべてのペアの最短経路問題 最短を見つけるパス すべてのペアの間 Gの頂点の数（ソースはすべてGの頂点です）。

単一ソースの最短経路問題

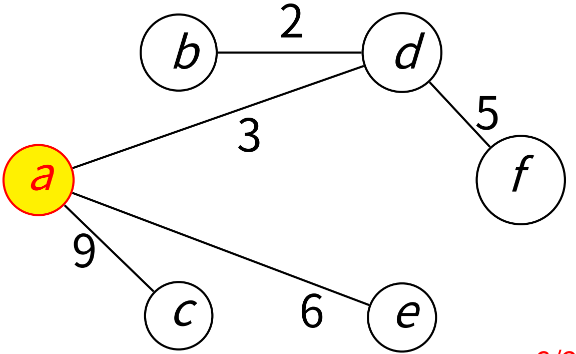
- しましょう G（V、E、W）で加重グラフになる **非負** エッジ距離、そしてしましょう s の頂点になる G そこからのすべての頂点 G 到達することができます。
- 次に、スパニングツリーが存在します T の G、として根付いた s、からの最短経路が含まれています s のすべての頂点に G。
- そのような木は呼ばれます **ツリーにまたがる最短経路**。

加重グラフとその最短経路スパニングツリー



加重グラフ G

最短経路
スパニング の木 G



ダイクストラのアルゴリズム

- アルゴリズムでは、グラフ G（V、E、W）によって表されます

距離行列 D。

/* s ソース頂点であり、 S からの最短経路が s の各頂点に S 完全に嘘をついている S。 各頂点について
$v \in V - S$、$d[v]$
からの現在の最短経路の距離が含まれます s
に v の頂点のみを通過する S。 */

ダイクストラのアルゴリズム（続き）

$S = \{s\}; d[s] = 0;$

にとって（$v \in V - S$）$d[v] = D[s$、$v];$

一方（$S \not= V$）$\{$

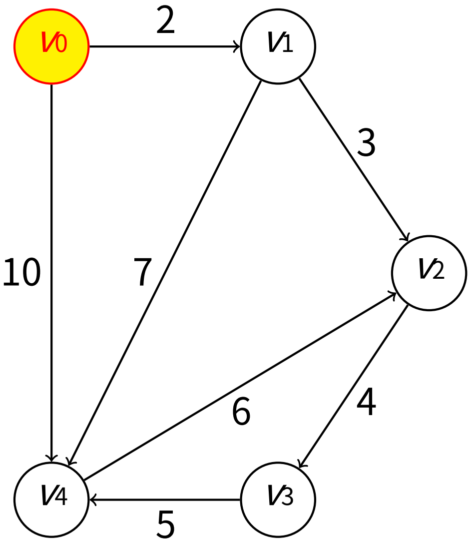
頂点を選択してください w に $V - S$ そのような

$d[w]$ 最小です。追加

w に $S;$

$\}$ にとって（$v \in V - S$）$d[v] = \min\{d[v]$、$d[w] + D[w$、$v]\};$

例



- 最初は：

  $S=\{v_0\}$、$d[v_0]=0$、

  $d\,[v$私$]$は2、$+\infty$、$+\infty$、 10、

  $i=1$、2、3、4;

- 最初の反復で：

  $w=v_1$が選択されているので

  $d\,[v_1]=2$が最小です。

- 次に：

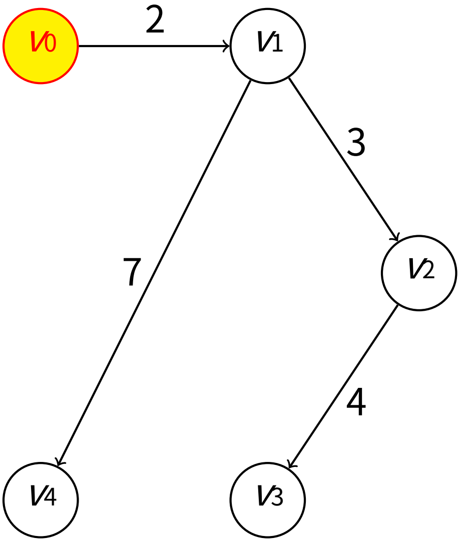
  $d\,[v_2]=\min\{+\infty、\ 2+3\}=5$

  $d\,[v_4]=\min\{10、2+7\}=9$

変数の遷移

Iter。	S	w	$d[w]$	$d[v_1]$	$d[v_2]$	$d[v_3]$	$d[v_4]$
初期化。	$\{v_0\}$	–	–	2	$+\infty$	$+\infty$	10
1	$\{v_0$、$v_1\}$	v_1	2	2	5	$+\infty$	9
2	$\{v_0$、v_1、$v_2\}$	v_2	5	2	5	9	9
3	$\{v_0$、v_1、v_2、$v_3\}$	v_3	9	2	5	9	9
4	すべて	v_4	9	2	5	9	9

「例」の単一ソース最短経路



ダイクストラのアルゴリズムの正しさ

のサイズを誘導することにより、ダイクストラのアルゴリズムの正しさを証明します。 S。

- 基礎。$|S/=1$.からの最短経路 s それ自体に長さ0とからのパス s に v、 完全に内 S

を除いて v、 シングルエッジで構成されています（s、 v）。 したがって、

$d[v]$ 正しく計算されました。（にとって

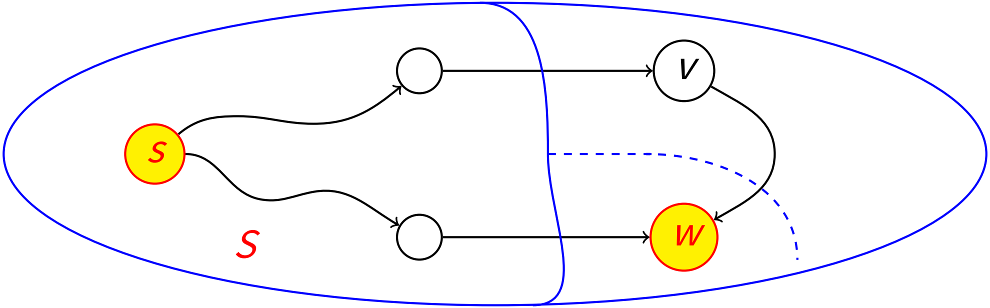
（$v \in V - S$）$d[v] = D[s$、 $v]$;）

ダイクストラのアルゴリズムの正しさ（続き）

- 帰納法の仮説。次のように仮定します
  ステートメントは|に当てはまります$S/=k\geq 1$： S からの最短経路が s の各頂点に S 完全に嘘をついている S。各頂点について
  $v\in V-S$、$d[v]$ からの現在の最短経路の距離です
  s に v の頂点のみを通過する S。

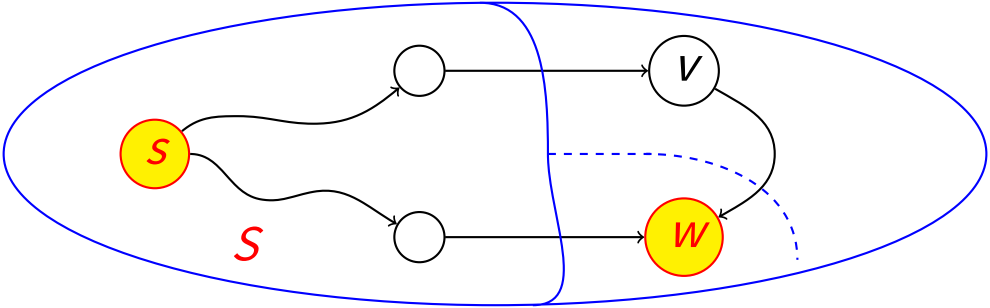
ダイクストラのアルゴリズムの正しさ（cntd2）

- 誘導ステップ。のために$|S| = k$、仮定する $w \in V - S$ そのような
  $d[w]$ 最小です が選択され、に追加されます S （$|S|$になります
  $k+1$）。場合$d[w]$からの最短経路の距離ではありません s
  に w、その後、より短いパスが必要です P そのような P
  以外の頂点が含まれています w ない S。しましょう v
  上の最初のそのような頂点になる P。



ダイクストラのアルゴリズムの正しさ（cntd3）

- しかし、その後の距離 s に v より短い $d[w]$、さらに、からの最短経路 s に v 完全に内にある S、を除いて v 自体。したがって、帰納的仮説により、$d[v] < d[w]$、a 矛盾。

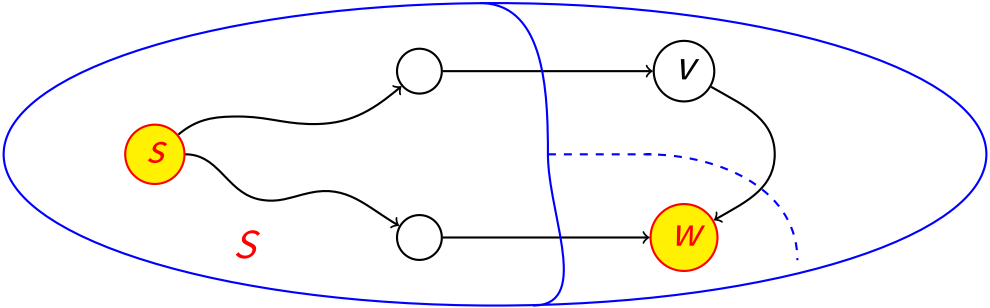


ダイクストラのアルゴリズムの正しさ（cntd4）

- 私たちは、その道は P 存在せず、 $d[w]$ それは
  からの最短経路の距離 s に w それは完全ににあります
  S。 更新操作から

$$d[v] = \min\{d[v]、d[w] + D[w、v]\}、$$

私たちは知っています $d[v]$ 正しく計算されます。



ダイクストラのアルゴリズムの効率的な実装

- グラフが隣接する（距離）で表される場合
  行列、ダイクストラのアルゴリズムの時間計算量は
  $O(|V|^2)$。
- ダイクストラのアルゴリズムは、次の方法でより効率的に作成できます。
  を使用してグラフを維持する 隣接（距離）リスト
  にないノードの優先キューを維持する S。
  次のスライドは、そのような実装の擬似コードを
  示しています。

ダイクストラのアルゴリズム（リストバージョン）

/ * s ソース頂点であり、 S からの最短経路が s の各頂点に S

完全に嘘をついている S。

各頂点について $v \in V - S$、$d[v]$ からの現在の最短経路の距離が含まれます s に v の頂点のみを通過する S。

ここに $D[i、j]$ 説明します 距離リスト* /

ダイクストラのアルゴリズム（リストver。）2

$S = \{s\}; d[s] = 0;$

にとって（$v \in V - S$）$d[v] = D[s$、$v];$

にとって（$v \in V - S$）構築する $d[v]$ に

最小ヒープ。

一方（$S \neq V$）$\{$

削除 $d[w]$ ヒープから追加します w に $S;$

にとって（$v \in V - S$）if（（w、v）$\in E$）

$\{\{d[v] = \min\{d[v]$、$d[w] + D[w$、$v]\};$

ヒープ状態を復元します;}

$\}$

ダイクストラのアルゴリズムの効率的な実装

- この実装では、の時間計算量

  ダイクストラのアルゴリズムは O（（$|V|+|E|$） ログ$|V|$）。

最小スパニングツリーと最短パススパニングツリーの違い

- プリムのアルゴリズムでは、頂点とエッジがに追加されます。
  1つずつツリーを作成し、各ステップで可能な限り最短の
  エッジを選択します。 Vに $V-T$たす。
- ダイクストラのアルゴリズムでは、エッジ（u、v）追加される
  最も近いとは言えないかもしれませんが、最小化 d（u）$+D[u$、$v]$
  ソースからへのすべてのパス内 v。

単位エッジ長のグラフ内の最短経路

- 与えられたグラフ G（V、E）エッジの距離は1、からの最短経路 u に v からのパスです u に v 最小の長さ（エッジの数）で。この問題を解決するには、ダイクストラのアルゴリズムを使用できます。

  ただし、この問題のより単純なアルゴリズムはBFSアルゴリズムです。

すべてのペアの最短経路問題

- ワーシャルのアルゴリズムフロイドのアルゴリズム