最短路

定義 2.1 有向グラフ（directed graph，digraph）
$V \cap E = \emptyset$ を満たす有限集合 V, E と，$e \in E$ を始点 $t(e) \in V$ と終点 $h(e) \in V$ に対応させる関数 $t, h : E \to V$ の組 $G := (V, E, t, h)$ を有向グラフという．

無向グラフで定義した用語や記法については，特に定義を書かない限り有向グラフに対しても同様に扱う．実際に，有向グラフにおける自己閉路や路，部分有向グラフや枝・頂点の削除といった概念は，枝の向きを無視して得られる無向グラフでの概念と対応する．しかし，用語や記法の定義が有向グラフと無向グラフで対応しない場合もある．

定義 2.2 並列（parallel）
異なる 2 つの枝 e_1, e_2 が $t(e_1) = t(e_2)$, $h(e_1) = h(e_2)$ をみたすとき，e_1 と e_2 は並列であるという．

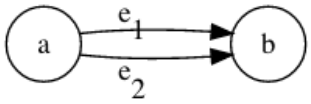

図 1　並列な枝

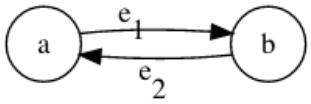

図 2　並列でない枝

定義 2.3 単純（simple）
有向グラフ G が自己閉路と並列な枝のどちらももたないとき，G は単純であるという．

定義 2.4 有向路（directed path，dipath）
$e \in E$ に対して $e = vw$ と書いたとき，$v = t(e), w = h(e)$ を意味するものとする．路 $P : v_0, e_1, v_1, ..., e_k, v_k$ の枝 e_i が $e_i = v_{i-1}v_i$ をみたすとき，前向き（*forward*）であるという．また，そうでないときは，後ろ向きや逆向き（*reverse*）であるという．すべての枝が前向きである路を有向路という．有向路でなおかつ単純閉路であるとき，有向単純閉路（*dicircuit*）という．各 $e \in E$ が実コスト c_e をもつとき，路 P のコストを $C(P) := \sum_{i=1}^{k} c_{e_i}$ と定義する．

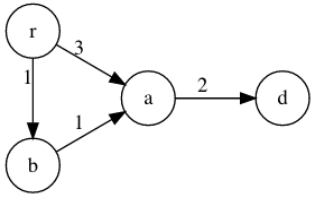

図 3　枝にコストのある有向グラフ

定義 2.5 最短路（shortest path）
頂点 $v, w \in V$ があって，有向路 P が v から w への有向路の中でコストが最小であるとき，P を v から w への最短路という．

最短路問題

入力：有向グラフ G, 1 つの頂点 $r \in V$ と実コストベクトル $(c_e : e \in E)$ が与えられる．
目的：各 $v \in V$ に対して，r から v への最短路を見つける．（もし存在すれば）

注意 2.1
十分大きなコストの枝を r から各 $v \in V$ へ付け足すことで，r から v への

有向路がしない場合を考えずにすむ．よって，r から任意の頂点への有向路は存在すると仮定する．なお，付け足す枝のコストは，想定される最短路のコストより大きい必要がある．具体的には $(|\max\{c_e|e \in E\}| + 1) \times |V|$ あれば十分である．

補足 2.1
P を最短路とするとき，コストを変えずに P を初等的な路へと変形できる．

証明
P が初等的な路でないとき，P は有向閉路 C を含む．C のコストが正であるとするとその閉路を取り除くことでよりコストを小さくでき，P が最短路であることに矛盾．また，C のコストが負であるとしても，C を何度も通ることでいくらでもコストを小さくでき矛盾．よって C はコストが 0 であり，C を取り除く変形ができる．この操作を繰り返すことで，P を初等的な路に変形できる．なお，変形によって P の長さは真に短くなるため，この操作は有限回で止まる． □

注意 2.2
有向グラフ G に，単純であるという仮定をおくことがある．実際に，並列な枝のコストが大きい方やコストが 0 以上のループは，取り除いて考えても求まる有向路のコストは変化しない．また，コストが 0 未満のループが存在し，なおかつそのループを経由できるとき，何度もループを通ることでいくらでもコストを小さくできるため最短路は定まらない．

次に，最短路問題を解く上で基本となる主張について書く．

主張 2.1
各 $v \in V$ について y_v が v への最短路のコストであるとき，以下が成立する．

$$\forall vw \in E,\ y_v + c_{vw} \geq y_w \tag{2.12}$$

証明
$y_v + c_{vw} < y_w$ をみたす $vw \in E$ が存在したと仮定すると，コストが y_v となる有向路に枝 vw を付け足すことで，最短路よりもコストが小さい有向路を作ることができ矛盾する． □

定義 2.6 実行可能ポテンシャル（feasible potential）
$y = (y_v : v \in V)$ が (2.12) と $y_r = 0$ をみたすとき，y を実行可能ポテンシャルという．

上の定義において，(2.12) が本質的な条件である．なぜなら，(2.12) をみたす $y = (y_v : v \in V)$ を与えられたとき，$y' = (y_v - y_r : v \in V)$ と定めることで (2.12) をみたしながら $y'_r = 0$ とできるからである．次の命題で示すように，実行可能ポテンシャルは最短路のコストの下界を与える．

命題 2.9
y を実行可能ポテンシャルとし，P を r から v への有向路とするとき，$c(P) \geq y_v$ が成立する．

証明
任意の r から v への有向路 $P = v_0, e_1, v_1, ..., e_k, v_k$ について，

$$c(P) = \sum_{i=1}^{k} c_{e_i} \geq \sum_{i=1}^{k} (y_{v_i} - y_{v_{i-1}}) = y_{v_k} - y_{v_0} = y_v.$$

□

他にもう 1 つ，シンプルだが有益な観察について書く．

定義 2.7 部分路（subpath)
路 $P = v_0, e_1, v_1, ..., e_k, v_k$ に対し，$P' = v_l, e_{l+1}, v_{l+1}, ..., e_m, v_m (0 \leq l \leq m \leq k)$ を P の部分路という．

主張 2.2

P が最短路であるとき，その部分路 P' も最短路．

証明

P' が最短路でないと仮定すると，P' をよりコストの小さい有向路に変更することで P のコストを小さくできる．これは P が最短路であることに矛盾する． □

主張 2.3

$G = (V, E, t, h)$ を頂点 r から各頂点への最短路をもつ有向グラフとする．頂点 r から各頂点への最短路を，次の条件をみたすようにとれる．

任意の $v \in V \setminus \{r\}$ について，
終点が v である枝 $e \in E$ はちょうど 1 個だけ用いる．

証明

まず，各頂点への最短路を初等的な路に変形しておく．次に，ある $v \in V \setminus \{r\}$ について，最短路に用いている枝の中に終点が v であるものが複数あったとする．用いた最短路のなかで，頂点 v を通るものを $P_1, ..., P_k$ とおく．$P_1, ..., P_k$ を頂点 v で分割して得られる r から v への路をそれぞれ $P'_1, ..., P'_k$ とすると，これらは最短路である．よって，各 $P_2, ..., P_k$ の $P'_2, ..., P'_k$ の部分を P'_1 に置き換えることで，終点が v である枝はちょうど 1 個だけ用いるように変形できる．この操作を繰り返すことで，条件をみたすことができる． □

定義 2.8 有向全域木（directed spanning tree）

G を有向グラフとし，$r \in V$ とする．G の全域木 T が r から任意の頂点への有向路を含むとき，T を r を根とする有向全域木という．

定義 2.9 最短路木（shortest path tree）

G を有向グラフとし，$r \in V$ とする．G の r を根とする有向全域木 T が頂点 r から任意の頂点への最短路を含むとき，T を r を根とする最短路木という．

主張 2.4

G を頂点 r から各頂点への最短路をもつ有向グラフとする．このとき，r を根とする最短路木が存在する．

証明

主張 2.3 を使って各頂点への最短路をとり，そこで用いた枝と頂点からなる部分グラフを T とする．すると，T の枝は枝の終点に対応させて数えることができ，$n-1$ 本とわかる．また，T は r から任意の頂点への最短路を含む全域部分グラフである．よって補題 2.2 により T は木であるとわかり，r を根とする最短路木となっている． □

注意 2.3

全域木は有向全域木であるとは限らない．

注意 2.4

図 4 の有向グラフからわかるように，r を根とする有向全域木で用いる枝のコストの総和が最小になるものであっても，最短路木とは限らない．また，最短路木であっても最小全域木とは限らない．

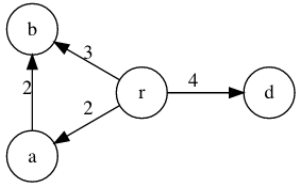

図 4　有向グラフの例

Ford のアルゴリズム

命題 (2.9) によって，実行可能ポテンシャル y とコストが y_v である各 $v \in V$ への有向路を得たときに，最短路問題を解くアルゴリズムを停止させればいいとわかる．そして今までの議論を踏まえると，「(2.12) をみたさないような枝 vw を見つけたときに y_w を $y_v + c_{vw}$ に置き換える」という手順を繰り返すことによって最短路問題が解けるのではないかと考えられる．この考え方を基本としているアルゴリズムが，次に記す Ford のアルゴリズムである．

Ford のアルゴリズムの擬似コード

```
1: for 各頂点 v ∈ V do                          ▷ 各頂点について y, p を初期化
2:     if v == r then
3:         y_v = 0
4:         p(v) = 0                                          ▷ 0 ∉ V
5:     else
6:         y_v = ∞
7:         p(v) = −1                                         ▷ −1 ∉ V
8: while y が実行可能ポテンシャルでない do
9:     for 各枝 vw ∈ E do
10:        if y_v + c_vw < y_w then
11:            y_w = y_v + c_vw                  ▷ 間違っている枝を改善する
12:            p(w) = v                  ▷ 改善したときの繋げ方を覚えておく
```

上記のコードが正しく動くかどうかは後で議論する．まずは実際の有向グラフを例に，このコードがどのように動くかについて観察する．