離散システム論 (第 12 回)

http://coconut.sys.eng.shizuoka.ac.jp/ds/

安藤和敏 (静岡大学工学部)

2006.06.26

2.1. 最短路問題 IV

2.2. ベルマン-フォード法とダイクストラ法との関係

ベルマン-フォード法を, 始点を $v_0 = s$ として図 2.1 のグラフに対して実行した結果は表 2.1 のようになる. さらに, 実行結果を図で表現すると, 図 2.2 のようになる.

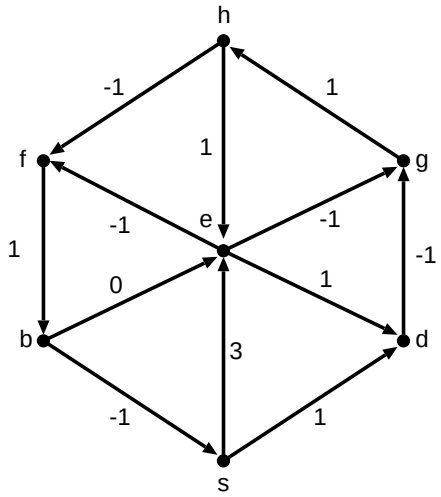

図 2.1: ネットワーク $\mathcal{N} = (G = (V, A), l)$

ただし, Step 2 における枝の選択の順番は,

s から出る枝, b から出る枝, d から出る枝, e から出る枝, f から出る枝, g から出る枝, h から出る枝.

とした.

こうやって, 結果を眺めてみるとなんとなく無駄な動きをしていると思うかも知れない. 例えば, b から出る枝よりも前に e から出る枝を調べていたらこんなにも繰り返しの数 (k) が大きくならなかったかも知れない. 各点から出る枝を調べるとき, (s, b, d, e, f, g, h) という順番がいけなかったのかも知れない. もっと良い順番で知らべれば, 繰り返しの数を小さくできるだろうか?

実はできる. 最適な順番は (s, d, g, h, f, b, e) である. この順番でベルマン-フォード法を動かした結果は表 2.2 のようになる. (各自で確認してみよ.)

表 2.1: ベルマン-フォード法の動き (II).

	s	b	d	e	f	g	h
p	0	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
q	—	—	—	—	—	—	—
$k=1$ p	0	3	1	2	0	0	1
q	—	f	s	h	h	d	g
$k=2$ p	0	1	1	2	0	0	1
q	—	f	s	h	h	d	g
$k=3$ p	0	1	1	1	0	0	1
q	—	f	s	b	h	d	g
$k=4$ p	0	1	1	1	0	0	1
q	—	f	s	b	h	d	g
$k=5$ p							
q							
$k=6$ p							
q							
$k=7$ p							
q							

表 2.2: ベルマン-フォード法の動き (III) (各自で記入せよ).

	s	b	d	e	f	g	h
p	0	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
q	—	—	—	—	—	—	—
$k=1$ p							
q							
$k=2$ p							
q							
$k=3$ p							
q							
$k=4$ p							
q							
$k=5$ p							
q							
$k=6$ p							
q							
$k=7$ p							
q							

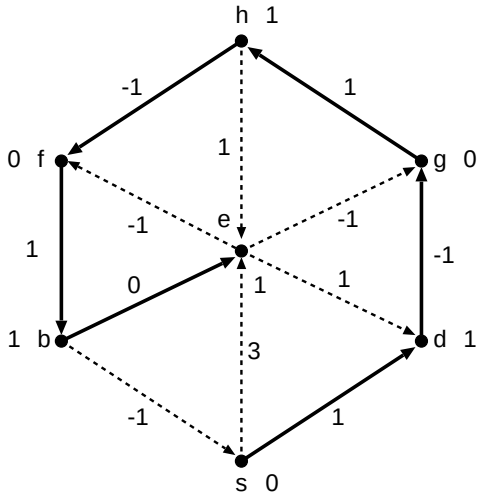

図 2.2: p と q の図的表現.

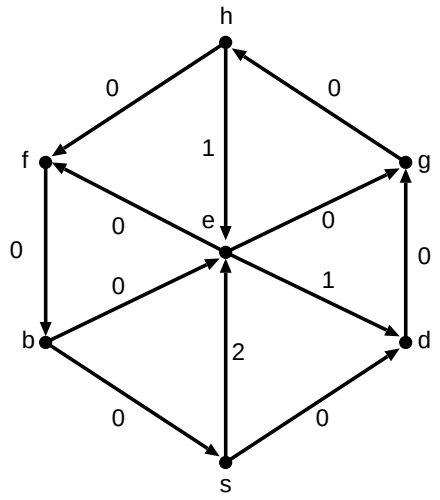

図 2.3: ネットワーク $\mathcal{N}_p = (G = (V, A), l_p)$

なんと, $k = 2$ で終了してしまった! 安藤は, どのようにしてこの順番: (s, d, g, h, f, b, e) を発見したのであろうか? また, このような「良い」順番を見付けることは常に可能であろうか?

このような「良い」順番を見付けることは常に可能である. この順番を見つける方法を以下に示そう. 図 2.3 は, ベルマン-フォード法が終了したときのポテンシャル p を用いて, 枝の長さを

$$l_p(v, w) = l(v, w) + p(v) - p(w)$$

で変更したネットワーク $\mathcal{N}_p = (G = (V, A), l_p)$ である. $l_p(a) \geq 0$ $(a \in A)$ であるから, $\mathcal{N}_p = (G = (V, A), l_p)$ に対する最短路問題はダイクストラ法を用いて解くことができる. ダイクストラ法を実行して, w として点が選ばれた順番が, (s, d, g, h, f, b, e) である.

(テキスト, あるいは, 6 月 5 日のプリントの) 補題 2.1 によって, $\mathcal{N} = (G = (V, A), l)$ に対する最短路問題は, $\mathcal{N}_p = (G = (V, A), l_p)$ に対する最短路問題と同値であるから, $\mathcal{N}_p$ に対する最短路問題を解けば元のネットワーク $\mathcal{N}$ に対する最短路問題を解けたことになることに注意しよう.

ただし, 注意すべき点は $\mathcal{N}_p$ を得るためには, ベルマン-フォード法を使わなければならないので, 点を調べる最適な順番というのは, 後になってみないと分からないのだ!

2.3. 非巡回的なネットワーク

ダイクストラ法は, 最適な順番で各点から出る枝を調べるベルマン-フォード法と考えることができる. ただし, ダイクストラ法が使えるのは, 全て枝の長さが非負であるという特別な場合だけである. 最適な順番最初から分かる, もう一つの特別なネットワークのクラスがある. そのような特別なネットワークについてこの節では考える.

有向グラフ $G = (V, A)$ は, もし有向閉路が存在しないとき**非巡回的**と呼ばれる. 有向グラフ $G = (V, A)$ が非巡回的であるようなネットワーク $\mathcal{N} = (G = (V, A), l)$ を, **非巡回的ネットワーク**と呼ぶ. ここで, 枝の長さは任意である. 非巡回的ネットワークには, 負の長さの枝があるかも知れないけれども, 有向閉路自体が存在しないので, 当然負長さの有向閉路も存在しない.

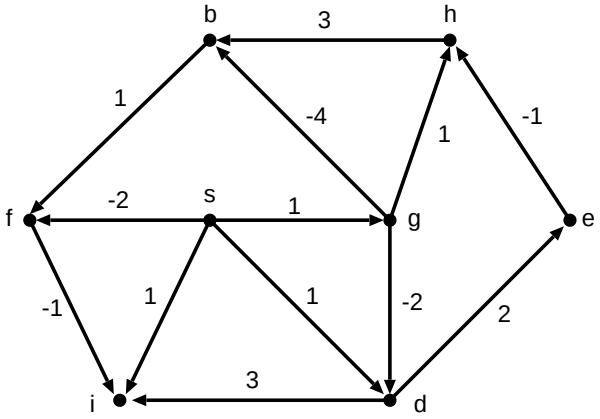

図 2.4: 非巡回的ネットワーク $\mathcal{N} = (G = (V, A), l)$

グラフ $G = (V, A)$ の点集合 V の**位相的順序** (topological order) とは, V の要素 (点) を

$$(v_i, v_j) \text{ が枝 } \Longrightarrow \ i < j$$

を満足するように並べた $(v_1, v_2, \cdots, v_n)$ である. グラフ $G = (V, A)$ が非巡回的であるとき, V の位相的順序は必ず存在する. 例えば, 図 2.4 のグラフの点の位相的順序の一つは, (s, g, d, e, h, b, f, i) である. 位相的順序の求め方は, 以下を繰り返すことで得られる.

1. 入る枝が無い点を一つ選ぶ.
2. その点から出る枝をすべて削除する.

アルゴリズム 1 非巡回的ネットワークに対する最短路アルゴリズム (始点を v_1 とする)

入力: 非巡回的ネットワーク $\mathcal{N} = (G = (V, A), l)$, $v_1 \in V$.
出力: v_1 からその他の各点 v への最短路, 及び, 最短路長.

1: $p(v_1) \leftarrow 0,\ p(u) \leftarrow +\infty\ (u \in V \setminus \{v_1\}),\ k \leftarrow 1.$
2: V の位相的順序 $(v_1, \cdots, v_n)$ を求める.
3: **for** $i = 1$ **to** n **do**
4: $\quad v_i$ から出る各枝 (v_i, w) に対して
$\quad\quad$ (*) $p(w) > p(v_i) + l(v_i, w)$ ならば, $\quad p(w) \leftarrow p(v_i) + l(v_i, w),\ q(w) \leftarrow v_i.$
5: **end for**

非巡回点ネットワークに対する最短路問題は, ベルマン-フォード法の Step 2 を V の位相的順序で各点から出る枝を順に調べていけば, 1 回の繰り返しで終了する. このアルゴリズムを形式的に述べたものがアルゴリズム 1 である.

$\mathcal{N} = (G = (V, A), l)$ が非巡回的ネットワークであるときは, 最短路問題だけでなく, 最長路問題も解くことができる. なぜならば, $\mathcal{N} = (G = (V, A), l)$ の最長路問題は, $\mathcal{N}$ の各枝の長さ $l(a)$ の符号を逆にしたネットワーク $-\mathcal{N} = (G = (V, A), -l)$ の最短路問題を解けばいいからである.

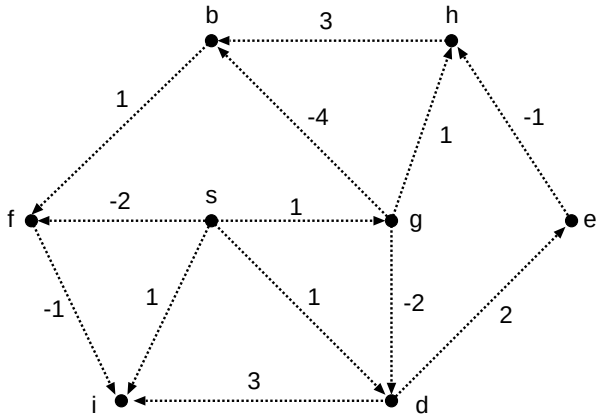

図 2.5: アルゴリズム終了後の p と q (各自で記入せよ).

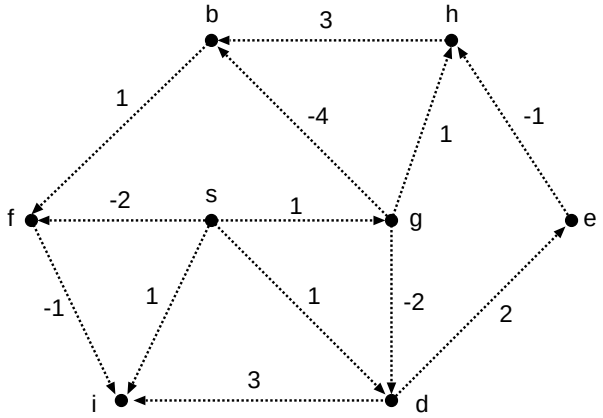

図 2.6: 最長路問題として解いたときの p と q (各自で記入せよ).