グラフとネットワーク (第12回)

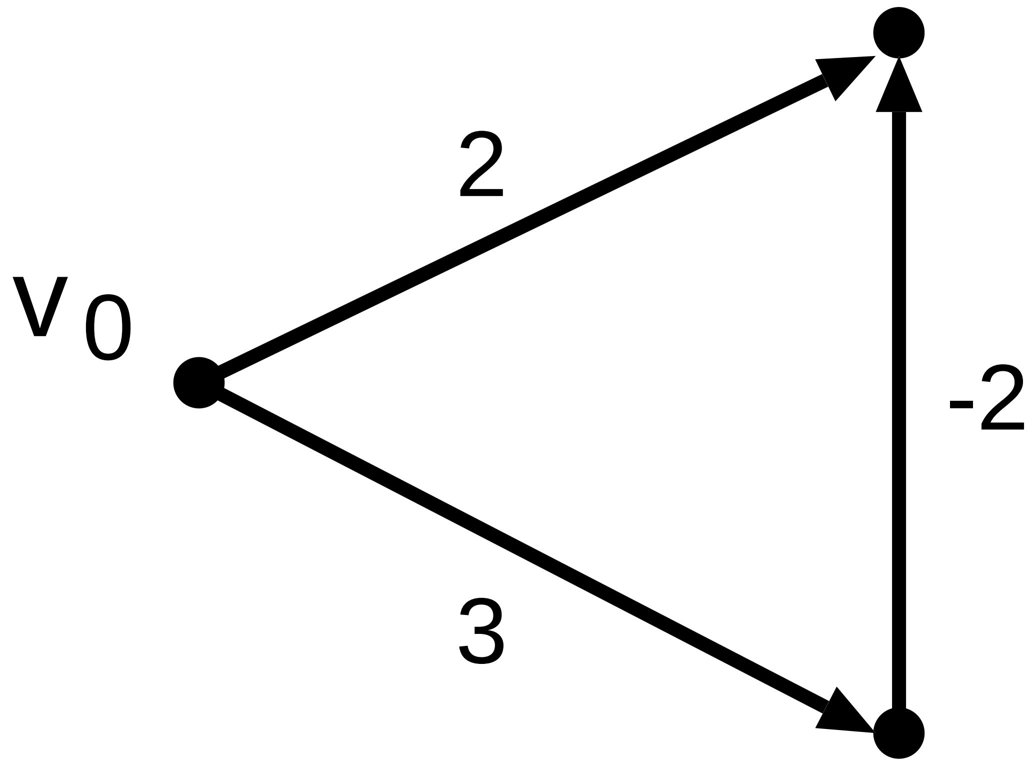
安藤 和敏

ando@sys.eng.shizuoka.ac.jp

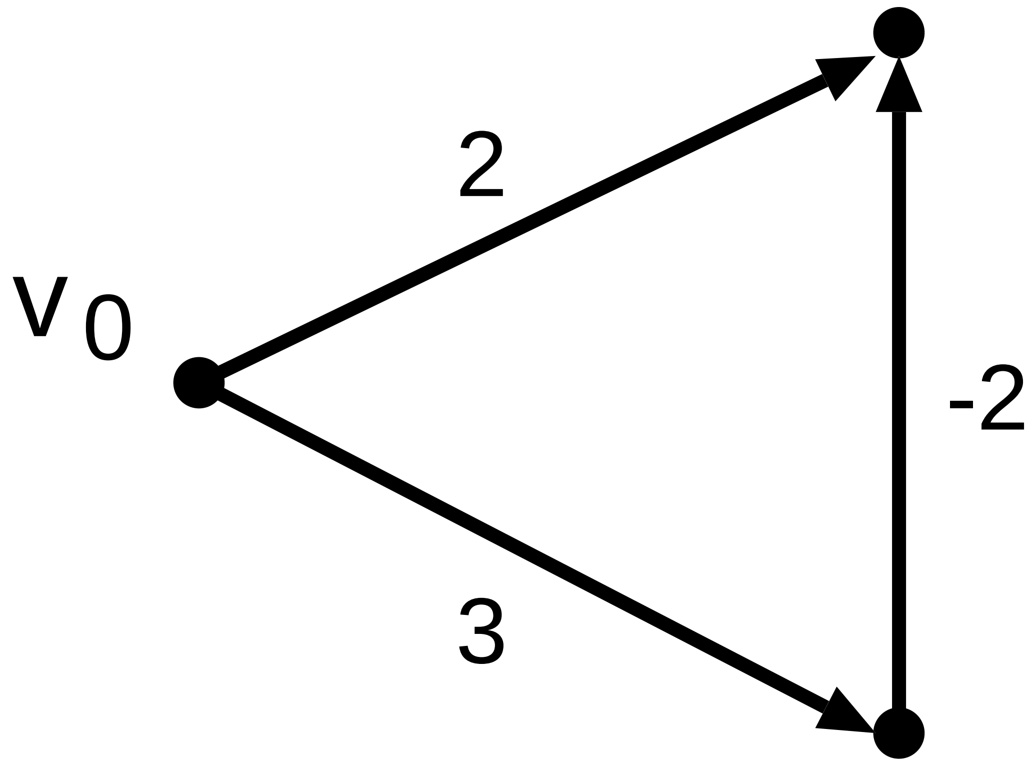
静岡大学工学部

2.1.2 最短路問題 (ベルマン-フォード法)

負の長さの枝があるネットワーク

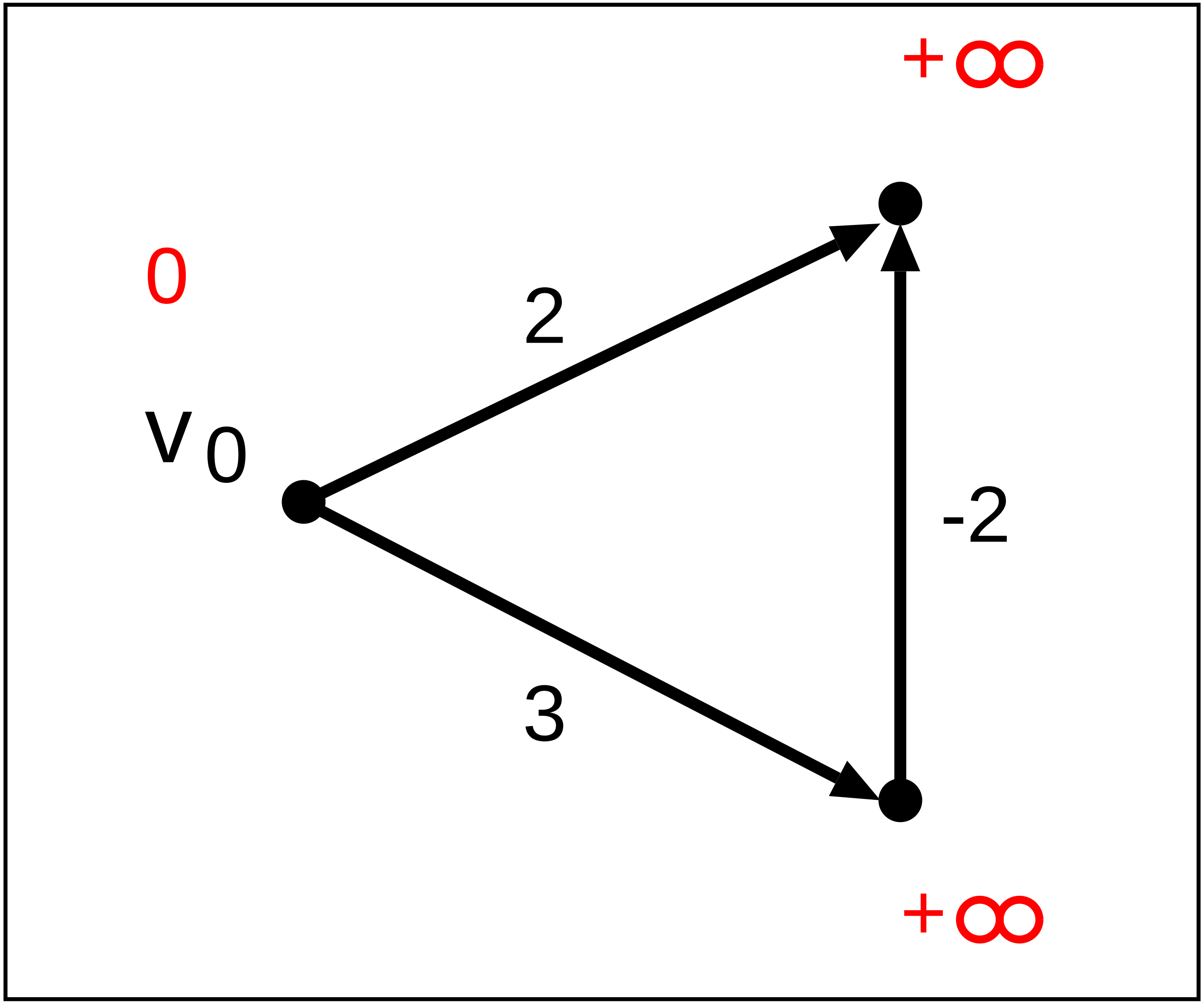


負の長さの枝があるネットワーク

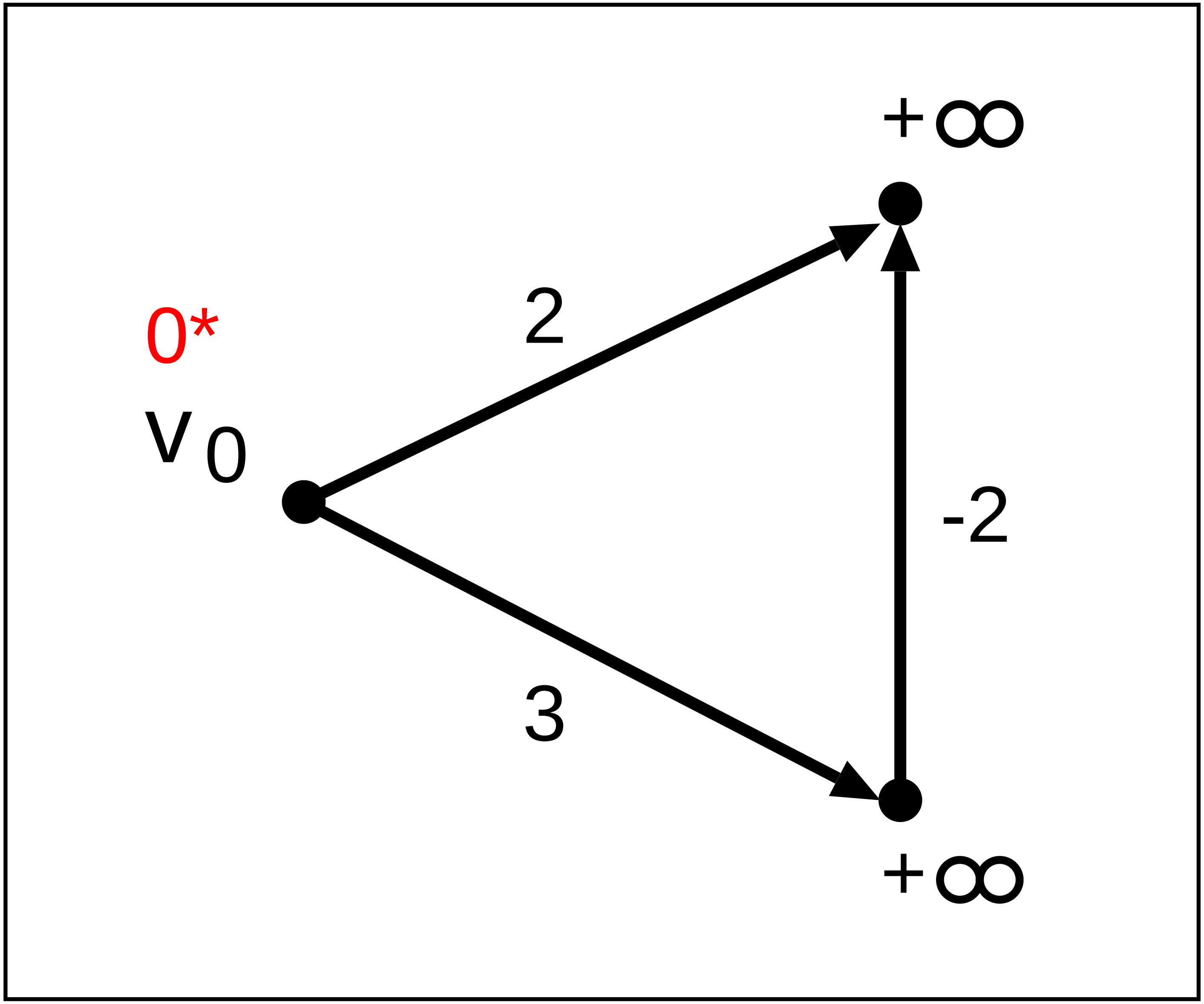


ダイクストラ法を動かしてみよう.

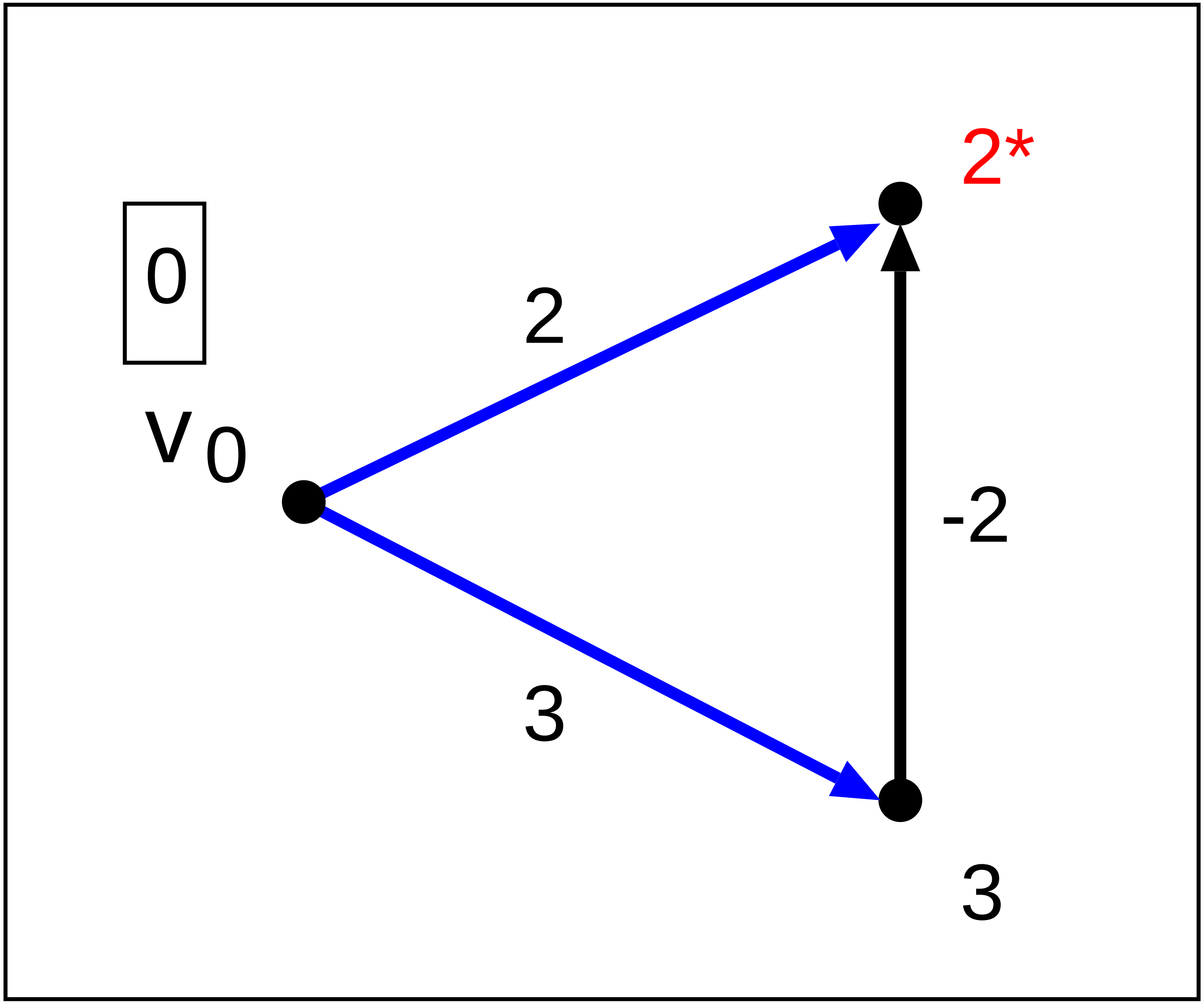
Step 1の終了時



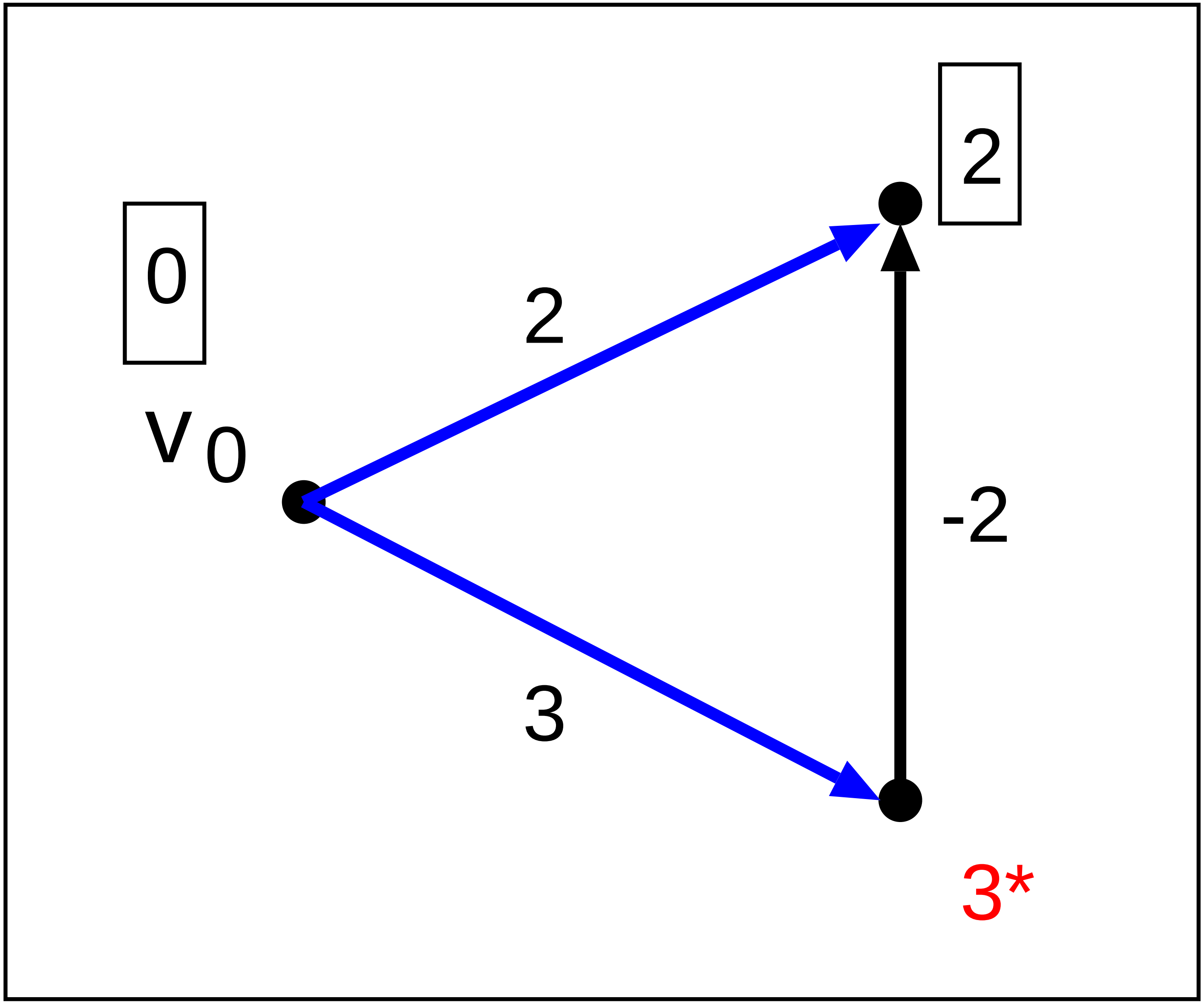
1回目の Step 2 で w が決まったとき



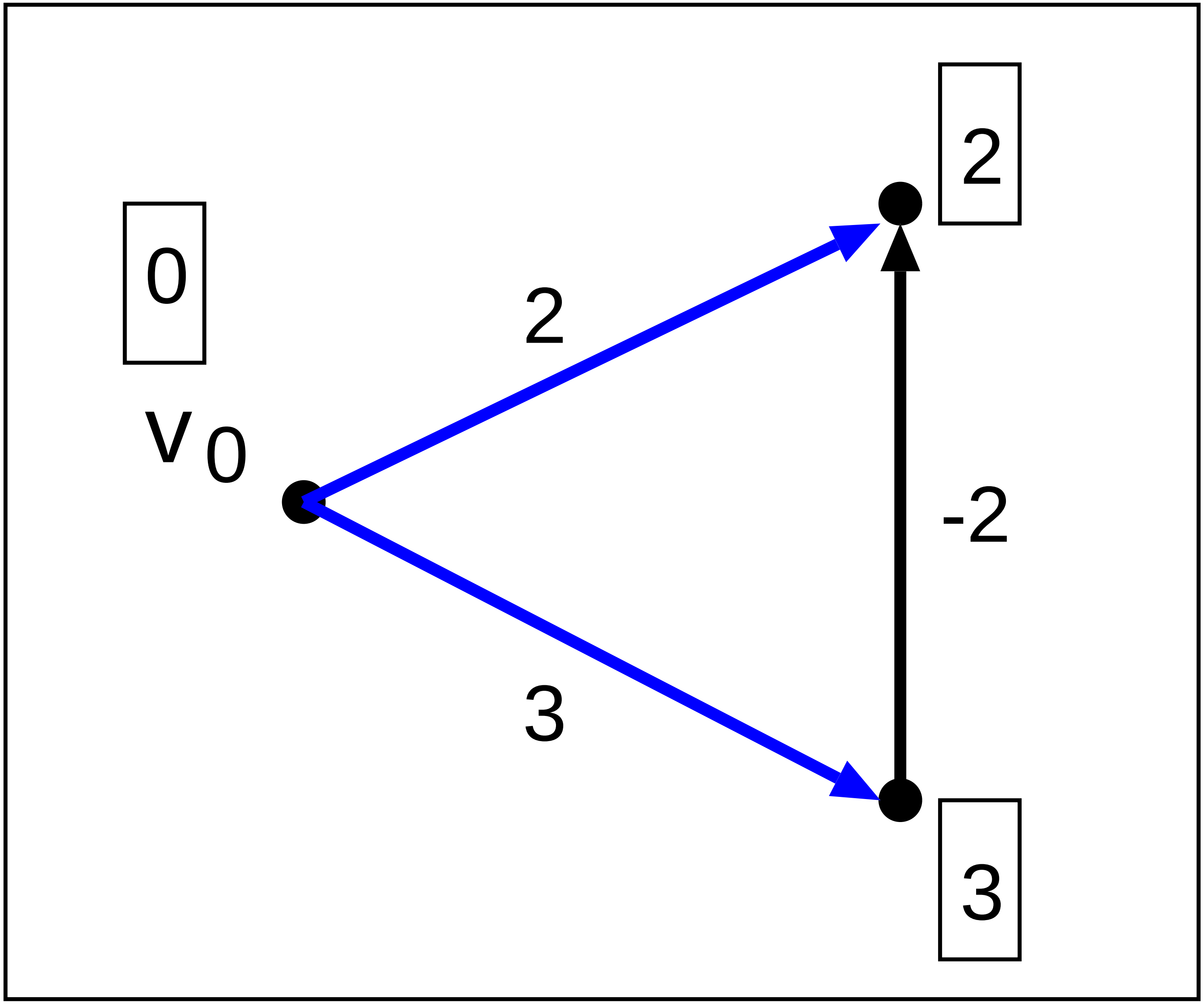
2回目のStep 2で w が決まったとき



3回目のStep 2でwが決まったとき



4回目の Step 2 で w が決まったとき

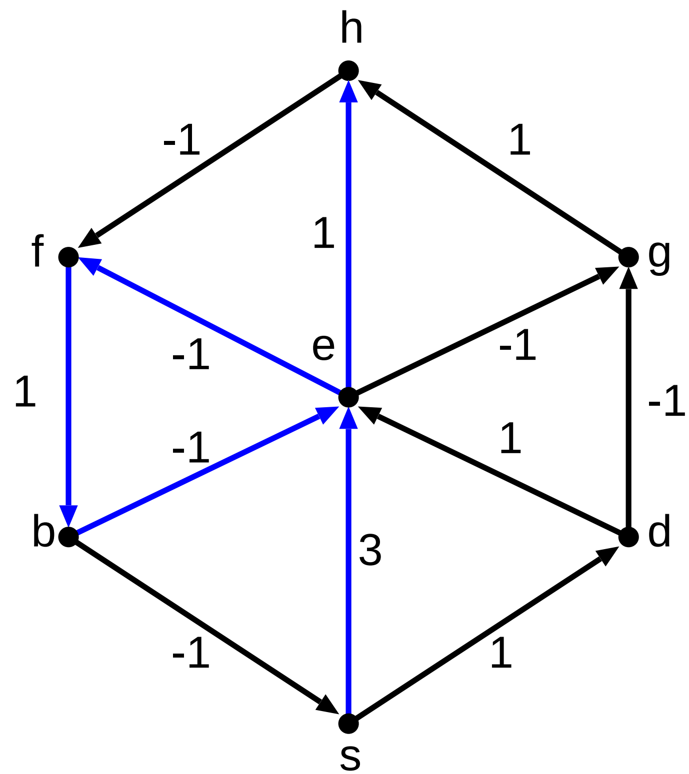


ベルマン-フォード法

本日の講義では, 負の長さの枝があるようなネットワークに対する最短路問題を解くアルゴリズムであるベルマン-フォード法を紹介する.

負の長さの枝の存在は非現実的に思えるかも知れないが, もっと複雑な問題を解くためにそのようなネットワークを扱う必要がある.

負の長さの有向閉路



与えられたネットワークに負の長さの有向閉路が存在するとき, 最短路問題は一般に解を持たない.

負の長さの有向閉路

では, 負の長さの有向閉路の存在は許すけれども, 問題を最短路として初等的な有向道 (⇒ p. 13) の中で最短なものを見付けることとしたらどうだろう.

負の長さの有向閉路

では, 負の長さの有向閉路の存在は許すけれども, 問題を最短路として初等的な有向道 ($\Rightarrow$ p. 13) の中で最短なものを見付けることとしたらどうだろう.

$\Longrightarrow$ 解くのが非常に困難な問題になる.

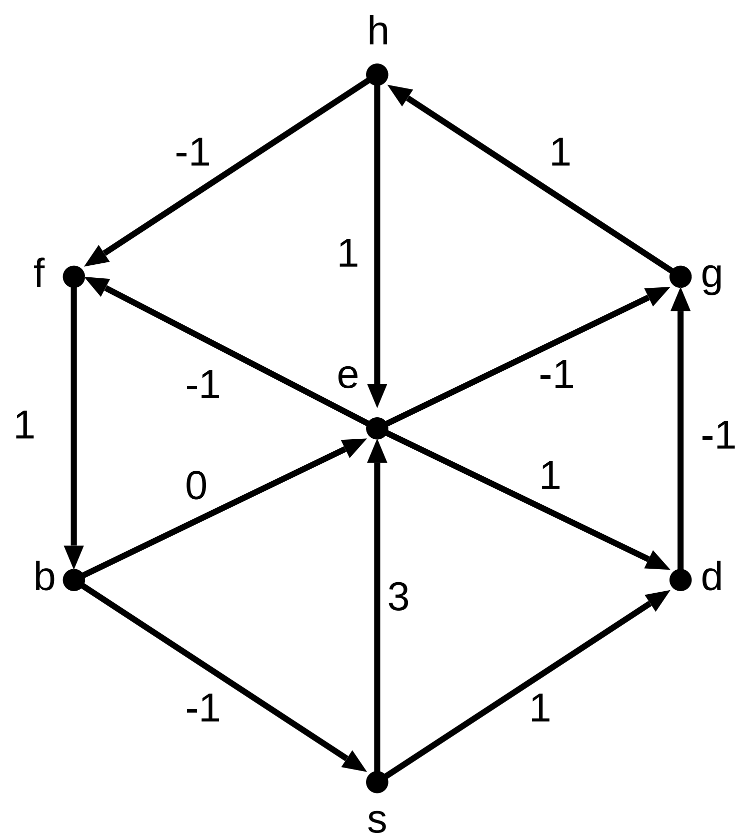
負の長さの有向閉路

では, 負の長さの有向閉路の存在は許すけれども, 問題を最短路として初等的な有向道 ($\Rightarrow$ p. 13) の中で最短なものを見付けることとしたらどうだろう.
$\Longrightarrow$ 解くのが非常に困難な問題になる.

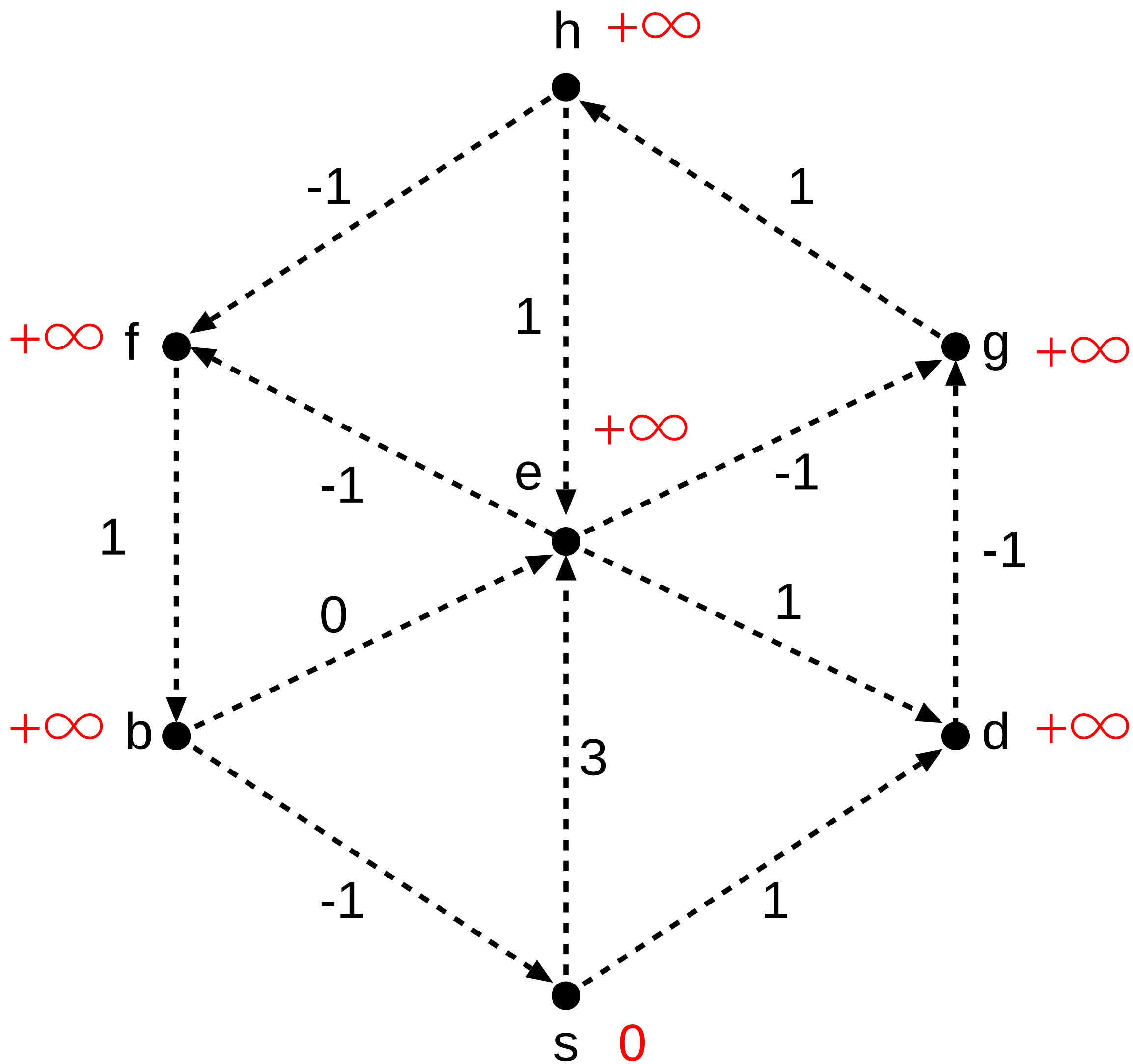
ベルマン-フォード法は, 負の長さの閉路を見付けるか, または, 始点からそれ以外の全ての点への最短路を見付けて停止する.

ベルマン-フォード法 (例題1)

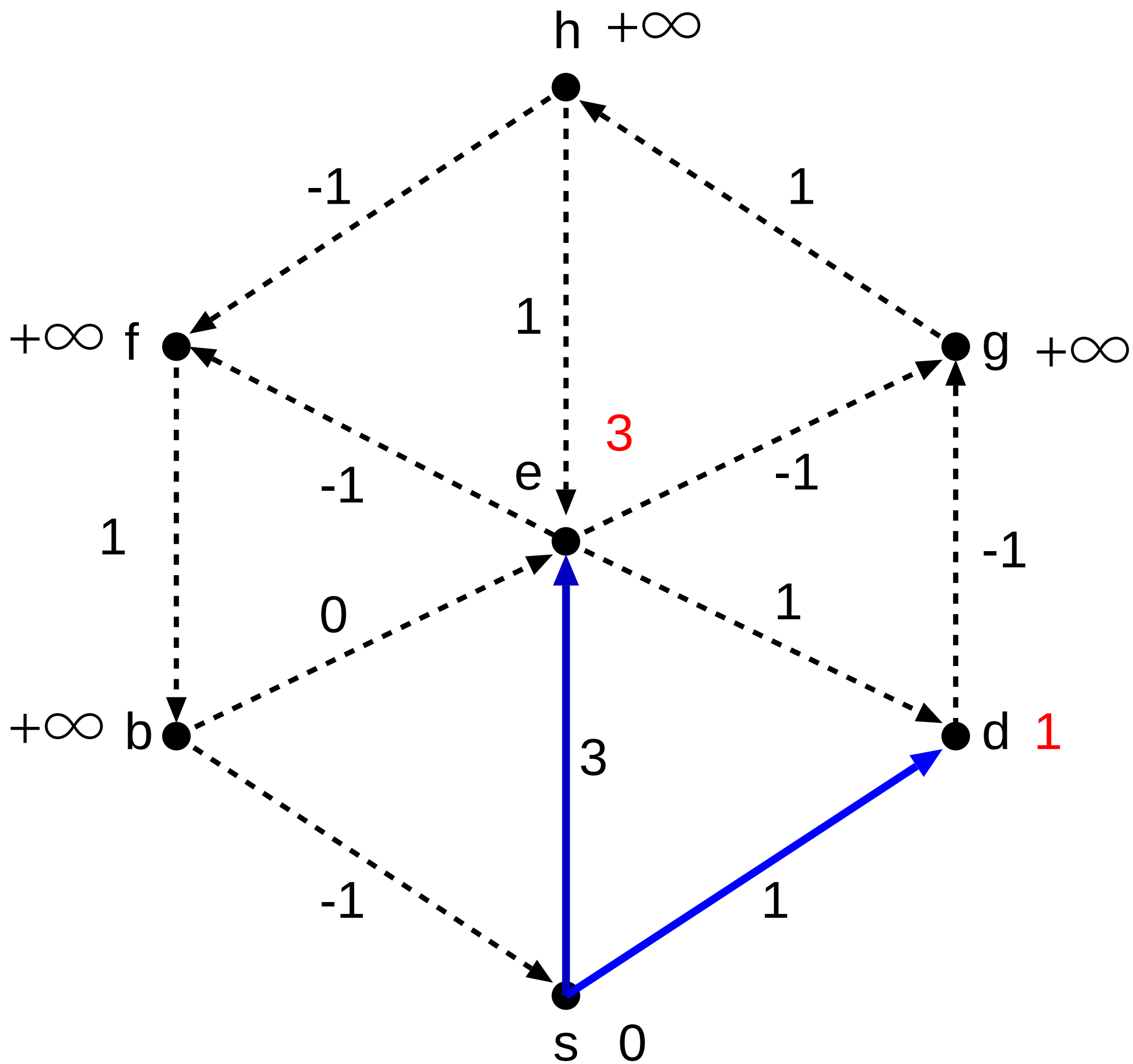
以下に示されるネットワークに対する, 始点を $v_0 = s$ とする最短路問題を, ベルマン-フォード法を用いて解いてみよう.



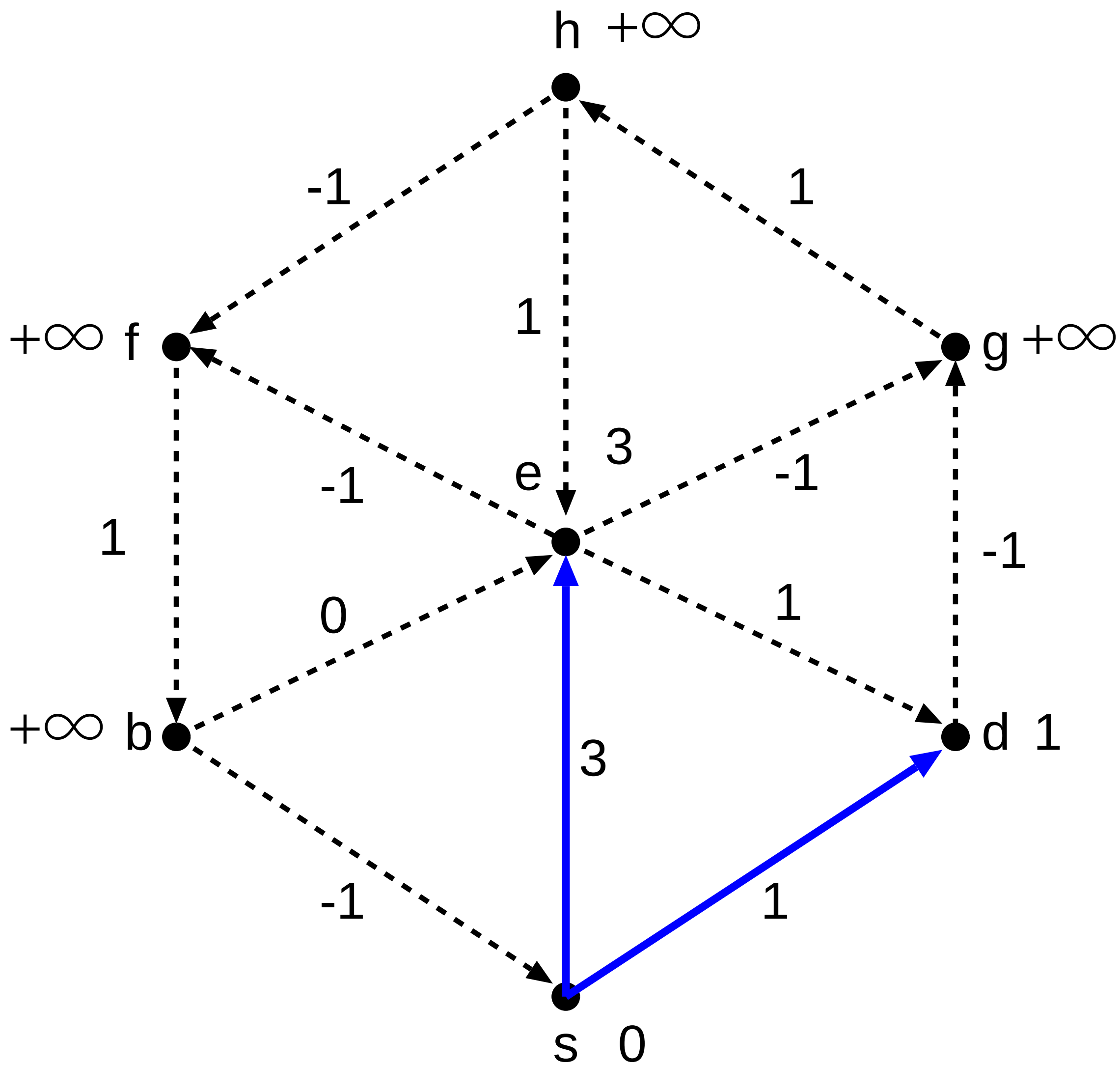
Step 1 終了時



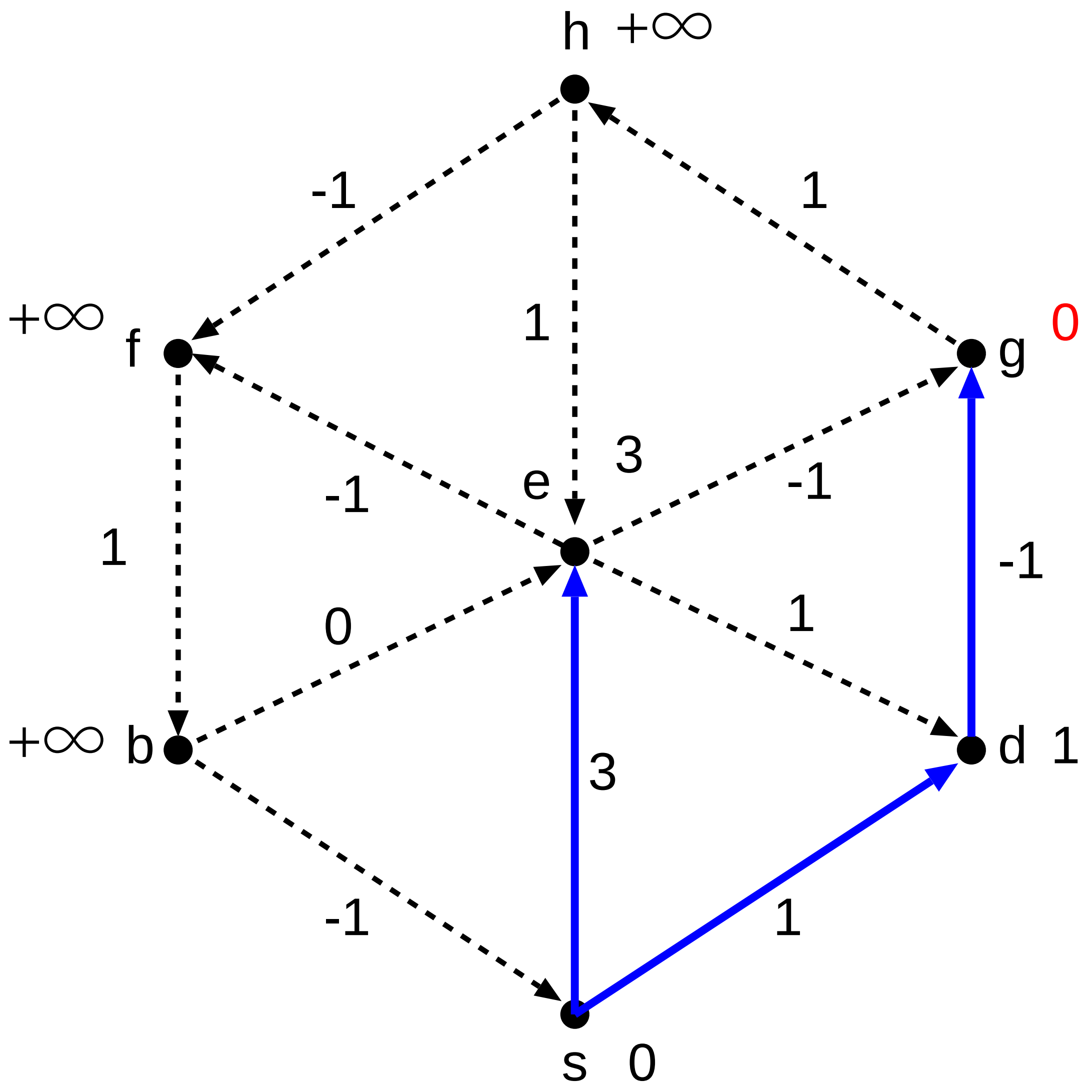
1回目の Step 2: s から出る枝 $(s,d),(s,e)$



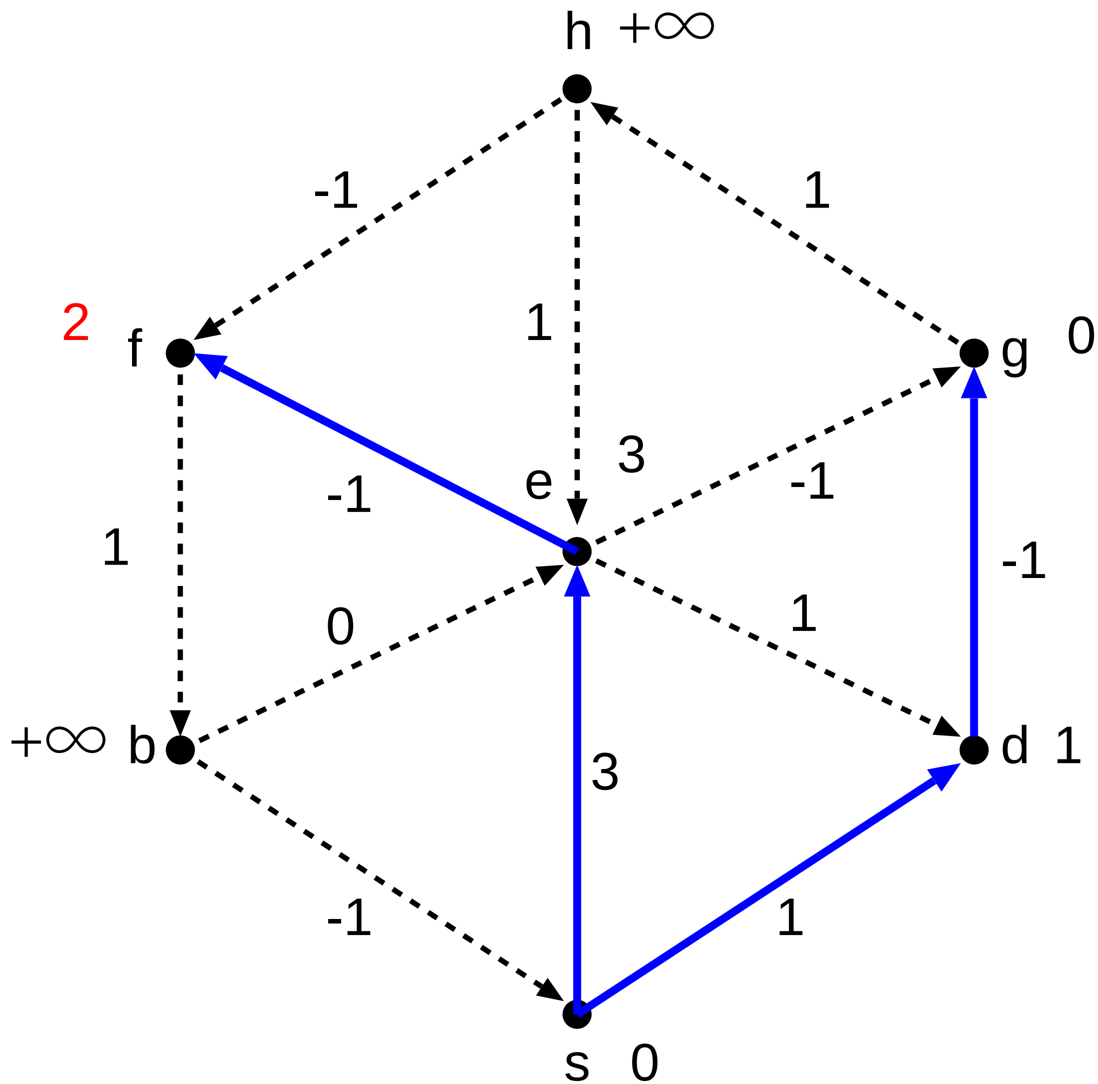
1回目の Step 2: bから出る枝$(b, e), (b, s)$



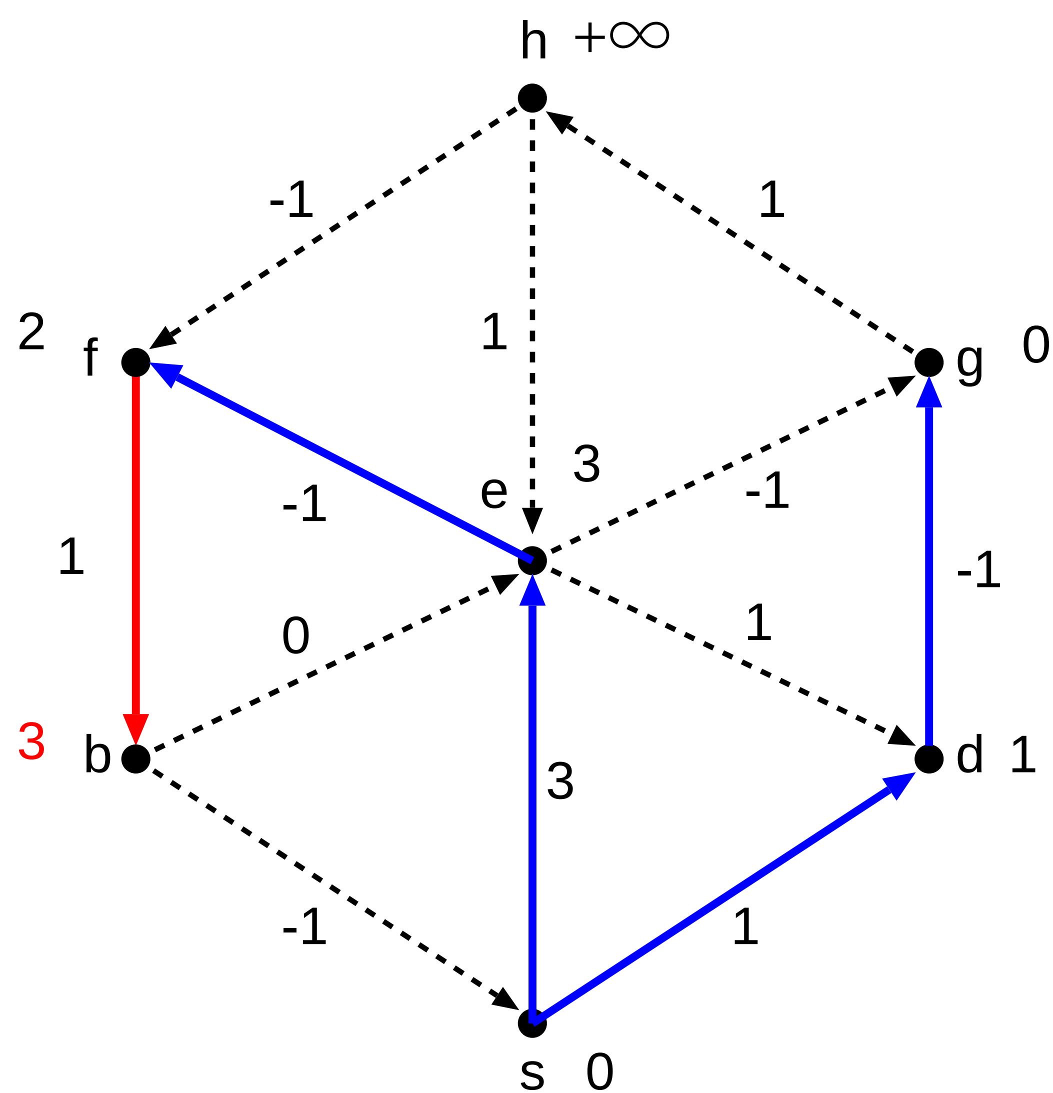
1回目の Step 2: dから出る枝(d, g)



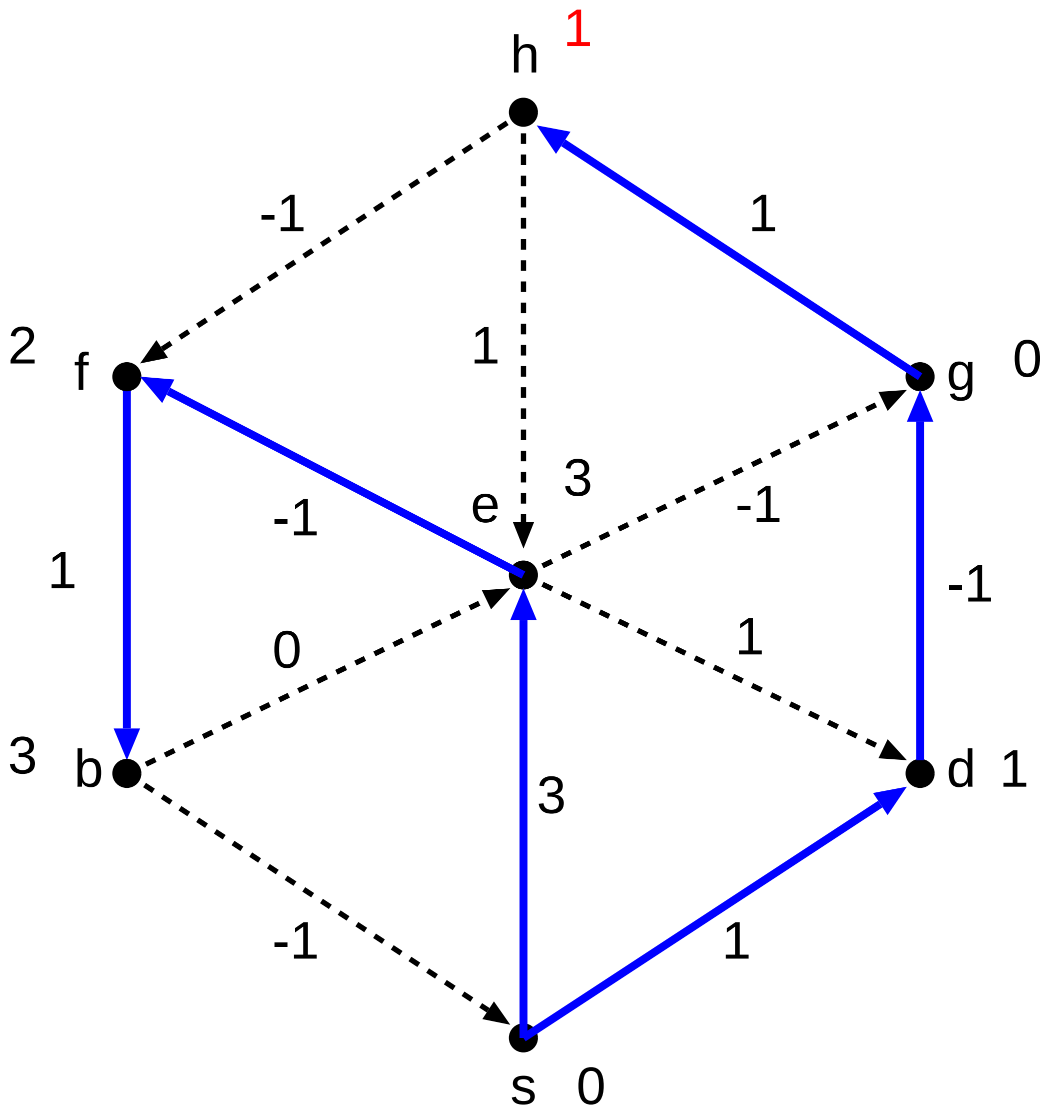
回目の Step 2: e から出る枝 $(e, f), (e, g), (e,$



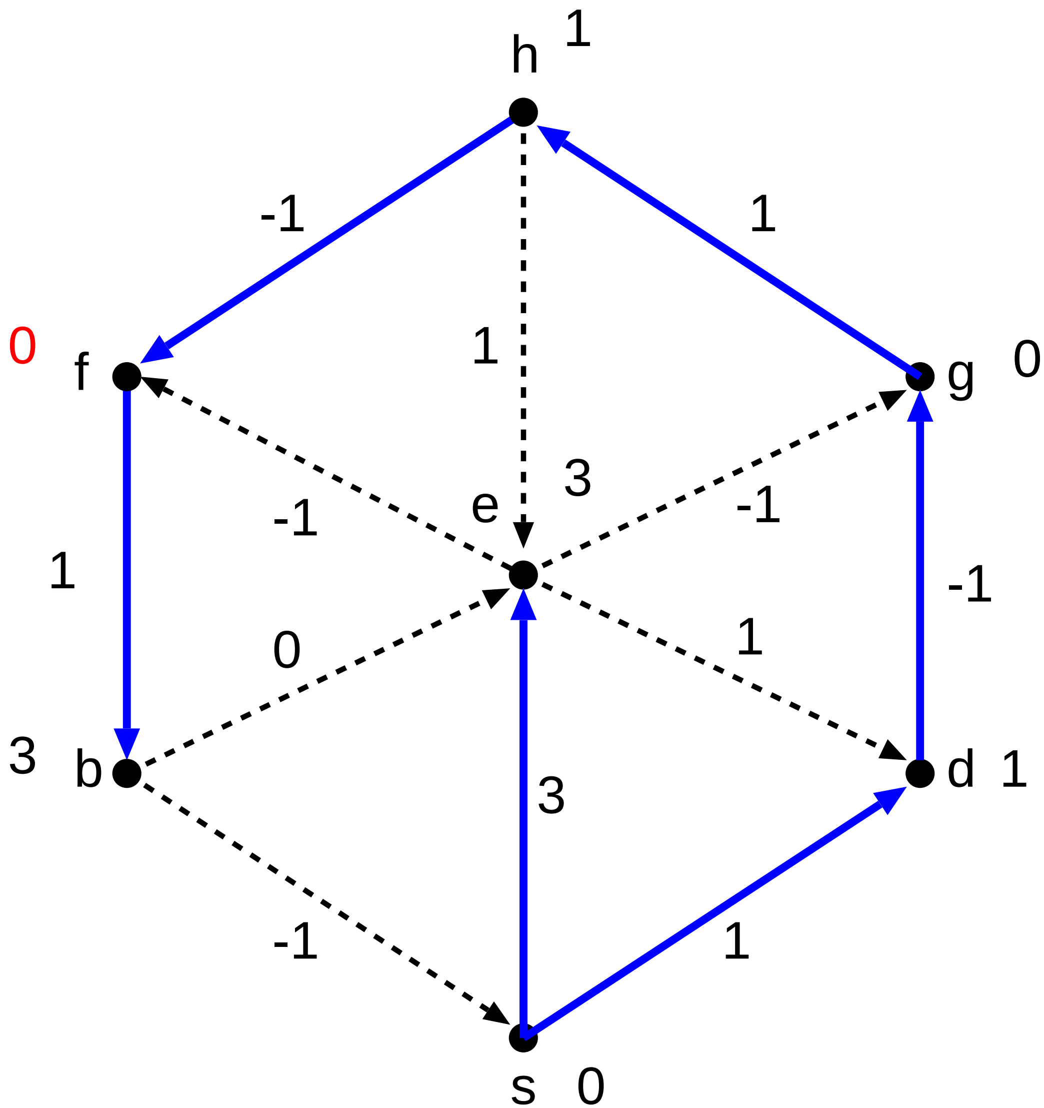
1回目の Step 2: f から出る枝 (f, b)



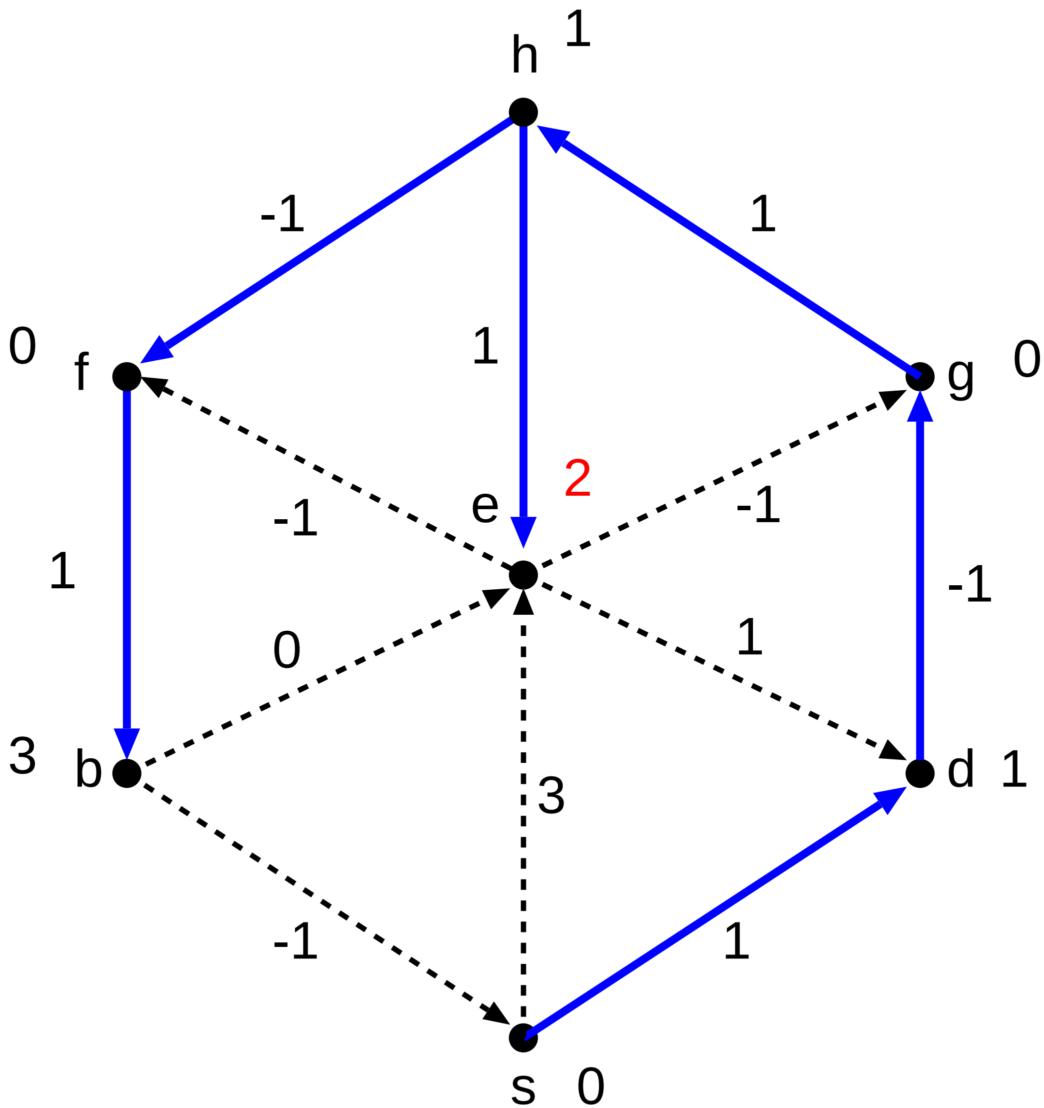
1回目の Step 2: g から出る枝 (g, h)



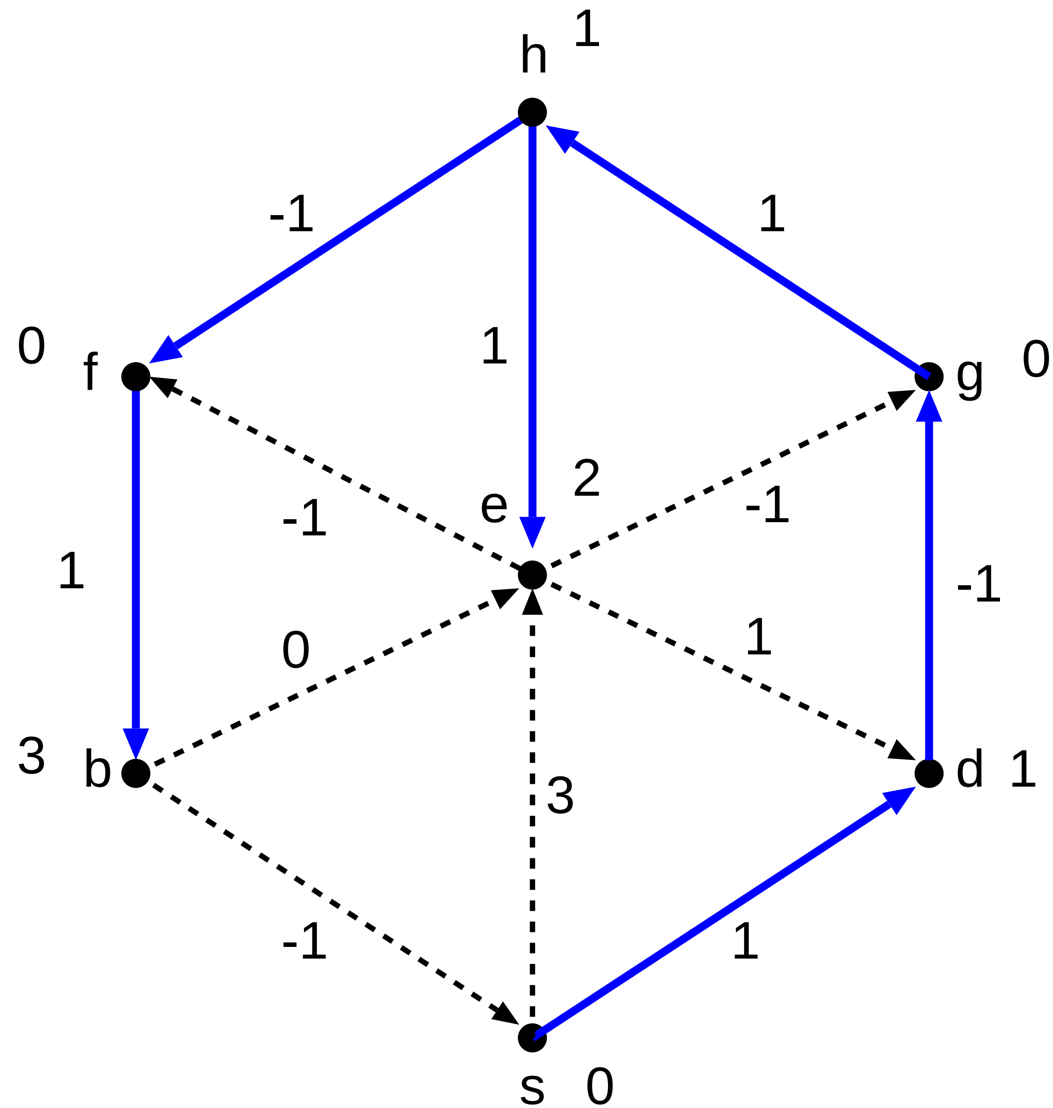
1回目の Step 2: h から出る枝 (h, f)



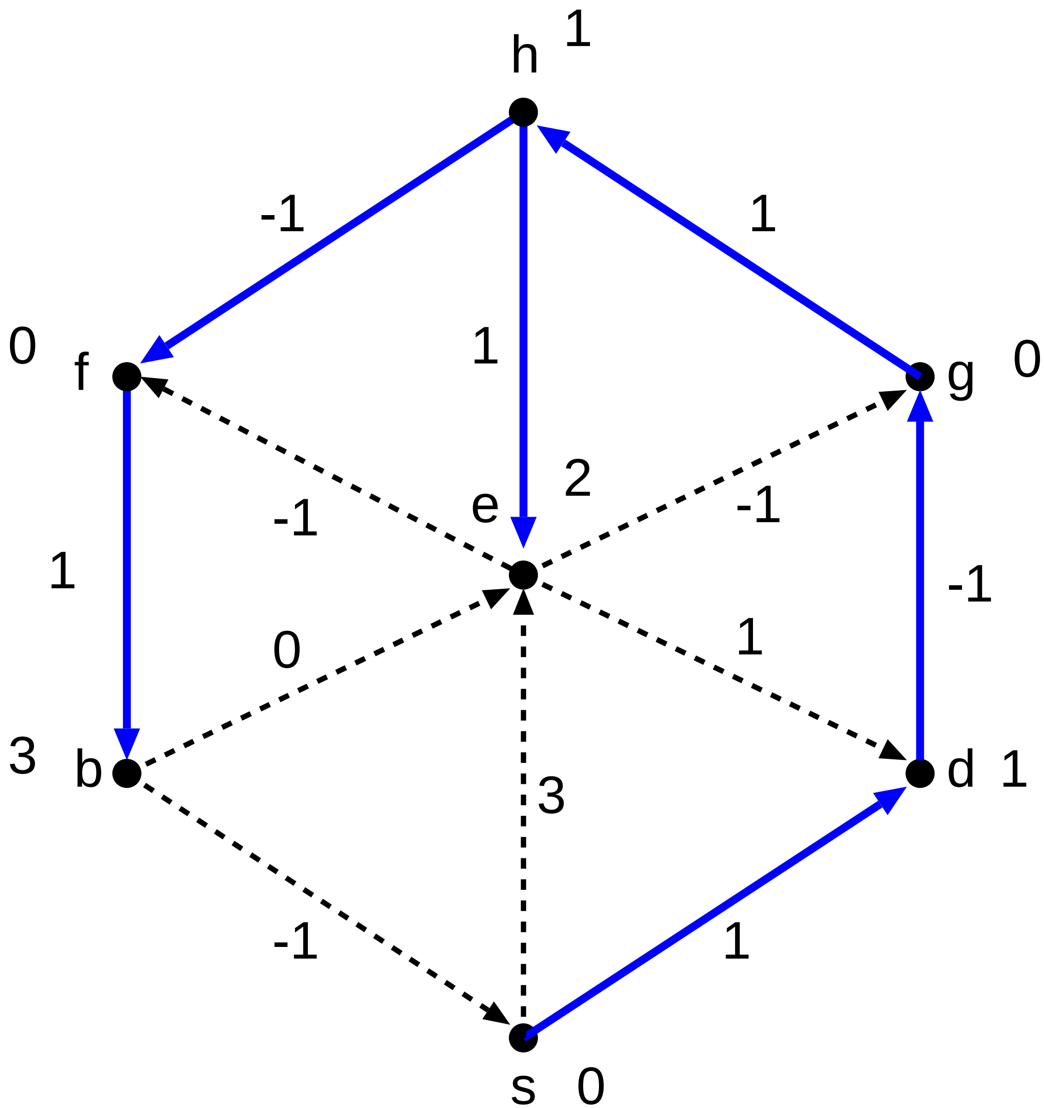
1回目の Step 2: h から出る枝 (h, e)



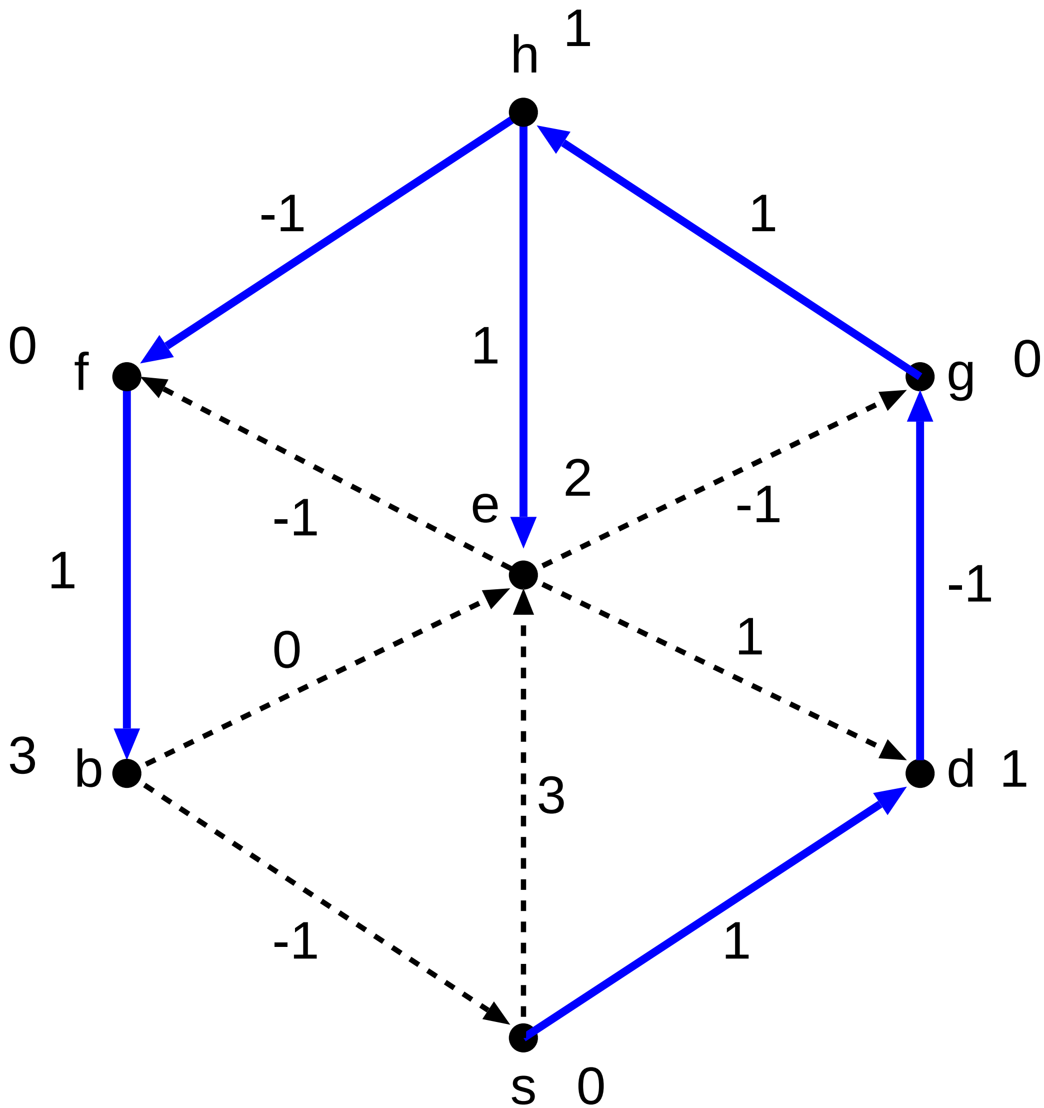
1回目の Step 2の終了時



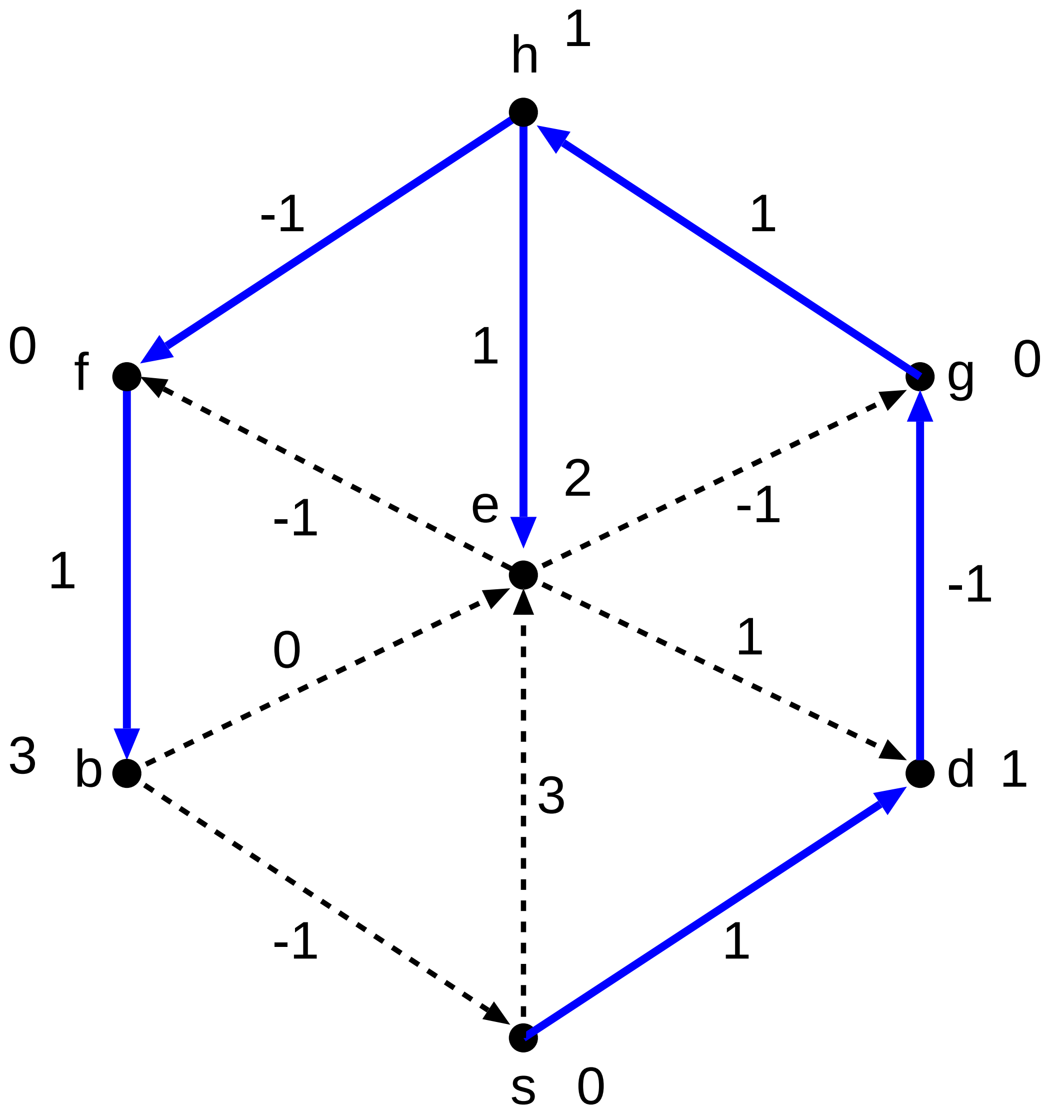
2回目の Step 2: s から出る枝 $(s, e), (s, d)$



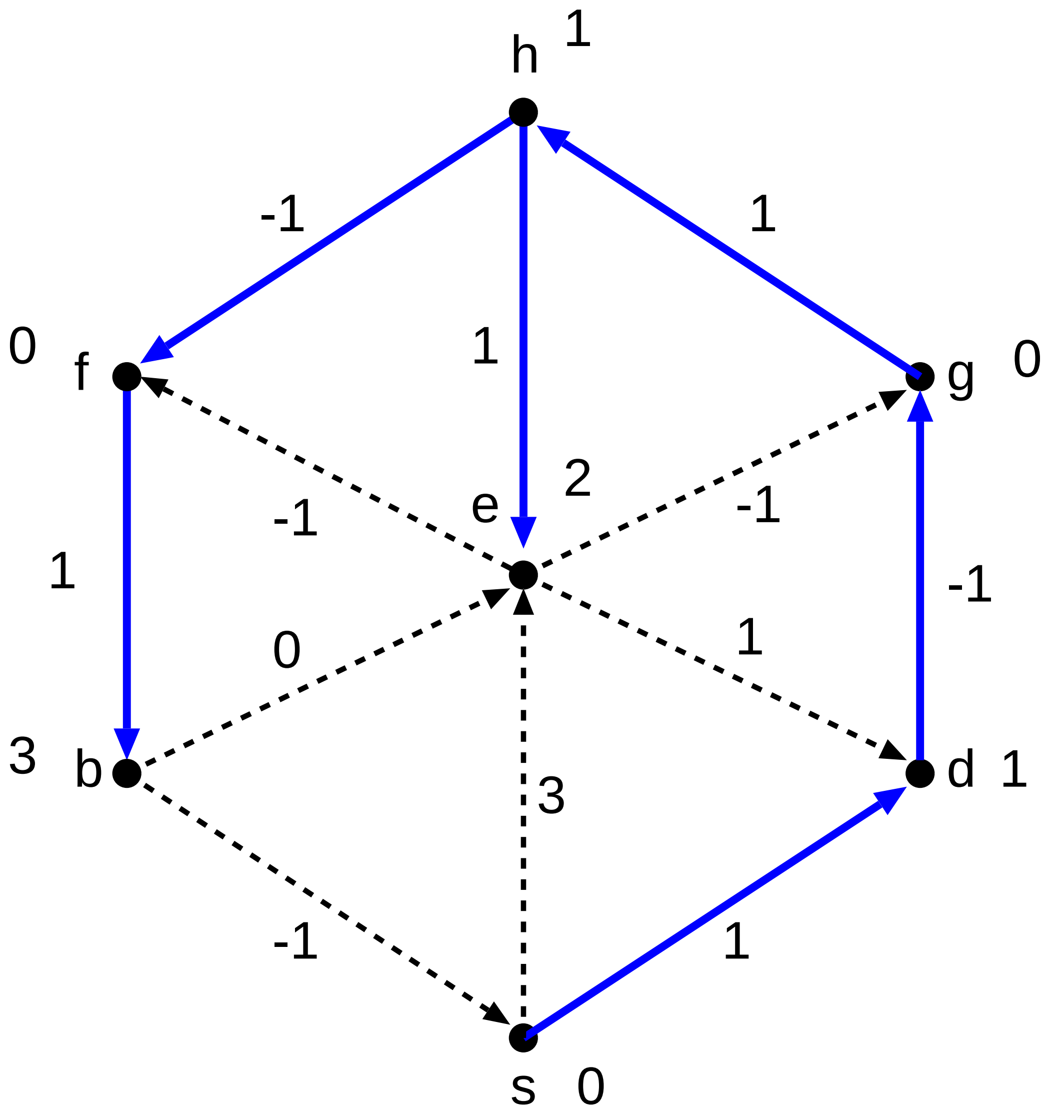
2回目の Step 2: bから出る枝$(b,e),(b,s)$



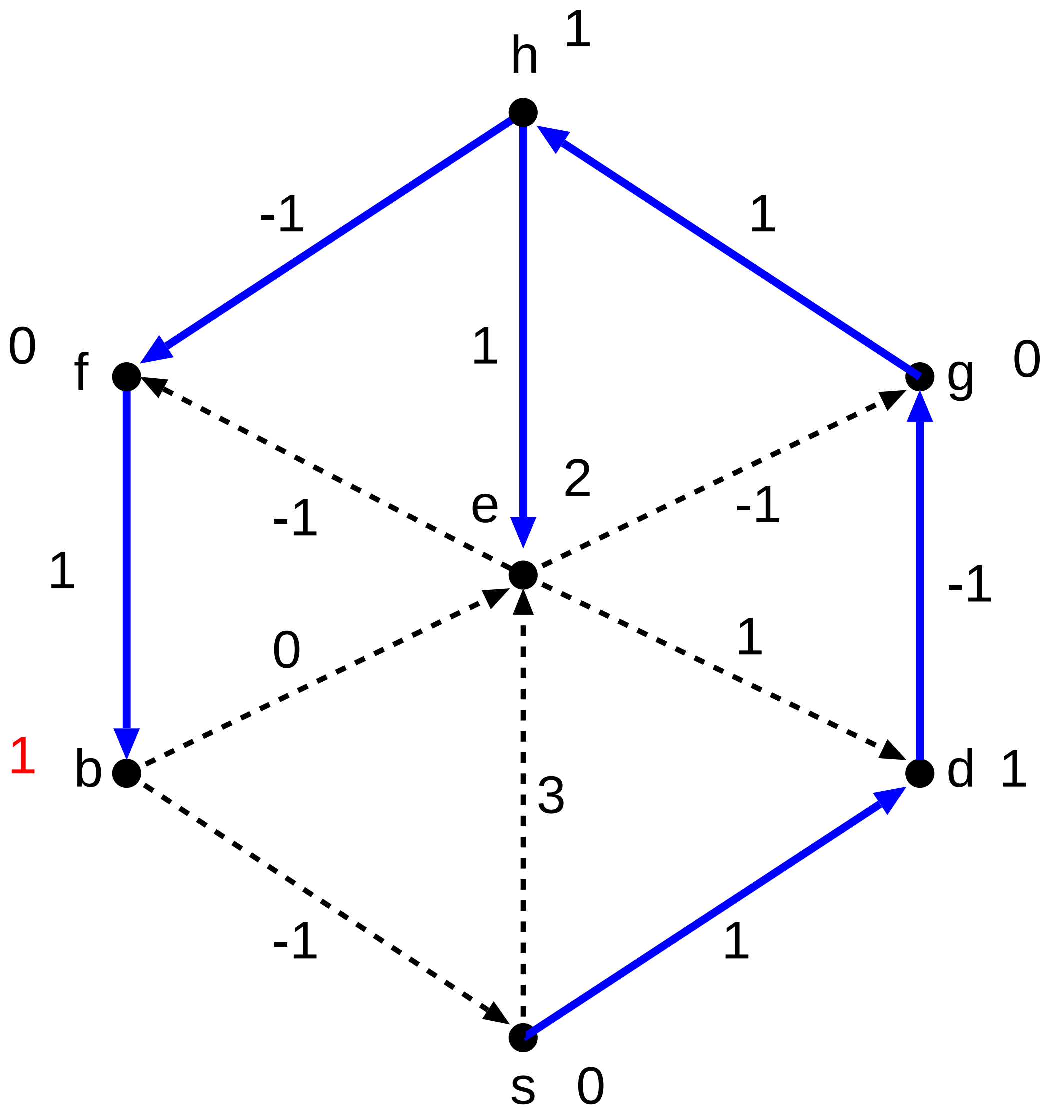
2回目の Step 2: dから出る枝(d, g)



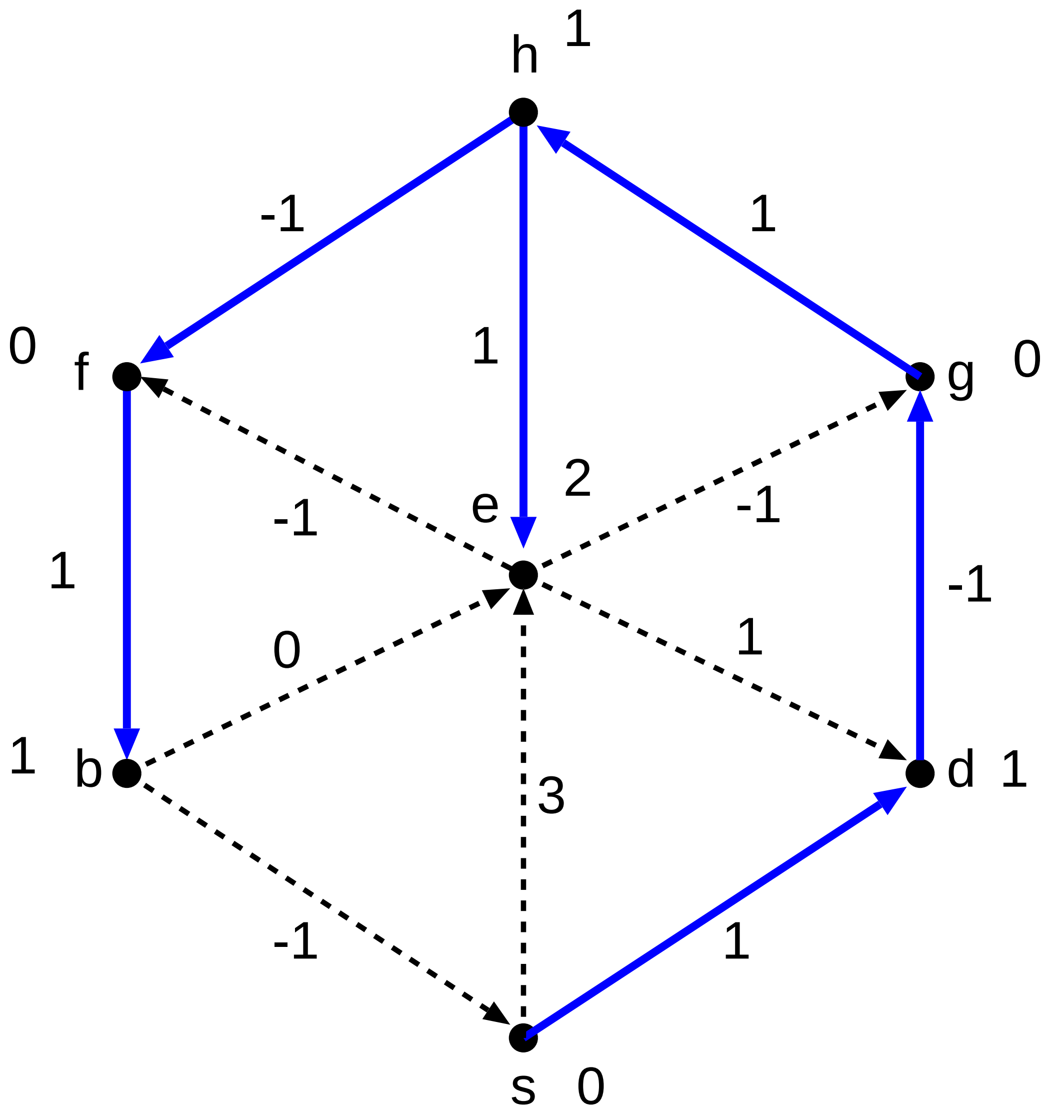
回目の Step 2: e から出る枝 $(e, f), (e, g), (e,$



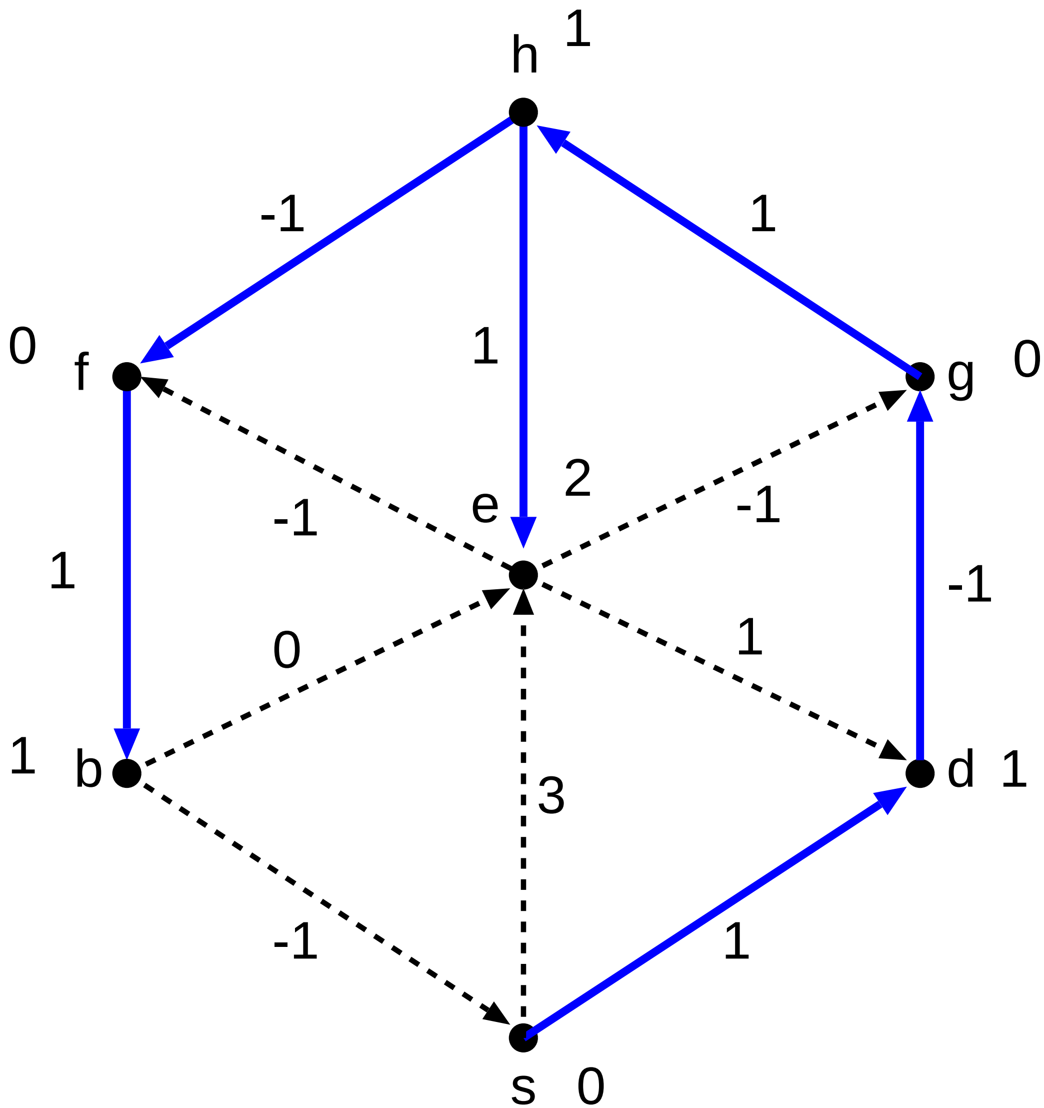
2回目の Step 2: f から出る枝 (f, b)



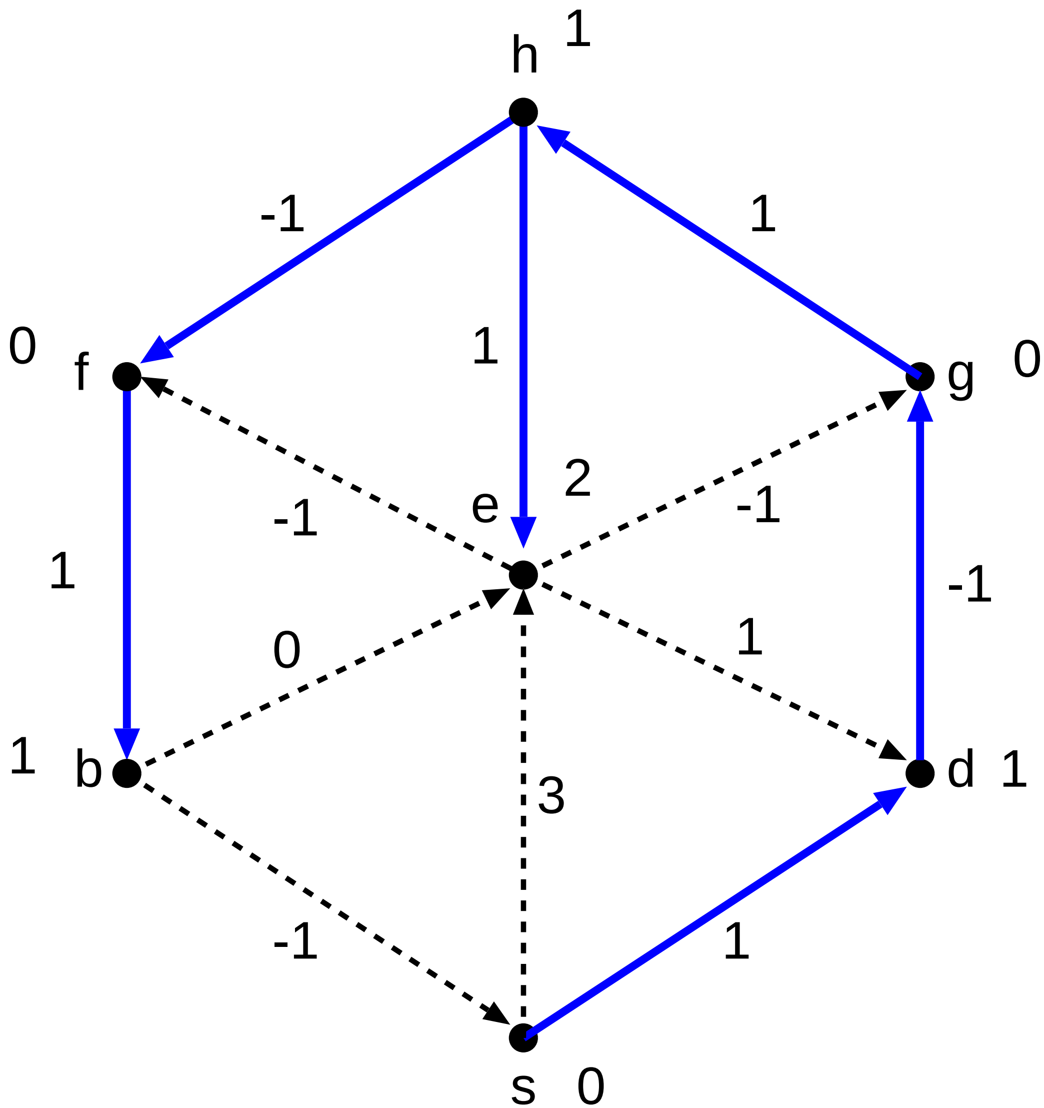
2回目の Step 2: gから出る枝(g,h)



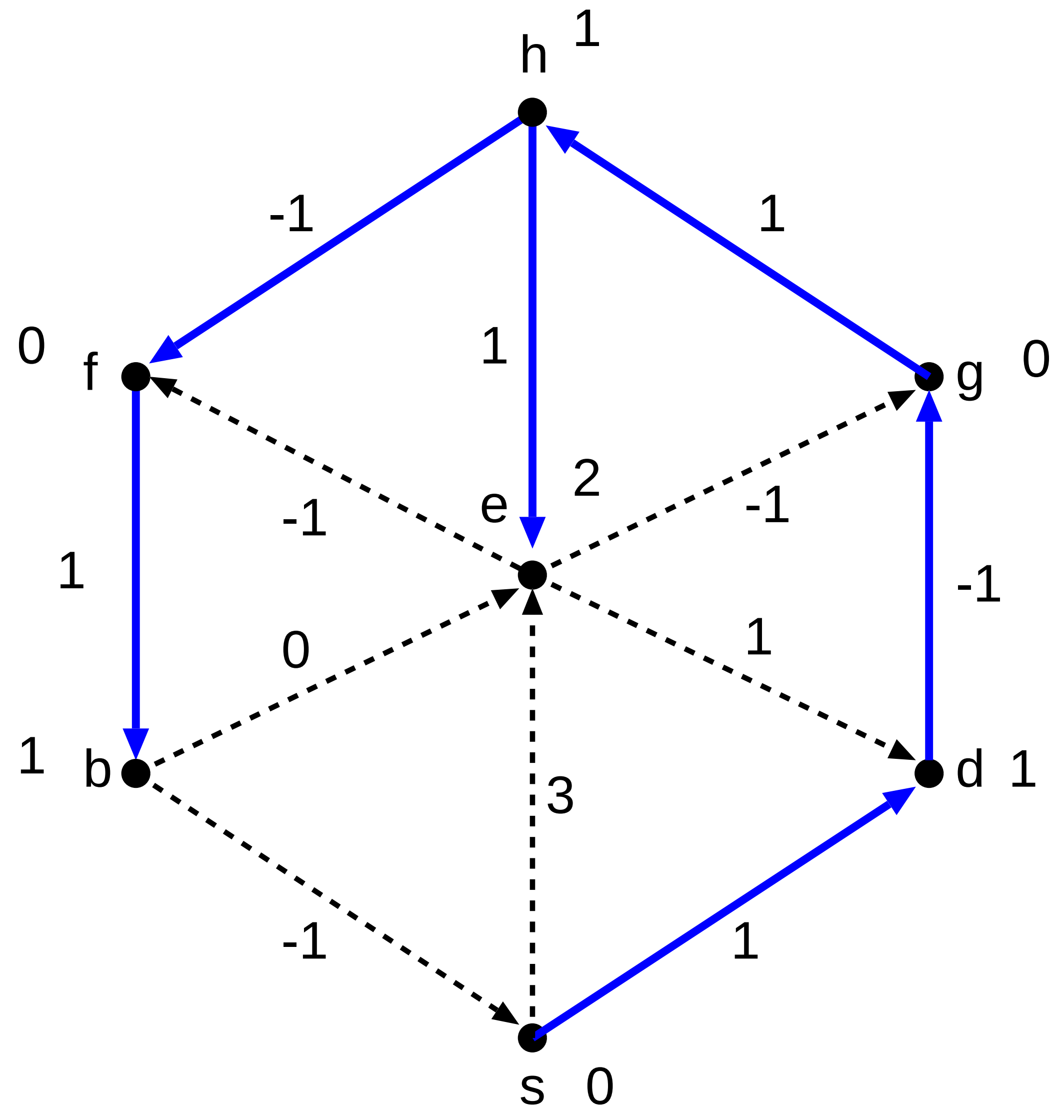
2回目の Step 2: h から出る枝 $(h, f), (h, e)$



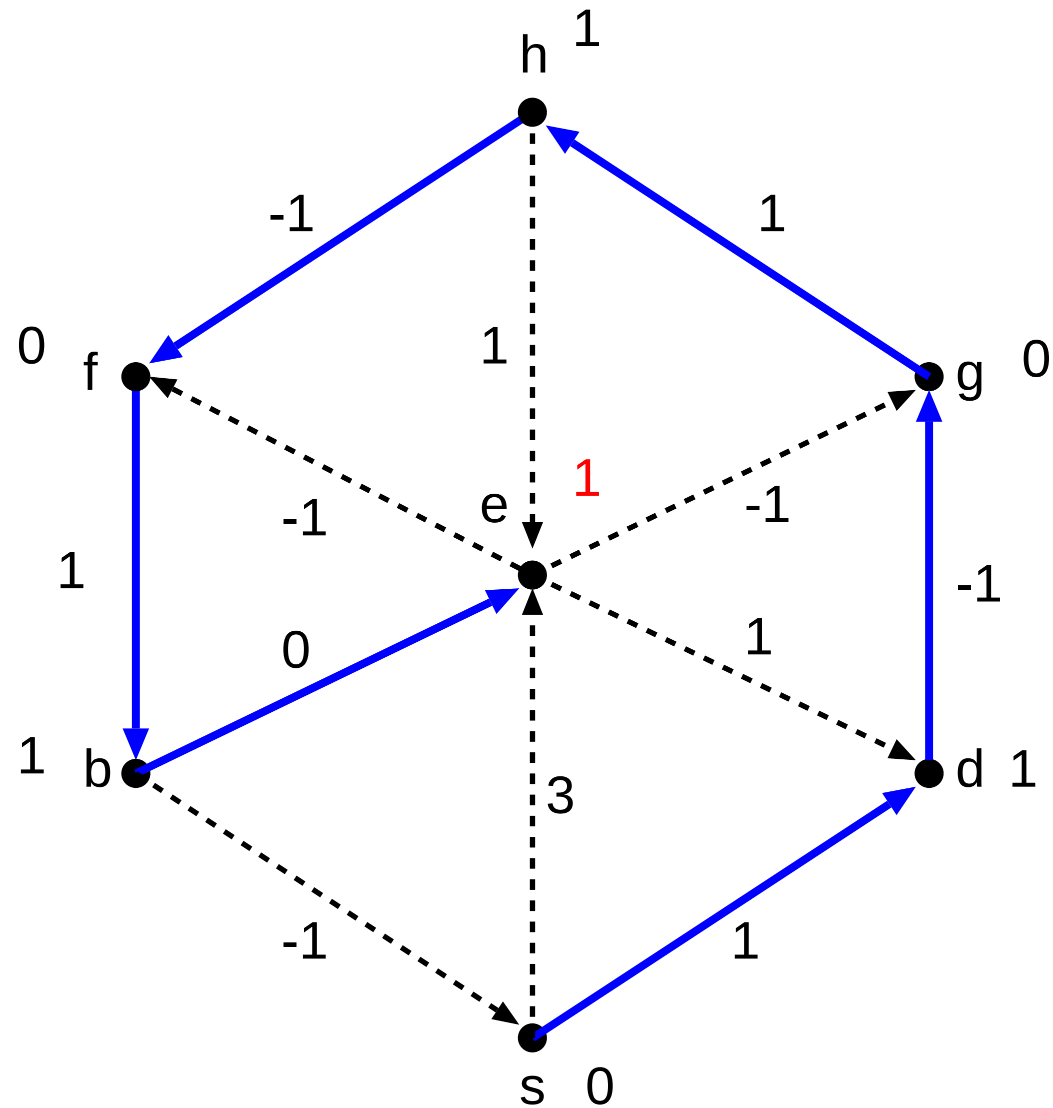
2回目の Step 2の終了時



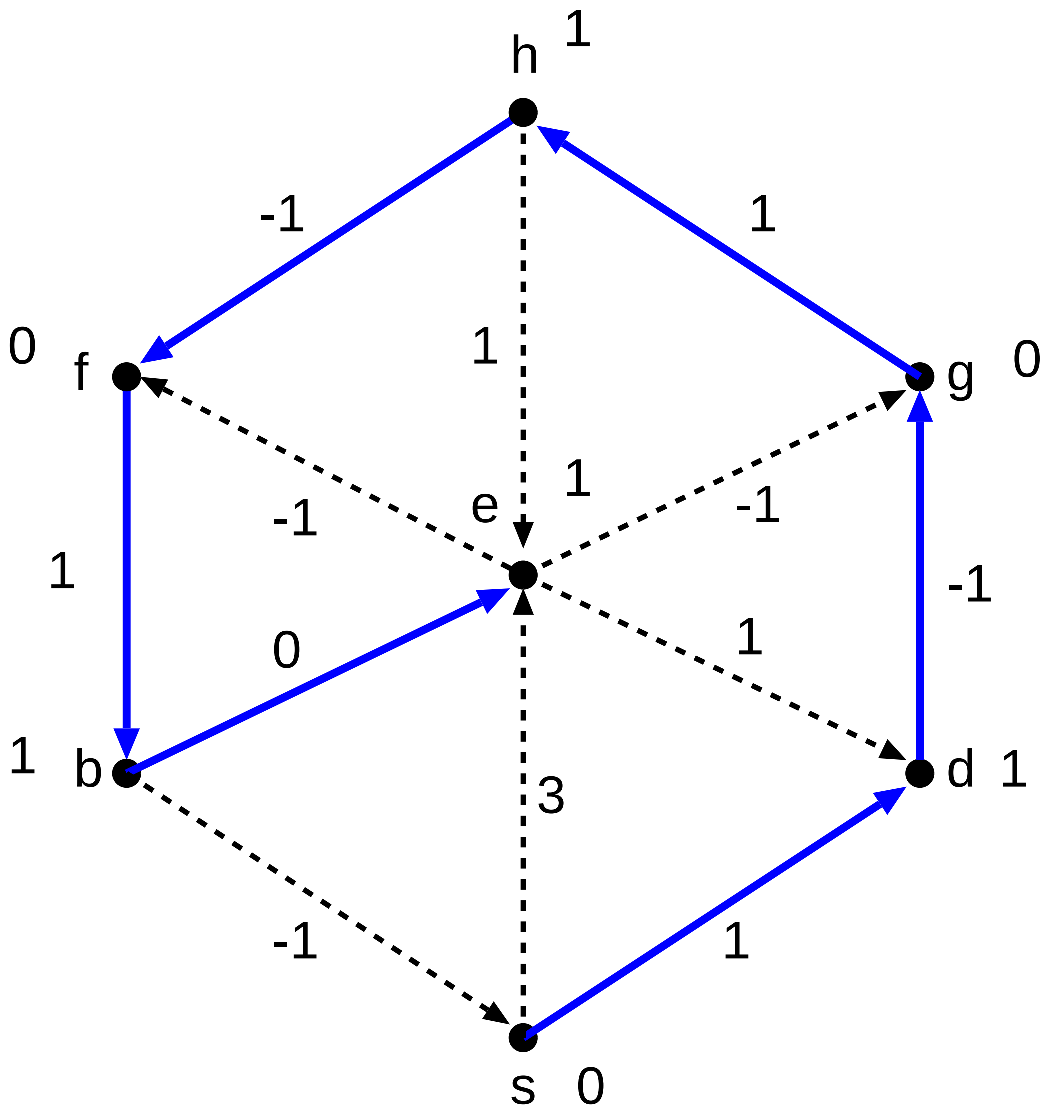
3回目の Step 2: sから出る枝$(h, f), (h, e)$



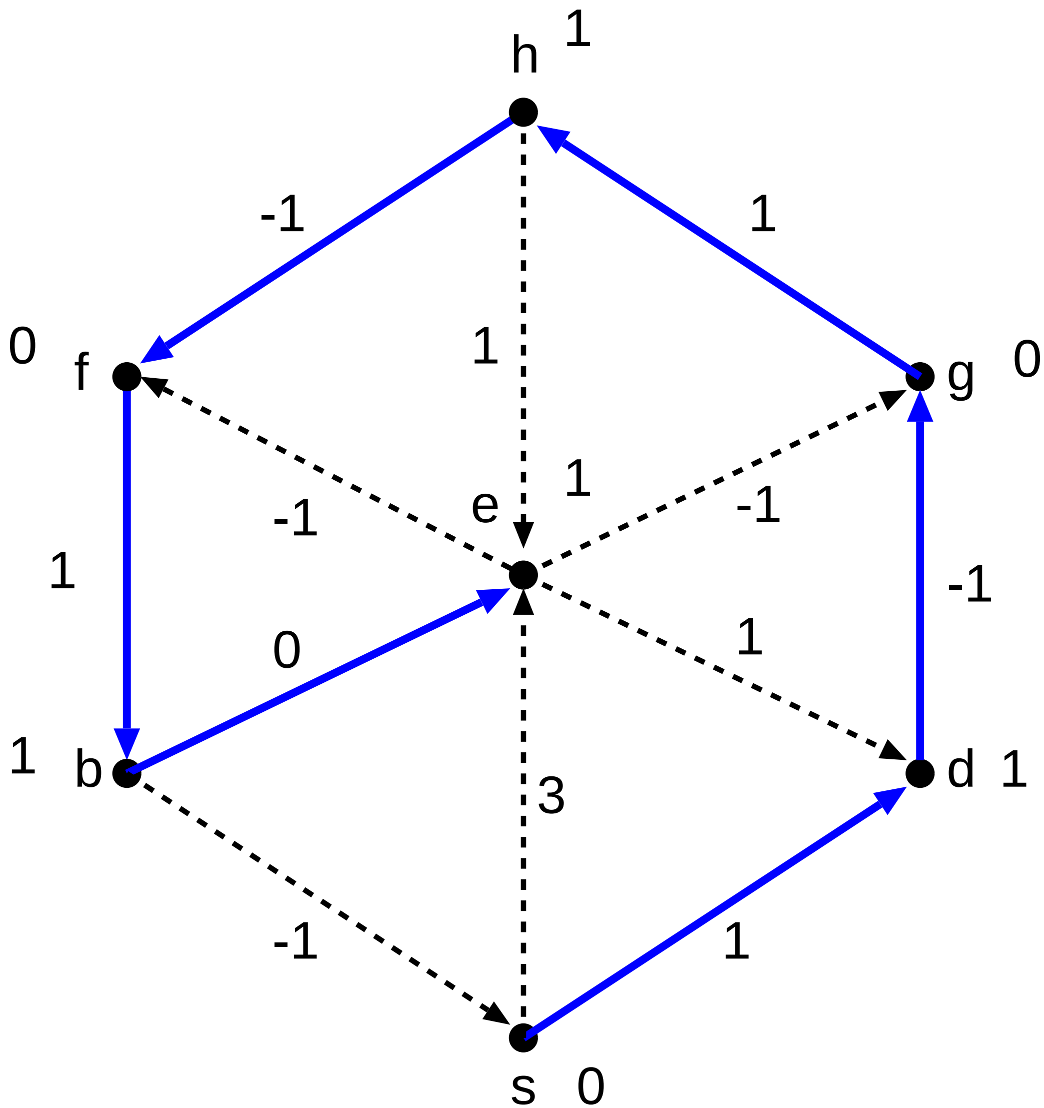
3回目の Step 2: bから出る枝$(b, e), (b, s)$



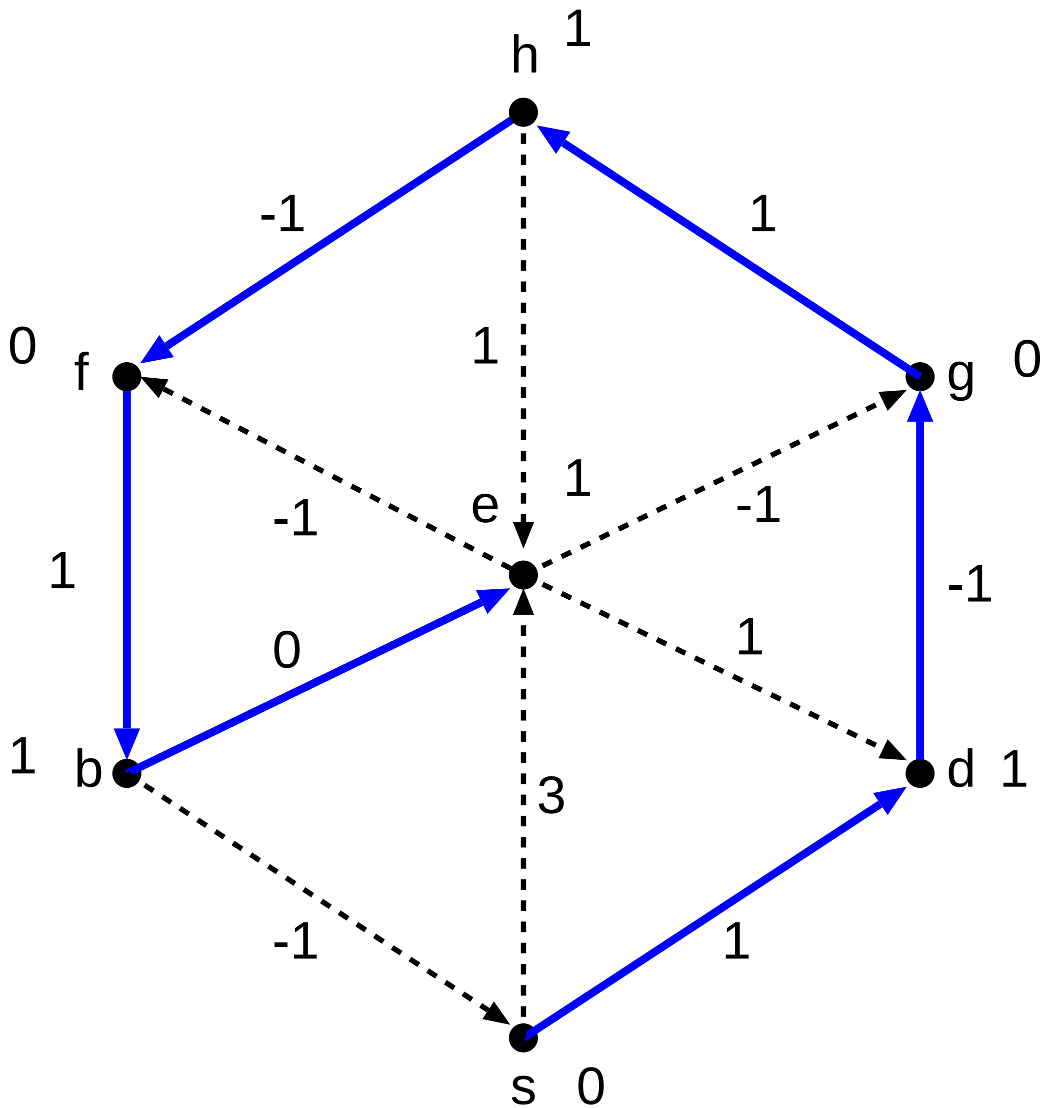
3回目の Step 2: d から出る枝 (d, g)



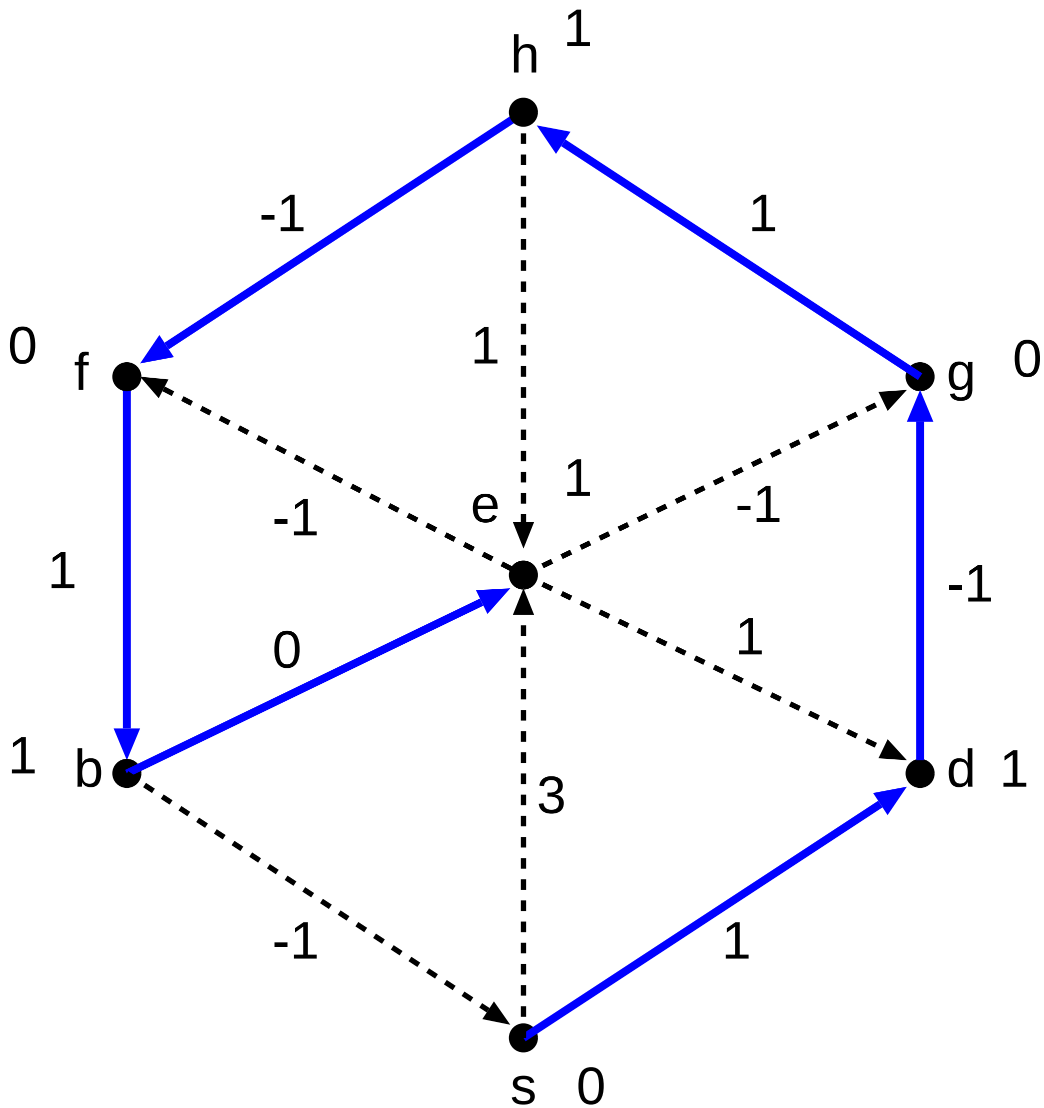
回目の Step 2: e から出る枝 $(e, f), (e, g), (e,$



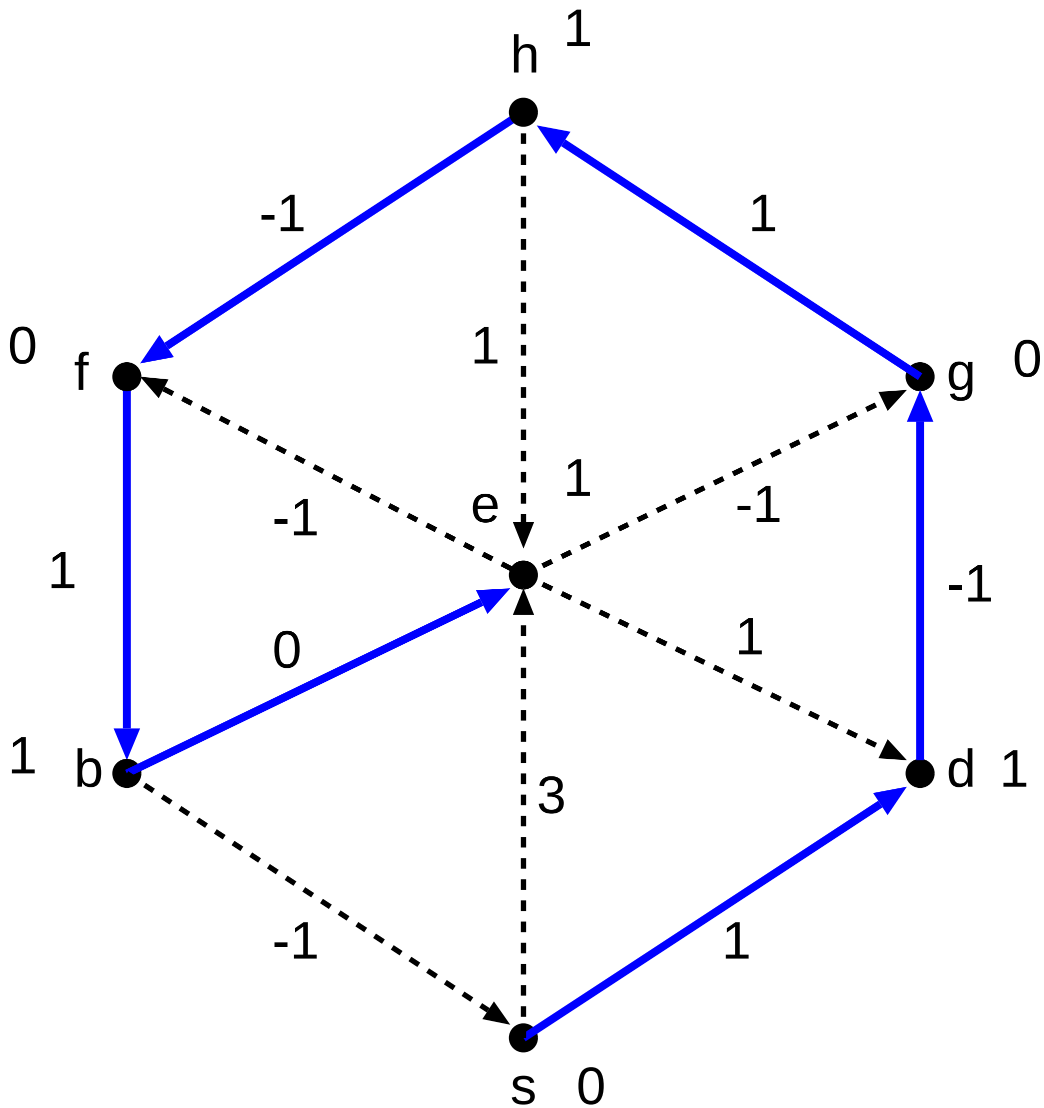
3回目の Step 2: f から出る枝 (f, b)



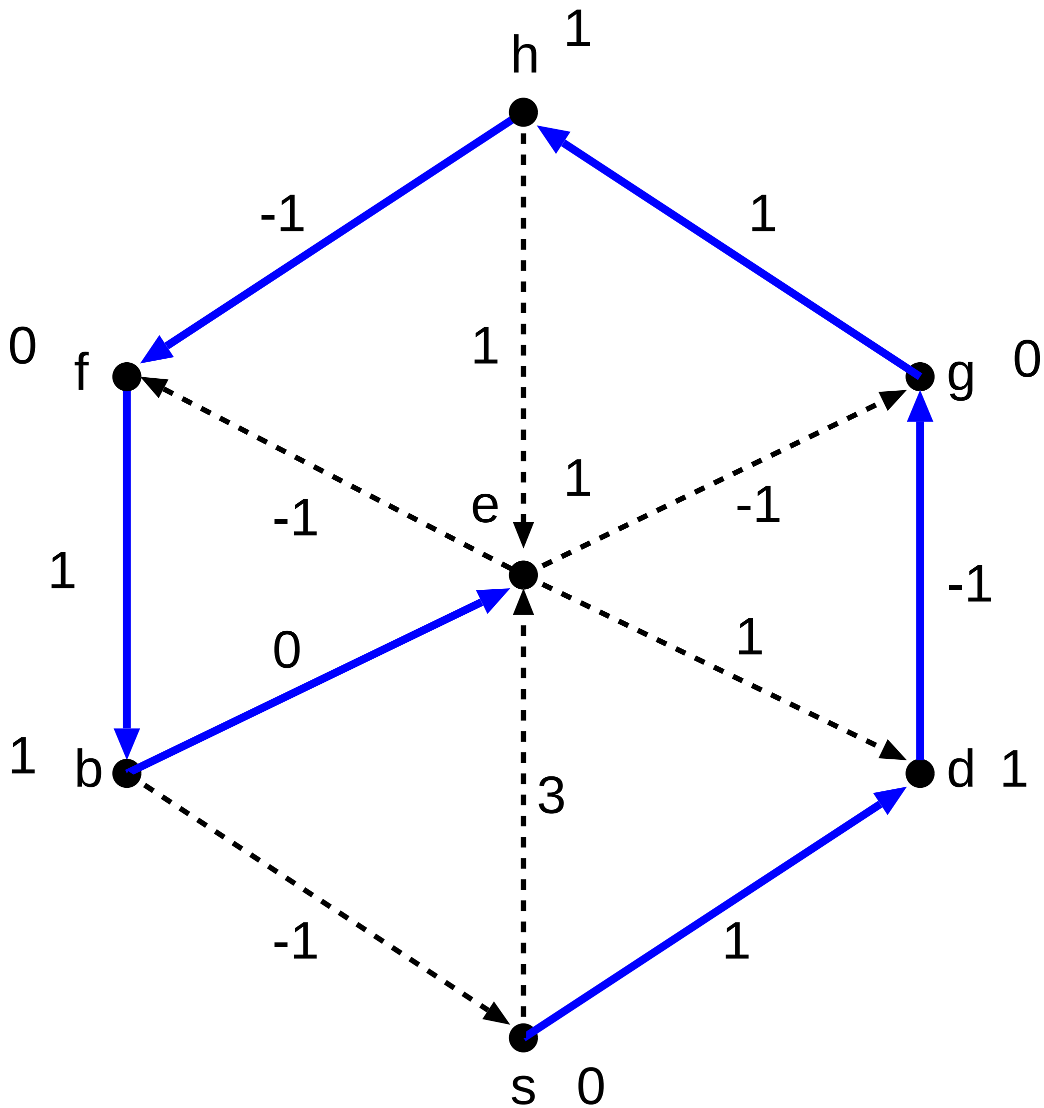
3回目の Step 2: g から出る枝 (g, h)



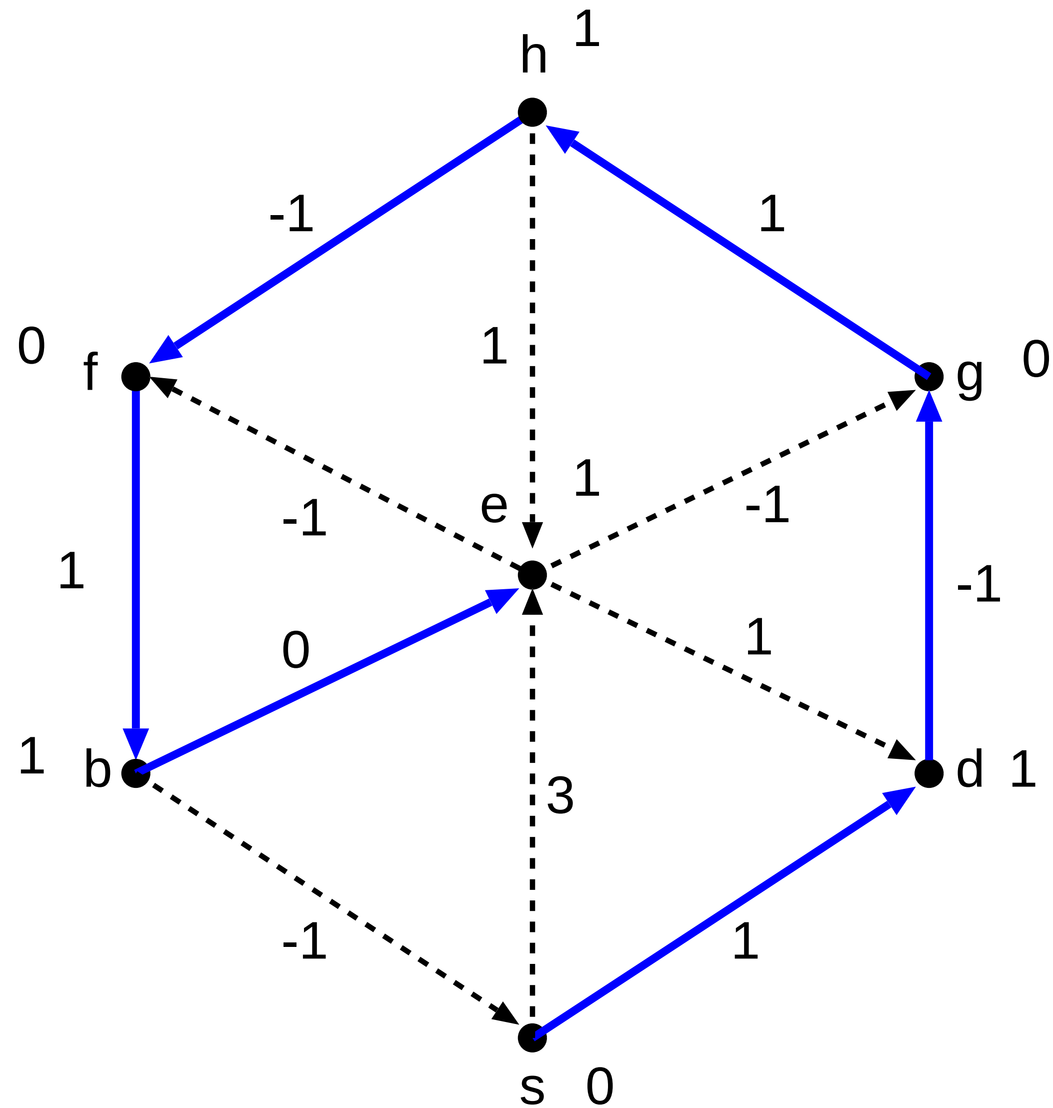
3回目の Step 2: h から出る枝 $(h, f), (h, e)$



3回目の Step 2の終了時



4回目の Step 2: p の更新はされない.



アルゴリズムの妥当性

Step 3 (i) でベルマン-フォード法が終了したとき, 全ての枝 $(v, w) \in A$ に対しては, $p(v) + l(v, w) \geq p(w)$ が成り立つ. 即ち,

$$l_p(v, w) = l(v, w) + p(v) - p(w) \geq 0.$$

さらに, 青い枝たち $(q(u), u)$ $(u \in V \setminus \{v_0\})$ は v_0 を根とする有向木を成し, この有向木上の枝 (v, w) に対しては,

$$l_p(v, w) = l(v, w) + p(v) - p(w) = 0.$$

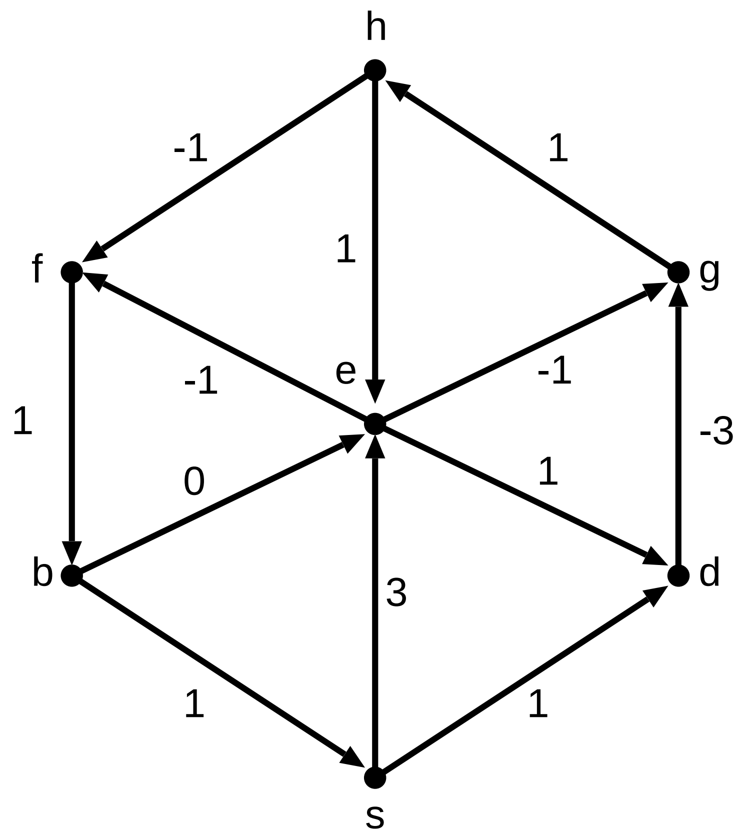
補題 2.2($\Rightarrow$ p. 43) から, v_0 からこの有向木上の道が最短路である.

アルゴリズムの妥当性

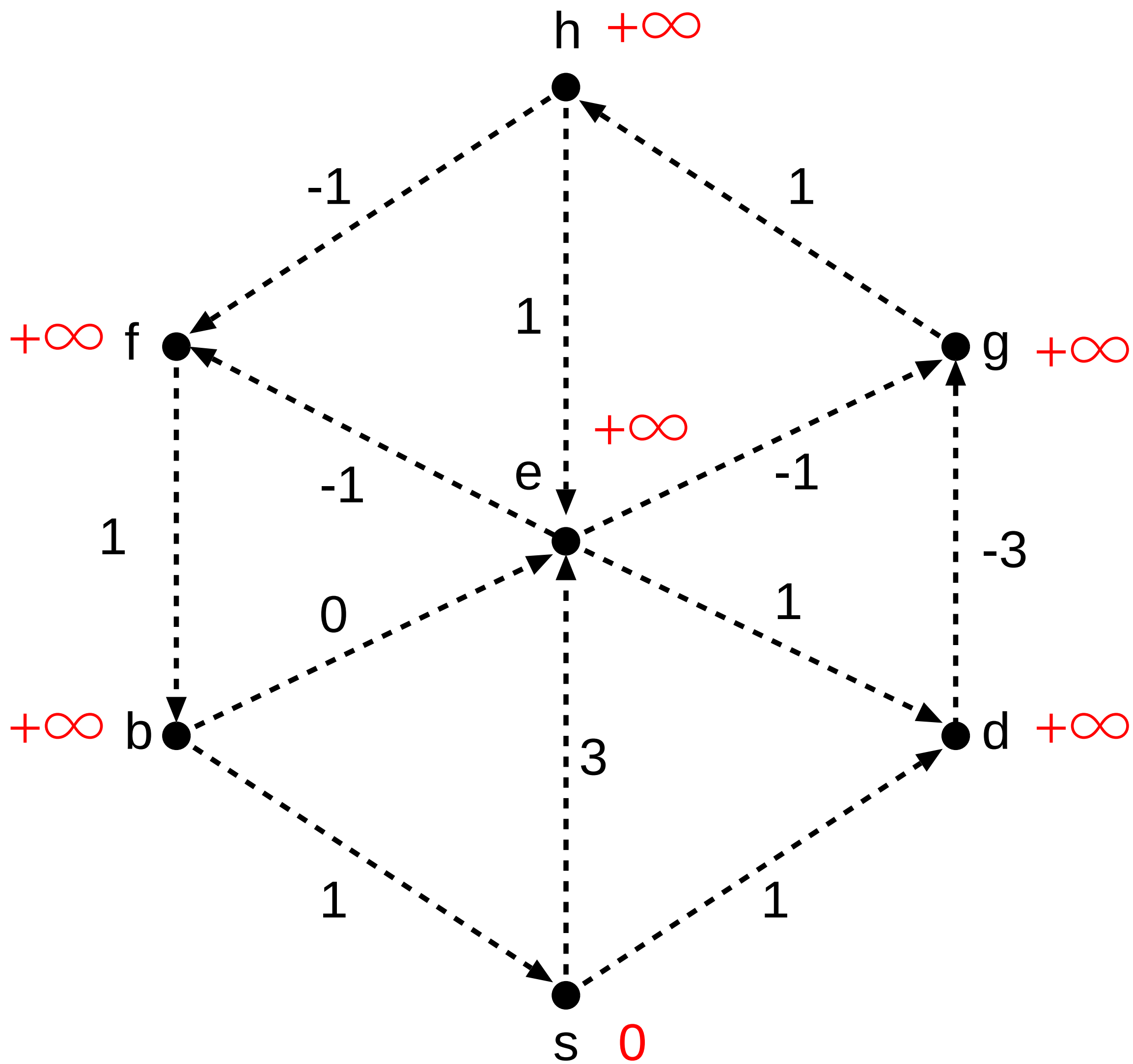
Step 3 (ii) で終了したときには負の長さの有向閉路が存在する. その証明は省略する.

ベルマン-フォード法 (例題2)

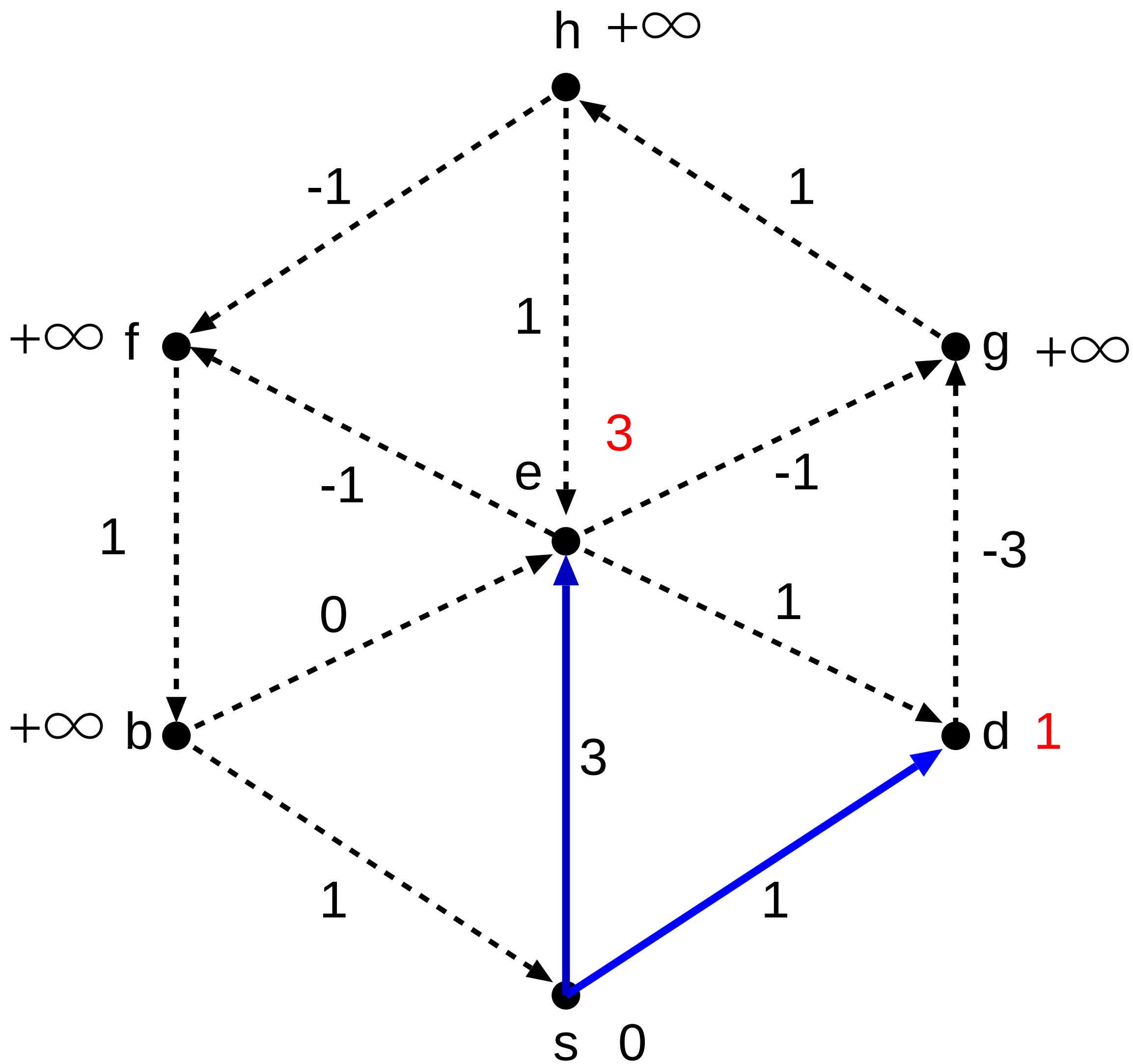
以下に示されるネットワークに対する, 始点を $v_0 = s$ とする最短路問題を, ベルマン-フォード法を用いて解いてみよう.



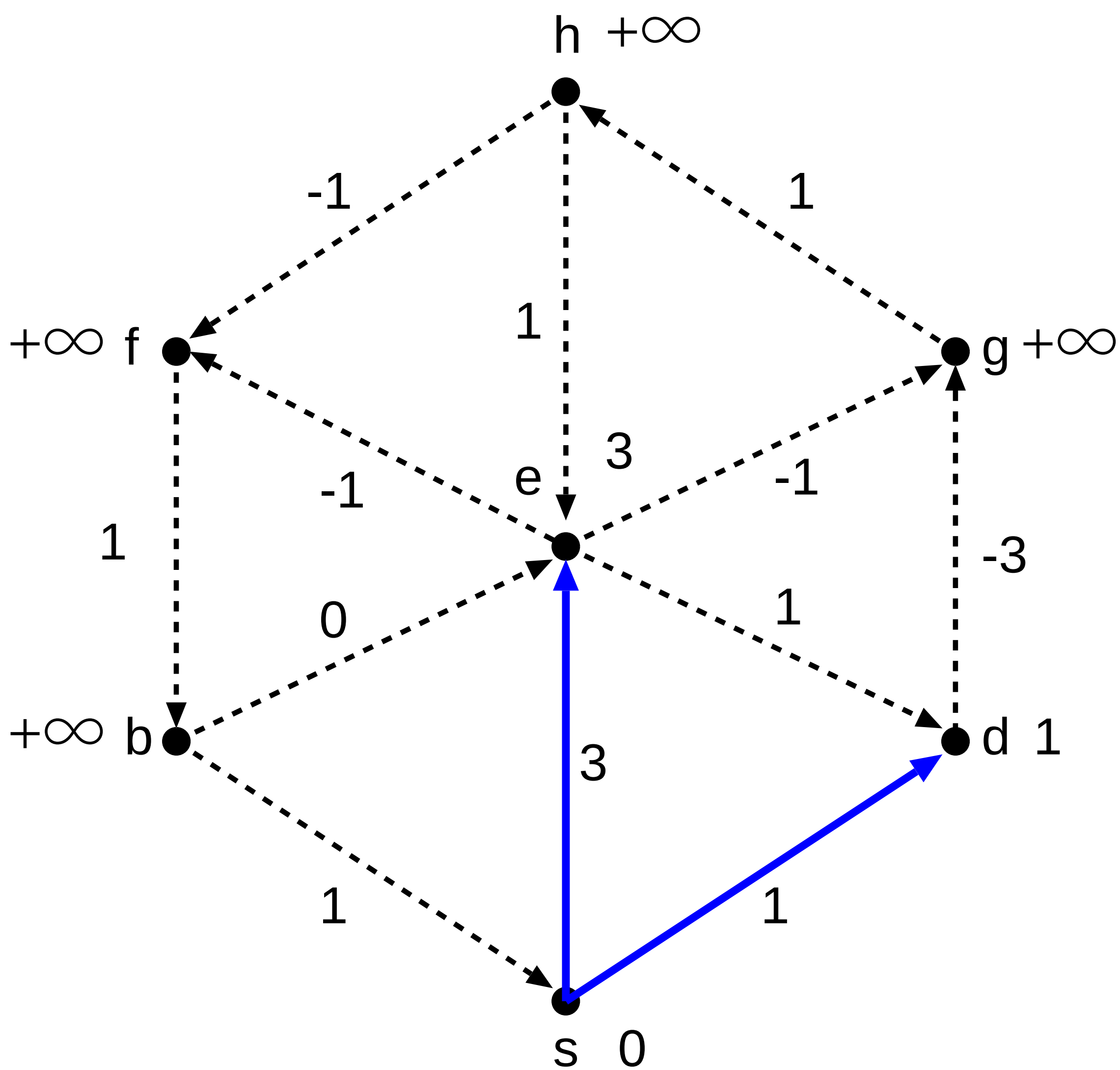
Step 1 終了時



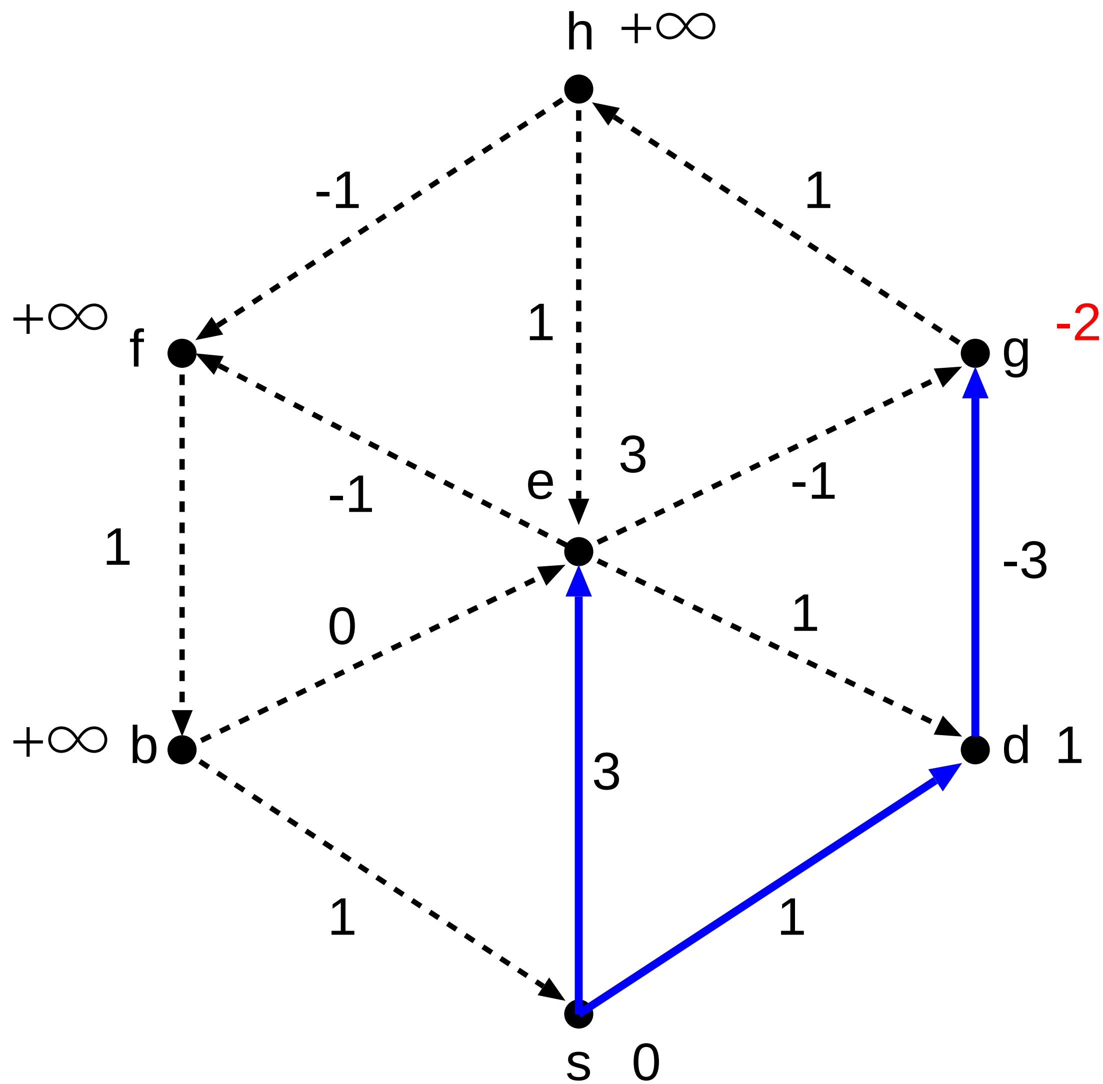
1回目の Step 2: s から出る枝 $(s,d),(s,e)$



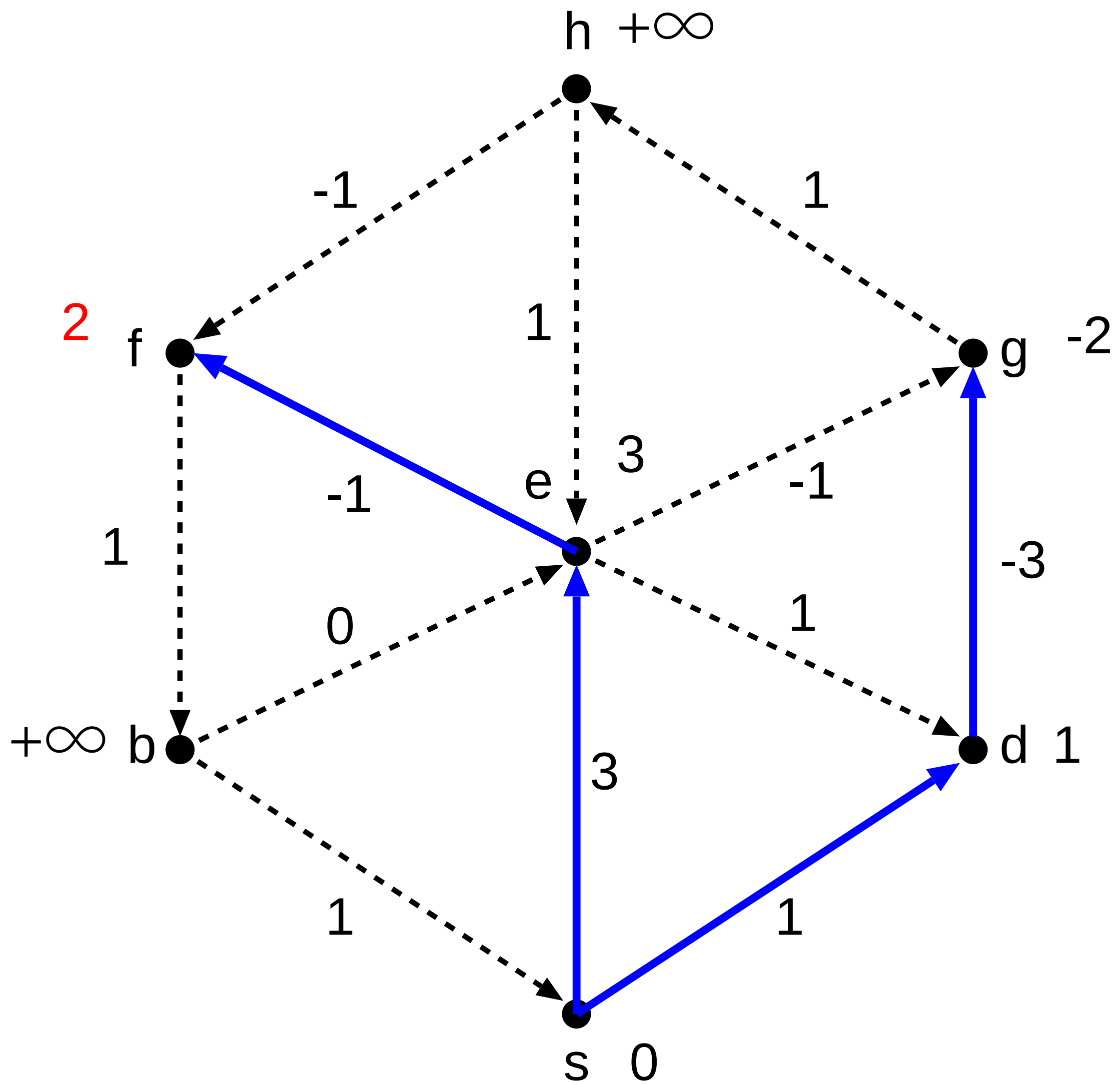
1回目の Step 2: bから出る枝$(b, e), (b, s)$



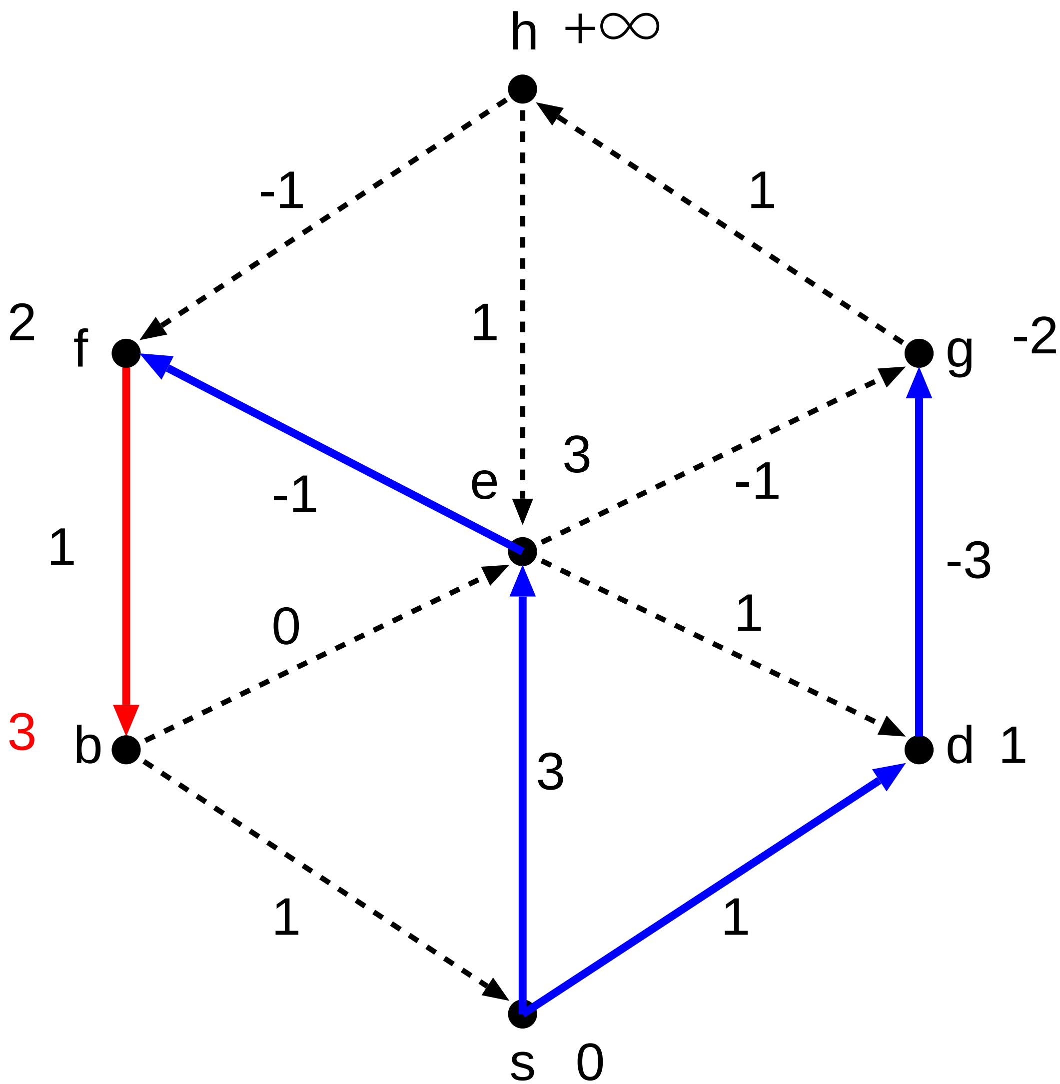
1回目の Step 2: d から出る枝 (d, g)



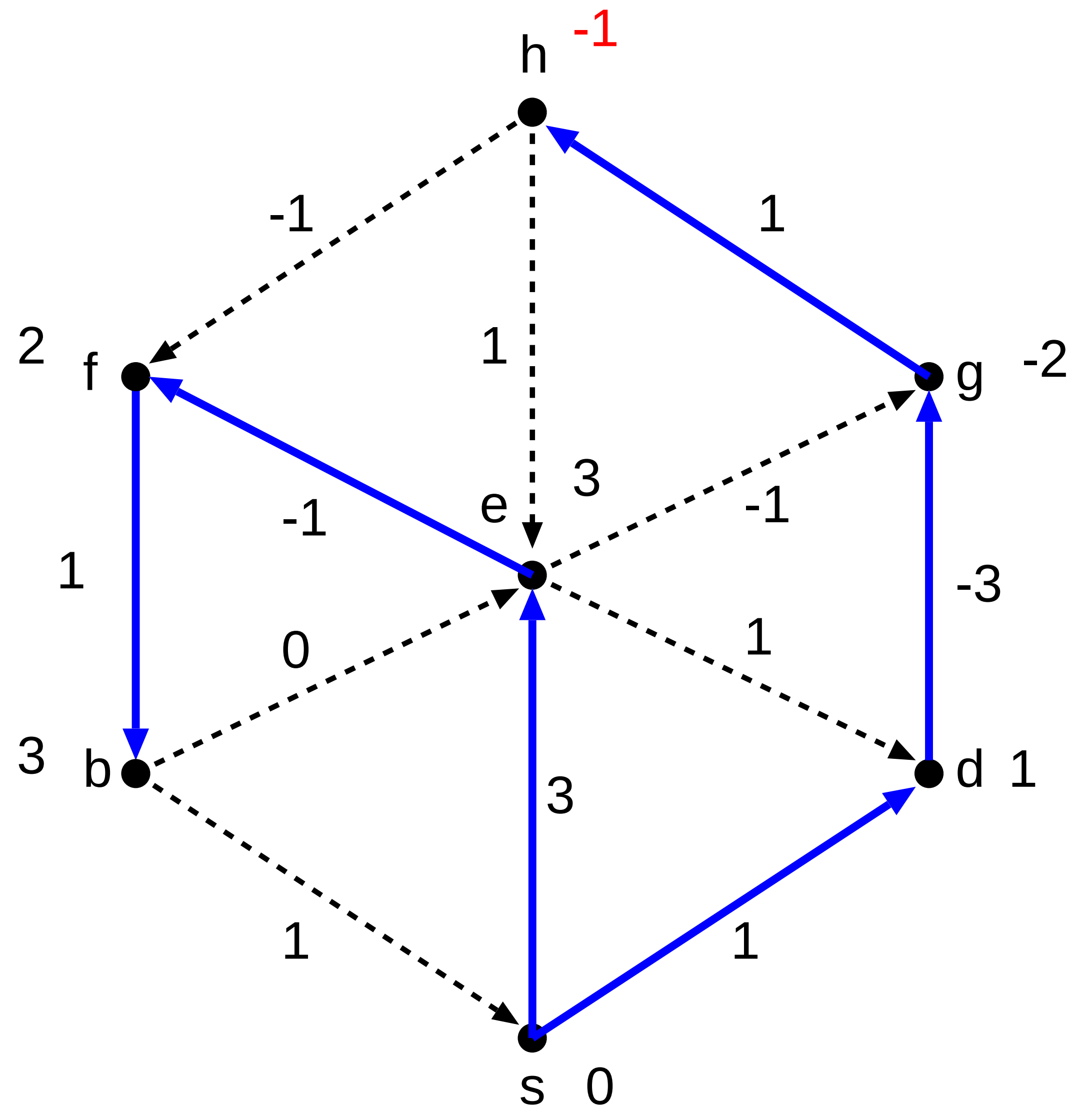
回目の Step 2: e から出る枝$(e, f), (e, g), (e,$



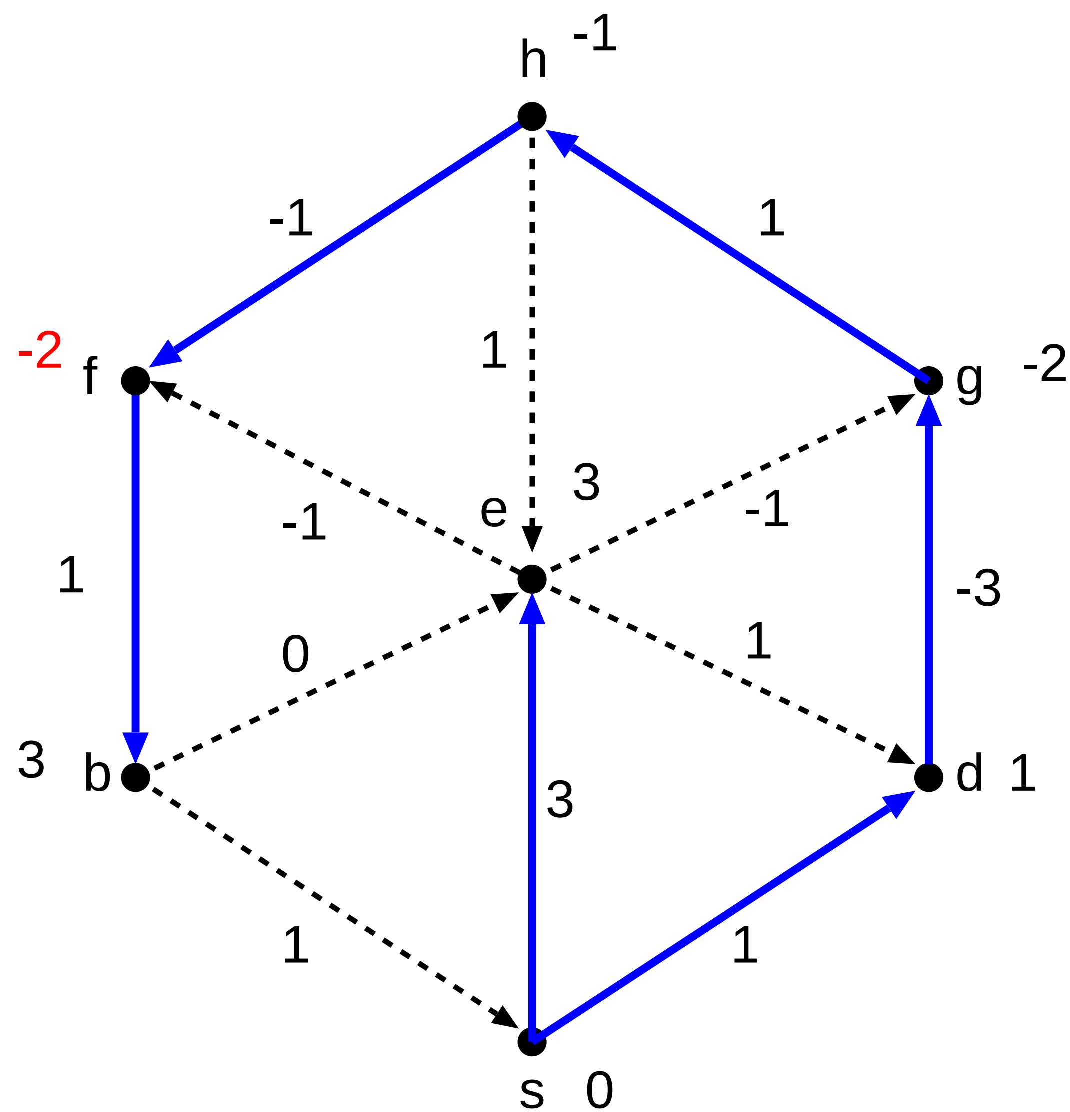
1回目の Step 2: f から出る枝 (f, b)



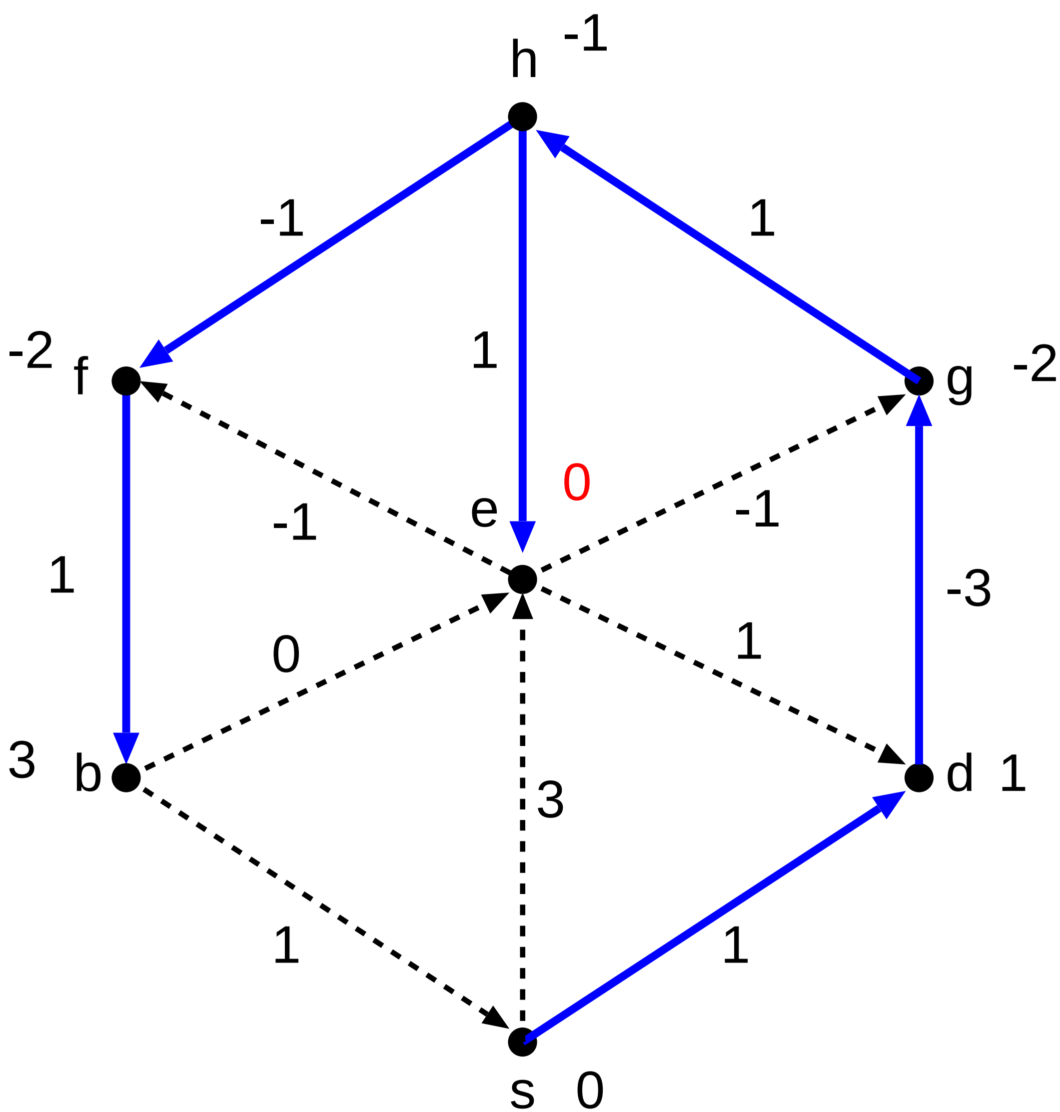
1回目の Step 2: g から出る枝 (g, h)



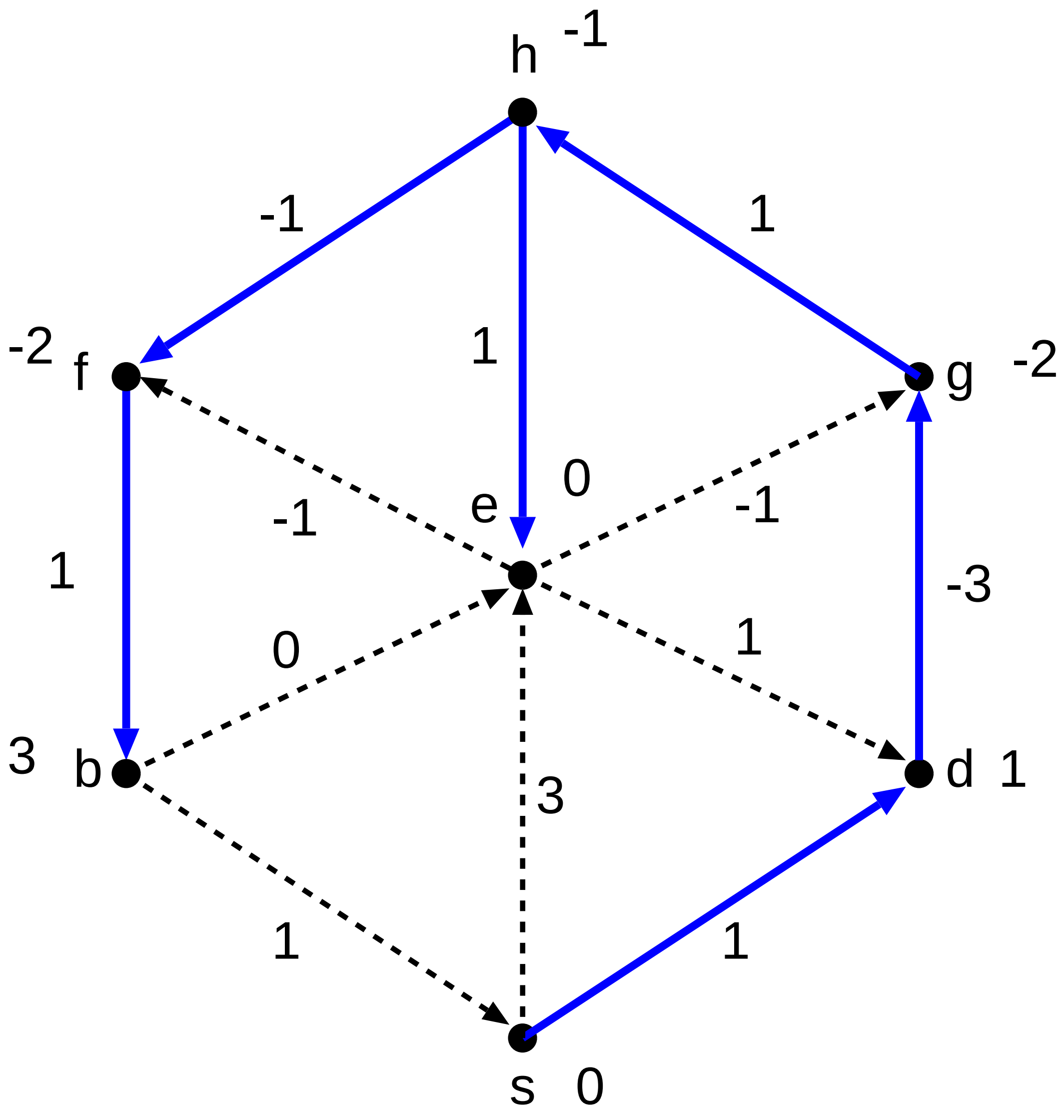
1回目の Step 2: h から出る枝 (h, f)



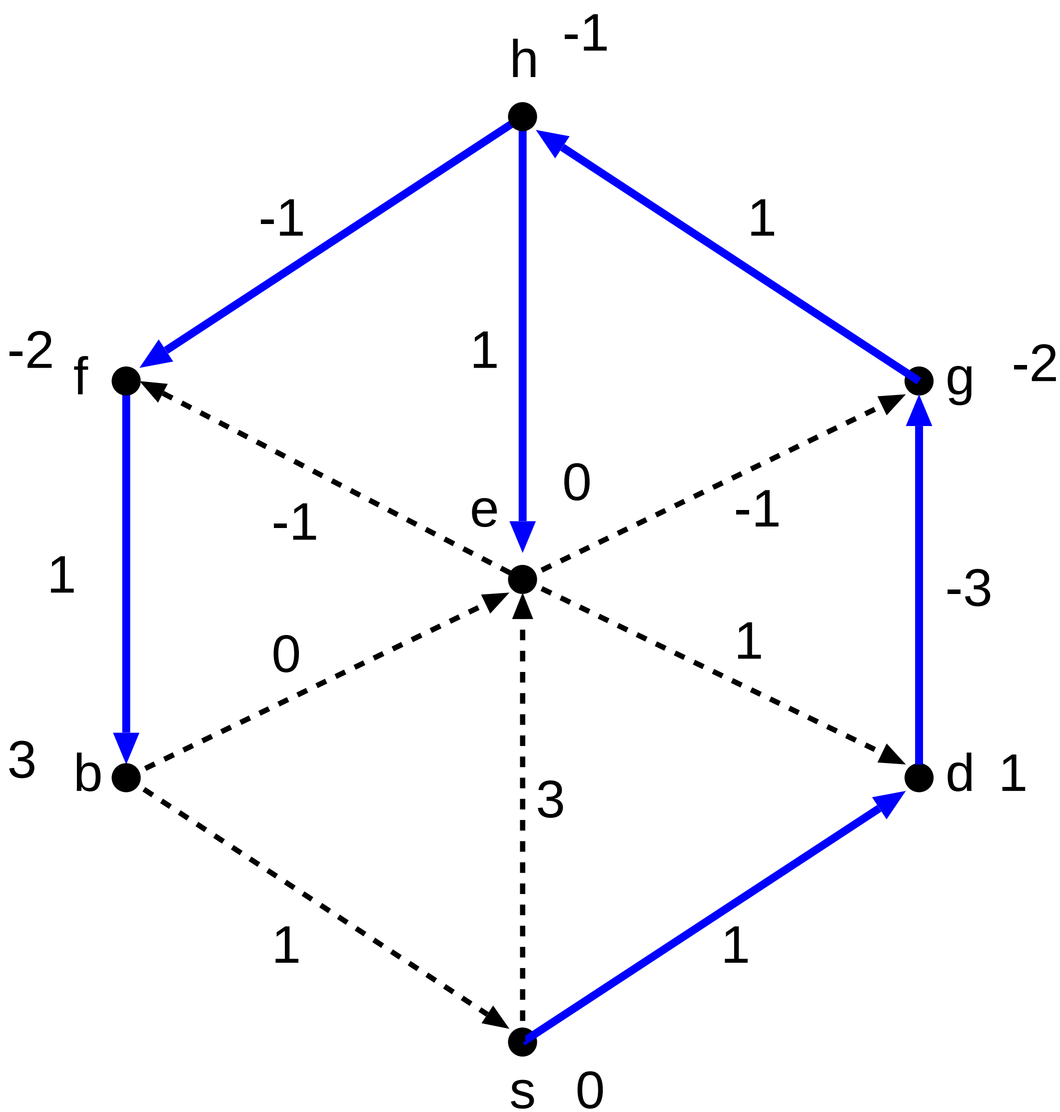
1回目の Step 2: h から出る枝 (h, e)



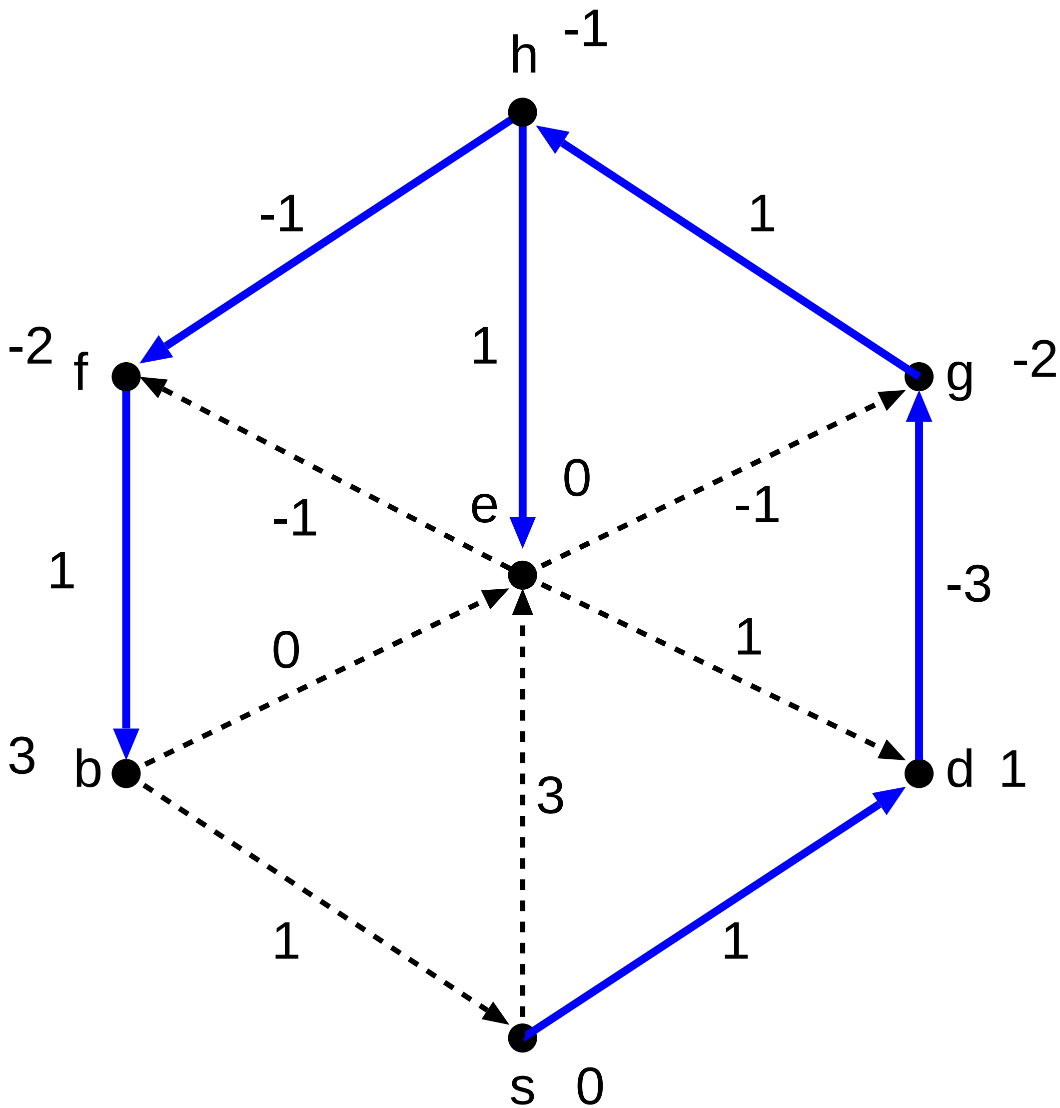
1回目の Step 2の終了時



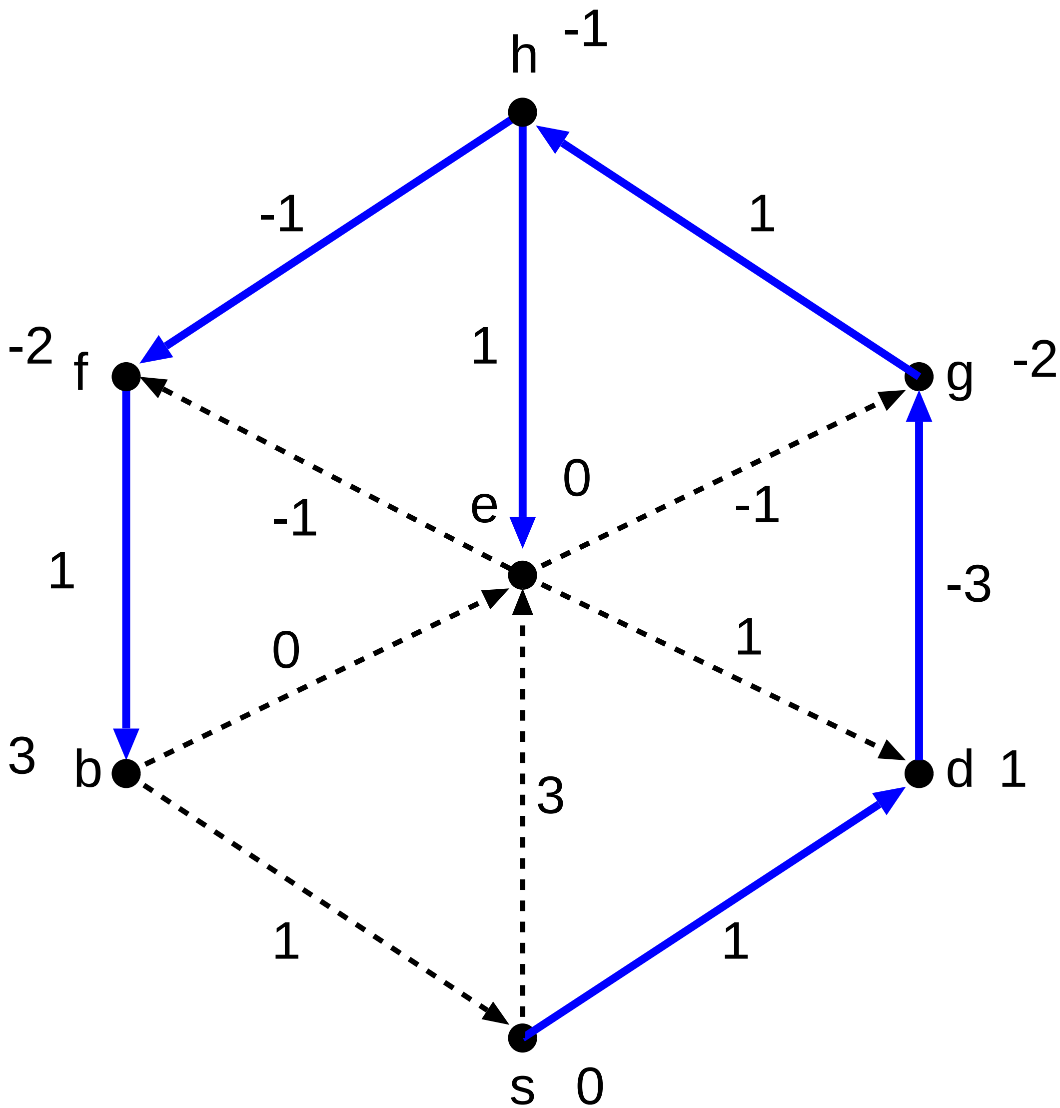
2回目の Step 2: s から出る枝 $(s, e), (s, d)$



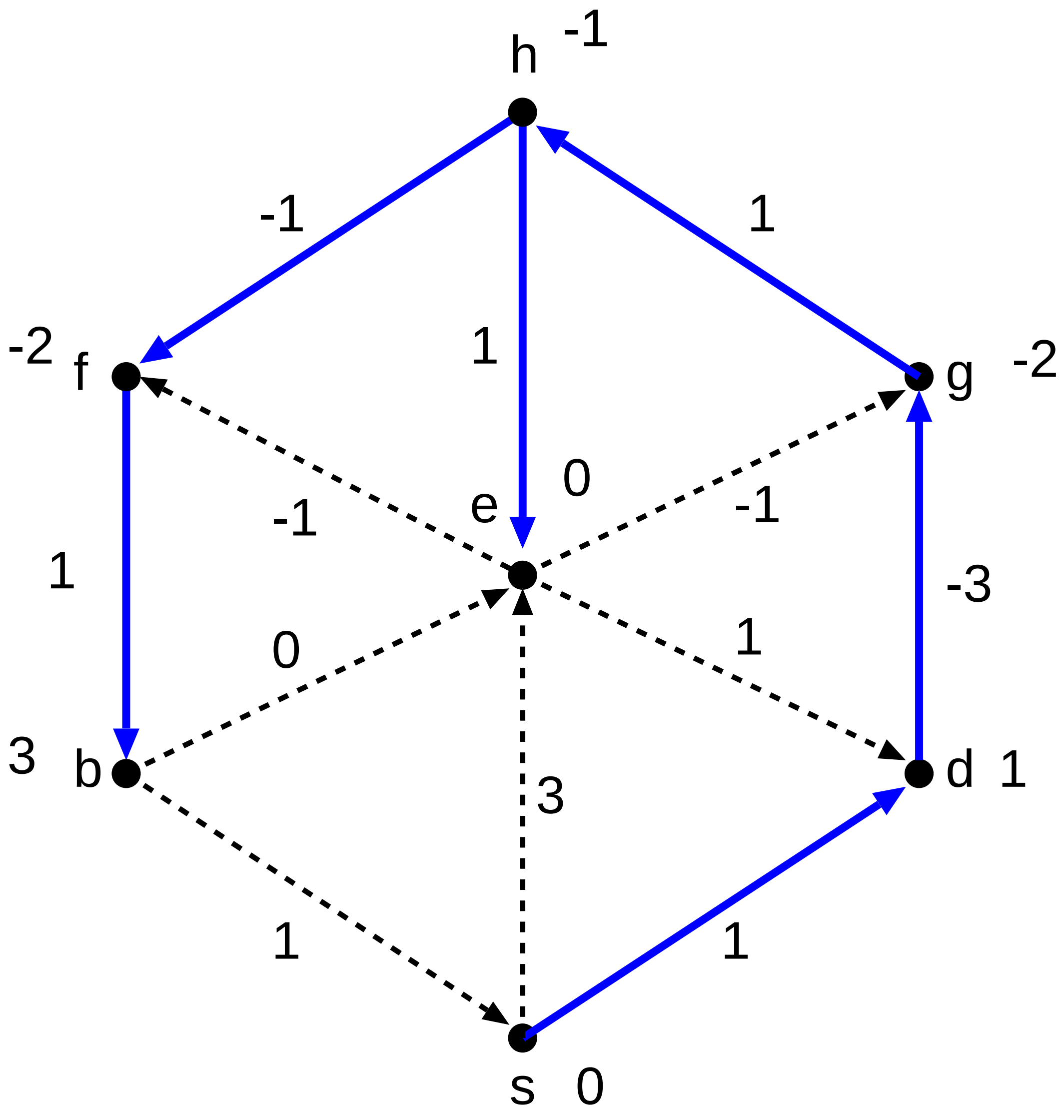
2回目の Step 2: bから出る枝$(b, e), (b, s)$



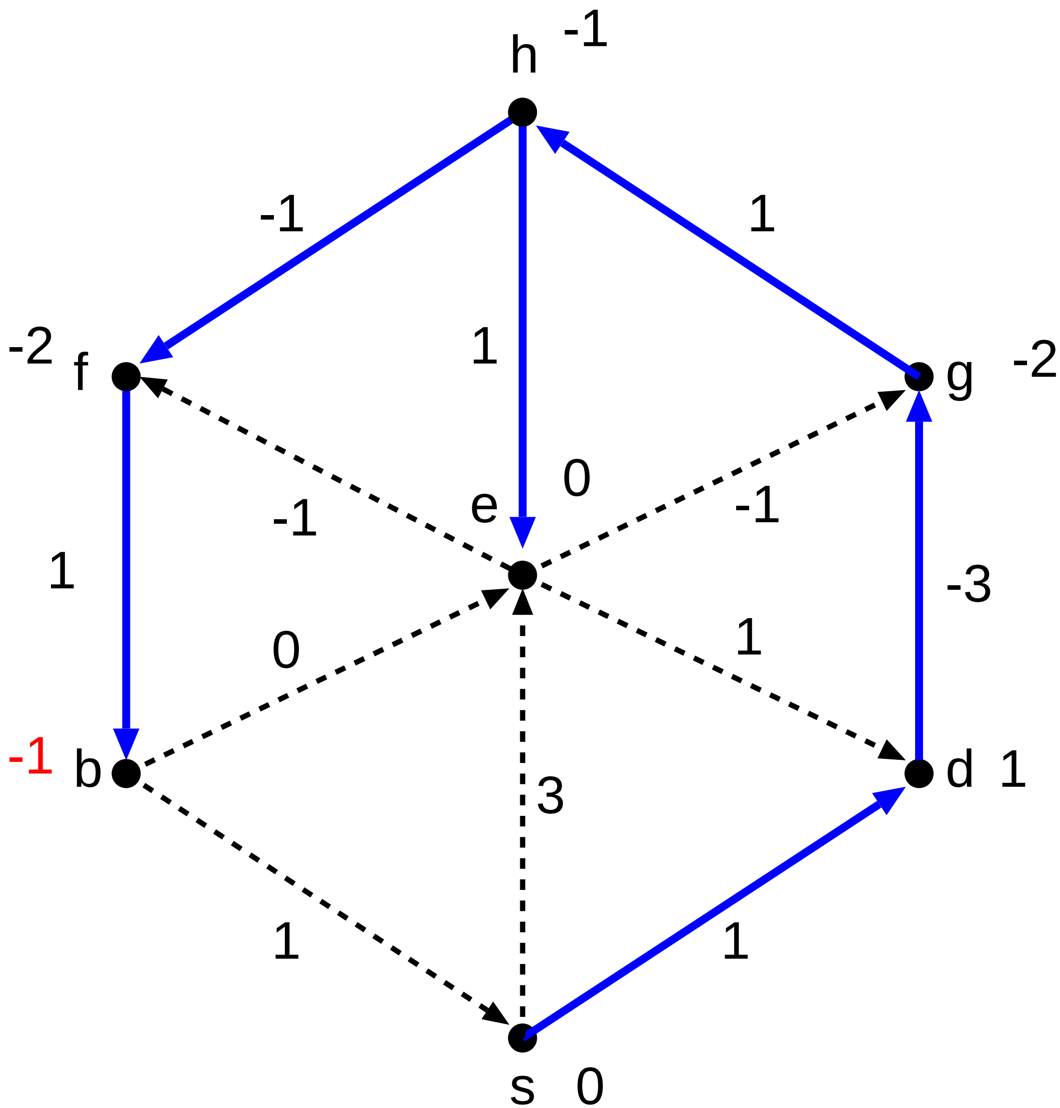
2回目の Step 2: dから出る枝(d, g)



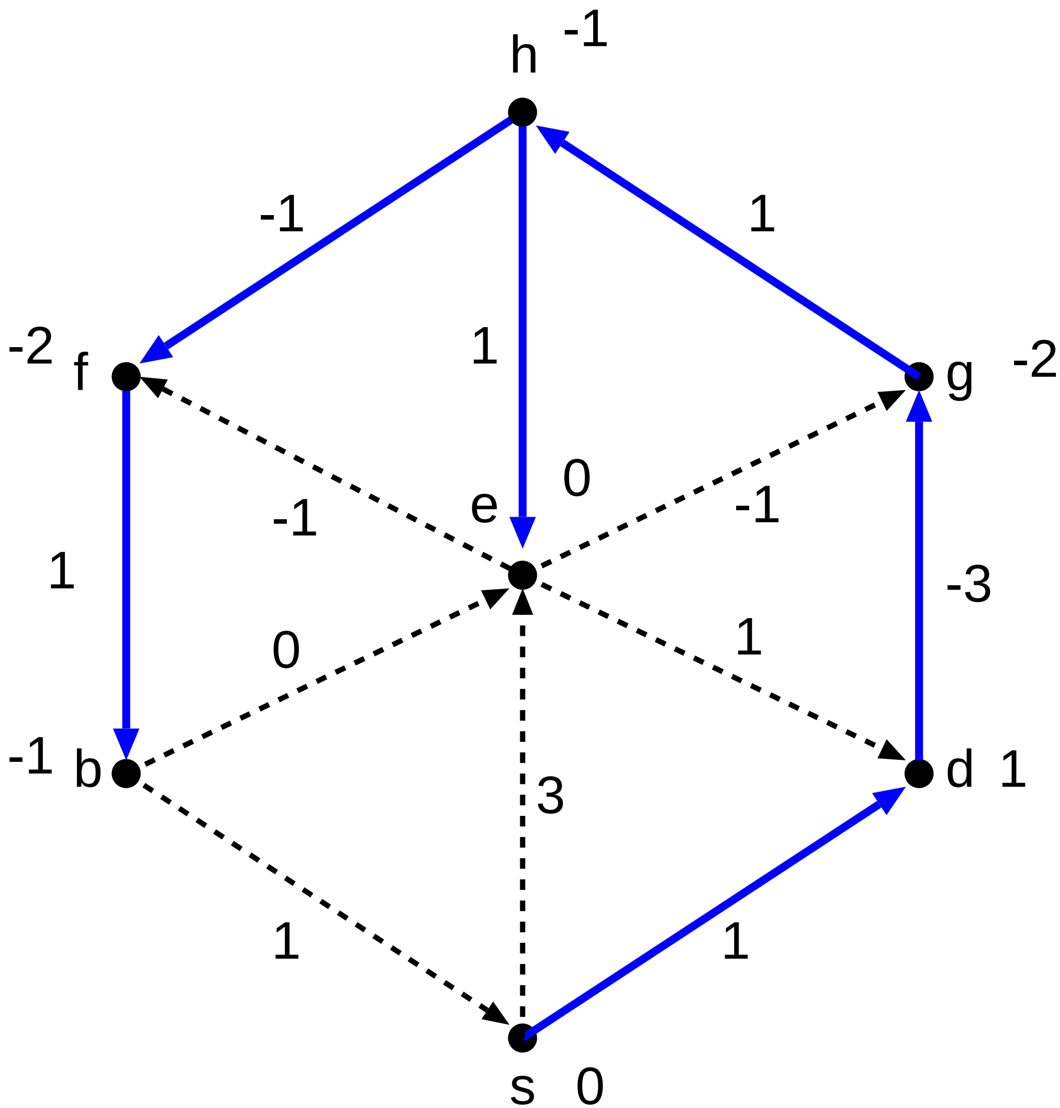
回目の Step 2: e から出る枝 $(e, f), (e, g), (e, $



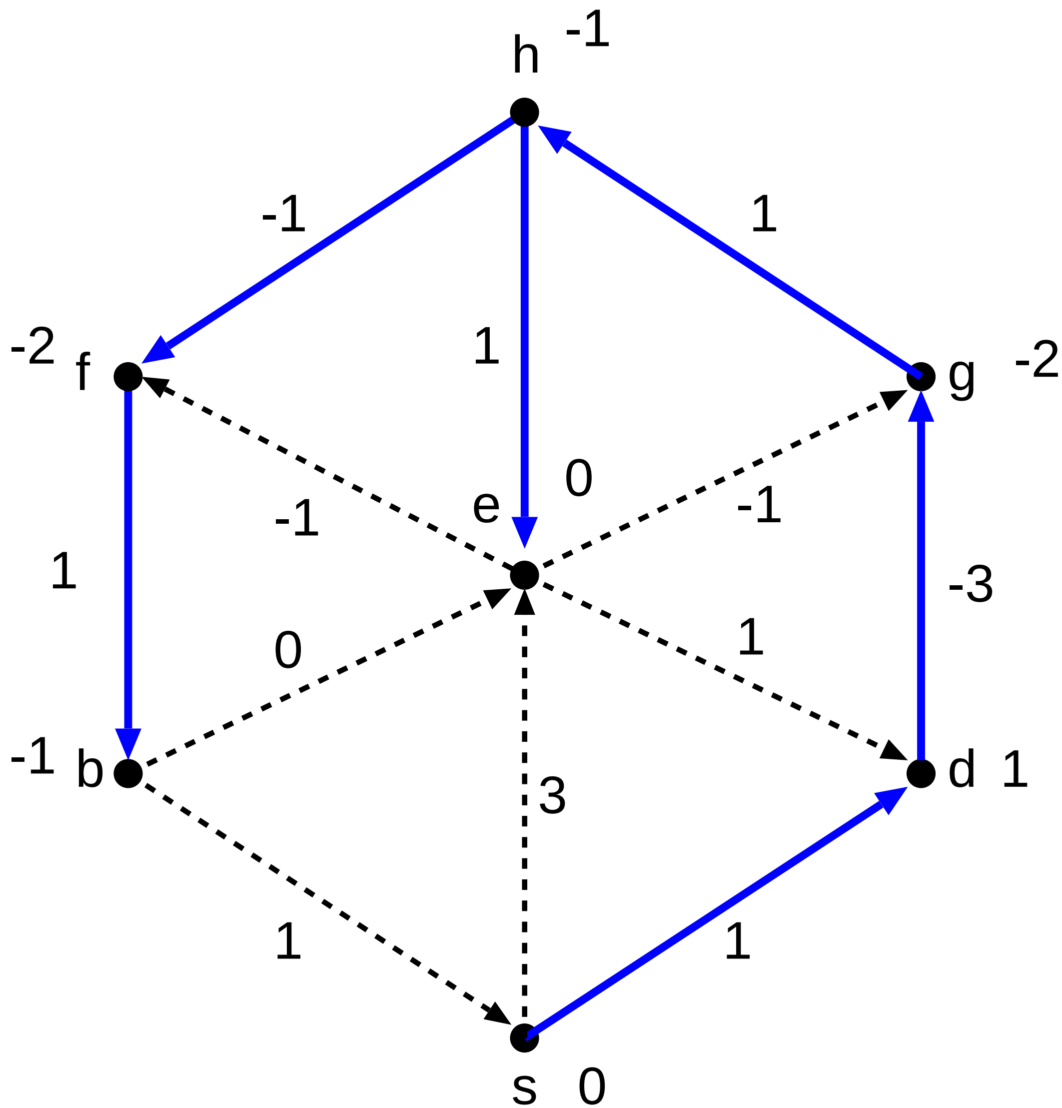
2回目の Step 2: f から出る枝 (f, b)



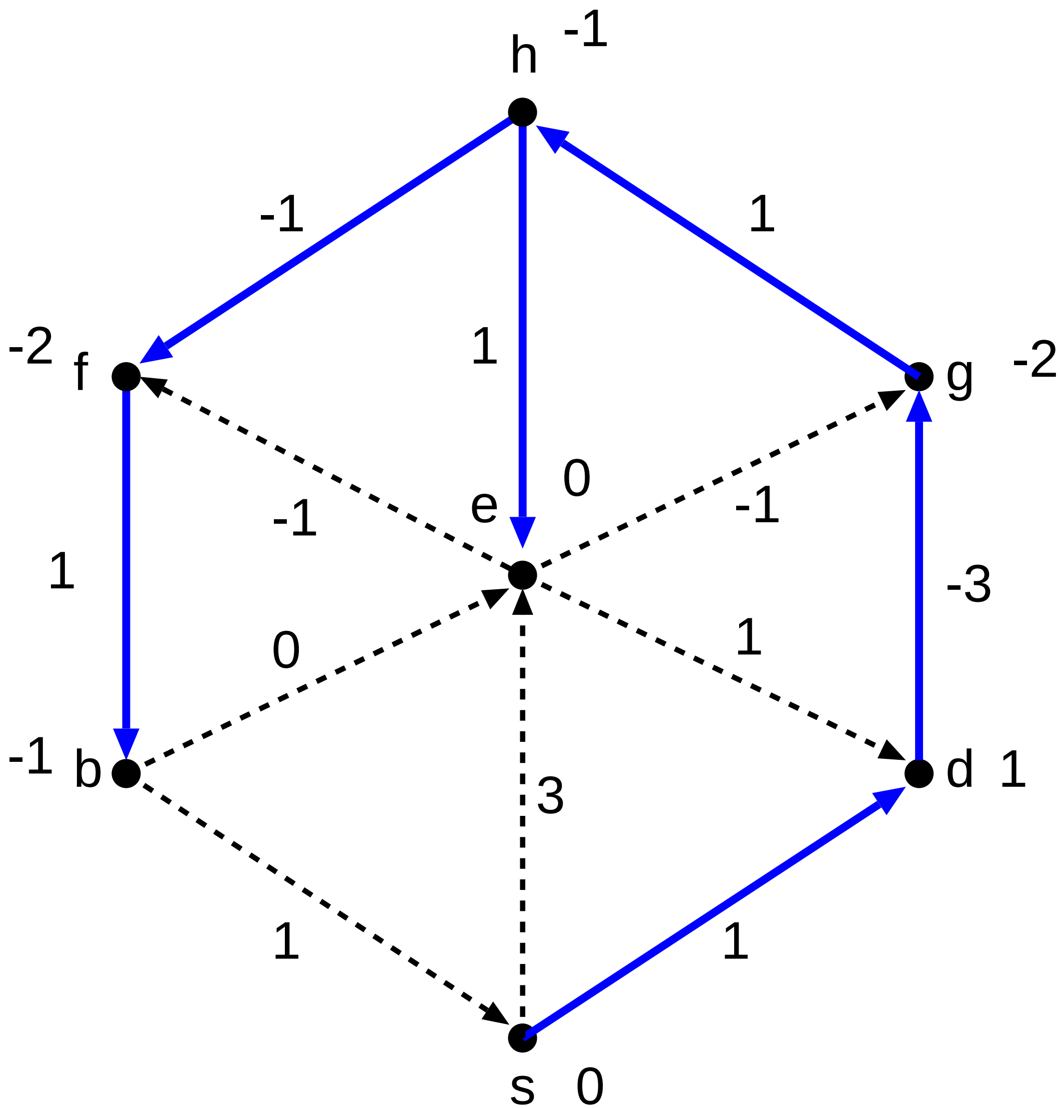
2回目の Step 2: g から出る枝 (g, h)



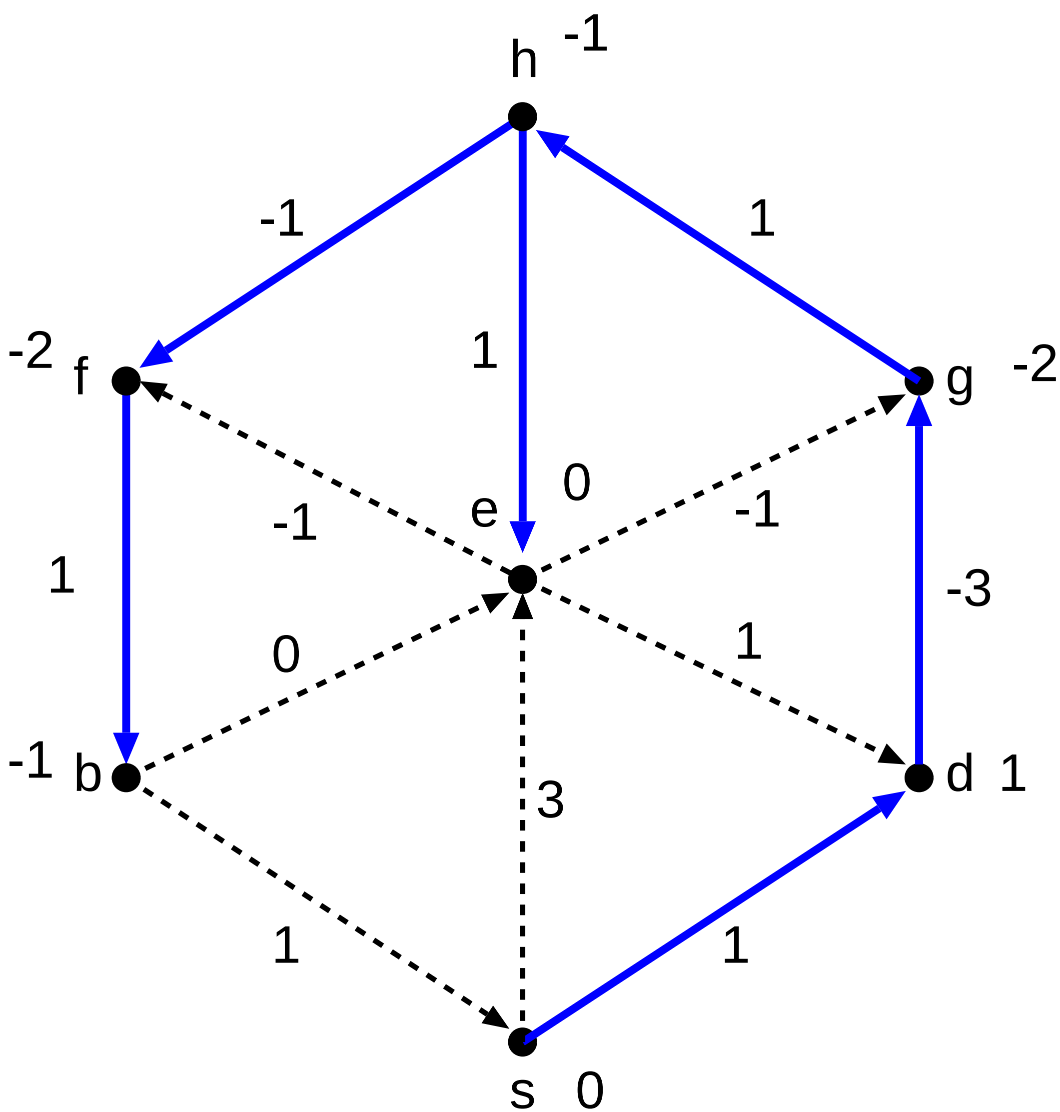
2回目の Step 2: h から出る枝 $(h, f), (h, e)$



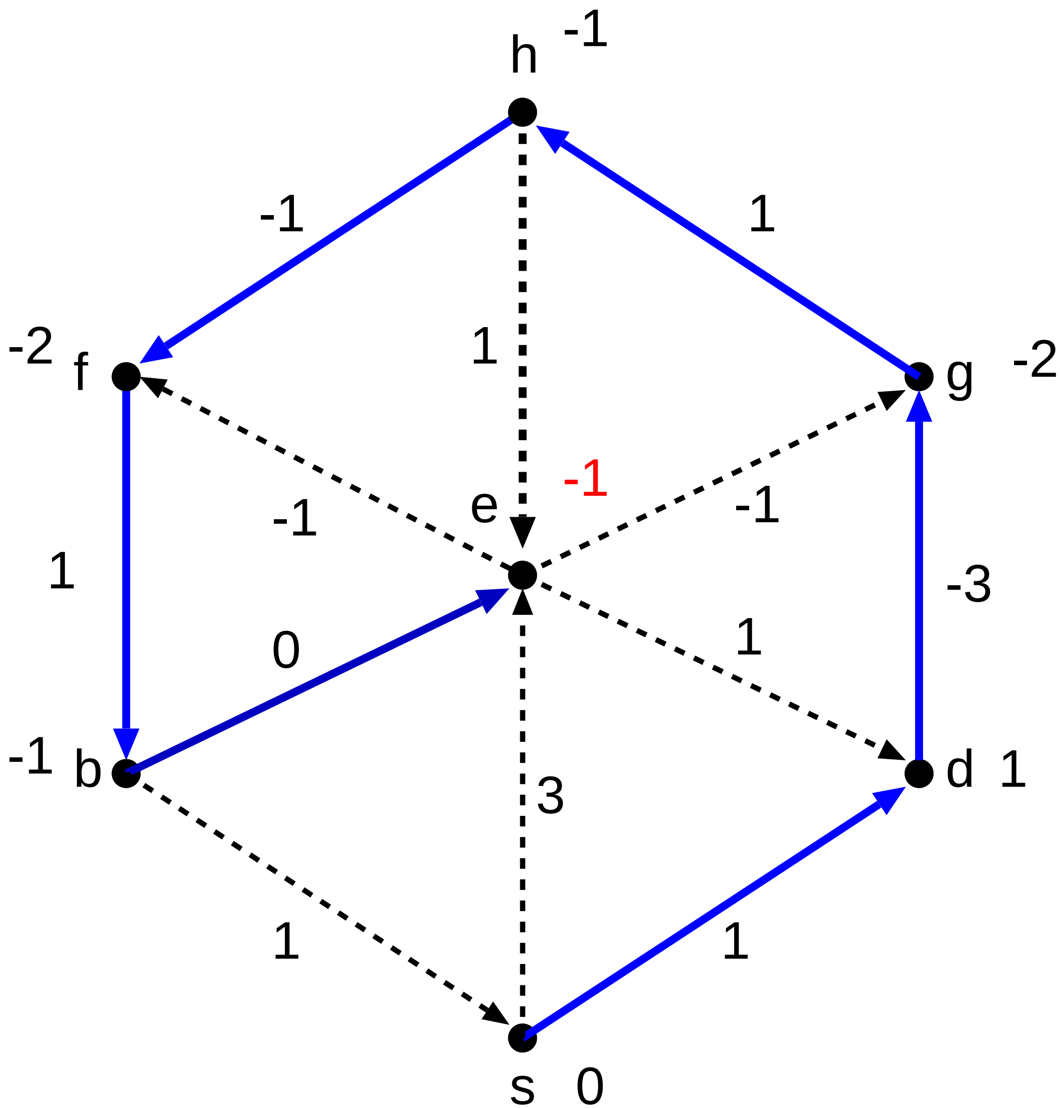
2回目の Step 2の終了時



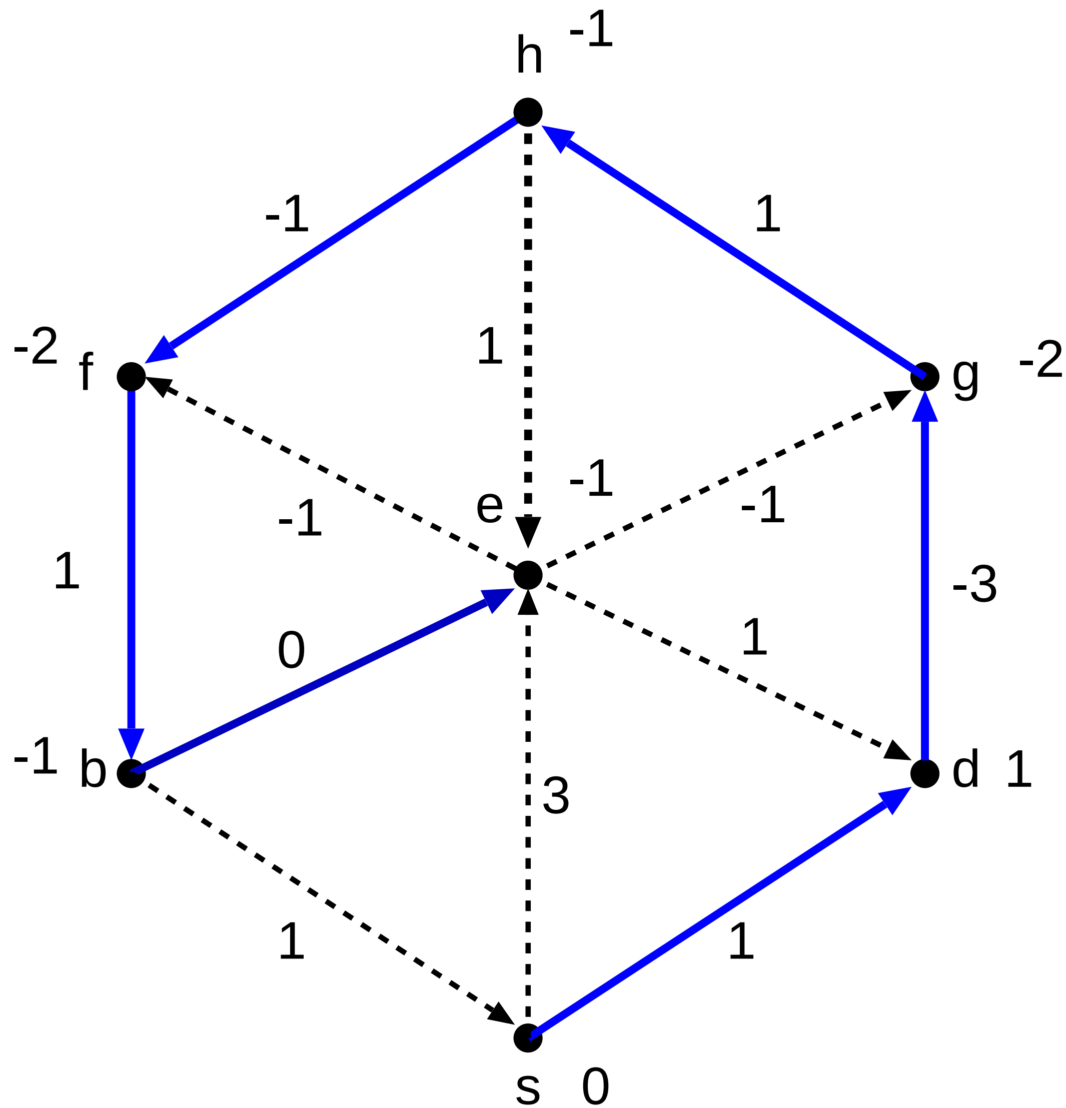
3回目の Step 2: s から出る枝 $(s, d), (s, e)$



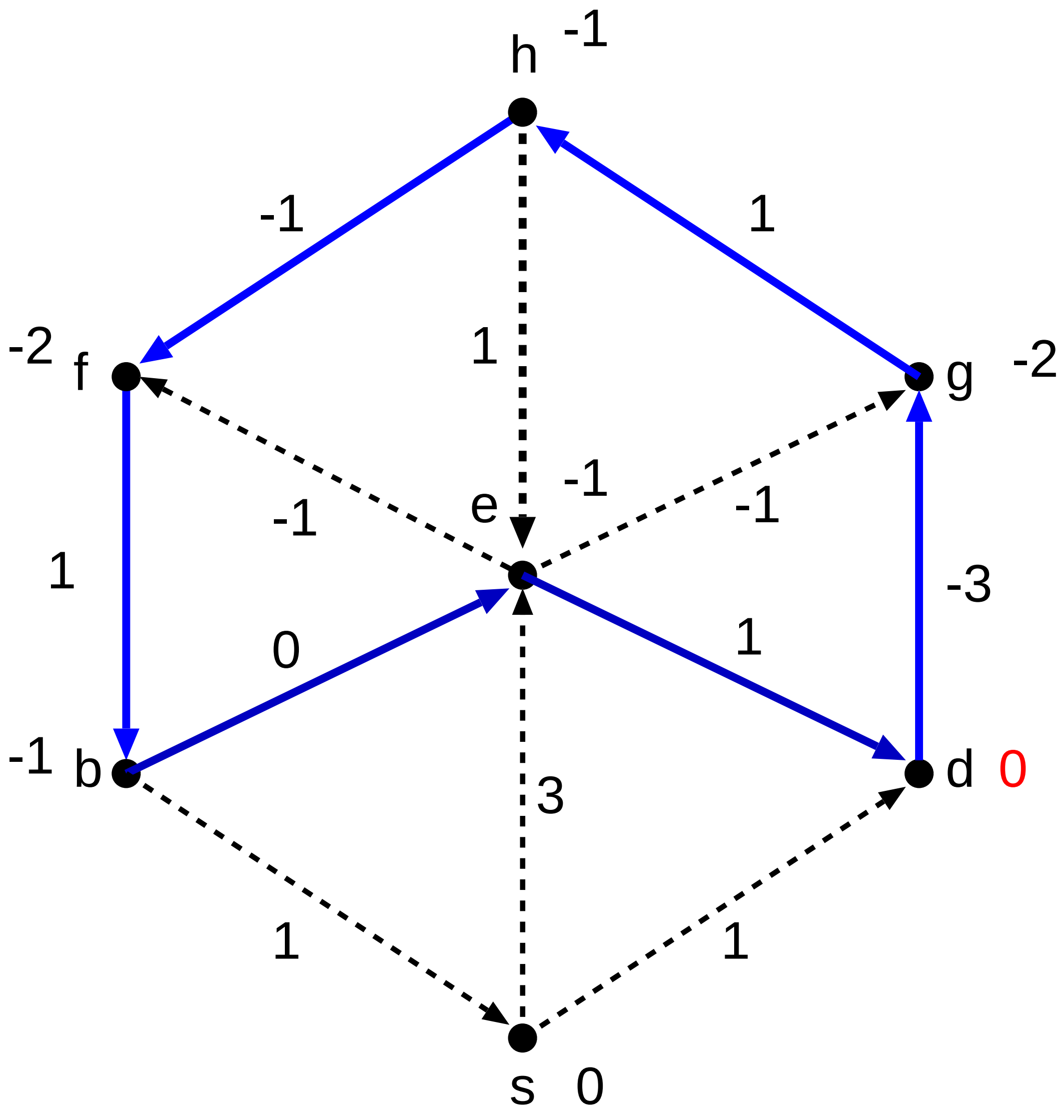
3回目の Step 2: bから出る枝$(b, e), (b, s)$



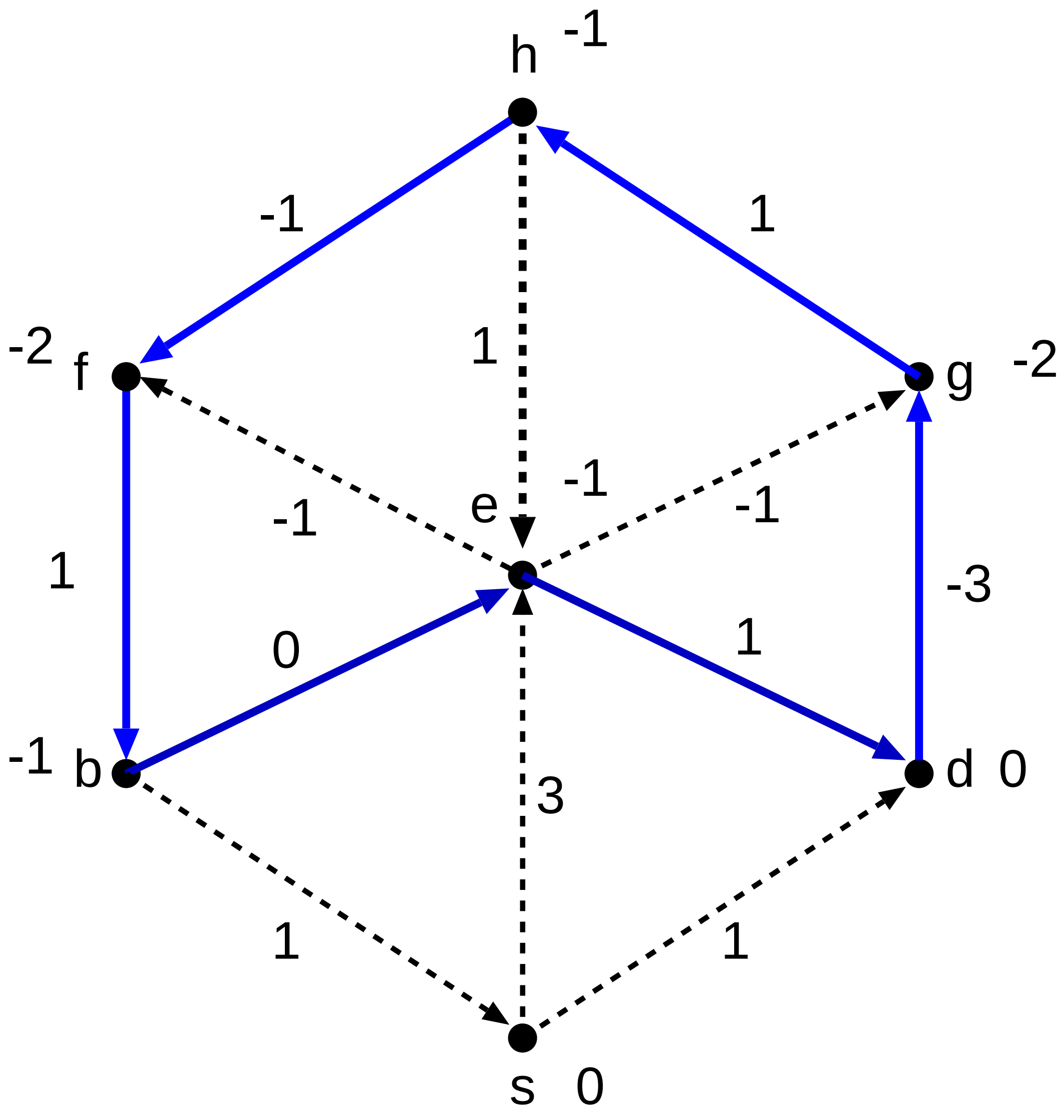
3回目の Step 2: dから出る枝(d, g)



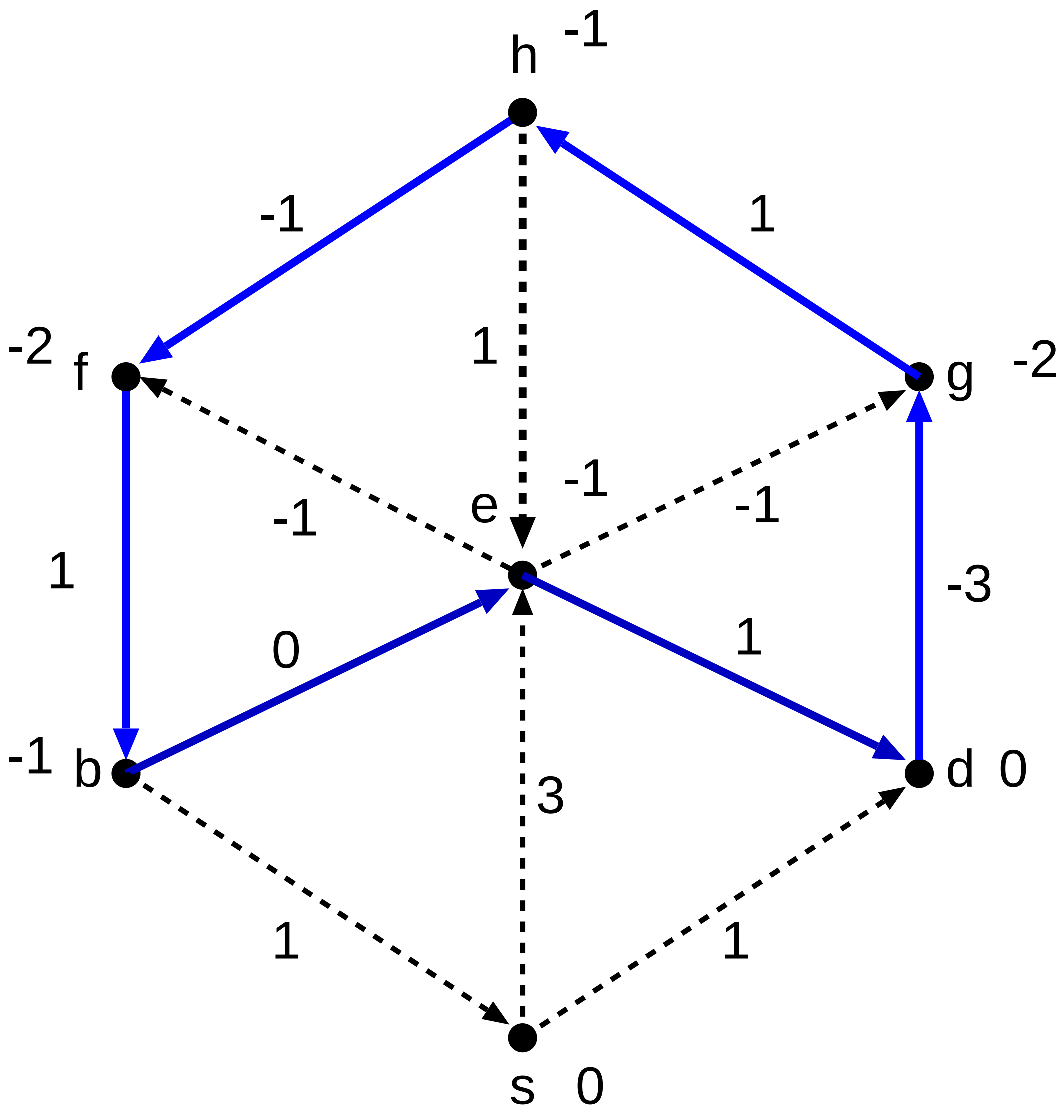
回目の Step 2: e から出る枝 $(e, f), (e, g), (e,$



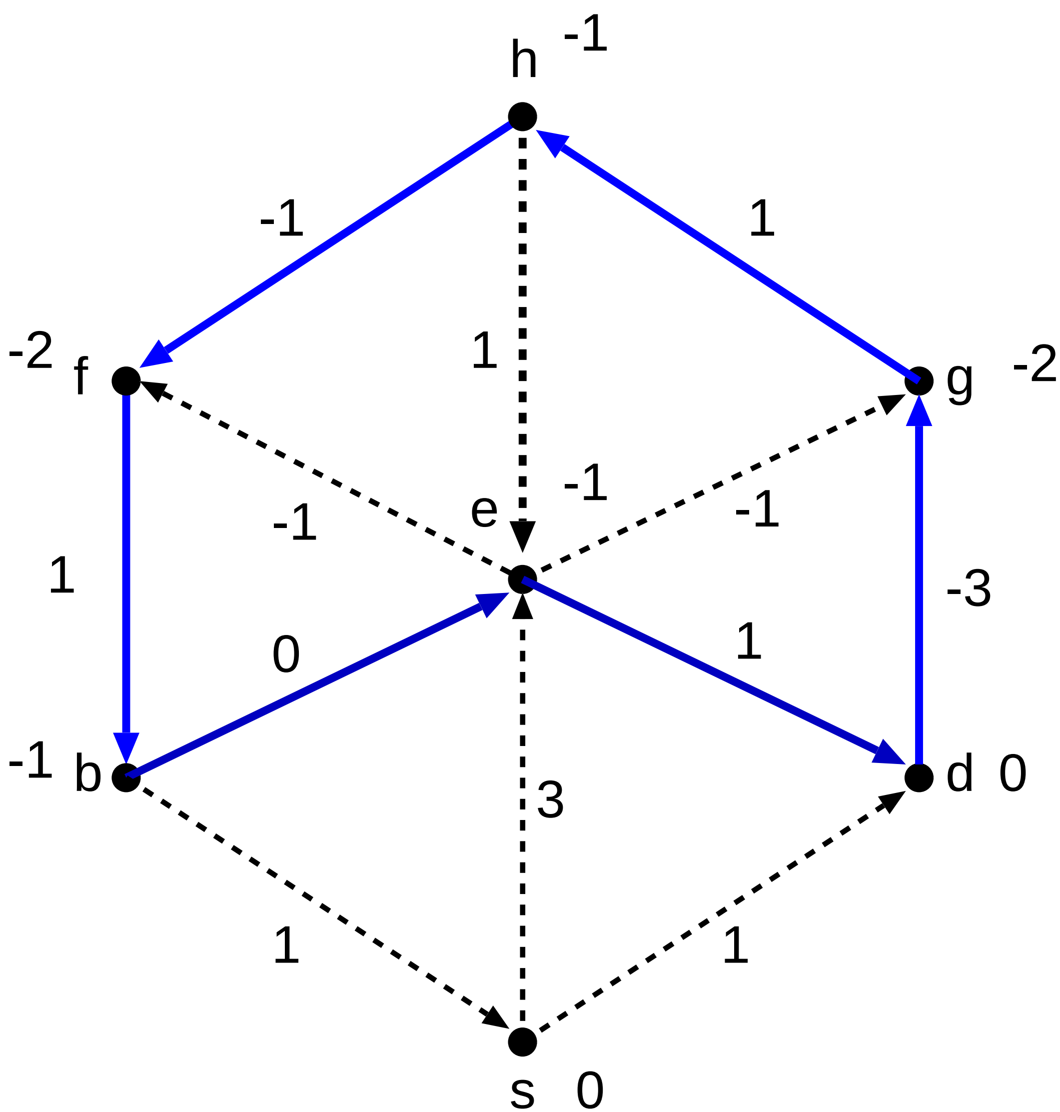
3回目の Step 2: f から出る枝 (f, b)



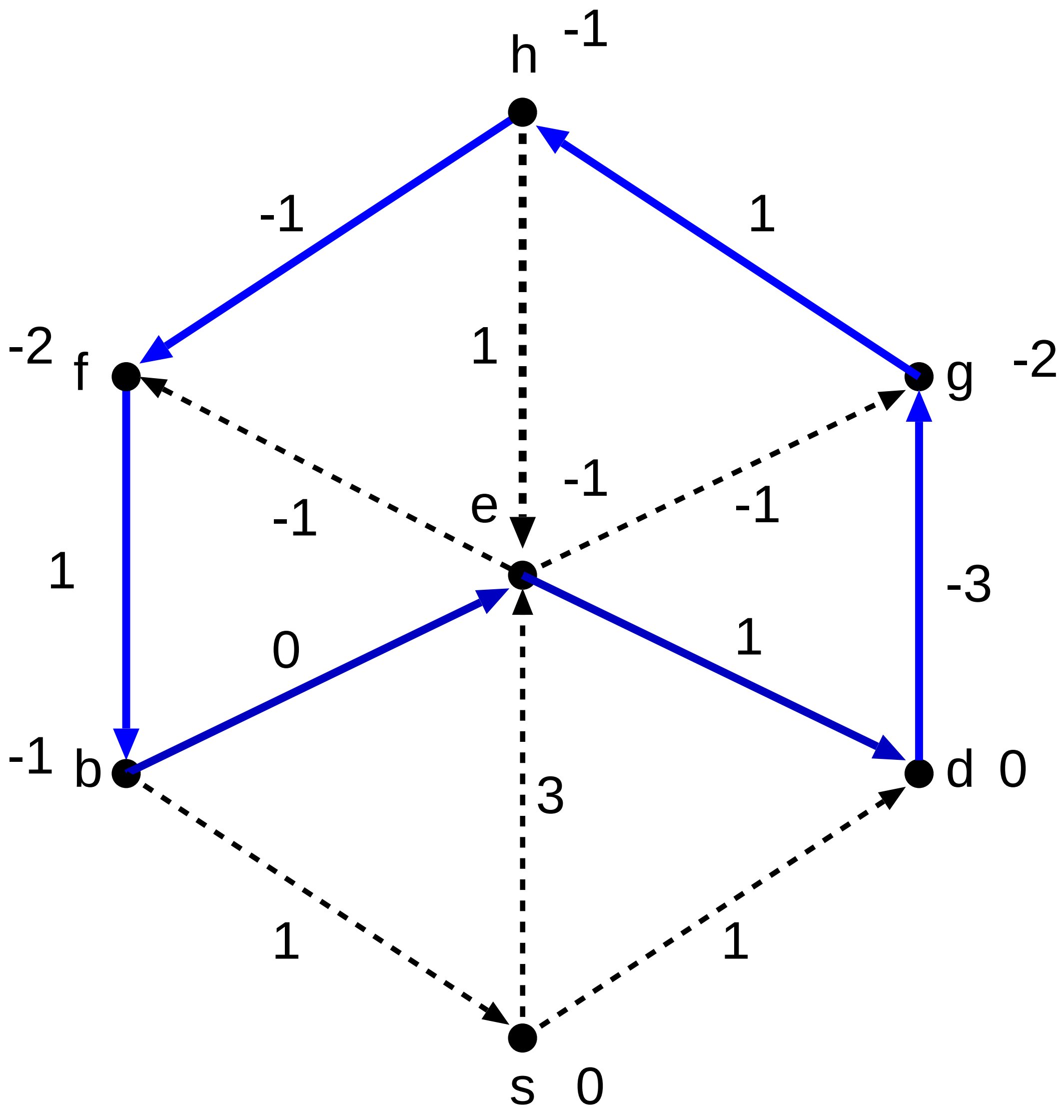
3回目の Step 2: g から出る枝 (g, h)



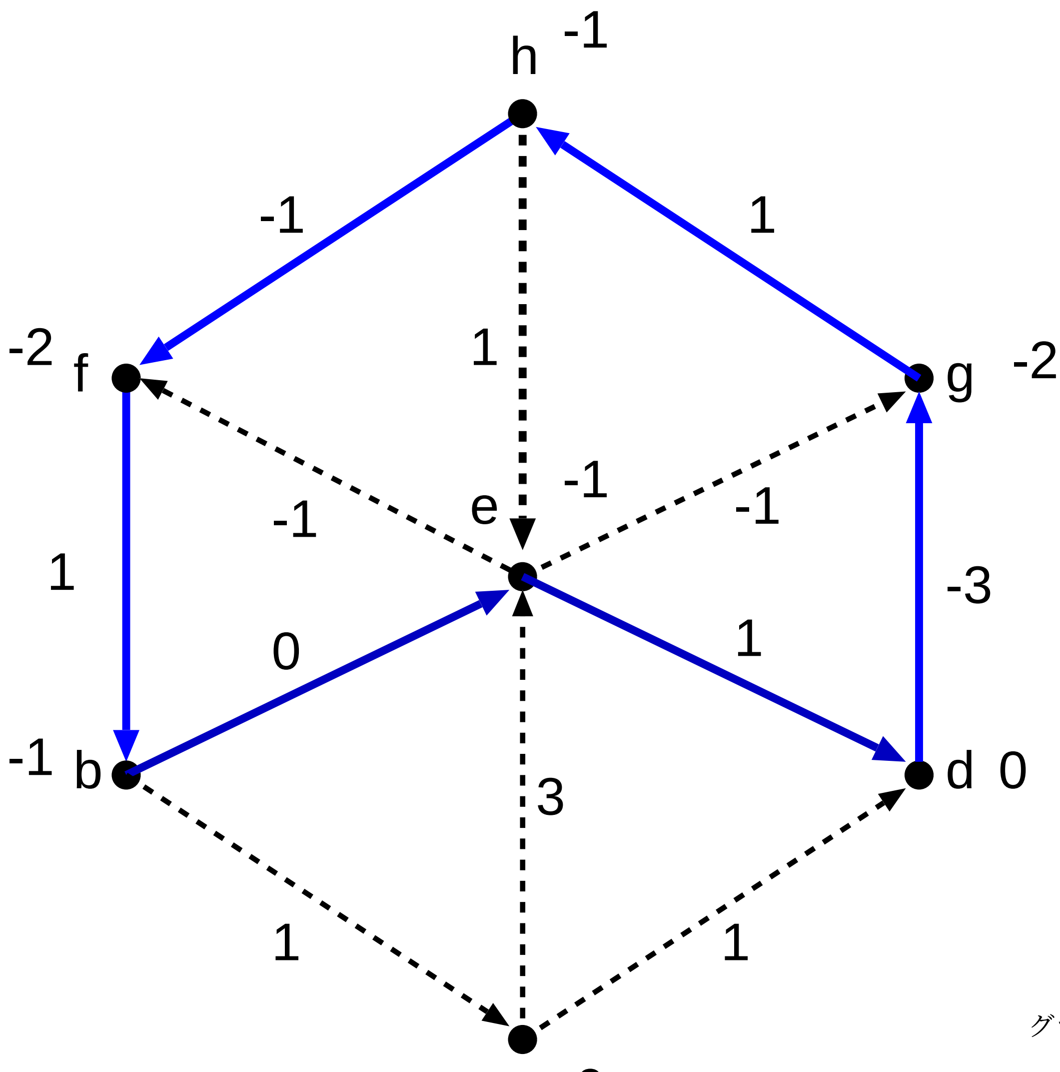
3回目の Step 2: h から出る枝 $(h, f), (h, e)$



3回目の Step 2の終了時



既に負の閉路が見つかっているが, アルゴリズムは $k = n$ となるまで繰返される.



負の長さ有向閉路でもうける方法

2004 年 1 月 8 日 18:10 頃の為替レート

レート	¥	$	Ł	EUR
¥	1	0.0094224	0.0051913	0.007482
$	106.13	1	0.55096	0.79409
Ł	192.63	1.815	1	1.4413
EUR	133.65	1.2593	0.69383	1

例えば...

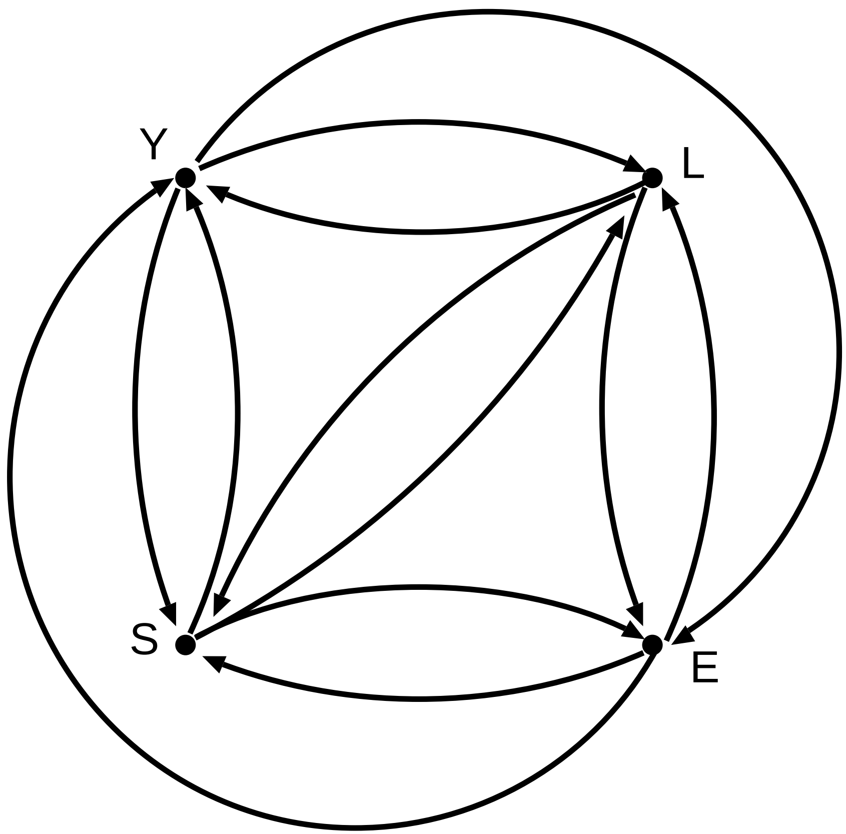
$$¥1 \xrightarrow[.0094224]{} \$\ .0094224 \xrightarrow[.79409]{} \text{EUR}$$

$$.0074822336 \xrightarrow[.69383]{} Ł\ .0051913981 \xrightarrow[192.63]{}$$

¥1.000019

だから, ¥ → $ → EUR → Ł → ¥というように換金すれば, 100 万から 19 円もうけることができる. (1 億円で, 1900 円)

点集合を $V = \{\mathrm{Y}, \mathrm{S}, \mathrm{L}, \mathrm{E}\}$ としてもつ完全有向グラフ $G = (V, A)$ を考える. 利益を生み出すような G の有向閉路を見付ければよい.



r	Y	S	L	E
Y	1	0.0094224	0.0051913	0.0074822
S	106.13	1	0.55096	0.79409
L	192.63	1.815	1	1.4413
E	133.65	1.2593	0.69383	1

$r_{CD} =$ 通貨 C を 通貨 D に換金するときのレート

とする. 例えば, $r_{\mathrm{SY}} = 106.13$.

枝の長さ

さらに, さきほど定義した完全グラフの枝の長さを,

$$l(C, D) = -\log r_{CD} \quad (CD \in V)$$

で定義し, ネットワーク $\mathcal{N} = (G = (V, A), l)$ を考える.

$-\log r$	Y	S	L	E
Y	0	4.6647	5.2608	4.8952
S	-4.6647	0	.59609	.23056
L	-5.2608	-.59609	0	-.36555
E	-4.8944	-.23056	.36553	0

有向閉路 $C_1 \to C_2 \to \cdots \to C_l \to C_1$ が利益を生み出す

$$\Updownarrow$$

$$r_{C_1C_2} \times r_{C_2C_3} \times \cdots \times r_{C_lC_1} > 1$$

$$\Updownarrow$$

$$\log r_{C_1C_2} + \log r_{C_2C_3} + \cdots + \log r_{C_lC_1} > 0$$

$$\Updownarrow$$

$$-\log r_{C_1C_2} - \log r_{C_2C_3} - \cdots - \log r_{C_lC_1} < 0$$

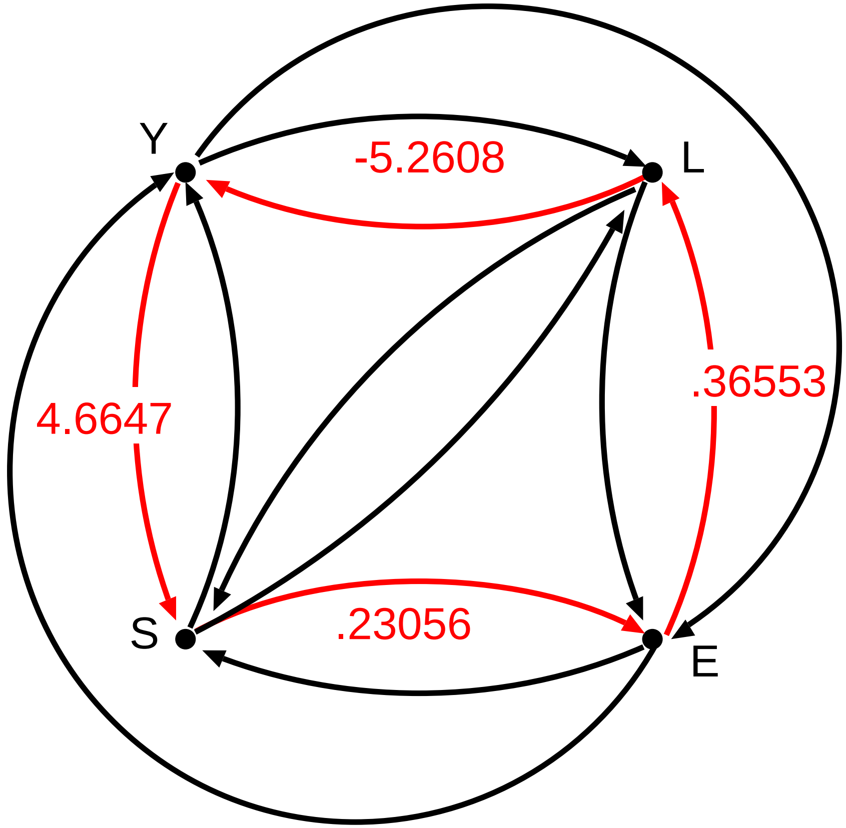
$$\Updownarrow$$

$$l(C_1, C_2) + l(C_2, C_3) + \cdots + l(C_l, C_1) < 0$$

$$\Updownarrow$$

$C_1 \to C_2 \to \cdots \to C_l \to C_1$ は, $\mathcal{N} = (G = (V, A), l)$ の負の長さの有向閉路

もうかる閉路



4.6647+.23056+.36553-5.2608=-0.00001