Eötvös Loránd Tudományegyetem

Természettudományi kar

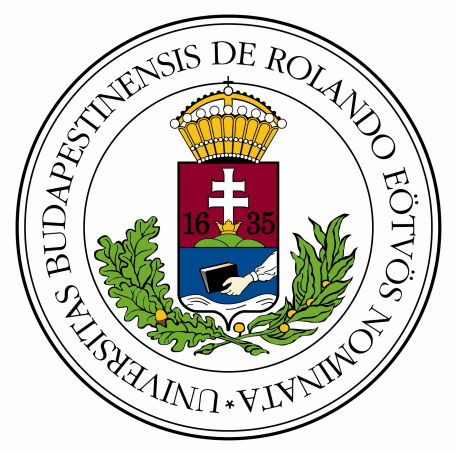
Erősen polinomiális algoritmusok a minimális költségű folyamfeladatra

Imre Balázs

Szakdolgozat, matematika BSc, alkalmazott matematikus szakirány

Témavezető:

Jüttner Alpár



Budapest, 2024

Tartalomjegyzék

Köszönetnyilvánítás

Szeretném kifejezni hálámat témavezetőmnek, Jüttner Alpárnak, aki szakértelmével, útmutatásával és támogatásával segítette munkámat. Köszönöm az értékes visszajelzéseket és a motiváló szavakat, melyek nélkül ez a dolgozat nem jöhetett volna létre. Szeretnék köszönetet mondani családomnak is, hisz mindig mellettem álltak és támogattak tanulmányaim során. Külön szeretnék köszönetet mondani szaktársaimnak, Blankának, Kincsőnek, Noéminek és Zoárdnak, akik nélkül az elmúlt 3 év közel sem lett volna ennyire élvezetes.

Előszó

A hálózati optimalizálás egyik alapvető feladata a minimális költségű folyamfeladat, melynek széles körű alkalmazásai vannak többek között a mérnöki tudományok számos ágában, az útvonaltervezésben, a telekommunikációban és a közlekedésben.

A minimális költségű folyamfeladatra számos polinomiális idejű algoritmus létezik, a gyakorlatban a hálózati szimplex algoritmust alkalmazzák a gyorsasága és az egyszerűsége miatt. Annak ellenére, hogy az általános szimplex algoritmushoz nem ismert olyan pivotálási szabály, amelyet alkalmazva az elméleti legrosszabb futásidő is polinomiális, a hálózati szimplex algoritmushoz találtak ilyet. Ezek közül kettőt is bemutatunk a dolgozatban.

Az első fejezetben definiáljuk a minimális költségű folyamfeladatot és bevezetjük a hozzá kapcsolódó fogalmakat. Megnézzük a lineáris programozás néhány alapvető tételét, a primál szimplex algoritmust, a duál szimplex algoritmust és a felsőkorlátos szimplex algoritmust, valamint definiáljuk a polinomiális és az erősen polinomiális futásidőt.

A második fejezetben James B. Orlin [1] cikke alapján megismerjük az általános primál hálózati szimplex algoritmust, és mutatunk ehhez egy olyan pivotálási szabályt, mellyel erősen polinomiális futásidejűvé tehető. Ezt alkalmazva a futásidő $O(\min\{n^2 m \log nC, n^2 m^2 \log n\})$ lesz. Ehhez definiáljuk a premultiplikátorokat, melyeken ez a pivotálási szabály alapszik.

A harmadik fejezetben Ronald D. Armstrong és Zhiying Jin [2] cikke alapján megismerjük az általános duál hálózati szimplex algoritmust, és hasonlóan az előző fejezethez, itt is mutatunk egy olyan pivotálási szabályt, mellyel erősen polinomiális futásidejűvé tehető. A futásidő ebben az esetben $O(mn(m + n \log n) \log n)$ lesz.

1. fejezet

Bevezetés

1.1 Alapfogalmak

Legyen $G = (N, A, u, c, b)$ egy **irányított hálózat**, ahol N a **csúcsok halmaza**, $|N| = n$, A az **irányított élek halmaza**, $|A| = m$, $u\colon A \mapsto \mathbb{R}^+ \cup \{+\infty\}$ a **kapacitásfüggvény**, $c\colon A \mapsto \mathbb{R}_0^+$ a **költségfüggvény**, valamint $b\colon N \mapsto \mathbb{R}$ az **igényfüggvény**, melyre $\sum_{v \in N} b(v) = 0$. Ezek által meghatározva minden $(v, w) \in A$ élhez tartozik egy c_{vw} költség és egy u_{vw} kapacitás, valamint minden $v \in N$ csúcshoz tartozik egy b_v igény. Ha a költségfüggvény egészértékű, akkor jelölje C a $\max\{c_{vw}\colon (v, w) \in A\}$ mennyiséget, különben legyen $C = +\infty$.

A minimális költségű folyamfeladat felírható az alábbi alakú optimalizálási feladatként. Keressük azt az $x\colon A \mapsto \mathbb{R}_0^+$ **folyamot**, amely minimalizálja a

$$\sum_{(v,w) \in A} c_{vw} x_{vw}$$

mennyiséget a következő feltételekkel:

$$\begin{aligned} &\forall v \in N\colon && \sum_{(v,w) \in A} x_{vw} - \sum_{(w,v) \in A} x_{wv} = b_v, \\ &\forall (v, w) \in A\colon && 0 \leq x_{vw} \leq u_{vw}. \end{aligned} \tag{1.1}$$

Vezessük be a $p\colon N \mapsto \mathbb{R}$ **potenciálfüggvényt**, amely minden $v \in N$ csúcshoz rendel egy p_v értéket. Minden p potenciálfüggvényhez és c költségfüggvényhez tartozik egy $c^p\colon A \mapsto \mathbb{R}$ **redukált költségfüggvény**, amely minden $(v, w) \in A$ élhez rendel egy c^p_{vw} értéket úgy, hogy $c^p_{vw} = c_{vw} - p_v + p_w$.

1.1.1. Lemma. *A minimális költségű folyamfeladatban egy adott c költségfüggvény és egy adott p potenciálfüggvény esetén a c^p redukált költségfüggvény szerinti minimalizálási feladat ekvivalens az eredeti c költségfüggvény szerinti minimalizálási feladattal.*

Legyen $D = (V, E)$ egy irányított gráf, $V = \{1, 2, \ldots, n\}$, $E = \{1, 2, \ldots, m\}$. Azt mondjuk, hogy az M mátrix ennek a gráfnak az **incidenciamátrixa**, ha

$$M_{ij} = \begin{cases} 1, & \text{ha az } i \text{ csúcs a } j \text{ élnek a kezdőpontja,} \\ -1, & \text{ha az } i \text{ csúcs a } j \text{ élnek a végpontja,} \\ 0, & \text{különben.} \end{cases}$$

Tehát az incidenciamátrix minden oszlopa egy élhez, minden sora egy csúcshoz tartozik. Minden oszlopban egy $+1$ és egy -1 szerepel, a többi helyen nullák állnak.

Adott $D = (V, E)$ irányított gráfban egy $(v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k)$ utat akkor hívunk **irányított útnak**, ha $e_1 = (v_0, v_1)$, $\ldots$, $e_k = (v_{k-1}, v_k)$ és minden csúcsa különböző. Az **irányított kör** definíciója hasonló az irányított út definíciójához, de a kör esetén $v_0 = v_k$. Legyen P egy irányított út. Ekkor $c(P)$ jelölje az irányított úton az éleknek a c költségfüggvény szerinti összköltségét. Hasonlóan, ha W egy irányított kör, akkor $c(W)$ jelölje az irányított körben az éleknek a c költségfüggvény szerinti összköltségét. Egy adott W irányított kör, p potenciálfüggvény és c költségfüggvény esetén $c(W) = c^p(W)$.

Egy A **algoritmus lépésszámán** azt a $t_A \colon \mathbb{N} \mapsto \mathbb{N}$ függvényt értjük, melyre

$$t_A(n) = \max\{\text{az } A \text{ algoritmus lépésszáma az } x \text{ inputon} \colon |x| = n\}.$$

Legyenek $f, g \colon \mathbb{N} \mapsto \mathbb{R}^+$ függvények. Ekkor $f = O(g)$, ha léteznek $n_0, c \in \mathbb{N}$ konstansok úgy, hogy minden $n \geq n_0$ egészre $f(n) \leq c \cdot g(n)$. Azt mondjuk, hogy az A algoritmus polinomiális futásidejű, ha a futásideje felülről korlátos az input méretének egy polinomjával, azaz létezik olyan $k \in \mathbb{N}$, melyre $t_A(n) = O(n^k)$, ahol n az input mérete.

A definíció függ a számítási modelltől, amely meghatározza, hogy a futásidőt, valamint az input méretét mi alapján számoljuk ki. Két számítási modellt veszünk ehhez alapul, a **Turing modellt** , valamint az **aritmetikai modellt**, melyeket Peter van Emde Boas [3] könyvében leírt módon definiálunk. Míg előbbi egy **Turing-**

gép, utóbbi egy **RAM (Random-Access Machine)** segítségével hajtja végre az algoritmusokat. Nézzük meg leegyszerűsítve ennek a két számítási modellnek néhány különbségét:

- Az aritmetikai modellben minden szám egy memóriacellát foglal el, ezzel szemben a Turing modellben minden szám annyi memóriacellát foglal el, ahány bit szükséges a bináris reprezentálásához.
- Az aritmetikai modellben minden alap aritmetikai művelet (összeadás, kivonás, szorzás, osztás) elvégzése egy lépés, míg a Turing modellben ezen műveletek végrehajtási ideje függ a számok méretétől, azaz a bináris reprezentálásukhoz szükséges bitek számától.

Ha egy algoritmus a Turing modell szerint polinomiális futásidejű, akkor a futásidő felülről korlátos az inputban levő számok bináris reprezentálásához szükséges bitek számának egy polinomjával. Ha egy algoritmus az aritmetikai modell szerint polinomiális futásidejű, akkor a futásidő felülről korlátos az inputban levő adatok számának egy polinomjával.

1.1.2. Példa. *Az Euklideszi algoritmus a Turing modell szerint polinomiális futásidejű, azonban az aritmetikai modell szerint nem. Az input egy* (a, b) *számpár, vagyis az aritmetikai modellben az input mérete* 2*, tehát konstans. Mivel a* 2 *minden polinomja konstans, ezért mindig tudunk úgy választani* a *és* b *számokat, hogy az algoritmus során az osztások száma, vagyis a futásidő, ennél nagyobb legyen.*

1.1.3. Példa. *Az algoritmus, amely olvas egy* $n \in \mathbb{N}$ *számot és ismételt négyzetre emeléssel kiszámolja* 2^{2^n} *értékét, az az aritmetikai modell szerint polinomiális futásidejű, míg a Turing modell szerint nem. A számok reprezentálásához szükséges bitek száma az egymás utáni szorzások miatt exponenciálisan nő, így nem becsülhető felülről polinommal.*

1.1.4. Definíció. *Azt mondjuk, hogy egy algoritmus erősen polinomiális futásidejű, ha a Turing-modell és az aritmetikai modell szerint is polinomiális futásidejű.*

Vagyis egy algoritmus akkor erősen polinomiális futásidejű, ha az inputbeli adatok számának és ezek bináris reprezentálásához szükséges bitek számának is egy polinomjával felülről becsülhető a futásideje mindkét számítási modellben. A minimális költségű folyamfeladatban meg kell adnunk a csúcsok halmazát, az élek

halmazát, minden csúcsnak az igényét, minden élnek a kapacitását, valamint a költségét. Célunk tehát olyan algoritmusokat mutatni a minimális költségű folyamfeladatra, melyeknek futásideje $O(\text{Poly}(n, m))$, azaz csak az irányított hálózat csúcs- és élszámától függ. Most nézzünk példát egy nem erősen polinomiális folyam algoritmusra.

1.1.5. Példa. *A Ford-Fulkerson algoritmus megoldja a maximális folyamfeladatot, ahol két kijelölt s és t csúcson kívül mindegyik csúcs igénye 0. Ha a kapacitások és a költségek egész számok, akkor algoritmus futásideje $O(m \cdot F)$, ahol F a maximális $s-t$ folyam értéke. Legrosszabb esetben minden javítóút csak 1 egységgel növeli a folyamértéket, míg az ilyen utak megtalálása $O(m)$ időt vesz igénybe.*

1.2 A lineáris programozás alapjai

A lineáris programozás az operációkutatás egyik ága, mely lineáris egyenlőtlenségrendszerekkel foglalkozik. Az ilyen egyenlőtlenség-rendszerekkel kapcsolatban több kérdés is felmerülhet, például hogy van-e egyáltalán megoldása, illetve ha van, akkor azok között egy előre meghatározott célfüggvény szerint tudunk-e találni optimálisat. Mi az utóbbi kérdést vizsgáljuk részletesebben ebben a szakaszban.

Legyen $A \in \mathbb{R}^{n \times m}$ mátrix, $b, y^T \in \mathbb{R}^n$, valamint $x, c^T \in \mathbb{R}^m$ vektorok. Az ezekre felírható **primál feladat** a következő:

$$\min\{cx \colon Ax = b, x \geq 0\}.$$

Az ehhez tartozó **duál feladat**:

$$\max\{yb \colon yA \leq c\}.$$

Vegyük a fent megfogalmazott primál és duál feladatot. Azt mondjuk, hogy egy x vektor **primál megengedett megoldás**, ha $Ax = b$ és $x \geq 0$. Egy y vektor **duál megengedett megoldás**, ha $yA \leq c$. Most nézzünk meg három fontos állítást ebből a témakörből.

1.2.1. Tétel (Gyenge dualitás tétel)**.** *Legyen x egy primál megengedett megoldás, y egy duál megengedett megoldás. Ekkor teljesül, hogy*

$$cx \geq yb.$$

A következő tétel kimondása előtt vezessük be, hogy mit értünk az üres halmaz minimuma, illetve maximuma alatt. Legyen $\min\{\emptyset\} = +\infty$ és legyen $\max\{\emptyset\} = -\infty$.

1.2.2. Tétel (Erős dualitás tétel)**.** *Ha a primál és a duál feladat közül legalább az egyik megoldható, akkor létezik a maximum és a maximum, továbbá teljesül, hogy*

$$\min\{cx\colon Ax = b, x \geq 0\} = \max\{yb\colon yA \leq c\}.$$

1.2.3. Állítás (Optimalitási feltétel)**.** *Legyen x egy primál megengedett megoldás, y egy duál megengedett megoldás. Az x és a y vektorok pontosan akkor optimálisak, ha minden $i \in \{1, 2, \dots m\}$ esetén, ha $x_i > 0$, akkor $ya_i = c_i$, ahol a_i az A mátrix i. oszlopát jelöli.*

A szimplex algoritmus segítségével megoldhatóak a korábban definiált optimalizálási feladatok, így a szakasz hátralevő részében ezt fogjuk vizsgálni. Szükségünk van arra, hogy az A mátrix sorai lineárisan függetlenek legyenek. A szimplex algoritmus bázisokon lépked, így minimalizálva a célfüggvényt a primál feladat megoldása közben, vagy maximalizálva a célfüggvényt duál feladat esetén. Egy bázis mindig teljes rangú és a hozzá tartozó oszlopok kijelölésével adható meg. A bázison kívüli oszlopokhoz tartozó változók értéke nulla. A B **bázishoz tartozó primál megoldás** az $x_B = B^{-1}b$ vektor nullákkal kiegészítve. A B **bázishoz tartozó duál megoldás** az $y_B = c_B B^{-1}$, ahol c_B a c vektor B bázishoz tartozó része. Azt mondjuk, hogy a B bázis **primál megengedett**, ha $x_B \geq 0$, valamint **duál megengedett**, ha $y_B A \leq c$. Azt mondjuk, hogy a B bázis optimális, ha x_B és y_B optimálisak.

1.2.4. Állítás. *A B bázis pontosan akkor optimális, ha egyszerre primál és duál megengedett.*

A szimplex algoritmus során kétféle kimenetel lehetséges. Vagy találunk egy optimumot, amit az optimalitási feltétel segítségével tudunk leellenőrizni, vagy találunk egy végtelen csökkentő/növelő irányt, ami azt jelenti, hogy a célfüggvény nem korlátos.

Vizsgáljuk meg először a primál szimplex algoritmust (1.1. eljárás). Ennek végrehajtása közben primál megengedett bázisokon lépkedünk. Minden lépésben egy oszlopot cserélünk ki a B bázisban. A primál célfüggvény értéke monoton csökken. Előfordulhat, hogy degenerált lépéseket teszünk, azaz olyan báziscseréket végzünk, melyek során a célfüggvény értéke nem csökken, a bázisok pedig ciklizálnak. Két

helyen tudunk pontosítani az algoritmuson; amikor kiválasztjuk a kilépő, illetve a belépő oszlopot.

1: **procedure** PRIMÁL_SZIMPLEX()
2: keressünk egy primál megengedett B bázist
3: $x_B \leftarrow B^{-1}b$
4: $y \leftarrow c_B B^{-1}$
5: **if** $yA \leq c$ **then**
6: **return** x és y optimális
7: **end if**
8: legyen $j\colon ya_j > c_j$
9: $x'_B \leftarrow -B^{-1}a_j$, $x'_j \leftarrow 1$ ▷ $Ax' = 0$ és $cx' < 0$
10: **if** $x' > 0$ **then** ▷ $x + tx' > 0$ és $Ax + tx' = b$ minden $t \in \mathbb{R}^+$ szám esetén
11: **return** a primál célfüggvény nem korlátos
12: **end if**
13: $t \leftarrow \min\left\{\frac{x_i}{x'_i}\colon i \in \{1, 2, \ldots, m\}, x'_i < 0\right\}$ ▷ létezik olyan $i\colon x'_i < 0$
14: $i \leftarrow \arg\min\left\{\frac{x_i}{x'_i}\colon i \in \{1, 2, \ldots, m\}, x'_i < 0\right\}$
15: $x \leftarrow x + tx'$
16: $B \leftarrow B - a_i + a_j$ ▷ B bázisban az i. oszlopot cseréljük a j. oszlopra
17: **go to** 4
18: **end procedure**

1.1. eljárás. primál szimplex

1.2.5. Állítás (Bland-szabály)**.** *Ha a lehetséges kilépő, illetve belépő oszlopok közül mindig a legkisebb indexű oszlopot választjuk, akkor a primál szimplex algoritmus véges sok lépésben leáll.*

Következőnek nézzük meg a duál szimplex algoritmust (1.2. eljárás). A duál szimplex algoritmus során duál megengedett bázisokon lépkedünk és minden lépésben egy oszlopot cserélünk ki a B bázisban. A duál célfüggvény monoton nő, továbbá itt is igaz, hogy a Bland-szabály alkalmazásával (1.2.5. állítás) véges sok lépésben leáll az algoritmus. A fő különbség a két eljárás között az, hogy a primál esetben előbb választjuk ki a bázisba belépő oszlopot, majd ehhez választunk kilépő oszlopot, míg a duál változatban a kilépő oszlophoz választunk belépő oszlopot.

1: **procedure** DUÁL_SZIMPLEX()
2: keressünk egy duál megengedett B bázist
3: $y \leftarrow c_B B^{-1}$
4: $x \leftarrow B^{-1}b$
5: **if** $x \geq 0$ **then**
6: **return** x és y optimális
7: **end if**
8: legyen $i\colon x_i < 0$
9: $y' \leftarrow I^i B^{-1}$ ▷ $I^{iT} \in \mathbb{R}^n\colon I^i_i = 1, I^i_j \neq 1$, ha $i \neq j$, $y'b < 0$
10: **if** $y'A \leq 0$ **then** ▷ ekkor $(y + ty')A \leq c$ minden $t \in \mathbb{R}^+$ szám esetén
11: **return** a duál célfüggvény nem korlátos
12: **end if**
13: $t \leftarrow \min\left\{\dfrac{ya_j - c_j}{y'a_j}\colon j \in \{1, 2, \ldots, n\}, y'a_j > 0\right\}$ ▷ létezik olyan $j\colon y'a_j > 0$
14: $j \leftarrow \arg\min\left\{\dfrac{ya_j - c_j}{y'a_j}\colon j \in \{1, 2, \ldots, n\}, y'a_j > 0\right\}$
15: $y \leftarrow y + ty'$
16: $B \leftarrow B - a_i + a_j$ ▷ B bázisban az i. oszlopot cseréljük a j. oszlopra
17: **go to** 4
18: **end procedure**

1.2. eljárás. duál szimplex

1.3 A felső korlátos szimplex algoritmus

Lineáris programozási feladatoknál a gyakorlatban sokszor előfordul, hogy egy változónak van alsó és felső korlátja is. Nézzük meg a következő általános alakú lineáris programozási feladatot:

$$\min\{cx\colon Ax = b, l \leq x \leq u\}.$$

Az alsó korlát egy $x' = x - l$ helyettesítéssel és a b vektor módosításával könnyen átírható nemnegativitási feltétellé. A felső korlát is átírható nemnegativitási feltétellé, azonban így a mátrix mérete nagy mértékben nőne. Ez motiválja a **felső korlátos szimplex algoritmust**.

Vegyük észre, hogy a minimális költségű folyamfeladat is felírható ilyen alakban. Ehhez először vegyük az irányított hálózathoz tartozó M incidenciamátrixot. Tekintsünk a költségfüggvény transzponáltjára és a folyamra egy $\mathbb{R}^m$-beli, az igényfüggvényre egy $\mathbb{R}^n$-beli vektorként. Ekkor a minimális költségű folyamfeladat a következő alakú:

$$\min\{cx\colon Mx = b, 0 \leq x \leq u\}.$$

Az eddig vizsgált szimplex algoritmusokban a bázison kívüli oszlopokhoz tartozó változók értékét mindig rögzítettük az alsó korlátjukhoz – nullához. A felső korlátos szimplex algoritmusban a bázison kívüli oszlopokhoz tartozó változók értékét vagy az alsó, vagy a felső korlátjukhoz rögzítjük. Ennek következtében az A mátrix felírható $A = (B \quad L \quad U)$ alakban, ahol az L részmátrix oszlopaihoz tartozó változók az alsó, míg az U részmátrix oszlopaihoz tartozó változók a felső korlátjukon vannak, a B pedig egy teljes rangú részmátrix. A B bázis oszlopaihoz tartozó változók is lehetnek valamelyik korlátjukon. Egy ilyen particionálást **bázisstruktúrának** hívunk. Az x vektor is particionálható $x = (x_B \quad x_L \quad x_U)^T$ alakban. Vezessük be a $z = cx$ jelölést az adott bázisstruktúrához tartozó célfüggvény értékére.

A következő összefüggés írható fel:

$$Bx_B + Lx_L + Ux_U = b.$$

Ebből x_B vektort kifejezve kapjuk, hogy

$$x_B = B^{-1}b - B^{-1}Lx_L - B^{-1}Ux_U. \tag{1.2}$$

Az $x_L = l_L$, $x_U = u_U$ helyettesítést elvégezve adódik

$$\beta = B^{-1}b - B^{-1}Ll_L - B^{-1}Uu_U.$$

Az $x = (x_B \quad x_L \quad x_U)^T = (\beta \quad l_L \quad u_U)^T$ megoldást **bázismegoldásnak** hívjuk. Ha az $l_B \leq x_B \leq u_B$ feltételek is teljesülnek, akkor **megengedett bázismegoldásról** beszélünk. A c vektort is hasonlóan felbonthatjuk $c = (c_B \quad c_L \quad c_U)$ partíciókra. Ekkor

$$z = c_Bx_B + c_Lx_L + c_Ux_U. \tag{1.3}$$

Az (1.3) kifejezésbe helyettesítve (1.2) kifejezést, hogy

$$z = c_BB^{-1}b - (c_BB^{-1}L - c_L)x_L - (c_BB^{-1}U - c_U)x_U. \tag{1.4}$$

Jelölje J_L és J_U azon indexhalmazokat, melyek elemeihez tartozó változók az alsó, illetve a felső korlátjukhoz rögzítettek. Legyenek $m = c_BB^{-1}L$ és $n = c_BB^{-1}U$ vektorok. Ezek dimenziói megegyeznek c_L és c_U vektorok dimenzióival. Ekkor az

(1.4) kifejezésből kapjuk, hogy

$$z = c_B B^{-1} b - \sum_{j \in J_L} (m_j - c_j) x_j - \sum_{j \in J_U} (n_j - c_j) x_j. \tag{1.5}$$

Jegyezzük meg, hogy itt az $m - c_L$ és az $n - c_U$ vektorok komponensei felelnek meg a hálózati szimplex algoritmusban az egyes élekhez tartozó redukált költségfüggvénnyel.

Most vizsgáljuk meg, hogy egy báziscsere (pivotálás) során hogyan változik a célfüggvény értéke. Vegyük az x_j bázison kívüli változót. Először nézzük meg azt az esetet, amikor $j \in J_L$. Mivel az x_j ekkor az alsó korlátján van, így az értékét csak növelni tudjuk. Az (1.5) kifejezés alapján ez pontosan akkor csökkenti a célfüggvény értékét, ha $m_j - c_j > 0$. Hasonlóan $j \in J_U$ esetén az x_j változó a felső korlátján van, így csak csökkenteni lehet az értékét. Szintén az (1.5) kifejezés alapján leolvasható, hogy ez pontosan akkor csökkenti a célfüggvény értékét, ha $n_j - c_j < 0$.

1.3.1. Állítás (Optimalitási feltétel a felső korlátos szimplex algoritmusra)**.** *Egy megengedett bázismegoldás pontosan akkor optimális, ha* $m_j - c_j \leq 0$ *minden* $j \in J_L$ *indexre, valamint* $n_j - c_j \geq 0$ *minden* $j \in J_U$ *indexre.*

2. fejezet

Az erősen polinomiális primál hálózati szimplex algoritmus

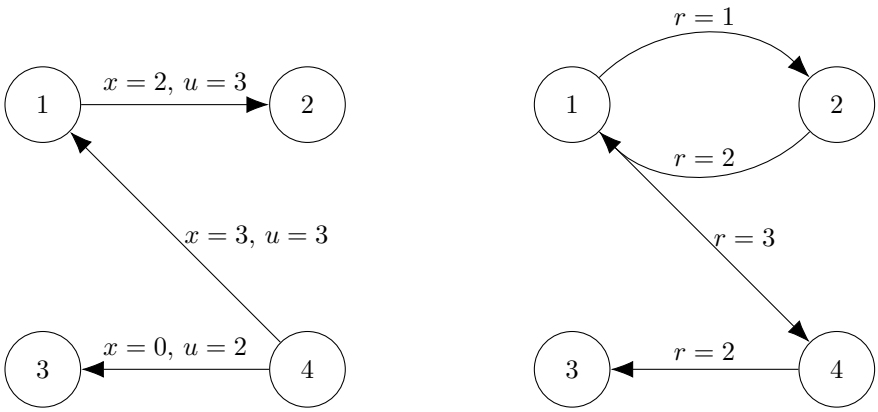
Legyen $G = (N, A, u, c, b)$ egy irányított hálózat az 1.1. szakaszban definiált módon, ahol n a csúcsok száma, m az élek száma. Ebben a fejezetben vizsgálunk egy $O(\min\{n^2 m \log nC, n^2 m^2 \log n\})$ futásidejű primál hálózati szimplex algoritmust a minimális költségű folyamfeladat megoldására.

Először megnézzük a primál hálózati szimplex algoritmust (2.1. eljárás), majd a premultiplikátor szimplex algoritmust (2.2. eljárás), amely az előbbinek egy speciális változata. Ennek mutatjuk meg azt a költség skálázós változatát (2.5. eljárás), amely megoldja a minimális költségű folyamfeladatot $O(\min\{nm \log nC, nm^2 \log n\})$ pivotálással. Ehhez egy olyan adatstruktúrát használunk majd, amelyben a pivotálások $O(n)$ lépésben végrehajthatóak, ami a fenti futásidőt adja.

2.1 A primál hálózati szimplex algoritmus

Legyen a csúcsok halmaza $N = \{1, 2, \ldots n\}$. Először módosítsuk a hálózatot és vegyünk fel segédéleket. Legyenek ezek az $(1, v)$ és $(v, 1)$ élek minden $v \neq 1$ csúcsra $+\infty$ kapacitással és $1+nC$ költséggel. Ezeket az éleket használhatjuk kezdeti megengedett bázisnak. Azért van szükségünk $(1, v)$ és $(v, 1)$ élekre is, hogy a később definiált nemdegenerált feltevéshez tudjunk alkalmazni perturbációs technikákat. Ha egy optimális megoldásban bármelyik segédélen pozitív a folyamérték, akkor az eredeti minimális költségű folyamfeladat nem megoldható.

Minden megengedett x folyam esetén az élekhez tartozik **reziduális kapacitás** a következőképpen: ha $(v, w) \in A$, akkor a (v, w) él reziduális kapacitása $r_{vw} = u_{vw} - x_{vw}$, költsége c_{vw}. Ha $(w, v) \in A$, akkor a (v, w) él reziduális kapacitása $r_{vw} = x_{wv}$, ennek költsége $-c_{wv}$. Használjuk a $+\infty - x_{vw} = +\infty$ konvenciót. Jelölje $G(x)$ a pozitív reziduális kapacitású élekből álló segédgráfot. A (v, w) jelölést használjuk az élekre, de a hálózatban lehetnek többszörös élek is, de ezek nem befolyásolják az algoritmusoknak sem a futását, sem a futásidejét.



2.1. ábra. példa reziduális hálózatra

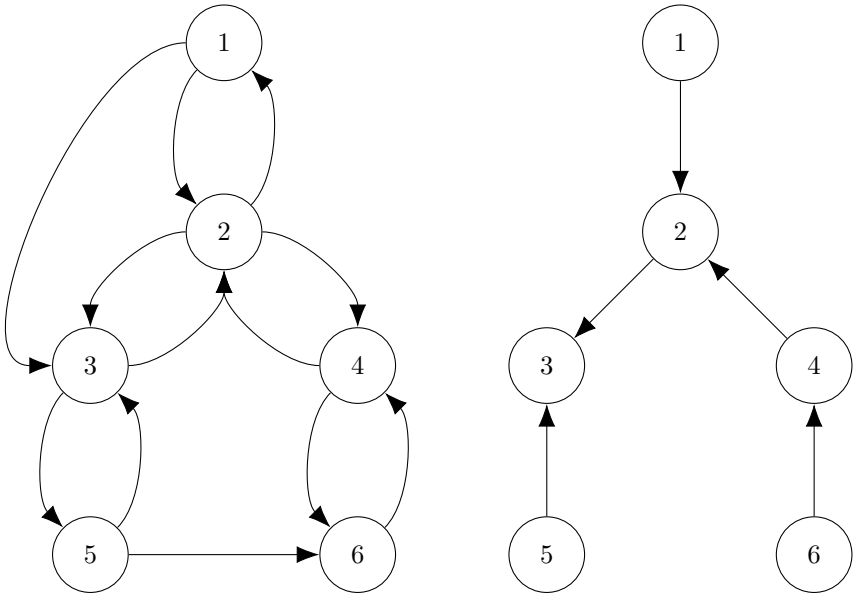
A minimális költségű folyamfeladat esetében – a felső korlátos szimplex algoritmushoz hasonlóan – egy bázisstruktúra az éleknek egy (T, L, U) particionálásával adható meg. Itt T jelöli a bázisbeli éleket, melyek feszítőfát alkotnak a hálózatban, az L és az U partíciókba pedig a bázison kívüli élek tartoznak . Előbbi halmaz elemei az alsó korlátjukon, utóbbi halmaz elemei pedig a felső korlátjukon vannak. Minden bázisstruktúrához tartozik egy x folyam, amely a következőképpen számolható ki: minden $(v, w) \in L$ élre $x_{vw} = 0$, minden $(v, w) \in U$ élre $x_{vw} = u_{vw}$, a T-beli élekhez rendelt folyamértékek pedig az első fejezetben leírt (1.1) feltételrendszerből egyértelműen megkaphatók, mivel a T-beli élek feszítőfát alkotnak. Azt mondjuk, hogy a (T, L, U) bázisstruktúra **megengedett bázisstruktúra**, ha $0 \leq x_{vw} \leq u_{vw}$ teljesül minden $(v, w) \in T$ élre. A megengedett bázisstruktúrából számolt x folyamot **megengedett megoldásnak**, vagy **megengedett folyamnak** hívjuk. A hálózati szimplex algoritmus (2.1. eljárás) minden lépésben megengedett bázisstruktúrát tart fenn.

Tegyük fel, hogy minden megengedett megoldás nemdegenerált, azaz nincs olyan T-beli él, amely vagy az alsó, vagy a felső korlátjára van állítva. Ezt hívjuk **nemdegeneráltsági feltevésnek**. Ez egy erős megkötésnek tűnik, de több perturbációs

technika is létezik, melyeket alkalmazva a perturbált feladat szerinti optimális bázisstruktúrák pontosan az eredeti feladat szerinti optimális bázisstruktúrák. Egy ilyen technika olvasható James B. Orlin [4] cikkében. Ennek alapja, hogy az 1 csúcs igényét kellően kicsi ε értékkel megnöveljük, a többi csúcs igényét pedig $\varepsilon/(n-1)$ egységgel csökkentjük. A nemdegeneráltsági feltevés egyben azt is jelenti, hogy a $G(x)$ segédgráf tartalmazza a T-beli éleket és azok megfordítottjait is, míg a bázison kívüli élekre $(v, w) \in G(x)$ esetén $(w, v) \notin G(x)$.

2.1.1. Definíció. *Jelölje $G^*(x)$ a $G(x)$ segédgráfnak azon részgráfját, amely nem tartalmazza a T-beli éleket és azok megfordítottjait.*

2.1.2. Definíció. *Egy adott T fa és egy kijelölt v gyökércsúcs esetén jelölje $T(v)$ a $G(x)$ segédgráfnak azt a T szerinti részfáját, melyben minden él a v csúcs felé irányított.*



2.2. ábra. példa $G(x)$-re és ebből kapott $T(3)$-ra

Legyen x egy megengedett megoldás és T az ehhez tartozó feszítőfa. Ha $(p, q) \in G^*(x)$, akkor ezt az élt a fához hozzávéve keletkezik egy W kör, melyet a (p, q) **él által generált alapkörnek** nevezünk, és a (p, q) élből, valamint a $T(p)$-beli irányított $q - p$ útból áll. Legyen $\varepsilon = \min\{r_{vw} \colon (v, w) \in W\}$. Küldjünk körbe a W körön ε egység folyamot, ezzel csökkentve az élek reziduális kapacitását és növelve a fordított élekét. A nemdegeneráltsági feltevés miatt pontosan egy olyan él lesz a körben, melynek a reziduális kapacitása 0-ra csökken. A pivotálás során hozzáadunk a fához egy (p, q) élt, az általa generált W kör mentén küldünk ε egység folyamot,

majd kivesszük a bázisból azt az élt, melynek a reziduális kapacitása 0-ra csökken. Ha a W kör összköltsége negatív, valamint ε pozitív, akkor a pivotálás során a célfüggvény értéke csökken.

2.1.3. Állítás (Optimalitási feltétel a primál hálózati szimplex algoritmusra)**.** *A (T, L, U) bázisstruktúra optimális, ha minden $(v, w) \in G^*(x)$ él által generált alapkör összköltsége nemnegatív.*

```
1: procedure PRIMÁL_HÁLÓZATI_SZIMPLEX()
2:     keressünk egy (T, L, U) megengedett bázisstruktúrát
3:     kiszámoljuk az ehhez tartozó x megengedett megoldást
4:     while x nem optimális do
5:         legyen (p, q) ∈ G*(x): (p, q) által generált W alapkör összköltsége negatív
6:         ε ← min{r_vw: (v, w) ∈ W}
7:         pivotálás ε értékű folyam küldésével W körön
8:         x és (T, L, U) frissítése
9:     end while
10: end procedure
```

2.1. eljárás. primál hálózati szimplex

A nemdegeneráltsági feltevéssel a primál hálózati szimplex algoritmus (2.1. eljárás) véges sok lépésben leáll, mert véges sok bázisstruktúra létezik a hálózatban, és a végrehajtása során kapott bázisstruktúrákon a célfüggvény értéke szigorúan monoton csökken.

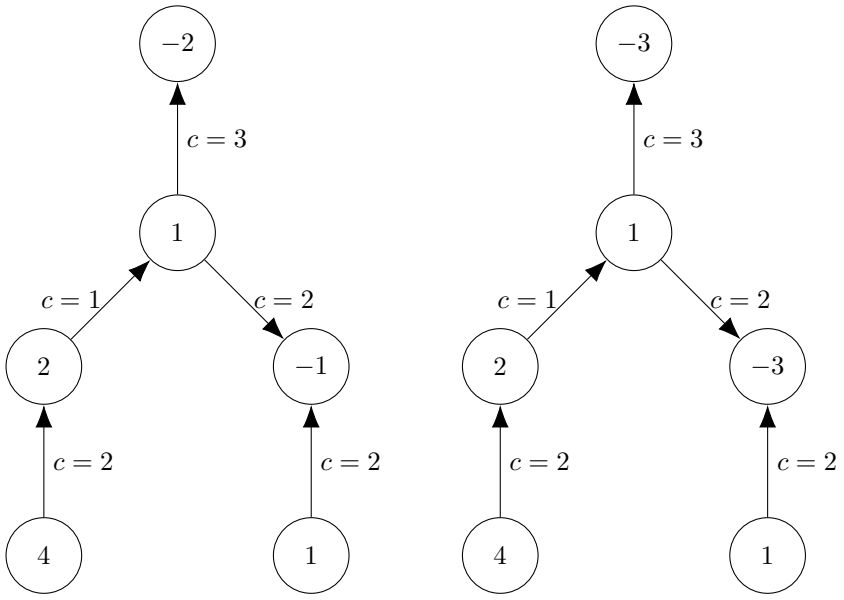
A p potenciálfüggvényt **szimplex változónak** hívjuk, ha a T feszítőfa minden (v, w) élére $c^p_{vw} = 0$. A primál hálózati szimplex algoritmusban (2.1. eljárás) fenntartunk egy ilyen tulajdonságú potenciálfüggvényt. Ennek előnye, hogy a $(p, q) \in G^*(x)$ él által generált alapkör összköltsége c^p_{pq}, így negatív összköltségű kör keresésénél csak azt kell leellenőrizünk, hogy létezik-e olyan $(p, q) \in G^*(x)$ él, melyre $c^p_{pq} < 0$. Minden báziscsere után a p potenciálfüggvényt $O(n)$ időben frissíthetjük, de erre vannak hatékonyabb módszerek is, Alexander Schrijver [5] könyvében olvashatunk is egy ilyet.

2.2 A premultiplikátor szimplex algoritmus

2.2.1. Definíció. *Azt mondjuk, hogy a p potenciálfüggvény **premultiplikátora** a $T(v)$ fának, ha minden $(v, w) \in T(v)$ élre $c^p_{vw} \leq 0$. A p potenciálfüggvény premultiplikátora a T fának, ha létezik olyan $v \in T$ csúcs, melyre p premultiplikátora a $T(v)$ fának.*

2.2.2. Lemma. *Legyen T egy fa és p egy premultiplikátora a $T(v)$ fának. Ekkor p pontosan akkor premultiplikátora a $T(i)$ fának, ha a T-beli egyértelmű $w - v$ út mindegyik élének a redukált költsége 0.*

Bizonyítás. Legyen P a $T(v)$-beli irányított $w-v$ út. Ekkor $T(w)$ megkapható $T(v)$-ből a P út éleinek megfordításával. Először tegyük fel, hogy a $T(v)$-beli P út minden élének a redukált költsége 0. Ekkor a fordított éleknek a redukált költsége is 0, azaz nemnegatív, a többi él pedig megegyezik a két fában. Most nézzük meg azt az esetet, amikor P-nek van negatív redukált költségű (p, q) éle. Ekkor a $T(w)$ fában a (q, p) él redukált költsége pozitív, mivel ellentettje (p, q) élének, ebből pedig következik, hogy p nem lehet premultiplikátora $T(w)$-nek. □



2.3. ábra. példa szimplex változóra és premultiplikátorra, csúcsokban p_v értékekkel

2.2.3. Definíció. *Legyen T egy fa és p egy premultiplikátora a $T(v)$ fának valamely $v \in T$ csúcsra. Azt mondjuk, hogy a w* ***csúcs megengedett****, ha p premultiplikátora a $T(w)$ fának is. Egy $(p, q) \in G^*(x)$ élt* ***megengedett élnek*** *hívunk, ha a p csúcs megengedett és $c^p_{pq} < 0$.*

2.2.4. Lemma. *Legyen T egy fa és p egy premultiplikátora a $T(v)$ fának valamely $v \in T$ csúcsra. Ekkor minden megengedett él által generált alapkör összköltsége negatív.*

Bizonyítás. Legyen (p, q) egy megengedett él és W az által generált alapkör. Legyen z a körnek a v csúcshoz T szerinti legközelebb eső csúcsa ($z = v$ is lehetséges). Ekkor

$W = \{(p,q) \cup P \cup P'\}$, ahol P a $T(v)$-beli irányított $q - z$ út, P' pedig a $T(p)$-beli irányított $z - p$ út. Tudjuk, hogy

$$c(W) = c^p(W) = c^p_{pq} + \sum_{(k,l)\in P} c^p_{kl} + \sum_{(k,l)\in P'} c^p_{kl}.$$

Egyrészt $c^p_{pq} < 0$ a megengedett él definíciója miatt. Másrészt $c^p_{kl} \leq 0$ minden $(k,l) \in P$ élre, mivel p premultiplikátor. Harmadrészt $c^p_{kl} = 0$ minden $(k,l) \in P'$ élre a 2.2.2. lemma miatt. Ezekből következik, hogy $c(W) < 0$. □

```
1: procedure PREMULTIPLIKÁTOR_SZIMPLEX()
2:     keressünk egy (T, L, U) megengedett bázisstruktúrát
3:     kiszámoljuk a (T, L, U)-hoz tartozó x megengedett megoldást
4:     legyen p premultiplikátora a T(v) dának
5:     while x nem optimális do
6:         if létezik megengedett él then          ▷ p premultiplikátor T(v)-re
7:             legyen (p, q) megengedett él
8:             szimplex_pivotálás(p, q)            ▷ 2.3. eljárás
9:         else
10:            premultiplikátor_frissítés()        ▷ 2.4. eljárás
11:        end if
12:    end while
13: end procedure
```

2.2. eljárás. Premultiplikátor szimplex

```
1: procedure SZIMPLEX_PIVOTÁLÁS(p, q)
2:     legyen W a (p, q) által generált alapkör
3:     ε ← min{r_vw : (v, w) ∈ W}
4:     ε értékű folyam küldése W körön
5:     legyen (k, l) a bázisból kilépő él
6:     T gyökerét állítsuk k-ra          ▷ p premultiplikátora a T(k) fának
7: end procedure
```

2.3. eljárás. szimplex pivotálás

```
1: procedure PREMULTIPLIKÁTOR_FRISSÍTÉS()
2:     legyen S a megengedett csúcsok halmaza
3:     Q ← {(v, w) ∈ T(v) : v ∉ S,   w ∈ S}
4:     δ ← min{−c^p_vw : (v, w) ∈ Q}          ▷ δ > 0, ha S ≠ N
5:     minden v ∈ S csúcsra p_v ← p_v + δ
6: end procedure
```

2.4. eljárás. premultiplikátor frissítés

A szimplex pivotálás (2.3. eljárás) során a (p, q) élt bevesszük a bázisba, a (k, l) élt pedig kivesszük, ezzel fenntartunk egy megengedett bázisstruktúrát. A fa gyökerét is módosítjuk a k csúcsra.

Most azt fogjuk belátni, hogy a premultiplikátor szimplex algoritmus (2.2. eljárás) helyesen megoldja a minimális költségű folyamfeladatot és véges sok lépésben leáll. Először ehhez azt látjuk be, hogy az algoritmus minden lépésében vagy megnöveljük a megengedett csúcsok számát, vagy egy nemdegenerált báziscserét hajtunk végre, ahol a célfüggvény értéke szigorúan csökken. Mivel legfeljebb n darab megengedett csúcs lehet, ezért legfeljebb n lépésenként javítunk a célfüggvény értékén. Minden nemdegenerált báziscsere egy új megengedett bázisstruktúrát ad szigorúan kisebb célfüggvény értékkel. Mivel a minimális költségű folyamfeladatnál a bázisstruktúrák száma véges, ezért a premultiplikátor algoritmus véges sok lépésben leáll.

2.2.5. Lemma. *A premultiplikátor szimplex algoritmus (2.2. eljárás) minden lépése során fenntartunk egy p premultiplikátort.*

Bizonyítás. Ezt a lemmát indukcióval látjuk be a lépések száma szerint. Tegyük fel, hogy p egy premultiplikátor a kezdeti megengedett bázisstruktúrára nézve. Először vizsgáljuk meg a premultiplikátor frissítés (2.4. eljárás) hatását. Legyen p premultiplikátora a $T(v)$ fának, valamint legyen p' premultiplikátor közvetlenül az eljárás végrehajtását követően. Azt kell igazolnunk, hogy $c^{p'}_{vw} < 0$ minden $(v, w) \in T(v)$ élre.

Legyen S halmaz a megengedett csúcsok halmaza a p premultiplikátorra nézve. Az eljárás minden S-beli csúcs értékét δ-val növeli. Világos, hogy ez nincs hatással azon élek redukált költségére, melyeknek a kezdőpontja és a végpontja is S-beli. Definíció szerint $T(v)$ nem tartalmaz olyan (v, w) élt, melyre $v \in S$ és $w \notin S$, így csak azokat a (v, w) éleket kell vizsgálnunk, melyekre $v \notin S$ és $w \in S$, vagyis a Q-val jelölt élek halmazát. Az S-beli csúcsok potenciálját δ-val növelve a Q-beli élek redukált költsége δ-val csökken, így azonban látható, hogy minden ilyen él redukált költsége nempozitív marad, és legalább egy él redukált költsége 0-ra nő.

Most nézzük meg a szimplex pivotálás (2.3. eljárás) hatását. Legyen (p, q) a belépő él. Mivel (p, q) megengedett él, ezért p megengedett csúcs, így p premultiplikátora a $T(p)$ fának, azaz minden $(v, w) \in T(p)$ élre $c^p_{vw} \leq 0$. A (p, q) által generált alapkör tartalmazza (p, q) élt és $T(p)$-beli irányított $q - p$ utat. A frissített folyam a nemdegeneráltsági feltevés miatt pontosan egy él reziduális kapacitását változtatja 0-ra. Ez

az él legyen (k, l). Legyen $T' = T-(k, l)+(p, q)$. A $T'(k)-(p, q)$ minden élének iránya megegyezik a $T(p)$-beli irányukkal, így ezekre az élekre a redukált költségfüggvény kisebb egyenlő nullával. Továbbá mivel $c^p_{pq} < 0$, így igazoltuk a lemmát. □

2.2.6. Lemma. *A premultiplikátor frissítés (2.4. eljárás) végrehajtása növeli a megengedett csúcsok számát.*

Bizonyítás. Az eljárás során egy Q-beli (v, w) él redukált költsége 0-ra nő. Mivel Q definíciója szerint w egy megengedett csúcs, így v is megengedett csúccsá válik, vagyis az S halmaz mérete nő. □

2.2.7. Tétel. *A premultiplikátor szimplex algoritmus (2.2. eljárás) egy speciális változata a primál hálózati szimplex algoritmusnak (2.1. eljárás). Mint ilyen, véges sok lépésben megoldja a minimális költségű folyamfeladatot.*

Bizonyítás. Az premultiplikátor szimplex algoritmus (2.2. eljárás) minden lépésében fenntartunk egy premultiplikátort. A megengedett élek által generált alapkörök összsúlya mindig negatív. Emiatt a premultiplikátor algoritmus egy speciális változata a primál hálózati szimplex algoritmusnak. A végesség a nemdegeneráltsági feltevésből következik, amely garantálja, hogy a bázisstruktúrák nem ciklizálnak. □

2.3 A skálázós premultiplikátor szimplex algoritmus

Ebben a szakaszban megismerünk egy erősen polinomiális futásidejű primál hálózati szimplex algoritmust. Az algoritmus során fenntartunk egy pozitív Δ értéket, amely a skálázási faktor lesz, és az algoritmust skálázási fázisokra osztja. Akkor lépünk egyik ilyen fázisból a másikba, ha a skálázási faktor megváltozik. Az algoritmus $O(\min\{\log nC, m \log n\})$ skálázási fázisból áll, és minden ilyen fázisban $O(nm)$ pivotálást végez, így a pivotálások száma $O(\min\{nm \log nC, nm^2 \log n\})$. Ennek egy olyan implementációját mutatjuk meg, melyben egy pivotálás $O(n)$ ideig tart, így az algoritmus futásideje $O(\min\{nm \log nC, nm^2 \log n\})$. Először vezessünk be négy új fogalmat. Mindegyik fogalom úgy értendő, hogy az aktuális skálázási faktor Δ.

2.3.1. Definíció. *Jelölje N^* azon csúcsok részhalmazát, melyeknek a potenciálja még nem változott az aktuális skálázási fázis során.*

2.3.2. Definíció. *Legyen p egy premultiplikátora a T fárnak és x egy megengedett megoldás. Azt mondjuk, hogy a p **Δ-premultiplikátor**, ha $c^p_{vw} \geq -\Delta$ teljesül minden $(v, w) \in G(x)$ élre.*

2.3.3. Definíció. *Egy v csúcsot **éber csúcsnak** hívunk, ha $v \in N^*$, vagy a hozzá tartozó p_v potenciál $\Delta/4$ egész többszöröse. A nem éber csúcsokat **alvó csúcsoknak** nevezzük.*

2.3.4. Definíció. *Azt mondjuk, hogy a $(p, q) \in G^*(x)$ él **választható él** a Δ-skálázási fázisban, ha p egy megengedett és éber csúcs, valamint $c^p_{pq} \leq -\Delta/4$.*

A javított közelítés (2.6. eljárás) bemenetnek kap egy p Δ-premultiplikátor az x megengedett megoldásra nézve és a kimenete egy p' $\Delta/2$-premultiplikátor az x' megengedett megoldásra nézve. Minden skálázási fázisnak akkor van vége, amikor $N^* = \emptyset$ lesz.

```
1: procedure SKÁLÁZÓS_PREMULTIPLIKÁTOR_SZIMPLEX()
2:     keressünk egy (T, L, U) megengedett bázisstruktúrát
3:     kiszámoljuk a (T, L, U)-hoz tartozó x megengedett megoldást
4:     legyen p premultiplikátora T(v) fának
5:     Δ ← max{|c^p_vw|: c^p_vw ≤ 0,   (v, w) ∈ G(x)}
6:     while x nem optimális do
7:         javított_közelítés(x, Δ, p)                    ▷ 2.6. eljárás
8:         Δ ← max{|c^p_vw|: c^p_vw ≤ 0,   (v, w) ∈ G(x)} ▷ Δ legalább a felére csökken
9:     end while
10: end procedure
```

2.5. eljárás. skálázós premultiplikátor szimplex

```
1: procedure JAVÍTOTT_KÖZELÍTÉS(x, Δ, p)
2:     N* ← N
3:     while N* ≠ ∅ do
4:         if létezik választható él then
5:             legyen (p, q) választható él                ▷ c^p_pq ≤ −Δ/4
6:             szimplex_pivotálás(p, q)                    ▷ 2.3. eljárás
7:         else
8:             Δ-premultiplikátor_frissítés()              ▷ 2.7.eljárás
9:         end if
10:    end while
11: end procedure
```

2.6. eljárás. javított közelítés

```
1: procedure Δ-PREMULTIPLIKÁTOR FRISSÍTÉS()
2:     legyen S a választható csúcsok halmaza
3:     N* ← N* − S
4:     if N* = ∅ then
5:         terminate javított_közelítés(x, Δ, p)          ▷ 2.6. eljárás
6:     end if
7:     δ1 ← min{−c^p_vw : (v, w) ∈ T(v),   v ∉ S,   w ∈ S}
8:     δ2 ← min{Δ/4 − p_v  mod Δ/4 : v ∈ S}
9:     δ ← min{δ1, δ2}
10:    for v ∈ S do
11:        p_v ← p_v + δ                                  ▷ δ > 0
12:    end for
13: end procedure
```

2.7. eljárás. Δ-premultiplikátor frissítés

Nézzük meg a skálázós premultiplikátor szimplex (2.5. eljárás) néhány jellemzőjét, melyeket a következő szakaszban igazoni fogunk:

- A csúcsokhoz tartozó p_v potenciálok monoton nőnek, valamint minden skálázási fázisban legalább egy csúcs potenciálja nő, de minden csúcs potenciálja legfeljebb $3n\Delta/2$ egységgel nőhet a Δ-skálázási fázis során
- Minden bázisba kerülő (p, q) megengedett élre $c^p_{pq} \leq -\Delta/4$, ezért minden megengedett él által generált alapkör összsúlya kisebb vagy egyenlő $-\Delta/4$-gyel.
- Minden (p, q) él legfeljebb $6n$ alkalommal kerülhet be a bázisba egy skálázási fázis során, így a pivotálások száma fázisonként $O(nm)$.
- A skálázási fázisok száma $O(\min\{m \log n, \log nC\})$.

Jegyezzük meg, hogy $p_v \mod \Delta/4 \in [0, \Delta/4)$, így $\delta_2 > 0$. A Δ-premultiplikátor frissítés() eljárásban úgy defniáltuk δ_2 értékét, hogy a legkisebb pozitív valós szám legyen valamely választható v csúcsra, hogy $p_v + \delta_2$ a $\Delta/4$ egész többszöröse legyen.

2.4 Futásidő vizsgálat

Ebben a szakaszban először belátjuk, hogy skálázós premultiplikátor szimplex algoritmusban (2.5. eljárás) egy pivotálás $O(n)$ időben el tudunk végezni, majd azt, hogy $O(\min\{\log nC, m \log n\})$ skálázási fázis alatt találunk egy optimumot, és minden skálázási fázis $O(nm)$ során $O(mn)$ alkalommal pivotálunk.

Azt szeretnénk, hogy $O(n)$ idő alatt tudjunk báziscseréket végezni. Ehhez figyelnünk kell arra, hogy a választható élek megtalálása ne vegyen el sok túl időt.

Ehhez bevezetjük az **aktuális él adatstruktúrát**. Ebben minden v csúcshoz tartozik egy éllista, amely a $G(x)$ segédgráfbeli v csúcsból kiinduló éleket tartalmazza egy előre meghatározott, rögzített sorrendben. A nevéből adódóan minden csúcshoz tartozik egy aktuális él ebből a listából. Ha a v csúcsból induló választható élt keresünk, akkor ellenőrizzük, hogy a csúcshoz tartozó aktuális él választható-e. Ha nem, akkor az éllista sorban következő elemét ellenőrizzük. Ezt addig ismételjük, amíg vagy találunk egy ilyen élt, vagy elfogynak az élek a listából. Utóbbi esetben az aktuális élnek állítsuk be az $\emptyset$ értéket, ami azt jelenti, hogy ebből a csúcsból nem indul választható él. Csak akkor módosítsuk ismét a v csúcshoz tartozó aktuális élt, ha a v csúcs ismét éberré válik, azaz a hozzá tartozó p_v potenciál $\Delta/4$ egész többszöröse lesz. Ekkor állítsuk be az éllistájának első elemét.

Választható él keresése során először a gyökérből indulva futtassunk egy mélységi keresést a T fán, hogy megtaláljuk a megengedett csúcsokat. Mivel a feszítőfában $(n-1)$ él van, a mélységi keresés futásideje pedig élszám nagyságrendű, ezért a megengedett csúcsokat $O(n)$ időben megtaláljuk. A szimplex pivotálás (2.3. eljárás), vagy a premultiplikátor frissítés (2.4. eljárás) végrehajtása után azon megengedett és éber csúcsok megtalálása, melyeknek az aktuális éle nem $\emptyset$, $O(n)$ lépésben megtehető. Jegyezzük meg, hogy a (v, w) élt a Δ-skálázási fázis során legfeljebb annyiszor vizsgálunk, ahányszor a v csúcs éberré váik.

Ha mindig leellenőriznénk a v csúcs aktuális élét, amikor a potenciálja nő, akkor nagyon sok időt venne igénybe ez a feladat. Amikor a v csúcs aktuális éle $\emptyset$ lesz, akkor legyen a v csúcs alvó, és egészen addig ne legyen újra éber, ameddig a potenciálja nem nő legalább $\Delta/4$ egységgel. Később megmutatjuk a a 2.4.5. lemmában, hogy a Δ-skálázási fázisban minden csúcs potenciálja $O(n\Delta)$ egységgel nőhet, így $O(n)$ alkalommal válhat éberré, azaz ennyiszer vizsgáljuk meg választható él keresése céljából.

Legyen p' premultiplikátor és x' megengedett megoldás, amikor a v csúcs alvó lesz, és legyen p premultiplikátor és x megengedett megoldás, amikor a v csúcs újra éberré válik. Ekkor minden $(v, w) \in G(x')$ élre $c^{p'}_{vw} > -\Delta/4$, és mivel $p_v = p'_v + \Delta/4$, így később a 2.4.2. lemmában látjuk majd, hogy minden $(v, w) \in G(x)$ élre $c^{p}_{vw} > -\Delta/2$, azaz teljesül a $\Delta/2$ optimalitás minden N^*-beli csúcsból kiinduló élre.

2.4.1. Lemma. *Legyen p egy Δ-premultiplikátor a Δ-skálázási fázis során, $v \notin N^*$ csúcs. Legyen p' közvetlenül a Δ-premultiplikátor frissítés (2.7. eljárás) legutóbbi*

végrehajtása előtti premultiplikátor, amikor v megengedett és éber csúcs volt. Ekkor $0 < p_v - p'_v \leq \Delta/4$.

Bizonyítás. Jelölje p^0 a premultiplikátort közvetlenül a Δ-skálázási fázis kezdetén. Amikor a p_v potenciál először növekszik, akkor a v csúcs egyszerre megengedett és éber, tehát p' jól definiált. Először nézzük meg azt az esetet, amikor $p_v - p_v^0 \leq \Delta/4$. Ekkor a lemma triviálisan igaz. Most tegyük fel, hogy $p_v - p_v^0 > \Delta/4$. Jelölje α a $\Delta/4$ legnagyobb egész többszörösét, amely szigorúan kisebb, mint p_v. Ebből következik, hogy $p_v - \alpha \leq \Delta/4$ és $p_v^0 < \alpha < p_v$. Most megmutatjuk, hogy $\alpha = p'_v$, amellyel igazoljuk a lemmát. Nézzük meg azt a lépést, melyben a v csúcs potenciálja először legalább α. Ekkor a Δ-premultiplikátor frissítés (2.7. eljárás) során a v csúcs potenciálja δ_2 definíciója miatt pontosan α-ra emelkedik. Most nézzük meg az első lépést, melyben a v csúcs potenciálja nagyobbra nő, mint α. Ebben a lépésben a v csúcs egyszerre megengedett és éber is. Ezt követően a v csúcs akkor lenne újra éber, ha a potenciálja $\alpha + \Delta/4 \geq p_v$-nél nagyobbra nőne, de α definíciója miatt ez nem történhet meg, így ebből következik, hogy $p'_v = \alpha$. □

2.4.2. Lemma. *Legyen x egy megengedett megoldás, p egy Δ-premultiplikátor a Δ-skálázási fázis során. Ekkor minden $(v, w) \in G(x)$ élre, ha $v \notin N^*$, akkor $c^p_{vw} \geq -\Delta/2$. Speciálisan, ha $N^* = \emptyset$, akkor p egy $\Delta/2$-premultiplikátor az x megengedett megoldásra nézve.*

Bizonyítás. Vizsgáljunk a $(v, w) \in G(x)$ élt. Legyen p' közvetlenül a Δ-premultiplikátor frissítés (2.7. eljárás) legutóbbi végrehajtása előtti premultiplikátor, amikor v megengedett és éber csúcs volt. Ekkor a (v, w) él vagy fa él, vagy egy nem választható nem fa él, amelyre $c^{p'}_{vw} > -\Delta/4$, vagy nincs reziduális kapacitása. Az első esetben a Δ-premultiplikátor frissítés (2.7. eljárás) meghívását követően teljesül, hogy $c^p_{vw} \geq \Delta/2$. Következőnek vizsgáljuk meg azt az esetet, amikor $c^{p'}_{vw} > -\Delta/4$. Ekkor

$$c^p_{vw} = c^{p'}_{vw} - (p_v - p'_v) + (p_w - p'_w).$$

A 2.4.1. lemma miatt $(p_v - p'_v) \leq \Delta/4$ és $p_w \geq p'_w$. Ebből következik, hogy

$$c^p_{vw} \geq -\frac{\Delta}{4} - \frac{\Delta}{4} + 0 = -\frac{\Delta}{2}.$$

Végül azt az esetet vizsgáljuk, amikor a (v, w) élnek nincs reziduális kapacitása abban a lépésben, amikor p' a premultiplikátor. Tegyük fel, hogy egy későbbi lépésben

kap reziduális kapacitást, amikor még mindig p' a premultiplikátor. Mivel a (v, w) él pontosan akkor kap reziduális kapacitást, ha a (w, v) él belép a bázisba, ezért $c_{wv}^{p'} \leq -\Delta/4$. Ebből következik, hogy $c_{vw}^{p'} = -c_{wv}^{p'} \geq \Delta/4$, továbbá

$$c_{vw}^{p} = c_{vw}^{p'} - (p_v - p'_v) + (p_w - p'_w) \geq -\Delta/4 - \Delta/4 + 0 = -\frac{\Delta}{2}.$$

Így leellenőriztük az összes lehetséges esetet, amelyből következik a lemma állítása. □

2.4.3. Lemma. *A Δ-premultiplikátor frissítés (2.7. eljárás) végrehajtását követően vagy a megengedett csúcsok száma nő, vagy valamelyik csúcs éberré válik.*

Bizonyítás. Legyen p közvetlenül a Δ-premultiplikátor frissítés (2.7. eljárás) legutóbbi végrehajtása előtti premultiplikátor, p' pedig az eljárást követő premultiplikátor. Jelölje S a p szerinti megengedett csúcsok halmazát. Ekkor mindegyik S-beli csúcs a p' szerint is megengedett. Az első esetben $\delta = \delta_1$. Ekkor a 2.2.6. lemma bizonyításában olvasható módon legalább egy olyan csúcs lesz, amely a p' multiplikátorra nézve megengedetté válik, de azt megelőzően nem volt, így a megengedett csúcsok száma nő. A második esetben $\delta = \delta_2$. Ekkor valamely v csúcs p_v potenciálja $\Delta/4$ egész többszörösére emelkedik, azaz a v csúcs felébred. □

Most belátjuk, hogy a Δ-skálázási fázisban az egyes csúcsok potenciálja legfeljebb $3n\Delta/2$ egységgel nőhet, ezzel korlátozva a pivotálások számát.

2.4.4. Lemma (James B. Orlin [1])**.** *Legyen x és x' két különböző, nemdegenerált, megengedett folyam. Ekkor tetszőleges v, w csúcspárra létezik olyan v csúcsból a w csúcsba vezető P út a $G(x)$ segédgráfban, hogy ennek megfordítottja egy w csúcsból v csúcsba vezető út a $G(x')$ segédgráfban.*

2.4.5. Lemma. *A Δ-skálázási fázisban minden csúcs potenciálja legfeljebb $3n\Delta/2$ egységgel nőhet.*

Bizonyítás. Legyen a Δ-skálázási fázis elején x egy megengedett folyam, p egy ehhez tartozó premultiplikátor. Ebben a skálázási fázisban néhány lépéssel később jelölje x' a megengedett megoldást és p' a hozzá tartozó premultiplikátort. A skálázási fázis véget ér, ha $N^* = \emptyset$, így egy köztes lépésben $N^* \neq \emptyset$. Definíció szerint minden $v \in N^*$ csúcsra $p'_v = p_v$. Vegyünk v és w csúcsokat úgy, hogy $v \in N \setminus N^*$ és $w \in N^*$. Jelöljön P egy utat a w csúcsból a v csúcsba $G(x)$-ben úgy, hogy ennek a P'-vel jelölt

megfordítottja $G(x')$-beli legyen. Ilyen út létezik a 2.4.4. lemma szerint. Feltehető, hogy w az egyetlen N^*-beli csúcs az úton. A Δ-optimalitásból következik, hogy

$$c^p(P) = c(P) - p_w + p_v \geq -(n-1)\Delta. \tag{2.1}$$

Mivel P' minden éle $N \setminus N^*$-beli csúcsból indul, ezért a 2.4.2. lemma miatt

$$c^{p'}(P') = c(P') - p'_v + p'_w \geq -(n-1)\frac{\Delta}{2}. \tag{2.2}$$

Adjuk össze a (2.1) és a (2.2) kifejezések ellentettjét. Mivel $p'_w = p_w$ és $c(P)+c(P') = 0$, így azt kapjuk, hogy $p'_v \leq p_v + 3(n-1)\Delta/2 \leq p_v + 3n\Delta/2$, amiből következik, hogy a v csúcs potenciálja legfeljebb $3n\Delta/2$ egységgel nőhet. □

2.4.6. Lemma. *A Δ-skálázási fázisban legyen p a premultiplikátor, amikor a (v,w), vagy a (w,v) él bekerül a bázisba, p' pedig a premultiplikátor, amikor a (v,w), vagy a (w,v) él legközelebb ismét bekerül bázisba. Ekkor $p'_v - p_v + p'_w - p_w \geq \Delta/2$.*

Bizonyítás. Tegyük fel, hogy amikor p a premultiplikátor, akkor a (v,w) él kerül be a bázisba. A bizonyítás hasonló, ha a (w,v) él kerül be a bázisba. Mivel a (v,w) él bekerült a bázisba, ezért $c^p_{vw} = c_{vw} - p_v + p_w \leq -\Delta/4$.

Az első esetben a (w,v) él kerül be a bázisba ezt követően. Ekkor $c^{p'}_{wv} \leq -\Delta/4$, ezért $c^{p'}_{vw} = -c^{p'}_{wv} \geq \Delta/4$. Ebből következik, hogy

$$\frac{\Delta}{2} \leq c^p_{wv} - c^{p'}_{wv} = -p_w + p_v + p'_w - p'_v = -(p'_v - p_v) + (p'_w - p_w) \leq 0 + (p'_w - p_w),$$

amiből következik, hogy $p'_v - p_v + p'_w - p_w \geq \Delta/2$.

A második esetben a $(v,w) \in G(x)$ él kerül be a bázisba legközelebb. A két pivotálás között kellett legyen egy olyan pivotálás, amelyben a (w,v) él kikerült a bázisból. Amikor a (v,w) él a korábbi pivotálásban bekerült a bázisba, akkor pozitív értékű folyamot küldtünk rajta keresztül a v csúcsból a w csúcsba, de ahhoz, hogy újra bekerülhessen, le kellett csökkentenünk rajta a folyam értékét 0-ra, vagyis ki kellett kerülnie a bázisból a nemdegeneráltsági feltevés értelmében. Tegyük fel, hogy ennél a köztes pivotálásnál p'' volt a premultiplikátor. Mivel p'' premultiplikátora $T(v)$-nek és $(w,v) \in T(v)$, ezért $c''_{wv} \leq 0$, továbbá a potenciálok szigorúan monoton nőnek, így $p \leq p'' \leq p'$. Ismerjük a következő összefüggéseket:

$$c_{vw} - p_v + p_w \leq -\frac{\Delta}{4}, \tag{2.3}$$

$$c_{vw} - p''_v + p''_w \geq 0, \tag{2.4}$$

$$c_{vw} - p'_v + p'_w \leq -\frac{\Delta}{4}. \tag{2.5}$$

A (2.4) kifejezésből kivonva a (2.3) kifejezést kapjuk, hogy

$$\frac{\Delta}{4} \leq p''_w - p_w + p_v - p''_v \leq p''_w - p_w \leq p'_w - p_w,$$

valamint a (2.4) kifejezésből kivonva a (2.5) kifejezést adódik, hogy

$$\frac{\Delta}{4} \leq p'_v - p''_v + p''_w - p'_w \leq p'_v - p''_v \leq p'_v - p_v.$$

Ezeket összeadva kapjuk a lemma állítását. □

2.4.7. Lemma. *A skálázós premultiplikátor szimplex algoritmusban (2.5. eljárás) egy skálázási fázis során legfeljebb* $6nm$ *pivotálást végzünk.*

Bizonyítás. Használjuk a 2.4.6. lemma bizonyítása során bevezetett jelöléseket p, p' és p'' premultiplikátoroka. Szintén ebből a lemmából tudjuk, hogy $p'_v - p_v + p'_w - p_w \geq \Delta/2$. Mivel a 2.4.5. lemma miatt ismert, hogy a p_v és a p_w potenciálok legfeljebb $3n\Delta/2$ egységgel nőhetnek, így a (v, w) él egy skálázási fázisban legfeljebb $6n$ alkalommal kerülhet be a bázisba. Ebből adódik, hogy egy skálázási fázis során legfeljebb $6nm$ pivotálást végezhetünk. □

2.4.8. Lemma. *A skálázós premultiplikátor szimplex algoritmus (2.5. eljárás)* $O(\min\{m \log n, \log nC\})$ *skálázási fázis után leáll és megtalálunk egy optimális folyamot.*

Bizonyítás. Először azt az esetet nézzük meg, amikor $C < +\infty$, azaz az inputbeli számok egészek. Bebizonyítjuk, hogy ekkor $\log nC$ skálázási fázis alatt eljutunk az optimális folyamhoz. A skálázási faktor kezdeti értéke a segédélek miatt $O(nC)$. A 2.4.2. lemma kimondja, hogy a skálázási faktor a skálázási fázisok között legalább a felére csökken, azaz $O(\log nC)$ fázis alatt szigorúan kisebb lesz, mint $1/n$. Most belátjuk, hogy x egy optimális folyam, ha $\Delta < 1/n$. Szintén az előző lemma miatt p egy Δ-premultiplikátor, azaz $c^p_{vw} \geq -\Delta > -1/n$ minden $(v, w) \in G(x)$ élre. Legyen W egy irányított kör $G(x)$-ben, $c(W) = c^p(W) > -1$. Mivel $C < +\infty$ miatt $c(W)$ egész, így $c(W) \geq 0$ minden $G(x)$-beli irányított körre, azaz x optimális az optimalitási feltétel miatt (a 2.1.3. állítás). Ezzel beláttuk, hogy $O(\log nC)$ skálázási fázis elég.

Most megmutatjuk, hogy $O(m \log n)$ olyan skálázási fázis van, melyben pivotálunk. Azt mondjuk, hogy egy él a Δ-skálázási fázisban **permanens bázison kívüli él**, ha nincs bent egy megengedett bázisban sem, sem az aktuális, sem bármely későbbi skálázási fázisban. A permanens bázison kívüli élek gondolata Éva Tardos [6] cikkéből származik. Minden olyan skálázási fázisban, melyben pivotálunk, ott egy feszítőfabeli él permanens bázison kívüli éllé válik $(4 + \lceil 2 \log n \rceil)$ skálázási fázison belül. Mivel minden él egyszer permanens bázison kívüli éllé válik, így azon skálázási fázisok száma, melyekben pivotálunk, felülről becsülhető $(4m + m\lceil 2 \log n \rceil)$-nel.

Mivel a Δ-skálázási fázisban a potenciálok legfeljebb $3n\Delta/2$ egységgel nőhet és minden skálázási fázisban Δ legalább a felére csökken, így minden potenciál a jelenlegi és a későbbi skálázási fázisok során legfeljebb

$$\frac{3n\Delta}{2} + \frac{3n\Delta}{4} + \frac{3n\Delta}{8} + \cdots = 3n\Delta$$

egységgel nőhet. Ebből következik, hogy ha a (v, w) élre $c^p_{vw} > 3n\Delta$, akkor (v, w) permanens bázison kívüli él, hiszen a redukált költségfüggvénye nem lesz negatív.

Legyen (v, w) egy bázisba kerülő él a Δ-skálázási fázis során és jelölje W az általa generált alapkört. Mivel a (v, w) él választható él, ezért $c(W) < -\Delta/4$. Jelölje Δ' a skálázási faktort $(4 + \lceil 2 \log n \rceil)$ skálázási fázissal később és legyen p' a multiplikátor a Δ'-skálázási fázis elején. Ekkor $\Delta' < \Delta/(16n^2)$, melyből következik, hogy

$$c^{p'}(W) = c(W) < -4n^2\Delta'.$$

Mivel W-nek legfeljebb n éle van, így biztosan van olyan (p, q) éle, melyre $c^{p'}_{pq} < -3n\Delta'$. Mivel p' egy Δ'-premultiplikátor, ezért $(p, q) \notin G(x)$, hiszen minden $(v, w) \in G(x)$ élre $c^{p'}_{vw} > -\Delta'$ kell teljesüljön. Ebből következik, hogy $(q, p) \in G(x)$, továbbá $c^{p'}_{qp} > 3n\Delta'$, tehát az (q, p) él permanens bázison kívüli él. Ebből következik, hogy $O(m \log n)$ olyan skálázási fázis van, melyben pivotálunk.

Végül adunk egy felső becslést azon skálázási fázisok számára, melyekben nem pivotálunk. Azt látjuk be, hogy nem következhet egymás után két pivotálás nélküli skálázási fázis. Tegyük fel, hogy a Δ-skálázási fázisban nem pivotálunk. Ekkor a Δ-premultiplikátor frissítést (2.7. eljárás) hívjuk meg egymás után addig, ameddig mindegyik csúcs nem lesz megengedett. Ezt követően a feszítőfa minden élére a redukált költségfüggvény 0. A következő fázis elején a skálázási faktor $\Delta = \max\{-c^p_{vw} \colon c^p_{vw} < 0\}$, és keletkezik egy (p, q) él $c^p_{pq} = -\Delta$ redukált

költségfüggvénnyel. Mivel p éber és megengedett csúcs, ezért a (p, q) él választható él, amiből következik, hogy a skálázási pivotálunk. Mivel nincs két egymást követő pivotálás nélküli skálázási fázis, a skálázási fázisok száma $O(m \log n)$. □

2.4.9. Lemma. *A Δ-premultiplikátor frissítést (2.7. eljárás) $O(nm)$ alkalommal hívjuk meg egy skálázási fázis során.*

Bizonyítás. A 2.4.3. lemma alapján a Δ-premultiplikátor frissítés (2.7. eljárás) végrehajtása során vagy egy csúcs éberré válik, vagy egy csúcs megengedett lesz. A 2.4.5. lemma szerint az előbbi eset csúcsonként $6n$ alkalommal fordulhat elő, így összesen $O(n^2)$ alkalommal egy skálázási fázisban. Most szeretnénk egy felső korlátot adni arra, hogy egy csúcs hányszor válhat megengedetté.

Minden alkalommal, amikor egy csúcs megengedetté válik az eljárás végrehajtását követően, akkor van legalább egy olyan $(v, w) \in T$ él, melynek a redukált költsége 0-ra nő, de ezt megelőzően negatív volt. Azt mondjuk, hogy ekkor ezt a fa élt **töröljük**. Belátjuk, hogy egy (v, w) élt annyiszor törölhetünk a fából, ahányszor bevesszük a bázisba. Amikor a (v, w) él bekerül a bázisba, akkor a redukált költsége negatív, és egészen addig negatív marad, ameddig nem töröljük, vagy nem kerül ki a bázisból. Ha a (v, w) élt töröljük, akkor a redukált költségfüggvénye egészen addig 0 marad, ameddig nem kerül ki a bázisból. Ebből következik, hogy egy (v, w) élt két egymást követő bázisba kerülése között csak egyszer törölhetünk, így egy élt $O(n)$ alkalommal törölhetünk egy skálázási fázisban, ezzel pedig bizonyítottuk a lemmát. □

2.4.10. Tétel. *A skálázási premultiplikátor algoritmus megoldja a minimális költség folyamfeladatot $O(\min\{m \log n, \log nC\})$ skálázási fázissal, minden fázisban $O(nm)$ pivotálással, továbbá a futásidő egy skálázási fázisban $O(n^2 m)$.*

Bizonyítás. A 2.4.8. lemma miatt az algoritmus $O(\min\{m \log n, \log nC\})$ skálázási fázis alatt megtalál egy optimális megoldást a minimális költségű folyamfeladatra. A 2.4.7. lemma miatt az algoritmus minden skálázási fázisban $O(nm)$ pivotálást végez. Az alapkörben ε egység folyam küldése és a feszítőfa frissítése elvégezhető $O(n)$ időben. Ha van választható el, akkor annak megtalálása $O(n)$ idő pivotálásonként, az aktuális élek frissítése pedig $O(nm)$ időt vesz igénybe egy skálázási fázis alatt. A 2.4.9. lemma miatt $O(nm)$ alkalomma módosítjuk a premultiplikátort a Δ-premultiplikátor frissítés (2.7. eljárás) végrehajtásával. Minden futáskor potenciálisan az összes csúcsot vizsgálja δ_1 vagy δ_2 kiszámolásakor. Ez, valamint a p_v

potenciálok frissítése szintén elvégezhető $O(n)$ időben. Ebből következik, hogy a futásidő egy skálázási fázis során $O(n^2m)$, amiből kapjuk az állítást. □

3. fejezet

Az erősen polinomiális duál hálózati szimplex algoritmus

Legyen $G = (N, A, u, c, b)$ egy irányított hálózat az 1.1. szakaszban definiált módon, n a csúcsok száma, m az élek száma. Ebben a fejezetben vizsgálunk egy erősen polinomiális futásidejű duál hálózati szimplex algoritmust a minimális költségű folyamfeladat megoldására, melynek a futásideje $O(mn(m + n \log n) \log n)$. Először a duál hálózati szimplex algoritmussal (3.2. eljárás) foglalkozunk, majd ehhez mutatunk egy olyan pivotálási szabályt, mellyel a fenti futásidőt kapjuk.

3.1 A duál hálózati szimplex algoritmus

Feltehető, hogy a hálózatban nincsenek párhuzamos élek. A párhuzamos élek megszüntetésére olvasható egy technika James B. Orlin, Éva Tardos és Stanley A. Plotkin [7] cikkében, melynek alapja az élek felbontása új csúcsok hozzávételével. Legyen a csúcsok halmaza $N = \{1, 2, \ldots n\}$. Jelölje G^a azt a G-ből kapott segédgráfot, melynek csúcshalmaza $N^a = N \cup \{0\}$, élhalmaza $A^a = A \cup \{(0, v) \colon v \in N\}$. Legyen $b_0 = 0$, $u_{0v} = +\infty$, $c_{0v} = 0$ minden $v \in N$ csúcsra. Azt mondjuk, hogy az $f \colon A \mapsto \mathbb{R}_0^+$ egy **előfolyam**, ha kielégíti a $0 \leq f_{vw} \leq u_{vw}$ feltételeket minden $(v, w) \in A$ élre. Egy adott f előfolyam esetén a v csúcsban az **eltérést** a következőképpen definiáljuk:

$$e_v^f = b_v + \sum_{(w,v)\in A} f_{wv} - \sum_{(w,v)\in A} f_{vw}.$$

Azt mondjuk, hogy a v csúcs **hiányos**, ha $e_v^f < 0$, illetve **többletes**, ha $e_v^f > 0$. Vegyük a korábban definiált potenciálfüggvényt, illetve az ebből és a

költségfüggvényből kapott redukált költségfüggvényt. Azt mondjuk, hogy a p potenciálfüggvény **duál megengedett**, ha $c^p_{vw} \geq 0$ minden (v, w) élre, ahol $0 \leq f_{vw} < u_{vw}$ és $c^p_{vw} \leq 0$ minden (v, w) élre, ahol $f_{vw} = u_{vw}$. Az **optimalitási feltételek** a következők: ha $c^p_{vw} > 0$, akkor $f_{vw} = 0$, illetve ha $c^p_{vw} < 0$, akkor $f_{vw} = u_{vw}$.

A minimális költségű folyamfeladatra a felső korlátos szimplex algoritmushoz hasonlóan egy bázisstruktúrát az éleknek egy (T, L, U) particionálásával adhatunk meg az előző fejezetben definiált módon. Legyen T egy feszítőfa a G^a segédgráfban, legyen a 0 csúcs a lerögzített gyökér. Azt mondjuk, hogy egy él **felfelé irányított él**, ha a gyökér felé irányított, azaz a feszítőfabeli úton a végpontja közelebb van a gyökérhez, mint a kezdőpontja. Hasonlóan beszélhetünk **lefelé irányított élekről** is. Minden bázison kívüli (v, w) élhez L és U partíciók alapján legyen $x_{vw} = 0$, vagy $x_{vw} = u_{vw}$. Miután a bázison kívüli élekre beállítottuk a folyamértéket, a bázisbeli élekhez tartozó folyamérték egyértelműen kiszámolható. Ha rögzítjük a p_0 potenciált 0-ra, akkor a többi csúcshoz tartozó potenciál is egyértelműen kiszámolható, ha megköveteljük, hogy minden $(v, w) \in T$ élre $c^p_{vw} = 0$ teljesüljön. Azt mondjuk, hogy a T **fa duál megengedett**, ha az ilyen módon kiszámolt potenciálfüggvény duál megengedett és teljesíti az optimalitási feltételeket. Egy **bázisstruktúra duál megengedett**, ha a struktúrához tartozó fa duál megengedett.

A duál hálózati szimplex algoritmus (3.2. eljárás) során minden lépésben fenntartunk egy duál megengedett bázisstruktúrát és minden pivotálás után a feszítőfa duál megengedett marad. Mindig bázisbeli éleken keresztül küldünk folyamot a csúcsok között.

3.1.1. Definíció. *Azt mondjuk, hogy a duál megengedett T fa erősen duál megengedett, ha $f_{vw} > 0$ minden felfelé irányított élre és $f_{vw} < u_{vw}$ minden lefelé irányított élre.*

Ez a definíció elvivalens azzal, hogy a gyökérből a levelekbe mindig tudunk pozitív mennyiségű folyamot küldeni.

Azt mondjuk, hogy a w csúcs a v csúcsnak az **őse**, ha a w csúcs eleme v csúcsot a gyökérrel összekötő T-beli útnak. Hasonló feltételekkel a v csúcs a w csúcsnak a **leszármazottja**. Jelölje $\mathrm{pred}(v)$ a v csúcs közvetlen ősét a T fában. A kezdeti erősen duál megengedett T fa előállítható úgy, hogy a 0 csúcsot választjuk a gyökérnek és minden $v \in N$ csúcsra legyen $\mathrm{pred}(v) = 0$.

Legyen a v csúcsra $e_v^f > 0$ és $w = \text{pred}(v)$. Az algoritmus során $\min\{e_v^f, f_{wv}\}$ értékű folyamot küldünk a v csúcsból a w csúcsba, ha a (w, v) egy lefelé irányított él, vagy $\min\{e_v^f, u_{vw} - f_{vw}\}$ értékű folyamot, ha a (v, w) egy felfelé irányított él. Minden lépésben minden $v \in N$ csúcsra fenntartunk egy nemnegatív eltérést, a gyökérben pedig addig lesz hiány, ameddig nem kapunk egy primál megengedett megoldást, azaz nem érünk egy optimumba.

Nézzük meg hogyan választunk bázisból kilépő élt. Egy lefelé irányított T-beli (g, h) él, $e_h^f > 0$ eltéréssel és $f_{gh} = 0$ előfolyammal, vagy egy felfelé irányított T-beli (g, h) él, $e_g^f > 0$ eltéréssel és $f_{gh} = u_{gh}$ előfolyammal lesz a kilépő él. Könnyen látható, hogy az első esetben $x_{gh} < 0$, míg a második esetben $x_{gh} > u_{gh}$.

Most nézzük meg hogyan választunk belépő élt minden kilépő (g, h) élhez. A (g, h) él eliminálását követően a T fa két részre esik szét, jelölje T^R a gyökeret tartalmazó részt, $T^N = T \setminus T^R$. A T^N-ből folyamot kell küldenünk a T^R-be, hogy T^N-ben csökkenjen a többletek összege. Legyen

$$\theta = \min\{c_{vw}^p \colon f_{vw} = 0, v \in T^N, w \in T^R; -c_{vw}^p \colon f_{vw} = u_{vw}, v \in T^R, w \in T^N\}. \quad (3.1)$$

Az olyan (p, q) élek közül kerül ki a belépő él, melyeken a (3.1) minimum felvétetik, így a báziscsere után ismét egy duál megengedett bázisstruktúrát kapunk.

A belépő él kiválasztása után frissítjük a potenciálfüggvényt, hogy duál megengedett maradjon. Minden $v \in T^N$ csúcsra $p_v \leftarrow p_v + \theta$. A fa frissítésekor töröljük a kilépő élt a fából és hozzáadjuk a belépő élt.

Végezzünk mélységi bejárást. A bejárás során minden csúcshoz rendeljünk egy **mélységi számot**, amely azt fejezi ki, hogy hányadikként értük el az adott csúcsot. A frissítések után folyamot küldünk a fa élein a csúcsokból a mélységi számok szerinti csökkenő sorrendben. Ez azt jelenti, hogy a levelekből indulva küldünk folyamot a gyökér felé, formális leírása a folyam küldés (3.1. eljárás), melynek futásideje $O(n)$. Ennek minden végrehajtását követően vagy egy optimális (primál megengedett) folyamot kapunk, vagy megtaláljuk a következő kilépő élt, továbbá annyi folyamot küldünk a levelekből a gyökér irányába, amennyi lehetséges úgy, hogy f előfolyam maradjon.

3.1.2. Lemma. *A duál hálózati szimplex algoritmus (3.2. eljárás) minden lépésben fenntartunk egy erősen duál megengedett fát.*

```
1: procedure FOLYAM_KÜLDÉS(T)
2:     for v ∈ N a mélységi számok szerinti csökkenő sorrendben do
3:         w ← pred(v)
4:         if (w, v) lefelé irányított él then
5:             δ ← min{e_v^f, f_wv}
6:             δ értékű folyam küldése v csúcsból w csúcsba
7:         else if (v, w) felfelé irányított él then
8:             δ ← min{e_v^f, u_vw − f_vw}
9:             δ értékű folyam küldése v csúcsból w csúcsba
10:        end if
11:    end for
12: end procedure
```

3.1. eljárás. folyam küldés

Bizonyítás. A lemmát indukcióval látjuk be. A kezdeti fa erősen duál megengedett, hiszen minden éle lefelé irányított és kapacitásuk $+\infty$. Tegyük fel, hogy a lemma igaz néhány lépést követően. Most nézzünk meg egy pivotálást, amikor a (g, h) él a bázisból kilépő él, a (p, q) él pedig a bázisba belépő él. Tegyük fel, hogy $g \in T^N$ és $p \in T^N$. A bizonyítás hasonlóan történik akkor is, ha $h \in T^N$ és $q \in T^N$. T frissítésekor a p csúcsból a gyökér felé vezető út, p csúcsot g csúccsal összekötő szakaszán, a felfelé irányított élek lefelé irányítottá válnak, míg a lefelé irányított élek felfelé irányítottá válnak. Mivel T egy erősen duál megengedett fa, ezért pozitív mennyiségű folyamot küldünk T^N-ből a gyökér felé a fa frissítését követően. A frissítést követően a g csúcsból a p csúcsba tudunk küldeni pozitív mennyiségű folyamot. Miután küldtünk folyamot, az út felfelé irányított éleihez tartozó előfolyam értékei pozitívak, míg a lefelé irányított éleihez tartozó előfolyam értékei szigorúan kisebbek, mint a kapacitásuk. Ha a q csúcsból küldünk előfolyamot a gyökér felé, az a felfelé irányított éleken növeli a folyamértéket, a lefelé irányított éleken pedig csökkenti, így nem rontja el az erősen duál megengedett tulajdonságot, hiszen a gyökérből továbbra is küldhető mindegyik levélbe pozitív mennyiségű folyam. □

3.1.3. *Megjegyzés.* A 3.1.2. lemma következménye, hogy mindig küldhető T^N-ből T^R-be pozitív mennyiségű folyam.

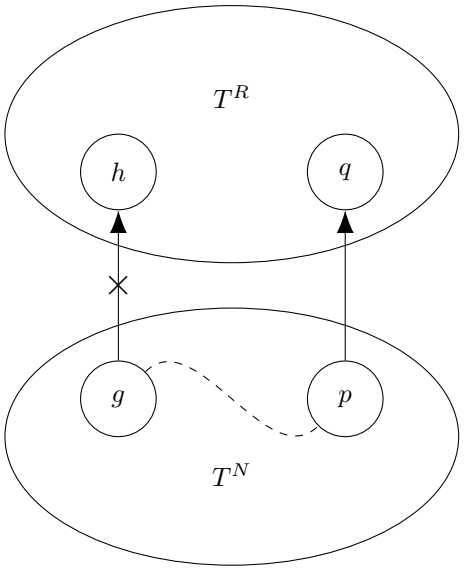
Legyen $B = \sum_{b_v:\, b_v>0} b_v$. A 3.1.3. megjegyzést felhasználva most megmutatjuk, hogy ha az igényfüggvény és a kapacitásfüggvény is egészértékű, akkor az algoritmus futásideje $O((m + n \log n)B)$.

Legyen $L = \sum_{v \in N} f_{0v}$. Ekkor $L \leq B$. A duál hálózati szimplex algoritmus (3.2. eljárás) 4. lépésében, ha nem létezik kilépő (g, h) él, akkor L csökken, mivel pozitív

```
1: procedure DUÁL_HÁLÓZATI_SZIMPLEX()
2:     T = {(0, v) : v ∈ N}                                  ▷ 0. lépés: Inicializáció
3:     for v ∈ N do
4:         p_v ← 0
5:         if b_v ≤ 0 then
6:             f_0v ← −b_v
7:         else if b_v > 0 then
8:             f_0v ← 0
9:         end if
10:    end for
11:    if nincs kilépő (g, h) él a T fában then          ▷ 1. lépés: Kilépő él választása
12:        return a (T, L, U) bázisstruktúra optimális
13:    else
14:        legyen (g, h) a kilépő él
15:    end if
16:    legyen θ a 3.1-ben definiált minimum              ▷ 2. lépés: Belépő él választása
17:    if nincs (p, q) él, melyen a θ minimum felvétetik then
18:        return a feladat nem megoldható
19:    else
20:        legyen (p, q) él, melyen a (θ) minimum felvétetik
21:    end if
22:    for v ∈ N do                                      ▷ 3. lépés: Frissítés
23:        p_v ← p_v + θ
24:    end for
25:    T ← (T − (g, h)) ∪ (p, q)
26:    folyam_küldés(T)                                  ▷ 3.1. eljárás
27:    if p ∈ T^N then                                   ▷ 4. lépés: Következő pivotálás
28:        keressünk kilépő élt a q csúcsból a gyökér felé haladva
29:    else if q ∈ T^N then
30:        keressünk kilépő élt a p csúcsból a gyökér felé haladva
31:    end if
32:    if nincs kilépő él jelölt then
33:        go to 11
34:    else
35:        legyen (g, h) a következő kilépő él
36:        go to 16
37:    end if
38: end procedure
```

3.2. eljárás. duál hálózati szimplex

mennyiségű folyamot küldünk a gyökérbe. Ha létezik kilépő él, akkor a T^N részgráf mérete nő. Ebből következik, hogy legfeljebb n pivotálást követően L csökken. Nem szükséges addig frissítenünk T-t, ameddig L nem csökken. Közvetlenül az algoritmus 2. lépését követően keressünk a q csúcsból a gyökér felé vezető úton, ha $p \in T^N$, vagy a p csúcsból a gyökér felé vezető úton, ha $q \in T^N$ egy (g, h) lefelé irányított élt, melyre $f_{gh} = 0$, vagy egy (g, h) felfelé irányított élt, melyre $f_{gh} = u_{gh}$ Ha



3.1. ábra. egédábra a 3.1.2. lemmához

nem létezik ilyen (g, h) él, akkor L csökken, hiszen T egy erősen duál megengedett fa, és pozitív mennyiségű folyamot küldhetünk T^N-ből a gyökér felé. Ha létezik ilyen (g, h) él, akkor legyen ez a következő kilépő él és ugorjunk az algoritmus 2. lépésére. Ameddig L nem csökken, addig sem a fát nem frissítjük, sem folyamot nem küldünk csúcsok között. Ezt hívjuk **blokk pivotálásnak**, ennek ötlete D. Goldfarb [8] cikkéből származik.

3.1.4. Lemma. *Egy blokk pivotálás futásideje* $O(m + n \log n)$.

Bizonyítás. Minden pivotálás azzal kezdődik, hogy kiválasztjuk a bázisból kilépő (g, h) élt, mely szétbontja T-t a korábban definiált T^R és T^N részfákra. Minden $w \in T^R$ csúcsra legyen

$$\theta_w = \min\{c^p_{vw} \colon f_{vw} = 0, v \in T^N; -c^p_{wv} \colon f_{wv} = u_{wv}, v \in T^N\},$$

továbbá legyen $\theta = \min\{\theta_w \colon w \in T^R\}$ a minimális redukált költsége a lehetséges belépő éleknek. Legyen (g^1, h^1) a következő kilépő él, és T^{R1}, valamint T^{N1} az élhez tartozó 2 részfa. Amikor kiszámoljuk az új θ_w, $w \in T^{R1}$ értékeket, akkor csak azokat a (v, w) éleket kell vizsgálnunk, melyekre $f_{vw} = 0$ és $v \in T^{N1} \setminus T^N$, valamint azokat a (w, v) éleket, melyekre $f_{wv} = u_{wv}$ és $v \in T^{N1} \setminus T^N$. Minden c^p_{vw}, melyre $v \in T^N$ és $w \in T^{R1}$ θ értékkel csökken, valamint hasonlóan minden c^p_{vw}, melyre $w \in T^N$ és $v \in T^{R1}$ θ értékkel nő. Egy él redukált költségét legfeljebb egyszer kell kiszámolnunk egy blokk pivotálás során. Egy blokk pivotálásban a minimális redukált költséget adó él kiválasztása (például Fibonacci kupaccal) $O(n \log n)$ időben megtehető. A

fa és a potenciálfüggvény frissítése $O(n)$ időben elvégezhető, például D. Goldfarb [8] cikkében leírt módon. A fa frissítését követően meghívjuk a folyam küldést (3.1. eljárás), melynek futásideje $O(m)$. Egy blokk pivotáláson belül minden élt legfeljebb egyszer vizsgálunk, így a futásidő $O(m + n \log n)$. □

Mivel minden blokk pivotálás során pozitív mennyiségű folyamot küldünk a gyökérbe, így ha az igényfüggvény és a kapacitásfüggvény is egészértékű, akkor az algoritmus futásideje $O((m + n \log n)B)$.

3.2 A skálázós duál hálózati szimplex algoritmus

Ebben a fejezetben mutatunk a duál hálózati szimplex algoritmushoz (3.2. eljárás) egy olyan pivotálási szabályt, mellyel erősen polinomiális futásidejűvé tehető. Ehhez alkalmazunk egy James B. Orlin [9] cikkén alapuló skálázási technikát.

Hasonlóan a skálázós premultiplikátor szimplex algoritmushoz (2.5. eljárás), itt is minden lépésben fenntartunk egy pozitív Δ értéket, ami a skálázási faktor, és az algoritmust itt is Δ-skálázási fázisokra osztjuk fel. Akkor lépünk az egyik ilyen fázisból a másikba, ha a skálázási faktor megváltozik. Minden Δ-skálázási fázisban az algoritmus olyan (g, h) élt választ kilépő élnek, mely vagy lefelé irányított él $e_h^f \geq \Delta$ többlettel és $f_{gh} = 0$ előfolyammal, vagy felfelé irányított él $e_g^f \geq \Delta$ többlettel és $f_{gh} = u_{gh}$ előfolyammal. Célunk Δ egység folyamot küldeni olyan v csúcsokból a gyökér felé, melyekre $e_v^f \geq \Delta$.

Legyen $w = \text{pred}(v)$. Előfordulhat, hogy tudunk pozitív mennyiségű folyamot küldeni a v csúcsból a w csúcsba, de Δ-nál kevesebbet. Például legyen $\Delta = 5$, $e_v^f = 5$, $e_w^f = 0$, $u_{vw} = 6$, $f_{vw} = 3$. Ekkor összesen 3 egység folyam küldhető a (v, w) élen. Miután a v csúcsból a w csúcsba küldünk 3 egység folyamot, $e_w^f = 3$ lesz, így viszont nem küldhetünk a w csúcsból a $\text{pred}(w)$ csúcsba folyamot, hiszen $e_w^f < \Delta$. Ekkor a többletek összege nem változik, de a gyökértől különböző csúcsok között újraosztódnak. Később azt szeretnénk belátni, hogy egy skálázási fázisban ilyen folyam javítás csak $O(m)$ alkalommal fordulhat elő. Ehhez először vezessük be az **éltúllépés** fogalmát.

Jelölje e_{vw}^{vf} az éltúllépést a v csúcsban, amely akkor keletkezik, amikor Δ-nál kevesebb folyamot küldünk a v csúcsból a w csúcsba. Hasonlóan jelölje e_{vw}^{wf} az éltúllépést

a w csúcsban, amikor Δ-nál kevesebb folyamot küldünk a w csúcsból a v csúcsba. Az éltúllépés csak akkor változhat egy Δ-skálázási fázison belül, ha az élen folyamot küldünk. Minden éltúllépés nemnegatív, valamint a skálázási fázisok kezdettén az összegük 0, a fázisok végén pedig átalakítjuk őket eltéséressé a következő módon:

$$e_v^f \leftarrow e_v^f + e_{vw}^{vf}, \quad e_{vw}^{vf} \leftarrow 0, \text{ és}$$
$$e_w^f \leftarrow e_w^f + e_{vw}^{wf}, \quad e_{vw}^{wf} \leftarrow 0.$$

Vizsgáljuk az éltúllépés frissítést (3.3. eljárás). Tegyük fel, hogy a Δ-skálázási fázisban vagyunk, $e_v^f \geq \Delta$ és $w = \text{pred}(v)$. Nézzük meg azt az esetet, amikor folyamot küldünk a v csúcsból a w csúcsba.

```
1:  procedure ÉLTÚLLÉPÉS_FRISSÍTÉS((v, w), Δ)
2:      if (v, w) felfelé irányított él then
3:          e_v^f ← e_v^f − Δ
4:          δ ← min{u_vw − f_vw, Δ}
5:          f_vw ← f_vw + δ
6:          e_w^f ← e_w^f + δ
7:          if 0 < δ < Δ then
8:              e_vw^fv ← Δ − δ
9:              e_w^f ← e_w^f + e_vw^wf
10:             e_vw^wf ← 0
11:         end if
12:     else if (w, v) lefelé irányított él then
13:         e_v^f ← e_v^f − Δ
14:         δ ← min{f_wv, Δ}
15:         f_wv ← f_wv − δ
16:         e_w^f ← e_w^f + δ
17:         if 0 < δ < Δ then
18:             e_wv^fv ← Δ − δ
19:             e_w^f ← e_w^f + e_wv^wf
20:             e_wv^wf ← 0
21:         end if
22:     end if
23: end procedure
```

3.3. eljárás. éltúllépés frissítés

Amikor a v csúcsból a w csúcsba Δ értékű folyamot küldünk, azaz $\delta = \Delta$, akkor az éltúllépések nem változnak. Ha viszont Δ-nál kevesebb folyamot küldünk, akkor $\Delta - \delta$ lesz e_{vw}^{vf} vagy e_{wv}^{vf}. Az első esetben e_{vw}^{wf}, a második esetben e_{wv}^{wf} olvad bele a w csúcs eltérésébe és válik nullává. Továbbá $0 \leq e_{vw}^{vf} < \Delta$ és $0 \leq e_{vw}^{wf} < \Delta$ is teljesül minden (v, w) élre. Azt mondjuk, hogy a (v, w) vagy a (w, v) él **megszorítja a folyamot**, ha a w csúcs kevesebb, mint Δ egység folyamot kap folyamjavításkor,

azaz δ és a w csúcshoz tartozó éltúllépés összege kevesebb Δ-nál. Ha folyam javítást követően $e_w^f < \Delta$, akkor azt mondjuk, hogy a (v, w) vagy a (w, v) él **Δ-korlátozó él**. Ha $e_w^f \geq \Delta$, akkor megpróbálunk folyamot küldeni a w csúcsból a gyökér felé. Előrdulhat, hogy egy él megszorítja a folyamot, de nem Δ-korlátozó, mivel a w csúcsbeli eltérés kellően nagy ahhoz, hogy a folyam javítását követően $e_w^f \geq \Delta$ teljesüljön. Fordított esetben azonban minden Δ-korlátozó él megszorítja a folyamot.

3.2.1. Lemma. *Egy skálázási fázisban egy él legfeljebb kétszer szoríthatja meg a folyamot.*

Bizonyítás. Legyen a skálázási faktor Δ. Vegyünk egy tetszőleges (v, w) élt és tegyük fel, hogy ez az él akkor szorított meg először a folyamot, amikor a v csúcsból küldünk folyamot a w csúcsba. A bizonyítás hasonlóan zajlik fordított esetben is. A közvetlenül megszorítást követően $f_{vw} = u_{vw}$, $0 < e_{vw}^{vf} = \Delta - \delta < \Delta$ és $e_{vw}^{wf} = 0$.

Ez az él szoríthatja meg legközelebb a folyamot, ha a w csúcsból küldünk folyamot a v csúcsba. Közvetlenül ez előtt legyen az előfolyam f'. Ekkor $f'_{vw} < \Delta$ és $u_{vw} - f'_{vw}$ a Δ egész többszöröse, hiszen az első megszorítás után mindig Δ egység folyamot küldtünk a v és a w csúcsok között. Közvetlenül a második megszorítást követően $e_{vw}^{wf} = \Delta - f'_{vw}$, $e_{vw}^{vf} = 0$ és $f_{vw} = 0$.

Az éltúllépések egészen addig változatlanok maradnak, amíg Δ egység folyamot küldünk a két csúcs között. Nézzük meg a következő esetet, amikor Δ-nál kevesebb folyamot küldünk. Közvetlenül ezt megelőzően legyen az előfolyam f'', így $u_{vw} - f''_{vw} < \Delta$. Mivel a második megszorítást követően ezt megelőzően Δ egység folyamot küldtünk az élen, ezért $u_{vw} - f''_{vw} = f'_{vw}$. A folyam küldését követően $u_{vw} - f''_{vw} + e_{vw}^{wf} = \Delta$, azaz a w csúcs Δ egység folyamot kapott, ez nem Δ-korlátozó él. Továbbá a folyam küldést követően $e_{vw}^{vf} = \Delta - (u_{vw} - f''_{vw}) = \Delta - f'_{vw}$, $f_{vw} = u_{vw}$ és $e_{vw}^{wf} = 0$. Ezentúl ebben a skálázási fázisban $e_{vw}^{vf} = \Delta - f'_{vw}$ és $e_{vw}^{wf} = 0$, vagy $e_{vw}^{wf} = \Delta - f'_{vw}$ és $e_{vw}^{vf} = 0$. Ebből pedig következik, hogy ebben a skálázási fázisban ez az él többször nem szoríthatja meg a folyamot, ugyanis ezt követően mindig Δ egység folyamot kapnak a csúcsai. □

Mivel minden él legfeljebb kétszer szoríthatja meg a folyamot egy skálázási fázison belül, így fázisonként $O(m)$ ilyen javítás történhet. A 3.2.1. lemmából következik az is, hogy ha egy él a skálázási fázis elején az alsó, vagy a felső határára állított él – az összes bázison kívüli él ilyen –, akkor legfeljebb egyszer szoríthatja meg a folyamot.

3.2.2. Definíció. *Legyen Δ a skálázási faktor. Azt mondjuk, hogy a duál megengedett T fa Δ-erősen duál megengedett, ha $f_{vw} + e^{vf}_{vw} + e^f_v \geq \Delta$ minden $(v, w) \in T$ felfelé irányított élre és $u_{wv} - f_{wv} + e^{vf}_{wv} + e^f_v \geq \Delta$ minden $(w, v) \in T$ lefelé irányított élre.*

A skálázós duál hálózati szimplex algoritmus (3.7. eljárás) minden lépésben fenntart egy Δ-erősen duál megengedett fát. A Δ-skálázási fázisban a kilépő (g, h) él vagy egy lefelé irányított él $e^f_h \geq \Delta$ többlettel és $f_{gh} = 0$ előfolyammal, vagy egy felfelé irányított él $e^f_g \geq \Delta$ többlettel és $f_{gh} = u_{gh}$ előfolyammal. A Δ-erősen duál megengedett fa garantálja, hogy a bázisba belépő éleken küldött folyamok és az éltúllépések az összege Δ legyen.

Alkalmazzuk az élek relaxálásának technikáját. Azt mondjuk, hogy a (v, w) él **relaxált**, ha a nemnegativitási vagy a kapacitás megkötést figyelmen kívül hagyhatjuk. Az első esetben azt mondjuk, hogy az él **alsó relaxált**, míg a második esetben **felső relaxált** élről beszélünk. Akkor alsó relaxálhatjuk a (v, w) élt egy skálázási fázis kezdetén, ha f_{vw} kellően nagy ahhoz, hogy a nemnegativitás megkötés az aktuális és a későbbi fázison során biztosított legyen. Hasonlóan ehhez akkor felső relaxálhatjuk a (v, w) élt, ha $u_{vw} - f_{vw}$ kellően nagy ahhoz, hogy a kapacitási megkötés teljesüljön az aktuális és a későbbi skálázási fázisok során. Ha egy él alsó és felső relaxált is egyben, akkor csak bázison belüli él lehet és nem Δ-korlátozó él.

3.2.3. Lemma. *Minden relaxált él egy skálázási fázison belül legfeljebb egyszer lehet Δ-korlátozó.*

Bizonyítás. Legyen a skálázási faktor Δ. Nézzük meg először azt az esetet, amikor a (v, w) él felső relxált. Csak akkor lehet Δ-korlátozó él, ha a w csúcsból küldünk folyamot a v csúcsba és $f_{vw} < \Delta$. A javítást követően $f_{vw} = 0$, továbbá f_{vw} mindig Δ egész többszöröse marad. A második esetben (v, w) egy alsó relaxált él. Csak akkor lehet Δ-korlátozó él, ha a v csúcsból küldünk folyamot a w csúcsba és $u_{vw} - f_{vw} < \Delta$. A javítást követően $f_{vw} = u_{vw}$, valamint $u_{vw} - f_{vw}$ mindig Δ egész többszöröse marad. □

3.2.4. Lemma. *Legyen (v, w) egy bázison kívüli relaxált él egy skálázási fázis kezdetén. Ekkor ebben a fázisban (v, w) nem lehet Δ-korlátozó él.*

Bizonyítás. Legyen a skálázási faktor Δ. Az első esetben a (v, w) bázison kívüli él legyen alsó relaxált a skálázási fázis kezdetén. Ebből következik, hogy $f_{vw} = u_{vw}$.

Nem lehet (v, w) Δ-korlátozó él, hiszen alsó relaxált és mindig Δ egység folyamot küldünk a két végpontja között. Ha (v, w) felső relaxált, akkor ebből következik, hogy $f_{vw} = 0$. Hasonlóan az előző érveléshez, most is mindig Δ egység folyamot küldünk az élen keresztül, így nem lehet Δ-korlátozó. □

Legyen a skálázási faktor Δ. A skálázós duál hálózati szimplex algoritmus (3.7. eljárás) során a skálázási fázis kezdetén akkor állítunk át egy (v, w) élt alsó relaxálttá, ha $f_{vw} \geq 12m\Delta$, valamint akkor felső relaxálttá, ha $u_{vw} - f_{vw} \geq 12m\Delta$. A feltétel helyességének bizonyítása a 3.3.3. lemmában olvasható.

Ha a (v, w) élt a skálázási fázis elején felső relaxálttá állítottuk, akkor e_{vw}^{vf} mindig 0, valamint e_{vw}^{wf} is legfeljebb egy Δ-korlátozott folyam javítást követően 0 marad, így ezeket nem szükséges tovább tárolnunk. Hasonlóan elmondható ez alsó relaxált él esetében is az e_{vw}^{wf} és az e_{vw}^{vf} éltúllépésekről. Minden skálázási fázis kezdetén az él relaxálással (3.4. eljárás) relaxáljuk az éleket.

```
1: procedure ÉL_RELAXÁLÁS(Δ, relax)
2:     for (v, w) ∈ A do
3:         if f_vw ≥ 12mΔ    and    (v, w) nem alsó relaxált then
4:             lower_relax(v, w)
5:             relax ← 1
6:         end if
7:         if u_vw − f_vw ≥ 12mΔ    and    (v, w) nem felső relaxált then
8:             upper_relax(v, w))
9:             relax ← 1
10:        end if
11:    end for
12: end procedure
```

3.4. eljárás. él relaxálás

Legyen a skálázási faktor Δ. Azt mondjuk, hogy a (v, w) él **aktív**, ha $u_{vw} \geq \Delta/(2m)$. A nem aktív éleket **passzív élnek** hívjuk. A skálázós duál hálózati szimplex algoritmus (3.7. eljárás) során csak aktív élekkel foglalkozunk. Ahogy a skálázási faktor csökken, úgy válik minden él aktívvá. Tegyük fel, hogy a (v, w) él aktív lett az aktuális skálázási fázis kezdetén. Előfordulhat, hogy $c_{vw}^{p} < 0$ és $f_{vw} = 0$, mivel korábban passzív él volt és nem foglalkoztunk vele. Ebben az esetben küldjünk u_{vw} egység folyamot a v csúcsból a w csúcsba, hogy $f_{vw} = u_{vw}$ legyen. Ekkor teljesülnek a duál megengedettségi és az optimalitási feltételek is. Ha a v csúcs hiányos lesz emiatt, akkor küldjünk folyamot a v csúcsba, hogy megszüntessük a hiányt. Később belátjuk, hogy minden ilyen esetben megszüntethető a hiány anélkül, hogy

bármely másik csúcsban hiány keletkezne. Ennek formális leírása az él hozzáadás (3.5. eljárás).

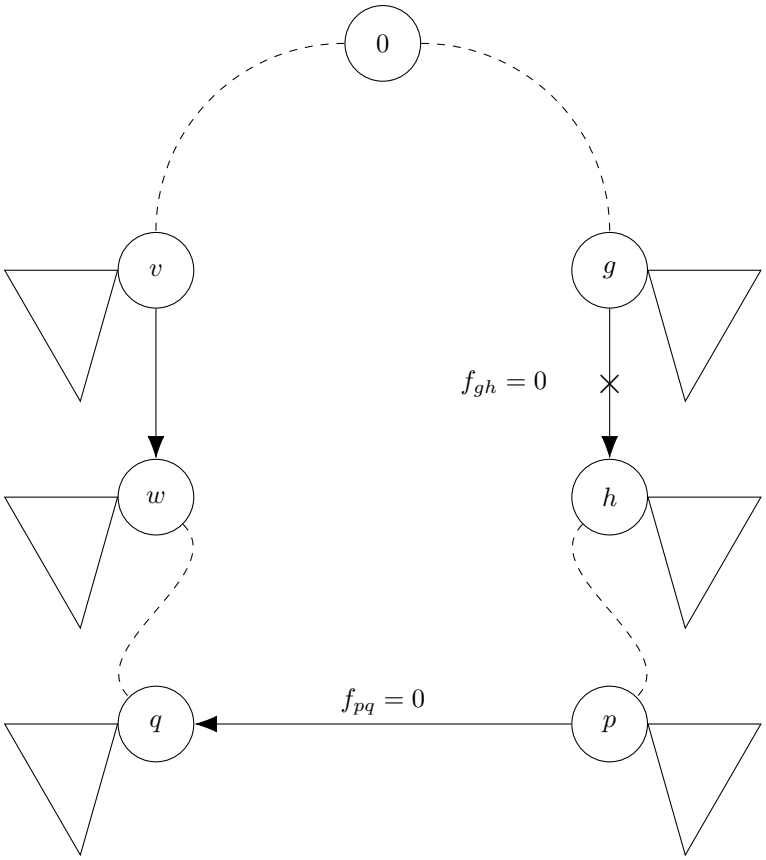
```
1: procedure ÉL_HOZZÁADÁS(T, Δ)
2:     for (v, w) ∈ A: (v, w) aktív, c^p_vw < 0, f_vw = 0 do
3:         u_vw értékű folyam küldése v csúcsból w csúcsba
4:         while e^f_v < 0 és v nem a gyökér do
5:             w ← pred(v)
6:             −e^f_v értékű folyam küldése w csúcsból v csúcsba
7:             v ← w
8:         end while
9:     end for
10: end procedure
```

3.5. eljárás. él hozzáadás

Legyen $E(\Delta) = \{(v,w)\colon (v,w) \in A$ aktív és nem relaxált él$\}$, azaz minden $(v,w) \in E(\Delta)$ élre $u_{vw} \geq \Delta/(2m)$ és $f_{vw} < 12m\Delta$, $u_{vw} - f_{vw} < 12m\Delta$ teljesül. A skálázós duál hálózati szimplex algoritmusban (3.7. eljárás) a skálázási faktort addig felezzük, ameddig legalább él nem válik felső, vagy alsó relaxálttá és $E(\Delta) = \emptyset$, majd ezt követően Δ a maximális többlet értékét veszi fel. Ha egy korábban nem relaxált él relaxálttá válik, akkor is folytatódhat úgy az algoritmus, hogy a skálázási faktor feleződik. Előfordulhat, hogy több élt relaxálunk mielőtt módosítjuk a skálázási faktort, illetve a 3.3.6. következmény azt is kimondja, hogy egy aktív él $O(\log n)$ skálázási fázison belül relaxálttá válik.

A korábban leírt blokk pivotálás alkalmazható most is. Legyen T egy Δ-erősen duál megengedett fa és legyen (g,h) a bázisból kilépő él. Akkor frissítsük a fát a blokk pivotálás során, amikor egy él Δ-korlátozott, vagy amikor Δ egység folyamot küldünk a gyökérbe. Az első esetben **részleges blokk pivotálásról**, a második esetben **teljes blokk pivotálásról** beszélünk. Most megvizsgáljuk, hogy egy pivotálásról hogyan dönthető el a végrehajtása előtt, hogy teljes blokk pivotálás, részleges blokk pivotálás, vagy nem küldhető folyam a gyökérbe, de megtaláljuk a következő kilépő élt. Legyen (p,q) él a belépő él. Mivel T egy Δ-erősen duál megengedett fa, ezért a frissítést követően legalább Δ egység folyam küldhető a (p,q) élen keresztül.

Először ellenőrizzük a (p,q) élt. Ha $p \in T^N$ és $e^f_q + e^{qf}_{pq} + u_{pq} < \Delta$, vagy $q \in T^N$ és $e^f_p + e^{pf}_{pq} + u_{pq} < \Delta$, akkor (p,q) egy Δ-korlátozó él. Ebben az esetben frissítsük a fát és a potenciálfüggvényt, a blokk pivotálás részleges. Másodszor nézzük meg azokat az éleket, amelyeken a folyamérték változna, ha Δ egység folyamot küldenénk a (p,q)



3.2. ábra. Segédábra a blokk pivotáláshoz

élen keresztül. Nézzük a q csúcsot a gyökérrel összekötő utat, ha $p \in T^N$, vagy a p csúcsot a gyökérrel összekötő utat, ha $q \in T^N$. Keressünk ezen az úton olyan (v, w) felfelé irányított él, melyre $e_w^f + e_{vw}^{wf} + u_{vw} - f_{vw} < \Delta$, vagy olyan lefelé irányított élt, melyre $e_v^f + e_{vw}^{vf} + f_{vw} < \Delta$. Ha nem találunk ilyen élt, akkor frissítsük a fát és a potenciálfüggvényt. Ez egy teljes blokk pivotálás, mivel Δ egység folyamot küldtünk a gyökérbe. Ha létezik ilyen él az úton, akkor legyen a gyökér felé haladva az első ilyen él (v, w). Vegyük észre, hogy ekkor Δ egység folyam küldhető a w csúcsba, ha (v, w) lefelé irányított él, vagy a v csúcsba, ha (v, w) felfelé irányított él, mivel ez az első él az úton, melyen Δ-korlátozott javítást végezhetünk. Ha (v, w) egy lefelé irányított él és $f_{vw} = 0$, vagy (v, w) egy felfelé irányított él és $f_{vw} = u_{vw}$, akkor ez a következő bázisból kilépő él. Ha nem, akkor részleges blokk pivotálással frissítsük a fát és a potenciálfüggvényt.

Legyen (g, h) él a blokk pivotálás során először kiválasztott bázisból kilépő él. Miután frissítettük a fát és a potenciálfüggvényt, próbáljunk meg Δ egység folyamot küldeni a g csúcsból, ha $f_{gh} = u_{gh}$, vagy a h csúcsból, ha $f_{gh} = 0$ a folyam küldéssel (3.1. eljárás).

```
1: procedure Δ-FOLYAM_KÜLDÉS(T, Δ, node)
2:     v ← node
3:     while e_v^f ≥ Δ do
4:         w ← pred(v)
5:         if (w, v) lefelé irányított él then
6:             δ ← min{f_wv, Δ}
7:             e_v^f ← e_v^f − δ
8:             f_wv ← f_wv − δ
9:             e_w^f ← e_w^f + δ
10:            if (w, v) nem relaxált and 0 < δ < Δ then
11:                e_v^f ← e_v^f − Δ + δ
12:                e_wv^vf ← Δ − δ
13:                e_w^f ← e_w^f + e_vw^wf
14:                e_vw^wf ← 0
15:            end if
16:        else if (v, w) felfelé irányított él then
17:            δ ← min{u_vw − f_vw, Δ}
18:            e_v^f ← e_v^f − δ
19:            f_vw ← f_vw + δ
20:            e_w^f ← e_w^f + δ
21:            if (v, w) nem relaxált and 0 < δ < Δ then
22:                e_v^f ← e_v^f − Δ + δ
23:                e_vw^vf ← Δ − δ
24:                e_w^f ← e_w^f + e_vw^wf
25:                e_vw^wf ← 0
26:            end if
27:        end if
28:        v ← w
29:    end while
30: end procedure
```

3.6. eljárás. Δ-folyam küldés

3.2.5. Lemma. *Legyen Δ^k a skálázási faktor a k. skálázási fázisban, Δ^{k+1} pedig a skálázási faktor a $(k+1)$. skálázási fázisban. Ekkor $\Delta^{k+1} \leq \Delta^k/2$.*

Bizonyítás. Ha Δ^{k+1} a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) 3. lépésében veszi fel az értékét, akkor $\Delta^{k+1} = \Delta^k/2$. Ha az 1. lépésében, akkor $\Delta^{k+1} \leq \Delta' = \Delta^k/2$. □

3.2.6. Lemma. *A skálázós duál hálózati szimplex algoritmus (3.7. eljárás) fenntart egy Δ-erősen duál megengedett fát a Δ-folyam küldés (3.6. eljárás) minden végrehajtását követően a 2. lépésben.*

Bizonyítás. Indukcióval látjuk be az állítást az algoritmus lépésszáma szerint. Legyen a skálázási faktor Δ. A kezdeti fában mindegyik él lefelé irányított él $+\infty$ ka-

```
1: procedure SKÁLÁZÁSI_DUÁL_HÁLÓZATI_SZIMPLEX()
2:     T = {(0, v): v ∈ N}                                   ▷ 0. lépés: Inicializáció
3:     Δ ← U                                                 ▷ U = Σ_{(v,w)∈A} u_vw
4:     for v ∈ N do
5:         p_v ← 0
6:         if b_v ≤ 0 then
7:             f_0v ← −b_v
8:         else if b_v > 0 then
9:             f_0v ← 0
10:        end if
11:    end for
12:    for (v, w) ∈ A do
13:        e^vf_vw ← 0
14:        e^wf_vw ← 0
15:    end for
16:    Δ′ ← Δ/2                                              ▷ 1. lépés: Skálázási faktor definiálása
17:    repeat
18:        folyam_küldés(T)                                  ▷ 3.1. eljárás
19:        Δ ← max{e^f_v : v ∈ N}
20:        Δ ← min{Δ, Δ′}
21:        él_hozzáadás(T, Δ)                                ▷ 3.5.eljárás
22:    until nem keletkezik aktív él az él_hozzáadás(T, Δ) hívásakor
23:    if Δ = 0 then
24:        return a (T, L, U) bázisstruktúra optimális
25:    end if
26:    relax ← 0
27:    while létezik g ∈ N csúcs, melyre e^f_g ≥ Δ do        ▷ 2. lépés: Duál pivotálás
28:        h ← pred(g)
29:        if (h, g) lefelé él, f_hg = 0 or (g, h) felfelé él, f_gh = u_gh then
30:            blokk_pivotálás()
31:            Δ-folyam_küldés(T, Δ, node)                   ▷ 3.6. eljárás
32:        end if
33:    end while
34:    for (v, w) ∈ A do                                     ▷ 3. lépés: Következő skálázási fázis
35:        e^f_v ← e^f_v + e^vf_vw
36:        e^vf_vw ← 0
37:        e^f_w ← e^f_w + e^wf_vw
38:        e^wf_vw ← 0
39:    end for
40:    if A(Δ) = ∅ and relax = 1 then go to 16
41:    end if
42:    Δ ← Δ/2
43:    él_relaxálás(Δ, relax)                                ▷ 3.4. eljárás
44:    él_hozzáadás(T, Δ)                                    ▷ 3.5. eljárás
45:    go to 27
46: end procedure
```

3.7. eljárás. skálázásós duál hálózati szimplex()

pacitással. Ebből következik, hogy a Δ-folyam küldés (3.6. eljárás) első végrehajtása előtt T egy Δ-erősen duál megengedett fa.

Tegyük fel, hogy T egy Δ-erősen duál megengedett fa az algoritmus 2. lépésének elején. A Δ-folyam küldés (3.6. eljárás) végrehajtását követően a g csúcsból a gyökérbe vezető egyértelmű úton minden felfelé irányított (v, w) élre $f_{vw}+e^{vf}_{vw}+e^f_v \geq \Delta$ és minden lefelé irányított (v, w) élre $u_{vw} - f_{vw} + e^{wf}_{vw} + e^f_w \geq \Delta$, mivel $e^f_g \geq \Delta$ az eljárás végrehajtása előtt. Nézzük a g csúcsot a gyökérrel összekötő úton kívüli éleket. A blokk pivotálást követően ezeknek az éleknek vagy megváltozott az irányítása, vagy nem. Először nézzük meg azokat az éleket, melyeknek megváltozott. Ezekre továbbra is teljesül a Δ-erősen megengedett tulajdonság, hiszen a blokk pivotálás során olyan éleket kerestünk, melyekre nem teljesül. Most nézzük meg azokat az éleket is, melyeknek nem változott az irányítása. Ezeknek az éleknek nem változott az előfolyam értéke és mivel T a frissítés előtt Δ-erősen duál megengedett volt, így az ilyen élekre továbbra is teljesül a Δ-erősen megengedett tulajdonság.

Következőnek azt mutatjuk meg, hogy ha a skálázós duál hálózati szimplex algoritmusban (3.7. eljárás) az 1., vagy 3. lépését követően ugrunk a 2. lépésre, akkor is fenntartunk egy Δ-erősen duál megengedett fát. Legyen Δ^k a skálázási faktor a 2. lépés végén a k. skálázási fázisban, Δ^{k+1} pedig a skálázási faktor a 2. lépés elején a $(k + 1)$. fázisban. A 3.2.5. lemma miatt tudjuk, hogy $\Delta^{k+1} \leq \Delta^k/2$. Ha az 1., vagy a 3. lépést követően ugrunk a 2. lépésre, akkor az él hozzáadást (3.5. eljárás) az első esetben többször, a második esetben egyszer hívhatjuk meg. Ez nem rontja el a Δ^k-erősen megengedett tulajdonságát a fának, mivel a gyökérből a csúcsokba küldött folyam összege legfeljebb $(m\Delta^k)/(2m) = \Delta^k/2$ és a fa Δ^k-erősen megengedett a k. skálázási fázisban. Ez azt is bizonyítja, hogy nem keletkezhet hiány semelyik gyökértől különböző csúcsban, ha egy él aktívvá válik.

Végül belátjuk, hogy ha a (v, w) élnek megszüntetjük az éltúllépését, akkor ez sem rontja el a Δ-erősen megengedett tulajdonságot. Mivel $f_{vw} + e^{vf}_{vw} + e^f_v$ nem változik, ha (v, w) felfelé irányított él, és $u_{vw} - f_{vw} + e^{wf}_{vw} + e^f_w$ sem változik, ha (v, w) lefelé irányított él. Ezzel igazoltuk a lemmát. □

3.2.7. Lemma. *Ha a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) 2. lépésében találtunk egy bázisból kilépő élt, akkor az aktív élek közül mindig választható hozzá belépő él, feltéve, hogy az eredeti feladat megoldató.*

Bizonyítás. Legyen a skálázási faktor Δ és legyen (v, w) a bázisból kilépő él. Mivel a passzív éleken a kapacitások összege szigorúan kisebb, mint $(m\Delta)/(2m) = \Delta/2$, így ha az eredeti feladat megoldható, akkor biztosan lesz egy olyan aktív (p, q) él, melyre $p \in T^N$, $q \in T^R$ és $f_{pq} = 0$, vagy $p \in T^R$, $q \in T^N$ és $f_{pq} = u_{pq}$. □

3.2.8. Lemma. *Amikor a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) 1. lépésről a 2. lépésre ugrunk, akkor mindig létezik bázisból kilépő él.*

Bizonyítás. Az 1. lépésben a folyam küldés (3.1. eljárás) meghívását követően vagy minden $v \in N$ csúcsra $e_v^f = 0$ és egy optimumban vagyunk, vagy létezik egy (g, h) felfelé irányított él, melyre $e_g^f \geq \Delta$ és $f_{gh} = u_{gh}$, vagy létezik egy (h, g) lefelé irányított él, melyre $e_g^f \geq \Delta$ és $f_{hg} = 0$, ahol $\Delta = \max\{e_v^f \colon v \in N\}$.

Ha az él hozzáadás (3.5. eljárás) meghívását követően egy él sem válik aktívvá, akkor a (g, h), vagy a (h, g) él lesz a kilépő él a 2. lépésben, különben az eljárást legfeljebb m alkalommal hívhatjuk meg az első lépésben, hiszen minden híváskor legalább egy él aktívvá válik. □

3.3 Futásidő vizsgálat

A futásidő vizsgálatot azzal kezdjük, hogy megmutatjuk, hogy az élek relaxálására adott feltételek helyesek. Először korlátozzuk a skálázási fázisok kezdetén a többletek összegét, majd korlátozzuk a skálázási fázisokban az egy élen keresztül küldhető folyam mennyiséget. Ezt követően korlátozzuk az egy élen keresztül küldhető folyam mennyiséget az algoritmus hátralevő részében.

3.3.1. Lemma. *Ha a skálázási faktor Δ a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) 2. lépésének kezdetén, akkor ilyenkor a többletek összege kisebb, mint $(2m + 2n + 1)\Delta$.*

Bizonyítás. Amikor a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) 3. lépéséről a 2. lépésére ugrunk, akkor $\Delta = \Delta'/2$, ahol Δ' a skálázási faktor a 2. lépés végén az előző skálázási fázisban. A 3.2.1. lemmában láttuk, hogy minden él legfeljebb kétszer szoríthatja meg a folyamot, és mivel minden $v \in N$ csúcsra $e_v^f \leq \Delta'$ a 2. lépés végén az előző iterációban, így ilyenkor a többletek összege kisebb, mint $(m + n)\Delta'$. A 3. lépésben az él hozzáadás (3.5. eljárás) nem hoz létre $\Delta'/2$-nél

nagyobb többletet, így az aktuális skálázási fázisban a többletek összege

$$(m + n + 1/2)\Delta' = (2m + 2n + 1)\Delta.$$

Amikor az 1. lépésről a 2. lépésre ugrunk, akkor vagy $\Delta = \Delta'/2$, vagy $\Delta = \max\{e_v^f \colon v \in N\}$. Az első esetet most bizonyítottuk, a második esetben pedig a többletek összege kisebb, mint $n\Delta$, amiből szintén következik az állítás. □

3.3.2. Lemma. *Legyen a skálázási faktor Δ. Ebben a skálázási fázisban az egy élen keresztül küldhető folyam mennyisége kevesebb, mint $6m\Delta$.*

Bizonyítás. Legyen a skálázási faktor Δ. Ebben a skálázási fázisban az él hozzáadás (3.5. eljárás) során az egy élen keresztül küldhető folyam mennyisége legfeljebb $\Delta/2$. A Δ-folyam küldés (3.6. eljárás) végrehajtását követően vagy Δ értékű folyamot küldünk a gyökérbe, vagy találunk egy Δ-korlátozó élt, vagy megtaláljuk a következő kilépő élt. Ez utóbbi esetet mindig egy blokk pivotálás követi, ahol vagy az többletek összegét csökkentjük Δ egységgel teljes blokk pivotálással, vagy találunk egy Δ-korlátozó élt és részleges blokk pivotálunk. A 3.3.1. lemma szerint a teljes blokk pivotálások száma legfeljebb $2m+2n+1$. A 3.2.1. és a 3.2.3. lemmák szerint minden nem relaxált él legfeljebb kétszer, míg a relaxált élek legfeljebb egyszer lehetnek Δ-korlátozó élek, így a Δ-korlátozott javítások száma kevesebb, mint $m + \hat{m}$, ahol $\hat{m}$ a nem relaxált élek száma. Ebből következik, hogy a Δ-skálázási fázisban az egy élen keresztül küldött folyam mennyisége kevesebb, mint

$$\left(\frac{1}{2} + 2m + 2n + 1 + m + \hat{m}\right)\Delta \leq 6m\Delta.$$

□

3.3.3. Lemma. *Legyen a skálázási faktor Δ. Ha a skálázási fázis kezdetén $f_{vw} \geq 12m\Delta$, vagy $u_{vw} - f_{vw} \geq 12m\Delta$, akkor az $f_{vw} > 0$, vagy az $u_{vw} - f_{vw} > 0$ feltétel teljesül minden későbbi skálázási fázisban.*

Bizonyítás. Nézzük meg azt az esetet, amikor $f_{vw} \geq 12m\Delta$. Az $u_{vw} - f_{vw} \geq 12m\Delta$ eset hasonan bizonyítható. A 3.2.4. és a 3.3.2. lemmák következménye, hogy a w csúcsból a v csúcsba küldhető folyam mennyisége az aktuális és a későbbi skálázási fázisok során kevesebb, mint

$$6m\left(1 + \frac{1}{2} + \frac{1}{4} + \dots\right)\Delta \leq 12m\Delta.$$

Ebből pedig adódik a lemma állítása. □

A következő két lemmában mutatunk egy felső korlátot arra, hogy legfeljebb hány skálázási fázisonként válik egy él relaxálttá. Ha egy él relaxálttá válik, akkor elhagyhatjuk az éltúllépését és csak akkor lehet Δ-korlátozó él, ha bázisbeli él. Legfeljebb $m+n$ alkalommal relaxálhatunk élt, hiszen ha egy él felső és alsó relaxált is, akkor csak bázisbeli él lehet.

3.3.4. Lemma. *Ha a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) 2. lépésében találunk egy kilépő élt, akkor legfeljebb $O(\log n)$ skálázási fázist követően relaxálunk egy élt.*

Bizonyítás. A skálázós duál hálózati szimplex algoritmus (3.7. eljárás) második lépésében legyen a bázisból kilépő él (g,h), továbbá legyen $h = \mathrm{pred}(g)$, $f_{gh} = u_{gh}$ és $e_g^f \geq \Delta^0$, ahol Δ^0 az aktuális skálázási faktor. Legyen T a frissítés előtti fa és $T(g)$ ennek a g gyökerű részfája a (g,h) él elhagyását követően. Ekkor a $T(g)$-beli csúcsokon a többletek összege legalább Δ^0. Legyenek a következő skálázási faktorok $\Delta^1, \Delta^2, \ldots, \Delta^L$. A Δ^L-skálázási fázis elején a csúcsokon a többletek, valamint az éltúllépések összege szigorúan kisebb, mint $(m+n)2\Delta^L$, ami azt jelenti, hogy legalább $\Delta^0 - (m+n)2\Delta^L$ egység folyamot küldtünk ki a $T(g)$ részgráfból. Ez azt jelenti, hogy legalább egy (v,w) él ment ki $T(g)$-ből $f_{vw} = 0$ előfolyam értékkel, vagy ment a $T(g)$-be $f_{vw} = u_{vw}$ előfolyam értékkel, melyre most $f_{vw} \geq (\Delta^0 - (m+n)2\Delta^L)/m$, vagy $u_{vw} - f_{vw} \geq (\Delta^0 - (m+n)2\Delta^L)/m$. Ha $L = O(\log n)$, akkor $f_{vw} \geq 12m\Delta^L$, vagy $u_{vw} - f_{vw} \geq 12m\Delta^L$, mivel $2^L\Delta^L = \Delta^0$. Az első esetben az él alsó relaxálttá válik, mivel korábban nem lehetett alsó relaxált $f_{vw} = 0$ miatt, a második esetben pedig az él felső relaxálttá válik, mivel korábban nem lehetett felső relaxált $f_{vw} = u_{vw}$ miatt. □

3.3.5. Lemma. *Ha $(v,w) \in E(\Delta)$, akkor $O(\log n)$ skálázási fázison belül a (v,w) él relaxálttá válik.*

Bizonyítás. Legyen Δ^0 a skálázási faktor, amikor $(v,w) \in E(\Delta^0)$. Ekkor $u_{vw} \geq \Delta^0/2m$. Legyenek a következő skálázási faktorok $\Delta^1, \Delta^2, \ldots, \Delta^L$. Mivel

$$f_{vw} + u_{vw} - f_{vw} = u_{vw} \geq \frac{\Delta^0}{2m} = \frac{2^L\Delta^L}{2m},$$

ezért az f_{vw} vagy az $u_{vw} - f_{vw}$ közül legalább az egyik nagyobb vagy egyenlő, mint

$$\frac{2^L \Delta^L}{4m} \geq 12m\Delta^L,$$

ha $L = O(\log m) = O(\log n)$. Így (v, w) relaxálttá válik legkésőbb a Δ^L-skálázási fázis elején. □

3.3.6. *Következmény.* A 3.3.4. és a 3.3.5. lemmákból következik, hogy $O(\log n)$ skálázási fázison belül legalább egy él relaxálttá válik.

Most megvizsgáljuk az algoritmus során végrehajtott blokk pivotálások számát.

3.3.7. Lemma. *Legfeljebb $O(nm \log n)$ részleges blokk pivotálást végezhetünk a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) során.*

Bizonyítás. Minden részleges blokk pivotáláshoz tartozik egy Δ-korlátozott folyam javítás. Válasszuk külön azokat az eseteket, amikor a javítások a skálázási fázis elején bázison belüli, vagy bázison kívüli éleken keresztül történnek. $O(\log n)$ skálázási fázison belül legalább egy él relaxálttá válik, és mivel legfeljebb $m+n$ alkalommal relaxálhatunk éleket, így a skálázási fázisok száma $O(m \log n)$. A skálázási fázisok kezdetén a bázisbeli élen keresztül történő Δ-korlátozott javítások száma $O(nm \log n)$, hiszen egy skálázási fázisban ilyen legfeljebb $O(n)$ alkalommal fordulhat elő. A 3.2.4. lemma kimondja, hogy bázison kívüli élek nem lehetnek Δ-korlátozó élek, továbbá a 3.3.5. lemma miatt egy aktív él legfeljebb $O(\log n)$ skálázási fázison keresztül nem lehet relaxált. Ebből következik, hogy a skálázási fázisok kezdetén bázison kívüli éleken keresztül történő Δ-korlátozott javítások száma $O(m \log n)$, ebből pedig már következik a lemma állítása. □

3.3.8. Lemma. *Legfeljebb $O(nm \log n)$ teljes blokk pivotálást végezhetünk a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) során.*

Bizonyítás. Minden teljes blokk pivotálással Δ egység többletet szüntetünk meg a hálózatban, így a Δ-skálázási fázisban nem lehet több a teljes blokk pivotálások száma, mint ahányszor Δ egység többlet van a csúcsokban összesen a fázis kezdetén. Vizsgáljuk külön az előző skálázási fázis után a csúcsokból származó többleteket és az éltúllépésekből kapott többleteket. Összesen $O(m \log n)$ skálázási fázis van, így az algoritmus során a korábbi fázisokból összesen $O(nm \log n)$ Δ egységnyi többlet származhat csúcsokból. Egy relaxált élnek nincs éltúllépése és a 3.3.5. lemma miatt

egy aktív él $O(\log n)$ skálázási fázison keresztül maradhat nem relaxált. Ebből következik, hogy az algoritmus során az éltúllépésekből kapott többlet összesen $O(m \log n)$ Δ egység, ezzel pedig igazoltuk $\square$

3.3.9. Tétel. *A skálázós duál hálózati szimplex algoritmus (3.7. eljárás) futásideje* $O(mn(m + n \log n) \log n)$.

Bizonyítás. A 3.3.7. és 3.3.8. lemma során beláttuk, hogy a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) összesen $O(nm \log n)$ blokk pivotálást végez, valamit a 3.1.4. lemma miatt tudjuk, hogy egy blokk pivotálás $O(m + n \log n)$ időben elvégezhető. Ebből következik, hogy az algoritmus futásideje $O(mn(m + n \log n) \log n)$, ami egyben azt is igazolja, hogy a skálázós duál hálózati szimplex algoritmus (3.7. eljárás) erősen polinomiális futásidejű. $\square$

Összegzés

Megvizsgáltuk a minimális költségű folyamfeladatot és bevezettük a hozzá kapcsolódó fogalmakat. Átismételtük a lineáris programozás néhány alapvető tételét, a primál szimplex algoritmust, a duál szimplex algoritmust és a felsőkorlátos szimplex algoritmust, valamint a polinomiális és az erősen polinomiális futásidőt.

A második fejezetben James B. Orlin [1]. cikke alapján megismertük a primál hálózati szimplex algoritmust, és mutattunk ehhez egy olyan pivotálási szabályt, mellyel $O(\min\{n^2 m \log nC, n^2 m^2 \log n\})$ futásidejű lett.

A harmadik fejezetben Ronald D. Armstrong és Zhiying Jin [2]. cikke alapján megismertük a duál hálózati szimplex algoritmust, és mutattunk egy olyan pivotálási szabályt, mellyel $O(mn(m + n \log n) \log n)$ futásidejű lett.

A. függelék

Irodalomjegyzék

[1] James B. Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming*, 78:109–129, 1997.

[2] Ronald D. Armstrong and Zhiying Jin. A new strongly polynomial dual network simplex algorithm. *Mathematical Programming*, 78:131–148, 1997.

[3] Peter van Emde Boas. *Machine models and simulations*. University of Amsterdam, Department of Mathematics and Computer Science, 1989.

[4] James B. Orlin. On the simplex algorithm for networks and generalised networks. *Mathematical Programming*, 24:7–10, 1985.

[5] Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley Series in Discrete Mathematics & Optimization, 1998.

[6] Éva Tardos. A strongly polynomial minimum cost circulation algorithm. *Combinatorica*, 5:247–255, 1985.

[7] James B. Orlin, Éva Tardos, and Stanley A. Plotkin. Polynomial dual network simplex algorithms. *Mathematical Programming*, 60:255–276, 1993.

[8] D. Goldfarb. Efficient dual simplex algorithms for the assignment problem. *Mathematical Programming*, 33:187–203, 1985.

[9] James B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations Research*, 41:338–350, 1993.