Eötvös Loránd Tudományegyetem

Természettudományi Kar

Jánó Róbert

SZEMÉLYZETÜTEMEZÉS TÖBBTERMÉKES FOLYAMOKKAL

Szakdolgozat

Matematika BSc

Témavezető:
Dr. Jordán Tibor
Operációkutatási Tanszék

Budapest, 2025

Köszönetnyilvánítás

Mindenekelőtt szeretném őszinte hálámat kifejezni *Dr. Jordán Tibornak*, aki témavezetőként a szakdolgozatom megírásának teljes folyamata során rendkívül sokat segített. Köszönöm, hogy elvállalta a témavezetést, valamint hogy türelemmel, hasznos tanácsokkal és kiváló meglátásokkal támogatott a téma kiválasztásától egészen a dolgozat véglegesítéséig. Tartalmi és technikai észrevételeivel nagyban hozzájárult ahhoz, hogy a dolgozat világosabbá és olvashatóbbá váljon.

Hálás vagyok a családomnak és barátaimnak a biztatásért és támogatásért. Külön köszönettel tartozom *Szabó Máténak* a dolgozat külalakjára vonatkozó ötletekért, valamint *Ferencz Barbarának* a folyamatos ösztönzésért és a dolgozat érthetőségét javító visszajelzésekért.

Végül köszönöm egykori munkahelyem menedzserének, hogy rendelkezésemre bocsátotta a gyakorlati alkalmazáshoz szükséges adatkészletet.

Tartalomjegyzék

Bevezetés

A személyzetütemezés, mint matematikai probléma azáltal válik lényegessé, hogy egyfelől a munkáltató vállalatok vagy szervezetek folyamatos és komplex munkaerő lefedettséget igényelnek; másfelől a munkaerőt képező alkalmazottak változó, nem feltétlenül a munkáltató számára optimális órarendben és preferenciákkal tudnak és akarnak dolgozni. A munkaerő egy, mindkét fél számára megfelelő beosztásba ütemezése elengedhetetlen úgy a munkavégzés hatékonyságának-, mint az alkalmazottak egészségének és elégedettségének szempontjából.

Egy személyzetütemezési feladatot különböző szektorokban eltérő kihívások érintenek. Közlekedés és szállítás területén egyes műszakokhoz térbeli- illetve időbeli korlátok társulnak. Kereskedelemben és ügyfélszolgálatban a kereslet előrejelzése, és az előrejelzett kereslethez szükséges erőforrások lefedése a fő szempont. Egészségügyben és katasztrófavédelemben a folyamatos lefedettség biztosítása kiemelt fontosságú.

Mieke Defraeye és *Inneke Van Nieuwenhuyse* cikkének **[1]** bevezetéséből kiindulva, a személyzetütemezés 4 fő lépésének a következőket tekintjük:

- (i) kereslet előrejelzése empirikus adatok alapján;
- (ii) a szükséges munkaerő mennyiségének és összetételének meghatározása;
- (iii) a műszakok kialakítása jogszabályok és belső szabályok figyelembevételével;
- (iv) az alkalmazottak, mint személyek műszakokba osztása.

A dolgozat folyamán a személyzetütemezés lépéseinek listáját további két elemmel bővítjük:

- (v) rövid távú újraütemezés váratlan kiesések esetén;
- (vi) feladatok és szünetek ütemezése alkalmazottakhoz.

Ezen dolgozat keretein belül a fenti lista **(iii)-(vi)** lépéseivel fogunk foglalkozni. Ezen lépések modellezésének lehetőségeiről részletes áttekintést nyújt *A.T. Ernst, H. Jiang, M. Krishnamoorthy* és *D. Sier* cikke **[2]**; mely a jelen dolgozat témájának kigondolásában is hasznos olvasmánynak bizonyult.

A személyzetütemezési problémák során számos egyéni és összesített korlátozással kell számolni; egyfelől az egyes alkalmazottak munkakörülményeire-, másfelől az összesített munkavégzési elvárásokra vonatkozóan. E komplex követelményrendszer modellezésére kiválóan alkalmasnak bizonyultak a többtermékes folyamokat vizsgáló optimalizálási modellek, amelyek képesek kezelni mind az egyéni-, mind az összesített kikötéseket egy közös hálózati struktúrában. Bár a többtermékes folyamok vizsgálata a dolgozat központi tárgya; a hagyományos folyamok, illetve a speciális struktúrájú többutas folyamok alkalmazását is vizsgáljuk.

A dolgozat első fejezete a dolgozat megértéséhez szükséges elméleti alapokat ismerteti, a lineáris programozás és a hagyományos folyamfeladatok bemutatásával. A második fejezet egynapos személyzetütemezési problémákat vizsgál hagyományos folyamok alkalmazásával. A harmadik és negyedik fejezet a kéttermékes, majd a többtermékes folyamok megoldásközpontú elméletét mutatja be; utóbbi esetében kitérve a modell gyakorlati alkalmazására is többnapos személyzetütemezési problémák környezetében. Az ötödik fejezet a többutas folyamok elméleti hátterét tárgyalja, majd egy ütemezési problémán keresztül mutatja be azok alkalmazását. A dolgozat utolsó fejezetében a megismert folyammodellek implementációját segítő eszközök áttekintője-; a többtermékes folyamok alkalmazásának egy implementált példájának bemutatása-; és a vizsgált modellek fejlesztési lehetőségeinek tárgyalása olvasható.

1. fejezet

Szükséges előismeretek

Ezen első fejezetben felelevenítünk két, a dolgozat későbbi részeinek megértését megalapozó témakört. Először is szó lesz röviden a lineáris programozásról, mint optimalizációs feladatról; bevezetjük a lineáris programozás legfontosabb fogalmait; vázlatosan bemutatjuk legjelentősebb megoldási módszerét, a szimplex-módszert; és röviden tárgyaljuk hasznos tulajdonságát, a dualitást. Másodszor, szó lesz három hagyományos gráf- és folyamfeladat definiálásáról és általános megoldási módszeréről; ezek lesznek a minimális költségű utak-, a minimális költségű folyam- és a maximális nagyságú folyam problémája.

Ezen fejezet teljes mértékben *Ravindra K. Ahuja, Thomas L. Magnanti* és *James B. Orlin* könyve **[3]** alapján készült.

1.1. Lineáris programozás

1.1.1. Bevezetés

Lineáris programnak vagy *LP* (Linear Programming) feladatnak nevezzük azokat az optimalizálási problémákat, amelyek rendelkeznek egy lineáris célfüggvénnyel, egy, az ismeretlen változókra értelmezett, lineáris kikötések halmazával, illetve az ismeretlen változók nem szigorú pozitivitását biztosító kikötések halmazával. q darab változó és p darab lineáris kikötés mellett, egy lineáris program *standard* formája a következő alakban írható fel:

$$\text{Minimalizáljuk} \qquad \sum_{j=1}^{q} c_j x_j; \tag{1.1.1a}$$

$$\text{ahol} \qquad \sum_{j=1}^{q} a_{ij} x_j = b_i, \quad \forall i = 1, \ldots, p \tag{1.1.1b}$$

$$x_j \geq 0, \quad \forall j = 1, \ldots, p. \tag{1.1.1c}$$

A probléma mátrixos alakban is felírható:

Minimalizáljuk	$cx;$	(1.1.2a)
ahol	$\mathcal{A}x = b,$	(1.1.2b)
	$x \geq 0.$	(1.1.2c)

Egészértékű lineáris programozás: Akkor beszélünk *egészértékű lineáris programozásról* vagy *ILP* (Integer Linear Programming) feladatról, amikor az x ismeretlen változóktól nemnegativitásuk mellett egészértékűségüket is elvárjuk. Az ILP feladatok óriási gyakorlati jelentőséggel bírnak, viszont általános eseteik polinomiális időben nem megoldhatóak. Egy ILP probléma optimális megoldásának polinomiális idejű közelítése lehet a probléma *LP-relaxáltjának* megoldása, ami a probléma egészértékűségi feltételének törlésével kapott LP feladat megoldását jelenti.

Feltételezhető, hogy $b_i \geq 0 \quad \forall i = 1, \ldots, p$, mivel az a_{ij} értékek negatívak is lehetnek. Minden lineáris program felírható standard alakban: $\sum_{j=1}^{q} c_j x_j$ maximalizálása egyenértékű $-\sum_{j=1}^{q} c_j x_j$ minimalizálásával; egy $\sum_{j=1}^{q} a_{ij} x_j \leq b_i$ alakú kikötés 0 költségű y_i feleslegváltozók a modellhez vételével $\sum_{j=1}^{q} a_{ij} x_j + y_i = b_i$ alakú kikötésekké tehetőek, az optimalizálandó célfüggvény értékét nem változtatva. Standard alakban a kikötések halmaza egy egyenletrendszert alkot, aminek megengedett megoldásainak halmaza változatlan marad, a rendszer sorain végzett lineáris kombinációs műveletek során.

Azt mondjuk, hogy egy lineáris program *kanonikus* alakban van, ha:

- kikötésenként egy változó izolálva van +1-es együtthatóval;
- minden izolált változó sem más kikötésekben, sem a célfüggvényben nem jelenik meg.

Minden lineáris program kanonikus alakba hozható. A kikötéseken végzett egyszerű sorműveletekkel, ahol *egyszerű sorművelet* egy sor többszörösének egy másik sorhoz adása) mindig elérhetjük a kikötésekre vonatkozó feltételek teljesülését; továbbá egy később részletezett módszerrel mindig elérhetjük a célfüggvényre vonatkozó feltételek teljesülését is. Az izolált változók lesznek a feladat *bázisváltozói*, együtt alkotják a feladat *bázisát*. A nem izolált változókat *nembázisváltozóknak* hívjuk.

Az $\mathcal{A}$ kikötésmátrix $\mathcal{A}_j$ oszlopa a lineáris kikötéseknek a j-ik változóhoz tartozó együtthatókat tartalmazza. Legyen $B = \{1, 2, \ldots, p\}$ a bázisváltozók-, $L = \{p+1, p+2, \ldots, q\}$ pedig a nembázisváltozók indexhalmaza. A lineáris program (B, L)-ként jelölt bázistruktúrája szerint definiáljuk a $\mathcal{B} = [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_p]$ és $\mathcal{L} = [\mathcal{A}_{p+1}, \mathcal{A}_{p+2}, \ldots, \mathcal{A}_q]$ mátrixokat, ahol $\mathcal{B}$ a bázismátrix és $\mathcal{L}$ a nembázismátrix; továbbá particionáljuk a változókat az $x_B = \{x_i : i \in B\}$ és az $x_L = \{x_i : i \in L\}$ halmazokba. Ezen jelölésekkel a $\mathcal{A}x = b$ egyenletrendszer átírható $\mathcal{B}x_B + \mathcal{L}x_L = b$ alakba; amiből a probléma kanonikus alakja $x_B + \mathcal{B}^{-1}\mathcal{L}x_L = \mathcal{B}^{-1}b$.

Egy kanonikus alakban felírt probléma *bázismegoldásához* legyen $x_L = 0$, amiből $x_B = \mathcal{B}^{-1}b$. Akkor beszélhetünk megengedett bázismegoldásról, ha minden változó értéke nemnegatív. Egy bázisstruktúra akkor *megengedett*, ha a struktúrához tartozó bázismegoldás megengedett. Mivel a bázismegoldás létezésének szükséges feltétele a $\mathcal{B}$ bázismátrix invertálhatósága, amihez a bázisvektoroknak lineáris függetlensége szükséges; ezek után *bázisnak* csak a lineárisan független, p elemszámú vektorhalmazokat hívjuk.

Legyen $\bar{\mathcal{A}}_j = \mathcal{B}^{-1}\mathcal{A}_j$ az $\mathcal{A}_j$ vektor $\mathcal{B}$ bázis szerinti *képviselője*, mivel igaz rá, hogy $\mathcal{B}\bar{\mathcal{A}}_j = \mathcal{A}_j$. Hasonlóan legyen $\bar{\mathcal{A}}_L = \mathcal{B}^{-1}\mathcal{L}$ és $\bar{b} = \mathcal{B}^{-1}b$ a nembázismátrix és a kikötés-egyenletrendszer

jobboldalának képviselője; legyen továbbá $c_B = (c_1, c_2, \ldots, c_p)$ és $c_L = (c_{p+1}, c_{p+2}, \ldots, c_q)$ a bázis- és nembázisváltozók *költségvektora*.

Kanonikus alakban minden bázisváltozó együtthatója 0 a célfüggvényben. Ezt úgy érjük el, hogy az i-edik kikötést megszorozzuk egy π_i tényezővel, majd kivonjuk a célfüggvényből, így kapva a következő célfüggvényt: $\sum_{j=1}^{q} c_j x_j - \sum_{i=1}^{p} \pi_i \left[\sum_{j=1}^{q} a_{ij} x_j - b_i\right]$. Az **(1.1.1b)** alapján $-\sum_{i=1}^{p} \pi_i \left[\sum_{j=1}^{q} a_{ij} x_j - b_i\right] = 0$; tehát az ezután $z(x)$-szel jelölt célfüggvény, rögzített π vektor és $z_0 = \sum_{i=1}^{p} \pi_i b_i$ jelölésű konstans mellett a következőre alakítható:

$$z(x) = \sum_{j=1}^{p} \left[c_j - \sum_{i=1}^{p} \pi_i a_{ij}\right] x_j \quad + \quad \sum_{j=p+1}^{q} \left[c_j - \sum_{i=1}^{p} \pi_i a_{ij}\right] x_j \quad + \quad z_0. \tag{1.1.3}$$

A kanonikus alak eléréséhez π-t úgy választjuk, hogy teljesüljön:

$$c_j - \sum_{i=1}^{p} \pi_i a_{ij} = 0 \quad \forall j \in B. \tag{1.1.4}$$

Az **(1.1.4)** mátrixos felírása $\pi\mathcal{B} = c_B$. Legyen $\pi = c_B\mathcal{B}^{-1}$ a $\mathcal{B}$ bázishoz tartozó *szimplex tényező*. Definiáljuk végül az x_j változó $\mathcal{B}$ bázis szerinti *leárazott költségét*: $c_j^{\pi} = c_j - \sum_{i=1}^{p} \pi_i a_{ij}$.

1.1.2. Szimplex módszer

A *szimplex módszer*, mint lineáris program megoldó módszer, egy megengedett bázismegoldásból indul; ellenőrzi ezen megengedett bázismegoldás optimalitását; megáll, ha optimális a megoldás; amennyiben pedig nem optimális a megoldás, bázisváltást végez egy nem nagyobb költségű másik bázisra.

Kezdeti bázisstruktúra keresés

Mint előbb említettük, nem minden bázismegoldás megengedett; továbbá ismert az is, hogy egy megengedett bázis keresése szinte hasonlóan nehéz, mint az eredeti feladat. Emiatt egy segédfeladat könnyen megkapható megengedett bázisát fogjuk használni kezdeti bázisnak.

Vezessünk be minden kikötéshez egy elegendően nagy M költségű x_{q+i} segédváltozót úgy, hogy:

$$\text{Minimalizáljuk} \qquad \sum_{j=1}^{q} c_j x_j + \sum_{j=q+1}^{q+p} M x_j; \tag{1.1.5a}$$

$$\text{ahol} \qquad \sum_{j=1}^{q} (a_{ij} x_j + x_{q+i}) = b_i, \quad \forall i = 1, \ldots, p \tag{1.1.5b}$$

$$x_j \geq 0, \quad \forall j = 1, \ldots, q. \tag{1.1.5c}$$

Az elegendően nagy M költség miatt, ha a segédfeladatnak van olyan megoldása, amiben minden segédváltozó értéke 0, egy ilyen megoldás lesz az optimális. Látható, hogy pontosan akkor van az eredeti feladatnak megengedett megoldása, ha a segédfeladatnak van olyan

megoldása, amiben minden segédváltozó 0; és ebben az esetben a két feladat ekvivalens. Ez azt jelenti, hogy az eredeti feladat helyett oldhatjuk a segédfeladatot; a segédfeladatnak pedig egy megengedett bázisa a $\mathcal{B} = [\mathcal{A}_{q+1}, \mathcal{A}_{q+2}, \ldots, \mathcal{A}_{q+p},]$.

Optimalitási feltétel

Legyen (B, L) egy megengedett bázisstruktúrája a feladatnak. Ezen bázisnak megfelelő kanonikus alakban a $\mathcal{B}$ bázis szerinti szimplex tényezővel a célfüggvény:

$$z(x) = \sum_{j=p+1}^{q} c_j^{\pi} x_j + z_0. \tag{1.1.6}$$

Állítás: Egy megengedett bázismegoldás optimális, ha $c_j^{\pi} \geq 0 \quad \forall j \in L$.

Bizonyítás: Optimalitás ellenőrzésekor z_0 a jelenlegi megengedett bázismegoldás. Egy megengedett megoldásban minden x_j változó értéke nemnegatív, azaz $x_j \geq 0 \quad \forall j \in L$. Tehát, ha $c_j^{\pi} \geq 0 \quad \forall j \in L$, akkor $z(x) \geq z_0 \quad \forall$ új bázis szerinti $z(x)$ célfüggvényre, azaz z_0 optimális. □

Bázisváltás

A szimplex módszer kiválaszt egy x_s nembázis-, ún. érkező változót, amire $c_s^{\pi} < 0$; ennek értékét növeli meg θ-ra és tartja a többi nembázisváltozó értékét 0-n. Ekkor a kikötés-egyenletrendszer $x_B + \theta \bar{\mathcal{A}}_s = \bar{b}$ alakú, tehát:

$$x_i = \bar{b}_i - \theta \bar{a}_{is} \quad \forall i = 1, \ldots, p. \tag{1.1.7}$$

Ha $\bar{a}_{is} > 0$, akkor x_i szükséges nemnegativitása miatt $\theta \leq \frac{\bar{b}_i}{\bar{a}_{is}}$; így $\theta = \min_{1 \leq i \leq p}\{\frac{\bar{b}_i}{\bar{a}_{is}} : \bar{a}_{is} > 0\}$ a maximális megengedett értéke x_s-nek. Ha $\bar{a}_{is} \leq 0 \quad \forall i = 1, \ldots, p$, akkor x_s értéke végtelenül nagy lehet; és mivel $c_s^{\pi} < 0$, ezért a célfüggvény értéke végtelenül kicsi lehet, azaz nem létezik véges optimális megoldás.

Legyen x_r az ún. távozó bázisváltozó, amelyre $\theta = \frac{\bar{b}_r}{\bar{a}_{rs}}$.Ekkor **(1.1.7)** alapján $x_r = 0$. Ezután x_s lesz az r-ik bázisváltozó, x_r-ből pedig nembázisváltozó lesz. Az új bázisváltozóhalmaz ismeretében létrehozzuk a feladat új kanonikus alakját:

- az r-ik sort osztjuk $\bar{a}_{rs}$-sel, így az x_s együtthatója $+1$ lesz;
- $\forall 1 \leq i \leq p, \quad i \neq r$ sorszámú sor esetén megszorozzuk az r-ik sort $-\bar{a}_{is}$-sel és hozzáadjuk az i-edik sorhoz, így $\bar{a}_{is} = 0$ lesz;
- az r-edik sort megszorozzuk $-c_s^{\pi}$-vel és hozzáadjuk a célfüggvényhez, így x_s együtthatója a célfüggvényben is 0 lesz.

Módosított szimplex módszer

A teljes $\mathcal{A}$ kikötésmátrix frissítésénél egy gyakorlatban lényegesen költséghatékonyabb módszer a következő: ha hozzátesszük a kikötéseinkhez a képzeletbeli y változókat úgy, hogy $\mathcal{A}x + \mathcal{I}y = b, x \geq 0, y \geq 0$, ahol $\mathcal{I}$ az identitásmátrix; akkor a $x_B + \mathcal{B}^{-1}\mathcal{L}x_L + \mathcal{B}^{-1}y = \mathcal{B}^{-1}b$ alakú kanonikus alakból látszik, hogy a képzeletbeli változók együtthatói a bázisváltás után az új bázis inverzének felelnek meg. Ezek alapján tehát elegendő $\mathcal{A}$ helyett az $\mathcal{I}$ identitásmátrixon végeznünk a bázisváltást. Vegyük észre, hogy a szimplex módszer egyes lépései során a $\pi = c_B\mathcal{B}^{-1}$ szimplextényező- és a $\bar{\mathcal{A}}_s = \mathcal{B}^{-1}\mathcal{A}_s$ érkezővektor képviselőjének kiszámításához csak $\mathcal{B}^{-1}$-et kell használnunk; míg az $\mathcal{A}$ mátrixot csupán a $c^\pi = c - \pi\mathcal{A}$ leárazott költségfüggvény kiszámításához kell használnunk.

Korlátos változós szimplex módszer

Tegyük fel, hogy egy olyan lineáris problémát akarunk megoldani, ami abban különbözik az eredeti **(1.1.1)** problémától, hogy rendelkezik egy további kikötéshalmazzal: $x_j \leq u_j \quad \forall j = 1, \ldots, q$. Előzőekben egy (B, L) bázis-struktúra esetén az x_L nembázisváltozók az alsókorlátjuk értékét vették fel. Ehhez képest, legyen most (B, L, U) a jelen korlátos probléma változópartíciója; amire x_B a bázisváltozók halmaza, x_L és x_U pedig azon nembázisváltozók halmazai, amik az alsó- illetve felsőkorlátjuk értékét veszik fel. Eddigihez hasonló jelölésekkel és a nembázisváltozók megfelelő értékadásával kapjuk, hogy $x_B = \mathcal{B}^{-1}b - \mathcal{B}^{-1}\mathcal{U}u_U$.

A probléma kanonikus alakjában a célfüggvény most $z(x) = \sum_{j\in L} c_j^\pi x_j + \sum_{j\in U} c_j^\pi x_j + z_0$. Előzőekhez hasonlóan, egy megengedett bázismegoldás akkor optimális, ha $c_j^\pi \geq 0 \quad \forall j \in L$ és $c_j^\pi \leq 0 \quad \forall j \in U$. A korlátos változós szimplex módszer érkező változónak választhatja bármelyik, az adott változónak megfelelő optimalitási feltételt nem teljesítő változót. Az érkező változó θ legnagyobb megengedett értéke $\min(|u_s|, \min\{\frac{\bar{b}_i}{\bar{a}_{is}} : \forall 1 \leq i \leq p\})$; a $\theta = |u_s|$ esetben a bázisváltozók halmaza változatlan marad és csupán az x_s változó értéke változik meg egyik eredeti határértékéről a másikra; a többi esetben pedig a bázisváltozók halmaza és a probléma alakja a **Bázisváltás** részben leírtak alapján történik.

1.1.3. Lineáris program duálisa

Egy lineáris program duálisáról akkor beszélhetünk, ha a *primál* feladatot *szimmetrikus* alakjában írjuk fel:

$$\text{Minimalizáljuk} \qquad \sum_{j=1}^{q} c_j x_j; \tag{1.1.8a}$$

$$\text{ahol} \qquad \sum_{j=1}^{q} a_{ij} x_j \geq b_i, \quad \forall i = 1, \ldots, p \tag{1.1.8b}$$

$$x_j \geq 0, \quad \forall j = 1, \ldots, q. \tag{1.1.8c}$$

Egy lineáris programot úgy írunk fel szimmetrikus alakba, hogy a kikötés-egyenlőtlenségrendszer ellenkező irányú egyenlőtlenségeit szorozzuk -1-gyel és egyes egyenleteit két, egyetlen kikötésként indexelt, ellenkező irányú egyenlőtlenségként kezeljük.

A szimmetrikus alakú lineáris program *duálisa* minden **(1.1.8b)** kikötésnek egy π_i duális változót feleltet meg:

Maximalizáljuk
$$\sum_{i=1}^{p} b_i \pi_i; \tag{1.1.9a}$$

ahol
$$\sum_{i=1}^{p} a_{ij} \pi_i \leq c_j, \quad \forall i = 1, \ldots, q \tag{1.1.9b}$$
$$\pi_i \geq 0, \quad \forall j = 1, \ldots, p. \tag{1.1.9c}$$

Ha az i-edik **(1.1.8b)** kikötés egy egyenlőséget leíró egyenlőtlenségpár, akkor az i-edik **(1.1.9b)** kikötést töröljük.

Dualitás-tételek

Gyenge dualitás-tétel: Ha x és π megengedett megoldások a primál és duál problémára, akkor $\sum_{i=1}^{p} b_i \pi_i \leq \sum_{j=1}^{q} c_j x_j$.

Erős dualitás-tétel: Ha a primál és duál feladat egyikének van véges optimuma; a másiknak is van, és a két optimális célfüggvény értéke megegyezik.

Komplementer slackness optimalitási feltétel: Egy (x, π) megengedett megoldáspár optimális a primál és duál feladatra, ha rendelkezik a *komplementer slackness* tulajdonsággal:

$$\pi(i)[\sum_{j=1}^{q} a_{ij} x_j - b(i)] = 0, \quad \forall i = 1, \ldots, p \tag{1.1.10a}$$
$$x_j[c_j - \sum_{i=1}^{p} a_{ij} \pi(i)] = 0, \quad \forall j = 1, \ldots, q. \tag{1.1.10b}$$

1.2. Hagyományos folyamfeladatok

A következő problémák általános megközelítése kombinatorikus szemléletű, azonban az alábbiakban mindegyik feladat lineáris programozási formában is felírásra kerül. Ennek célja a dolgozat egységességének tartása, illetve ezen problémák, a dolgozatban később tárgyalt feladatokhoz viszonyított kisebb komplexitásának hangsúlyozása.

1.2.1. Minimális költségű utak

A feladat egy hálózatban kiszámolni egy kijelölt s csúcsból a minimális költségű utat az irányított gráf minden további csúcsába. További tárgyalásainkhoz elegendő a probléma azon esetét vizsgálnunk, ahol minden élhez rendelt költség nemnegatív és egész, s-ből pedig minden csúcs elérhető:

Minimalizáljuk $$\sum_{(i,j)\in A} c_{ij}x_{ij}; \tag{1.2.1a}$$

ahol $$\sum_{\{j:(i,j)\in A\}} x_{ij} + \sum_{\{j:(j,i)\in A\}} x_{ji} = \begin{cases} n-1, & i=s \\ -1, & \forall i \in N-\{s\} \end{cases}, \quad \forall i = 1,\ldots,p \tag{1.2.1b}$$

$$x_j \geq 0, \quad \forall (i,j) \in A. \tag{1.2.1c}$$

Dijkstra algoritmus

Az alap algoritmus minden i csúcshoz rendel egy $d(i)$ címkét, ahol kezdetben $d(s) = 0$, míg $d(i) = \infty \quad \forall i \neq s$. Minden időpillanatban $d(i)$ az i csúcsba vezető minimális költségű út jelenleg ismert legélesebb felsőbecslése. A vizsgált csúcsok halmaza kezdetben üres. Minden iterációban kiválasztjuk azt a még nem vizsgált csúcsot, amely s-hez a legközelebb található, majd ezen csúcs kimenő élei mentén frissítjük a végpontok d címkéit, ha egy olcsóbb, s-ből az él végpontjába menő, úthoz jutunk. Ezután az adott csúcs átvizsgált státuszt kap. Az algoritmus addig fut, amíg minden csúcs átvizsgálásra nem került. A futásidő $O(n^2)$, ahol n a csúcsok száma. A megoldás gyorsítható alternatív algoritmusok alkalmazásával.

1.2.2. Minimális költségű folyamok

Legyen $G = (N, A)$ egy irányított hálózat s és t kijelölt csúcsokkal, ahol s a G hálózathoz rendelt x folyam forrása, t pedig a G hálózathoz rendelt x folyam nyelője. Legyen c_{ij} költségfüggvény és u_{ij} kapacitásfüggvény minden $(i,j) \in A$ élre, feltételezve, hogy a költségfüggvény- és a kapacitásfüggvényértékek nemnegatívak. Továbbá legyen b_i kínálat-kereslet-függvény minden $i \in N$ csúcsra. Ez a b_i *kínálat-kereslet-függvényérték* a folyam s forrása esetén egy előre meghatározott b_s pozitív szám, a folyam t nyelője esetén a $b_t = -b_s$ negatív szám, a G hálózat minden egyéb $i \in N$ csúcsa esetén pedig $b_i = 0$. A feladat megkeresni az élkapacitásokat betartó, csúcs-kínálat-keresleteket kielégítő, minimális költségű megengedett folyamot. Feltételezzük, hogy létezik véges optimális megoldás, és létezik végtelen kapacitású irányított út minden két csúcs között:

Minimalizáljuk $$\sum_{(i,j)\in A} c_{ij}x_{ij}; \tag{1.2.2a}$$

ahol $$\sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i, \quad \forall i \in N \tag{1.2.2b}$$

$$0 \leq x_{ij} \leq u_{ij}, \quad \forall (i,j) \in A. \tag{1.2.2c}$$

Reziduális hálózat: Egy x folyamhoz tartozó $G(x)$ *reziduális hálózatban* minden $(i,j) \in A$ él helyett vegyük fel a (i,j) és (j,i) éleket egyaránt. Egy $G(x)$-beli (i,j) él költsége a G-beli c_{ij} érték, reziduális kapacitása pedig $r_{ij} = u_{ij} - x_{ij}$. A $G(x)$-beli (j,i) él költsége $c_{ji} = -c_{ij}$ és reziduális kapacitása $r_{ji} = x_{ij}$. $G(x)$ élei közül csak a pozitív reziduális

kapacitású éleket tartsuk meg.

Körmentesítő algoritmus

Negatív kör optimalitási feltétel: Egy x^* megengedett megoldás akkor és csakis akkor optimális, ha a megoldáshoz rendelt $G(x^*)$ reziduális hálózat *konzervatív*, azaz nem tartalmaz negatív összköltségű irányított kört.

Az algoritmus egy kezdeti megengedett megoldásból indul. Minden iterációban keres az aktuális $G(x)$ reziduális hálózatban egy negatív kört; ha nem talál, akkor optimális megoldáshoz jutott és befejeződik. Ha talál a reziduális hálózatban negatív kört, javítja a megoldást a kapott kör mentén; és a javított megoldással lép a következő iterációba. Az algoritmus futásideje $O(nm^2CU)$, ahol C és U a megfelelő kisbetűvel jelölt élfüggvényértékek véges értékeinek maximuma. A megoldás gyorsasága lényegesen javítható más algoritmusokkal.

Egészértékűség: Egészértékű adatok mellett a körmentesítő algoritmus során kapott optimális megoldás egészértékű lesz.

1.2.3. Maximális nagyságú folyamok

Legyen $G = (N, A)$ hálózat nemnegatív u_{ij} élenkénti kapacitásokkal, és legyen $U = \max\{u_{ij} : u_{ij} < \infty\}$. A feladat keresni egy maximális nagyságú megengedett folyamot G-ben a kijelölt s és t pontok között; feltéve, hogy nem létezik s és t közötti végtelen kapacitású irányított út:

$$\text{Maximalizáljuk} \qquad b_s; \tag{1.2.3a}$$

$$\text{ahol} \qquad \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = \begin{cases} b_s, & i = s \\ 0, & \forall i \in N - \{s,t\} \\ b_t = -b_s, & i = t \end{cases}, \tag{1.2.3b}$$

$$0 \le x_{ij} \le u_{ij}, \quad \forall (i,j) \in A. \tag{1.2.3c}$$

s-t vágás: Az N csúcshalmaz kettéválasztása S és $\bar{S} = N - S$ halmazokra, $[S, \bar{S}]$ jelöléssel úgy, hogy $s \in S$ és $t \in \bar{S}$. Alternatív definícióként $s - t$ vágásnak nevezhető az S és $\bar{S}$ csúcshalmazokat összekötő élek halmaza is. $[S, \bar{S}]$ vágásban $(i, j) \in A$ *előreél*, ha $i \in S$ és $j \in \bar{S}$; illetve $(i, j) \in A$ *visszaél*, ha $i \in \bar{S}$ és $j \in S$. Legyen $(S, \bar{S})$ és $(\bar{S}, S)$ a vágás előreéleinek és visszaéleinek halmaza. Legyen egy *vágás kapacitása* $u[S, \bar{S}] = \sum_{(i,j)\in(S,\bar{S})} u_{ij}$; és legyen egy *minimális vágás* egy olyan vágás, melynek kapacitása minimális G összes vágása között.

Címkézéses algoritmus

Vegyük észre, hogy minden egészértékű kapacitásokkal rendelkező hálózat lebontható egészértékű kapacitásokkal rendelkező $s-t$-utak és egészértékű kapacitásokkal rendelkező körök pozitív lineáris kombinációjára. Címkézést használva, minden iterációban s-ből szélességi keresést indítunk a reziduális hálózaton, minden csúcsnál mentve az oda küldhető legtöbb folyamot, illetve a csúcsnak a legtöbb folyam küldését biztosító útbeli ősét. Mikor elérjük t-t, visszafejtve t-től, a t-be legtöbb folyam küldését biztosító javító utat az elmentett őscsúcsok segítségével; elküldjük a maximálisan küldhető folyamot a megfelelő úton. Egy javítóút kapacitása az út élein felvett minimális reziduális kapacitás, ez lesz a javítóúton küldhető

folyam maximális mennyisége. x^* megengedett folyam akkor és csakis akkor maximális, ha a $G(x^*)$ reziduális hálózat nem tartalmaz pozitív kapacitású $s-t$ javítóutat; akkor áll meg az algoritmus, amikor megkapta ezt az optimális folyamot. Az algoritmus futásideje $O(nmU)$. A megoldás gyorsasága lényegesen javítható más algoritmusokkal.

Egészértékűség: Egészértékű adatok mellett a címkézéses algoritmus során kapott optimális megoldás egészértékű lesz.

Maximális folyam minimális vágás tétel: A maximális $s-t$ folyam nagysága megegyezik a minimális $s-t$ vágás kapacitásával.

2. fejezet

Személyzetütemezés hagyományos folyamokkal

A következő fejezetben a maximális nagyságú folyam- és a minimális költségű folyamfeladatok személyzetütemezési alkalmazásainak két példáját láthatjuk .

Ezen fejezet első alfejezete *Peter Brucker* és *Rong Qu* cikke **[4]** alapján-, második alfejezete *M. Segal* cikke **[5]** alapján készült.

2.1. Egynapos ütemezés: feladatok dolgozókhoz rendelése

Minden e alkalmazott esetén adott a műszakjának megfelelő $[V_e, W_e]$ időintervallum. Legyen n feladat, ahol $\forall j \in \{1, \ldots, n\}$ esetén adott egy p_j időtartam és egy $[R_j, D_j]$ végrehajtási időablak: $D_j - R_j \geq p_j$. Egyidőben egyetlen alkalmazott dolgozhat egy feladaton; minden feladat megszakítható és később folytatható más alkalmazott által is. Egy e alkalmazott csak a $j \in J_e \subseteq \{1, \ldots, n\}$ halmazbeli feladatokat végezheti el, ahol J_e az ennek megfelelően definiált feladathalmaz.

Minden alkalmazottnak szerződésétől függően jár adott mennyiségű és hosszúságú szünet, amelyek időablakai a munkás műszakja szerint van meghatározva. Az e alkalmazott szüneteit úgy is felfoghatjuk, mint egy-egy feladat, amit csak ő tud elvégezni; a továbbiakban tehát elégséges a munkák ütemezésével foglalkoznunk.

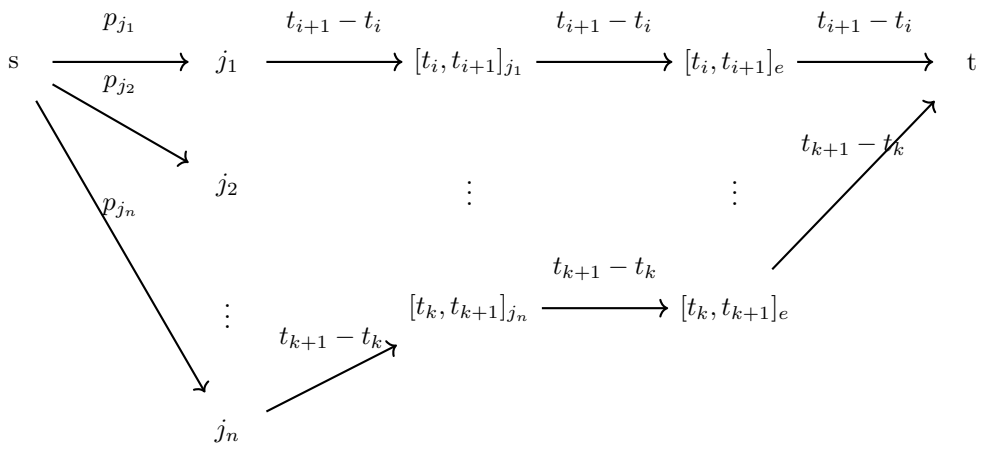
Legyen $T = \{R_j, D_j : \forall j\} \cup \{V_e, W_e : \forall e\}$; jelölje $t_1 < t_2 < \cdots < t_s$ a T elemeinek rendezett sorozatát. Tekintsük a következőképpen konstruálható $G = (N, A)$ hálózatot. Az N csúcshalmaz tartalmazza:

- a $j = 1, \ldots, n$ feladatcsúcsokat,
- a $\{[t_i, t_{i+1}]_j : [t_i, t_{i+1}] \subseteq [R_j, D_j]\}$ $\forall j$ feladat-időintervallum-csúcsokat,
- a $\{[t_i, t_{i+1}]_e : [t_i, t_{i+1}] \subseteq [V_e, W_e]\}$ $\forall e$ alkalmazott-időintervallum-csúcsokat,
- az s forrást és t nyelőt.

Az A irányított élhalmaz pedig tartalmazza:

- az (s, j) éleket, amelyek felső kapacitása p_j,
- a $(j, [t_i, t_{i+1}]_j)$ éleket, amelyek felső kapacitása $t_{i+1} - t_i$,
- a $\{([t_i, t_{i+1}]_j, [t_i, t_{i+1}]_e) : j \in Q_e\}$ $\forall e$ éleket, amelyek felső kapacitása $t_{i+1} - t_i$,
- a $([t_i, t_{i+1}]_e, t)$ éleket, amelyek felső kapacitása $t_{i+1} - t_i$.

Egy f folyam értéke a $([t_i, t_{i+1}]_j, [t_i, t_{i+1}]_e)$ éleken az e alkalmazott j feladattal töltött ideje az adott időintervallumban. A folyam megmaradási kikötések miatt egy j feladatot egyidőben



Ábra a G hálózatról.

egyetlen alkalmazott végez, összesen p_j ideig; illetve egy e alkalmazott a műszakjának minden pillanatában egyetlen feladatot végez. Akkor és csakis akkor létezik megengedett ütemezés, ha a hálózat maximális folyamának értéke $\sum_{j=1}^{n} p_j$.

Egyes feladatok megszakíthatósága kiiktatható, ha az adott feladat időablaka ugyanolyan hosszú, mint a feladat elvégzési ideje. A feladatot elvégző dolgozó személyének megváltoztathatósága továbbra is megengedett. Olyan megengedett ütemezés keresése, amelyben egy-egy feladat végrehajtása során minél kevesebb munkáscsere történik, NP-teljes probléma; ennek részletes tárgyalása azonban nem része a jelen dolgozat témájának.

Fejlesztési lehetőség: Megeshet, hogy egy feladatot különböző minőségekben el lehet végezni aszerint, hogy hány egymásra épülő részfeladatát oldjuk meg. Ütemezés szempontjából az a fontos, hogy az alapfeladat el legyen végezve, de törekednénk minél jobb minőségű összmunkára. Másképp megfogalmazva, lehetnek kötelezően elvégzendő feladatok és olyan feladatok, melyekből jó lenne minél többet megoldani, azaz minél többet hozzárendelni egy alkalmazotthoz. Ezen probléma megoldásához vegyünk egy $c : A \to \mathbb{R}_+$ költségfüggvényt:

$$c_{(ij)} = \begin{cases} p_j, & \text{ha } i = s \text{ és } j = j \text{ egy nem kötelező feladat,} \\ M, & \text{ha } i = s \text{ és } j = j \text{ egy kötelező feladat, illetve } M \text{ egy elég nagy szám} \\ 0, & \text{egyébként.} \end{cases}$$

Ezen költségfüggvénnyel a fejlesztett probléma ekvivalens egy minimális költségű folyamfeladattal a G hálózaton és a c élköltségfüggvénnyel; ahol akkor és csakis akkor van megengedett megoldás, ha a kapott folyam költsége M-nél kisebb.

2.2. Egynapos ütemezés: munkásigény lefedése szünetes műszakokkal

Legyenek különböző hosszúságú és szünetstruktúrájú műszakok, melyek szünetei az előre meghatározott időpontjaiktól 30 percnyit térhetnek el.

Probléma egészértékű programozás alapú felírása

Indexeljük egy nap negyedóráit i szerint, ahol $1 \leq i \leq 96$. Az összes lehetséges szünetes műszakot, indexeljük j szerint, $1 \leq j \leq J$. Definiáljuk az a_{ij} mátrixot:

$$a_{ij} = \begin{cases} 1, & \text{ha az } i. \text{ negyedóra dolgozó periódus a j szünetes műszakban,} \\ 0, & \text{egyébként.} \end{cases}$$

Legyen x_j a j-edik műszakhoz rendelt alkalmazottak száma, és c_j a j szünetes műszakból visszafejthető szünetmentes műszak egységköltsége. Legyen s_i az i-edik negyedórában szükséges alkalmazottak száma. A feladat ekkor a következő:

$$\text{Minimalizáljuk} \qquad \sum_{j \in J} c_j x_j; \tag{2.1.1a}$$

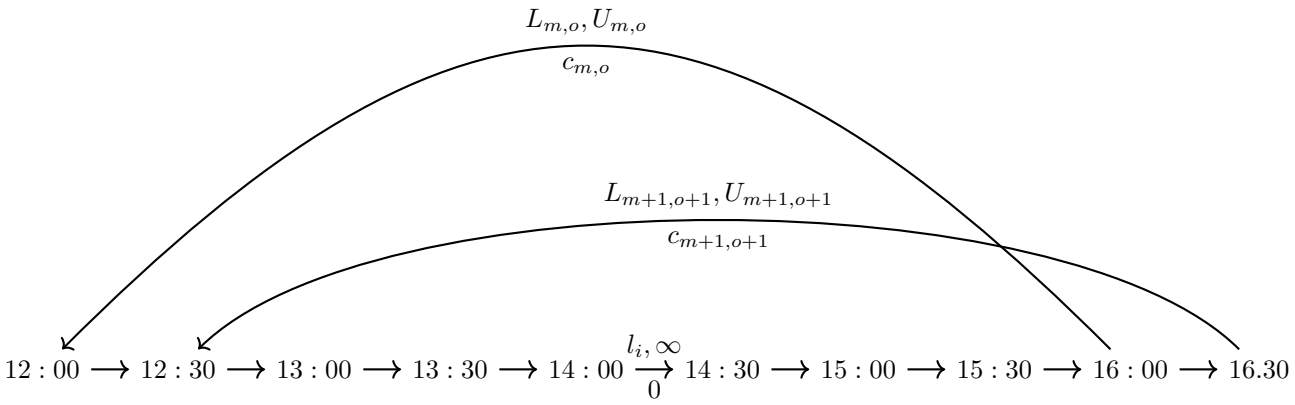
$$\text{ahol} \qquad \sum_{j \in J} a_{ij} x_j \geq l_i, \tag{2.1.1b}$$

$$x_j \geq 0 \quad \text{és} \quad x_j \in \mathbb{Z}, \quad \forall j \in J. \tag{2.1.1c}$$

Alkalmazottak szünetmentes műszakokhoz rendelése

Először a szünetmentes műszakok ütemezését oldjuk meg. Feltételezzük, hogy létezik igényt lefedő ütemezés. Ezen segédfeladathoz legyen az időegység 30 perc. Jelöljék a G gráf i pontjai az időegységeket, a i és $i+1$ közötti előreélek a megfelelő időintervallumokat, az m és o, $o > m$ közötti visszaélek pedig a szünetmentes műszakokat. Jelöljünk minden mást az előzőhöz hasonlóan.

Az előreélek kapacitásának felső korlátja legyen ∞, alsó korlátja l_i, költsége 0. A visszaélek kapacitásának felső korlátja legyen az adott szünetmentes műszakkal kompatibilis alkalmazottak száma, alsó korlátja azon dolgozók száma, akiknek kötelezően ebben a szünetmentes műszakban kell dolgozniuk; költsége pedig a szünetmentes műszak költsége.



Példa a G hálózat egy részére.

Az így kapott G hálózatban megoldott minimális költségű folyamfeladat x optimális megoldása megadja a visszaélekben az adott szünetmentes műszakokban dolgozók számát,

az előreélekben pedig egy adott időintervallumban elérhető alkalmazottak számát. Jelölje $y_i = x_{ii+1} - l_i$ a felesleg alkalmazottmennyiséget az adott időintervallumban.

Szünetek beütemezése, szünetek miatt keletkezett alkalmazotthiány mérése

Ismerve a jelenleg beütemezett szünetmentes műszakok szüneteinek optimális elhelyezését, kiszámoljuk minden időegységre a h_i munkaerőhiányt. Legyen a G hálózat, melyben egy s forrásból megy él i időegységekbe, minden i időegység össze van kötve $(i' - 1, i', i' + 1)$ csúcsokkal; és ezek mindegyike össze van kötve $t_0, t_1, \ldots, t_k$ nyelőkkel. A G hálózatban az $s - i$ élek felső- és alsó kapacitása h_i, minden más élnek pedig 0 illetve ∞; az $i' - t_f$ élek költsége f, minden más élnek pedig 0. Megoldva G-re a minimális költségű folyamfeladatot, megkapjuk a felesleg alkalmazottmennyiség optimális felhasználását, illetve a szükséges további alkalmazottak számát és ütemezését.

Megoldó algoritmus

Az első segédfeladat megoldása után kiszámítjuk a második részfeladat eredményét, és ennek alapján szükség esetén növeljük az alkalmazottak számát. Az iterációt mindaddig folytatjuk, amíg további alkalmazottakra nincs szükség.

A dolgozat folyamán ezután, többnapos ütemezés tekintetében a szünetek ütemezése nem lesz elemezve. Gyakorlatban a szünetek olyan részletek, melyeket elegendő csak az adott napon beütemezni; továbbá egyes munkanapok szünetütemezései függetlenek egymástól. Ezután mindig feltételezni fogjuk, hogy elérhető elegendő munkaerő, azaz létezik megengedett ütemezés. Ebbe a feltételbe könnyen bevehető, hogy a műszakok tartalmazzák a szüneteket is; és a szünetek figyelembe vételével is van elegendő munkaerő.

3. fejezet

Kéttermékes folyamok

Legyen $G = (N, A)$ hálózatban 2 darab termék. Mindkét *termék* rendelkezik egy forrás- és egy nyelőcsúccsal, egy saját, minden $(i, j) \in A$ élre értelmezett költség- és kapacitásfüggvénnyel; illetve az adott termékhez tartozó folyammal. Legyen egy k termék $(i, j) \in A$ élen értelmezett kapacitásfüggvény értéke $u_{ij}^k \geq 0$, költségfüggvény értéke $c_{ij}^k \geq 0$ és folyamértéke x_{ij}^k. Ezen felül, G hálózat éleihez tartozik egy összkapacitásfüggvény; mely értékeit *kötegkorlátoknak* nevezzük, és minden $(i, j) \in A$ él esetén u_{ij}-vel jelöljük. Ez utóbbi az egy élen minden termékből összesen átvihető maximális folyamnagyságot jelzi. A *folyammegmaradási kikötések* minden csúcsra előírják, hogy a beérkező és kimenő élek folyamértékeinek különbsége egyezzen meg az adott csúcs kínálat-kereslet-függvényértékével, vagy *termelési értékével.* Ez az érték egy k termék és a termékhez tartozó forrás esetén egy pozitív b^k szám, egy k termék és a termékhez tartozó nyelő esetén a negatív $-b^k$ szám, egyéb termék-csúcs kombinációk esetén pedig 0. A *minimális költségű kéttermékes folyamfeladatnak* megfelelő lineáris program a következő:

$$\text{Minimalizáljuk} \qquad \sum_{k\in\{1,2\}} \sum_{(i,j)\in A} c_{ij}^k x_{ij}^k; \tag{3.0.1a}$$

$$\text{ahol} \qquad \sum_{\{j:(i,j)\in A\}} x_{ij}^k - \sum_{\{j:(j,i)\in A\}} x_{ji}^k = \begin{cases} b^k, & i = s_k \\ 0, & \forall i \in N - \{s_k, t_k\} \\ -b^k, & i = t_k \end{cases} \quad , \quad \forall k \in \{1, 2\} \tag{3.0.1b}$$

$$0 \leq x_{ij}^1 + x_{ij}^2 \leq u_{ij}, \quad \forall (i, j) \in A. \tag{3.0.1c}$$

Az egészértékű minimális költségű kéttermékes folyamfeladat megoldása bizonyítottan NP-teljes; továbbiakban viszont láthatjuk ezen probléma két fontos speciális esetének eredményeit. A következő állításokban és kombinatorikus bizonyításokban a $c_{ij}^k = 1 \quad \forall (i, j) \in A \quad \forall k \in \{1, 2\}$ esetet tekintjük, azaz a maximális nagyságú kéttermékes folyam keresésével foglalkozunk.

Ezen fejezet teljes mértékben *Alexander Schrijver* jegyzete **[6]** alapján készült.

3.1. Hu tétele kéttermékes folyamokról

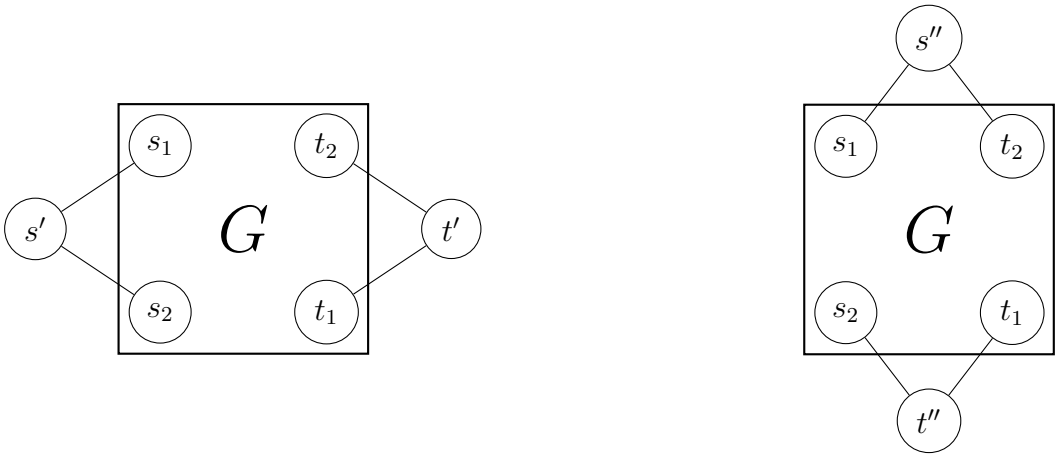
Tétel: A maximális nagyságú kéttermékes folyamfeladatnak egészértékű kapacitások és termelési értékek mellett akkor és csakis akkor létezik optimális félegész megoldása, ha

$$\forall M \subseteq N \quad \text{vágásra} \sum_{(i,j)\in\delta_A(M)} u_{ij} \geq \sum_{k\in\delta_K(M)} b^k \tag{3.1.1}$$

ahol $\delta_A(M)$ és $\delta_K(M)$ az M vágásból kimenő élek és kimenő termékek halmaza.

Bizonyítás: Legyen $f(x,v) = \sum_{(i,j)\in\delta_A(v)} x_{ij} - \sum_{(i,j)\in\delta_A^-(v)} x_{ij}$ a v csúcs termelési értéke x folyam esetén, ahol $\delta_A^-(v)$ a v csúcsba bemenő G hálózatbeli élek halmaza. Vegyük fel a következő segédcsúcsokat:

- s' legyen összekötve s_1 és s_2 csúcsokkal; t' legyen összekötve t_1 és t_2 csúcsokkal. Legyen $u_{s's_1} = b^1$ és $u_{t_1t'} = b^1$, illetve $u_{s's_2} = b^2$ és $u_{t_2t'} = b^2$.
- s'' legyen összekötve s_1 és t_2 csúcsokkal; t'' legyen összekötve s_2 és t_1 csúcsokkal. Legyen $u_{s''s_1} = b^1$ és $u_{t_1t''} = b^1$, illetve $u_{s''t_2} = b^2$ és $u_{s_2t''} = b^2$.



Tegyük fel, hogy a G hálózatra teljesül a **(3.1.1)** feltétel. Továbbá tegyük fel, hogy a kapacitásfüggvény- és termelési értékek egészek. A maximális folyam - minimális vágás tétel alapján ekkor rendre léteznek $b^1 + b^2$ nagyságú $s' - t'$ illetve $s'' - t''$ közötti x' és x'' egészértékű folyamok, amikre:

$$f(x', s_1) = b^1, \quad f(x', t_1) = -b^1, \quad f(x', s_2) = b^2, \quad f(x', t_2) = -b^2, \tag{3.1.2a}$$
$$f(x', v) = 0 \quad \forall v \in N - \{s_1, t_1, s_2, t_2\}, \tag{3.1.2b}$$
$$0 \leq x'_{ij} \leq u_{ij} \quad \forall (i,j) \in A; \tag{3.1.2c}$$

és

$$f(x'', s_1) = b^1, \quad f(x'', t_1) = -b^1, \quad f(x'', s_2) = -b^2, \quad f(x'', t_2) = b^2, \tag{3.1.3a}$$

$$f(x'', v) = 0 \quad \forall v \in N - \{s_1, t_1, s_2, t_2\}, \tag{3.1.3b}$$

$$0 \leq x''_{ij} \leq u_{ij} \quad \forall (i,j) \in A. \tag{3.1.3c}$$

Ekkor legyen $x^1 = \frac{1}{2}(x' + x'')$ és $x^2 = \frac{1}{2}(x' - x'')$ a két terméknek megfelelő folyam. Ezekre teljesülnek az $f(x^1, s_1) = b^1, \quad f(x^1, t_1) = -b^1, \quad f(x^1, v) = 0$ és $f(x^2, s_2) = b^2, \quad f(x^2, t_2) = -b^2, \quad f(x^2, v) = 0$ folyammegmaradási kikötések; továbbá x^1 és x^2 félegész megoldások. Ezen felül x^1 és x^2 megengedett megoldások a kötegkorlátok tekintetében is, hiszen minden $(i,j) \in A$ élre $x^1_{ij} + x^2_{ij} = \frac{1}{2}x'_{ij} + x''_{ij} + \frac{1}{2}x'_{ij} - x''_{ij} = \max\{x'_{ij}, x''_{ij}\} \leq u_{ij}$. □

Megoldó algoritmus: Feltételezzük, hogy a folyamfeladat adatai egészértékűek, illetve a G hálózatra teljesül a **(3.1.1)** feltétel. A fenti gondolatmenet alapján vegyük fel az s', t', s'' és t'' segédcsúcsokat és a hozzájuk tartozó éleket. Oldjuk meg G hálózatban az $s' - t'$ és $s'' - t''$ forrás-nyelő párok közötti maximális méretű folyamfeladatokat; és az ezekből kapott x' és x'' megoldások használatával készítsük el az eredeti probléma megoldásaiul szolgáló x^1 és x^2 folyamokat.

3.2. Egészértékűségi tétel Euler feltétellel

Euler-feltétel:

$$\sum_{(i,j)\in\delta_A(v)} u_{ij} \equiv \begin{cases} 0 & (\text{mod } 2) \quad v \notin \{s_1, t_1, s_2, t_2\}, \\ b^1 & (\text{mod } 2) \quad v \in \{s_1, t_1\}, \\ b^2 & (\text{mod } 2) \quad v \in \{s_2, t_2\}. \end{cases} \tag{3.2.1}$$

Tétel: Egészértékű kapacitások és termelési értékek mellett, ha a G hálózatra egyidőben teljesül a **(3.1.1)** és a **(3.2.1)** feltétel; a G hálózaton értelmezett kéttermékes folyamfeladatnak létezik egészértékű megoldása.

Bizonyítás: Tegyük fel, hogy a G hálózatra egyidőben teljesül a **(3.1.1)** és a **(3.2.1)** feltétel. Vegyük az előző bizonyításban definiált x' folyamot, amire teljesülnek a **(3.1.2)** feltételek. Legyen $A' = \{(i,j) \in A : x'_{ij} \not\equiv u_{ij} \pmod 2\}$, aminek minden v elemére az Euler-feltétel és a **(3.1.2)** feltételek mellett $f(x', v) - f(u, v) \equiv 0 \pmod 2$, tehát A' minden csúcsának A'-beli foka páros kell, hogy legyen. Így, ha eredetileg $A' \neq \emptyset$, akkor is A' irányítatlan körök uniója, amik mentén vezetve további folyamegységeket, az A'-n kívüli éleken vezetett folyammennyiségeket nem változtatva, megszüntethetjük A'-t. Hasonlóan járjunk el az előző bizonyításban definiált x'' folyammal is.

Az eddigiek alapján léteznek az x' és x'' segédfolyamok, melyekre teljesülnek a **(3.1.2)** és **(3.1.3)** feltételek; továbbá $x'_{ij} \equiv u_{ij} \pmod 2$ és $x''_{ij} \equiv u_{ij} \pmod 2 \quad \forall (i,j) \in A$. Ebből következik, hogy $x'_{ij} \equiv x''_{ij} \quad \forall (i,j) \in A$; az x' és x'' folyamokból származó $x^1 = \frac{1}{2}(x' + x'')$ és $x^2 = \frac{1}{2}(x' - x'')$ folyamok pedig egészértékűek. □

Megoldó algoritmus: Tegyük fel, hogy a folyamfeladat adatai egészértékűek, illetve a G hálózatra egyaránt teljesül a **(3.1.1)** és a **(3.2.1)** feltétel. Az előző megoldó algoritmushoz hasonlóan vegyük fel az s', t', s'' és t'' segédcsúcsokat és hozzájuk tartozó éleket. Oldjuk meg G hálózatban az $s'-t'$ és $s''-t''$ forrás-nyelő párok közötti maximális nagyságú folyamfeladatokat; és az ezekből kapott x' és x'' megoldások használatával készítsük el az eredeti probléma megoldásaiul szolgáló x^1 és x^2 folyamokat.

4. fejezet

Többtermékes folyamok

Ezen negyedik fejezet témája a többtermékes folyamfeladat lesz. Először bevezetjük a többtermékes folyam fogalmát; felírjuk a hozzá tartozó problémát és előkészülünk a probléma megoldásának megértésére. Másodszor be fogunk mutatni két, egymáshoz nagyon közel álló módszert a többtermékes folyamfeladat megoldására. Végül pedig két példát adunk a többtermékes folyamok alkalmazására személyzetütemezésben.

Ezen fejezet bevezető része és a probléma megoldásával foglalkozó része egyaránt *Ravindra K. Ahuja, Thomas L. Magnanti* és *James B. Orlin* könyve **[3]** alapján készült. A többtermékes folyamok személyzetütemezésben történő alkalmazására hozott példák közül az első példa bemutatása *Ercsey Zsolt* és *Kovács Zoltán* cikke **[7]** alapján-, a második példa bemutatása *Margarida Moz* és *Margarida Vaz Pato* cikke **[8]** alapján készült.

4.1. Bevezetés

4.1.1. Definíció

Legyen $G = (N, A)$ hálózatban K darab termék. Minden *termék* rendelkezik egy forrás- és egy nyelőcsúccsal, egy saját, minden $(i, j) \in A$ élre értelmezett költség- és kapacitásfüggvénnyel; illetve az adott termékhez tartozó folyammal. Legyen egy k termék $(i, j) \in A$ élen értelmezett kapacitásfüggvény értéke $u_{ij}^k \geq 0$, költségfüggvény értéke $c_{ij}^k \geq 0$ és folyamértéke x_{ij}^k. Ezen felül, G hálózat éleihez tartozik egy összkapacitásfüggvény; mely értékeit *közös korlátoknak* nevezzük, és minden $(i, j) \in A$ él esetén u_{ij}-vel jelöljük. Ez utóbbi az egy élen minden termékből összesen átvihető maximális folyamnagyságot jelzi. A *folyammegmaradási kikötések* minden csúcsra előírják, hogy a beérkező és kimenő élek folyamértékeinek különbsége egyezzen meg az adott csúcs kínálat-kereslet-függvényértékével, vagy *termelési értékével*. Ez az érték egy k termék és a termékhez tartozó forrás esetén egy pozitív b^k szám, egy k termék és a termékhez tartozó nyelő esetén a negatív $-b^k$ szám, egyéb termék-csúcs kombinációk esetén pedig 0.

Feltételezzük, hogy minden termék folyamegysége egységnyi kapacitást igényel; és hogy minden költségfüggvény lineáris a folyamegységek számában. Megjegyezzük, hogy az optimális megoldásban kaphatunk egy élen nem egészértékű egységnyit egy termékből, és az optimális összköltség sem lesz garantáltan egészértékű; az adatok egészértékűségi kikötése mellett sem. A feladat megkeresni a minimális költségű megengedett megoldást:

Minimalizáljuk
$$\sum_{1\le k\le K} c^k x^k; \tag{4.1.1a}$$

ahol
$$\sum_{1\le k\le K} x_{ij}^k \le u_{ij} \quad \forall (i,j)\in A \tag{4.1.1b}$$

vagy
$$\sum_{1\le k\le K} x_{ij}^k + s_{ij} = u_{ij} \quad \forall (i,j)\in A, \quad s_{ij}\ge 0 \text{ feleslegváltozó} \tag{4.1.1b'}$$

$$\mathcal{N}x^k = b^k, \quad \forall k = 1,\dots,K, \quad \mathcal{N} \text{ a } G \text{ szomszédsági mátrixa} \tag{4.1.1c}$$

$$0 \le x_{ij}^k \le u_{ij}^k, \quad \forall (i,j)\in A, \quad \forall k = 1,\dots,K. \tag{4.1.1d}$$

4.1.2. Optimalitási feltételek

Duális felírás: Ebben a részben feltételezzük, hogy minden él–termék kombináció esetén $u_{ij}^k = \infty$, vagyis a termékek élenkénti folyamnagyságai nincsenek egyedileg korlátozva; kizárólag az élenkénti kötegkorlátok szabályozzák az átvitelt a G hálózaton. A probléma duálisának két változófajtája legyen w_{ij} költség minden élre, és $\pi^k(i)$ potenciál minden csúcs-termék kombinációra; ezek segítségével definiáljuk a $c_{ij}^{\pi,k} = c_{ij}^k + w_{ij} - \pi^k(i) + \pi^k(j)$ leárazott költséget minden él-termék kombinációra. A leárazott költség mátrixos felírása $c^{\pi,k} = c^k + w - \pi^k\mathcal{N}$ alakú. A duális feladat a következő:

Maximalizáljuk
$$-\sum_{(i,j)\in A} u_{ij}w_{ij} + \sum_{k=1}^{K} b^k\pi^k; \tag{4.1.2a}$$

ahol
$$c_{ij}^{\pi,k} \ge 0, \quad \forall (i,j)\in A, \quad \forall k = 1,\dots,K, \tag{4.1.2b}$$

$$w_{ij} \ge 0, \quad \forall (i,j)\in A. \tag{4.1.2c}$$

Komplementer slackness tulajdonság többtermékes folyamokra: Egy x_{ij}^k megengedett megoldása a primál problémának az $u_{ij}^k = \infty \quad \forall (ij)\in A$ feltétel mellett akkor és csakis akkor optimális, ha létezik olyan nemnegatív w_{ij} duális költségfüggvény és π_i^k potenciálfüggvény, amelyekre igazak a következő tulajdonságok:

$$w_{ij}\Big(\sum_{1\le k\le K} x_{ij}^k - u_{ij}\Big) = 0, \quad \forall (i,j)\in A, \tag{4.1.3a}$$

$$c_{ij}^{\pi,k} \ge 0, \quad \forall (i,j)\in A, \quad \forall k = 1,\dots,K, \tag{4.1.3b}$$

$$c_{ij}^{\pi,k}x_{ij}^k = 0, \quad \forall (i,j)\in A, \quad \forall k = 1,\dots,K. \tag{4.1.3c}$$

Parciális dualizációs tétel: A megfelelő többtermékes folyamfeladatra optimális x_{ij}^k folyamok és optimális w_{ij} duális élköltségek minden k termékre teljesítik a következő minimális költségű folyamfeladatot:

Minimalizáljuk
$$\sum_{(i,j)\in A} (c_{ij}^k + w_{ij}) x_{ij}^k; \tag{4.1.4a}$$

ahol
$$\mathcal{N} x^k = b \tag{4.1.4b}$$
$$x_{ij}^k \geq 0 \quad \forall (i,j) \in A. \tag{4.1.4c}$$

4.2. Megoldási módszerek

4.2.1. Oszlop generálás

Irányított utas felírás

Feltételezzük, hogy minden terméknek egyetlen forrása és egyetlen nyelője van. Ismeretes, hogy minden megengedett, egy termékre értelmezett folyam szétbontható irányított $s-t$ utakon és irányított körökön átfolyó folyamok összegére. Feltételezzük továbbá azt is, hogy G hálózat *konzervatív* minden termékre, azaz egyik termékre sem létezik negatív összköltségű irányított kör. Ez utóbbi feltevés mellett minden optimális megoldásra feltehető, hogy minden termék esetén G minden irányított köre mentén az adott termék folyamának nagysága 0; így a potenciális optimális megoldásokat elegendő irányított $s_k - t_k$ utak nemnegatív lineáris kombinációjaként keresni, ahol s_k és t_k a k termék forrása és nyelője.

Legyen P^k azon P irányított utak halmaza, amelyeken k termék pozitív mennyiségű folyamot juttat el $s_k - t_k$ között. Minden él-termék kombináción felvett folyamérték felírható $x_{ij}^k = \sum_{P\in P^k} \delta_{ij}(P) f^k(P)$ alakba az $f^k(P)$ termékenkénti útfolyamértékekkel, ahol $\delta_{ij}(P) = 1$ ha (i,j) él $\in P$ útnak, 0 különben. Legyen továbbá $c^k(P) = \sum_{(i,j)\in P} c_{ij}^k$ a P út egységköltsége a k termék esetén. Ekkor az eredeti többtermékes folyamfeladat a következőképpen írható fel útfolyamokkal:

Minimalizáljuk
$$\sum_{1\leq k\leq K} \sum_{P\in P^k} c^k(P) f^k(P); \tag{4.2.1a}$$

ahol
$$\sum_{1\leq k\leq K} \sum_{P\in P^k} \delta_{ij}(P) f^k(P) \leq u_{ij}, \quad \forall (i,j) \in A \tag{4.2.1b}$$
$$\sum_{P\in P^k} f^k(P) = d^k, \quad \forall k = 1,\ldots,K \tag{4.2.1c}$$
$$f^k(P) \geq 0, \quad \forall (i,j) \in A, \quad \forall k = 1,\ldots,K, \quad \forall P \in P^k. \tag{4.2.1d}$$

Ezen felírásban a folyammegmaradási kikötések termékenként egyetlen kikötésként jelennek meg termék-csúcs kombinációkénti kikötések helyett, így a jelen lineáris program az eddigi $m+nK$ darab kikötés helyett csupán $m+K$ darab kikötéssel rendelkezik; viszont a módosított problémának sokkal több változója van - minden irányított útra egy. Kimutatható, hogy egy optimális megoldásban legfeljebb $K+m$ darab irányított útnak lesz az útfolyamértéke pozitív; ezt ezen dolgozat keretein belül nem bizonyítjuk. Ezáltal tehát egy optimális úthalmaz ismeretében, azaz a lineáris program oszlopainak egy optimális részhalmazának ismeretében;

a **(4.2.1)** feladat gyorsan megoldható lenne.

Irányított utas felírás optimalitási feltételei

A primál feladatbeli u_{ij} élenkénti kötegkorlátok és d^k termékenkénti kínálat-kikötések duális megfelelői a w_{ij} élenkénti költségértékek és a σ^k termékenkénti változók; amikkel az $f^k(P)$ útfolyamok leárazott költsége:

$$c_P^{\sigma,w} = c^k(P) + \sum_{(i,j)\in P} w_{ij} - \sigma^k = \sum_{(i,j)\in P} (c_{ij}^k + w_{ij}) - \sigma^k. \tag{4.2.2}$$

Komplementer slackness tulajdonság: $f^k(P)$ megengedett megoldás a **(4.2.1)**-beli problémára akkor és csakis akkor optimális, ha a hozzá tartozó duális változókkal együtt birtokolja a következő tulajdonságokat:

$$w_{ij}[\sum_{1\le k\le K}\sum_{P\in P^k} \delta_{ij}(P) f^k(P) - u_{ij}] = 0, \quad \forall (i,j)\in A \tag{4.2.3a}$$

$$c_P^{\sigma,w} \ge 0, \quad \forall k = 1,\dots,K, \quad \forall P \in P^k \tag{4.2.3b}$$

$$c_P^{\sigma,w} f^k(P) = 0, \quad \forall k = 1,\dots,K, \quad \forall P \in P^k. \tag{4.2.3c}$$

(4.2.2), **(4.2.3b)** és **(4.2.3c)** alapján az optimális megoldásnak megfelelő σ^k duális útváltozó a legolcsóbb $s_k - t_k$ út árával egyenlő a $c_{ij}^k + w_{ij}$ módosított élköltségre; továbbá minden, egy optimális megoldásban pozitív útfolyamértékkel rendelkező $s_k - t_k$ út minimális költségű az adott termékre nézve.

Optimalitási feltétel: $f(P)$ megengedett megoldás akkor és csakis akkor optimális, ha minden k termék esetén $f^k(P)$ értéke csak a G hálózatban a $c_{ij}^k + w_{ij}$ módosított árazásra minimális költségű $s_k - t_k$ utak mentén pozitív. Az optimális megoldásnak megfelelő σ^k duális útváltozó értéke ezzel a minimális költséggel egyenlő.

Megoldási módszer

Az **1.1.2** részben felelevenített módosított szimplex módszer minden iterációban elment egy $\mathcal{B}$ bázist, aminek segítségével kiszámolja a π szimplex tényezőket úgy, hogy a báziselemeknek megfelelő $c_B^\pi = c_\mathcal{B} - \pi\mathcal{B}$ leárazott költségek értéke 0 legyen; mindezt a nembázisoszlopok ismerete nélkül. Ezután a szimplex tényezőkkel kiértékeli a nembázisváltozókat, azaz kiszámolja a leértékelt költségüket; és ha van egy negatív leárazott költséggel rendelkező változó köztük, becseréli azt a bázisba. A következő megoldási módszer alapötlete az oszlopok struktúrájának olyan fajta kihasználása, ami mentén a nembázisoszlopok kiértékelésekor nem kell teljesen végignézni őket.

A probléma irányított utas felírását tekintve a módosított szimplex módszer minden iterációban a w_{ij} duális költségeket és σ^k szimplextényezőket számolja minden báziselemre; vagyis egy $s_k - t_k$ közötti P-vel jelölt bázisbeli útra a megoldandó egyenlet $c_P^{\sigma,w} = 0$, azaz $\sum_{(i,j)\in P}(c_{ij}^k + w_{ij}) = \sigma^k$. Az előbbiek alapján a bázist $K+m$ irányított út alkotja, továbbá minden egyenletnek $K+m$ változója van, K darab σ^k minimális útköltsége és m darab w_{ij} élváltozója; tehát a tényezők számolásához szükséges egyenletrendszer megoldása egyértelmű.

A **(4.2.3c)** komplementer slackness tulajdonságot tekintve egy bázisbeli P út esetén $c_P^{\sigma,w} = 0$; egy nembázisbeli út esetén $f^k(P) = 0$. A **(4.2.3a)** tulajdonságot tekintve az $s_{ij} = \sum_{1 \le k \le K} \sum_{P \in P^k} \delta_{ij}(P) f^k(P) - u_{ij}$ feleslegváltozó értéke 0-val egyenlő minden nembázisváltozóra; és minden bázisváltozó leárazott értékére $c_P^{\sigma,w} = 0 - w_{ij} = 0$, vagyis $w_{ij} = 0$. Tehát, a módosított szimplex módszer minden iterációjában látható bázis rendelkezik az első és harmadik komplementer slackness tulajdonsággal; vagyis az optimalitás ellenőrzéséhez csak a második tulajdonság ellenőrzésére van szükség: $c_P^{\sigma,w} \ge 0, \quad \forall k = 1, \ldots, K, \quad \forall P \in P^k$, azaz $\min_{P \in P^k} \sum_{(i,j) \in P} (c_{ij}^k + w_{ij}) \ge \sigma^k$. Ez azt jelenti, hogy a megengedett megoldás optimalitásának ellenőrzéséhez minden k termékre meg kell oldanunk a minimális költségű $s_k - t_k$ út problémát G hálózaton a $c_{ij}^k + w_{ij}$ módosított költséggel. A megengedett megoldás optimalitásához minden k termék esetén a megfelelő legrövidebb út költségének legalább akkorának kell lennie, mint σ^k. Ha van olyan legrövidebb út, ami ezt nem teljesíti, az definíció szerint egy negatív leárazott költségű nembázisbeli út; így egy ilyen utat cserélünk be a bázisba a következő iterációhoz. Ezen változtatásokon kívül az algoritmus a módosított szimplex módszert követi.

4.2.2. Dantzig-Wolfe felbontás

Képzeljük el, hogy a feladat megoldásához rendelkezésünkre áll K darab döntéshozó és 1 darab igazgató. Az *igazgató* feladata megoldani a többtermékes folyamfeladat irányított utas felírását, amit ezek után *mester problémának* fogunk nevezni, a k-adik *döntéshozó* feladata pedig meghatározni a k-adik terméknek megfelelő útfolyamértékeket a mester probléma megoldásában. Az igazgató számára minden pillanatban a mester probléma oszlopainak csak egy részhalmaza érhető el, így ő csak az ezeknek megfelelő lineáris programot képes megoldani, nevezzük ezt *megszorított mester problémának*.

Az igazgató minden iterációban megoldja a megszorított mester problémát, vagyis kiszámolja az optimális szimplex tényezőket a jelen bázisra. Ezután az igazgató ezeket a tényezőket megosztja a döntéshozókkal, akik a maguk rendjén megoldják a G hálózatban a minimális költségű $s_k - t_k$ út problémát az ő felelősségük alatt álló terméknek megfelelő módosított $c_{ij}^k + w_{ij}$ költségekre. Ha a k-adik döntéshozó kapott egy legolcsóbb utat, aminek költsége kisebb az igazgatótól kapott σ^k-nál, akkor visszajelzi ezt az utat az igazgatónak; ha pedig a legolcsóbb út költsége megegyezik σ^k-val, akkor nem küld vissza semmit. A σ^k definíciója alapján a k-adik döntéshozó által kapott legolcsóbb út költsége nem lehet nagyobb, mint σ^k. A döntéshozók feladatait nevezzük *alproblémáknak*.

A jelen megközelítése az előző megoldási módszernek rávilágít arra a tényre, hogy a többtermékes folyamfeladat oszlopgenerálást- vagy Dantzig-Wolfe felbontást alkalmazó megoldásai gyorsíthatóak párhuzamos processzorok használatával; hiszen a K darab alprobléma egymástól függetlenül, egyidőben végezhető minden iteráció során. A Dantzig-Wolfe felbontás egy általános megoldási irányzat alfeladatokra bontható problémák esetén.

4.3. Személyzetütemezés többtermékes folyamokkal

4.3.1. Többnapos ütemezés: munkásigény lefedése

Alapfeladat egyféle pozicióval

Legyen n a munkások száma, legyen D a tervezési időintervallum hossza, ahol a tervezési időintervallum azon periódus, amire készítjük a beosztást, napban; és legyen P a tervezési időintervallum építőköveiként szolgáló, nem feltétlenül azonos hosszúságú, időszakaszok száma. Legyen n_i az i-edik időszakasz munkaigénye és P_i az i-edik időszakasz hossza percben. Legyen L és U minden műszak hosszának alsó- és felsőkorlátja percben; továbbá legyen R a, minden alkalmazottra érvényes, két műszak közötti minimális pihenési idő percben. Legyen δ a teljes tervezési időintervallumra értendő alkalmazottankénti összmunkaidő, a tervezési időintervallumra megegyezett univerzális alkalmazottankénti összmunkaidőtől való eltérésének megengedett maximuma percben.

A következőkben a jelen ütemezési problémát egy többtermékes folyamfeladatként modellezük, amiben a termékek a munkásoknak felelnek meg; az egyes termékek folyamát alkotó pozitív folyamértékű irányított utak halmaza az alkalmazottak beosztásának vagy ütemezésének-; a hálózat iránya pedig az időnek felel meg. Az ezen modellhez épített $G = (N, A)$ hálózat csúcsai két szinten találhatóak: $N = \{q_0, q_1, \ldots, q_P\} \uplus \{p_0, p_1, \ldots, p_P\}$; élhalmaza pedig $A = \{(p_i, p_{i+1}), (q_i, q_{i+1}) : i = 0, \ldots, P-1\} \uplus \{(p_i, q_i), (q_i, p_i) : i = 0, \ldots, P\}$. Minden termék esetén q_0 a forrás és q_P a nyelő. Ha a q pontok sora alkotja a G hálózat alsó szintjét és a p pontok sora a G hálózat felső szintjét, akkor, ha az i-edik időszakaszban egy termék pozitív folyammennyiséggel rendelkező irányított útja egy alsó élen halad, a terméknek megfelelő alkalmazott az i-edik periódusban pihen; ha pedig ugyanezen irányított út egy j-edik időszakaszban egy felső élen halad, akkor a terméknek megfelelő alkalmazott a j-edik időszakaszban dolgozik. Egy termék pozitív folyammennyiségű irányított útjának felfelé irányuló élen haladása azt jelzi, hogy a termékhez tartozó alkalmazott a függőleges él utáni vízszintes éleknek megfelelő időszakaszban kezd dolgozni; hasonlóan az irányított útjának egy lefelé irányuló élen haladása az alkalmazott pihenőjének kezdetét jelzi.

Egészértékű lineáris program

A feladatnak megfelelő egészértékű lineáris program változói:

- $x_i^k \in \{0, 1\}, i \in \{1, \ldots, P\}$ jelzi, hogy az i-edik időszakaszban k alkalmazott dolgozik, vagy sem;
- $w_i^k \in \mathbb{N}, i \in \{1, \ldots, P\}$ az i-edik időszakasz végéig összegyűlt, megszakítás nélküli munkaideje k alkalmazottnak;
- $r_i^k \in \mathbb{N}, i \in \{1, \ldots, P\}$ az i-edik időszakasz végéig összegyűlt, megszakítás nélküli pihenőideje k alkalmazottnak;
- $d_i^k \in \{0, 1\}, i \in \{0, \ldots, P\}$ jelzi, hogy az i-edik időszakasz kezdete előtt k alkalmazott dolgozott, vagy sem.

Legyen M elég nagy, például $M = 1 + \sum_{i=1}^{P} P_i$. Az ütemezésnek megfelelő egészértékű lineáris program a következő:

Legyen $\forall k \in \{1, 2, \ldots, K\} \quad \forall i \in \{1, 2, \ldots, P\};$

Minimalizáljuk
$$\sum_{i=1}^{P} \sum_{k=1}^{K} x_i^k \tag{4.3.1a}$$

Műszakhossz kikötések:
$$w_i^k \leq x_i^k \cdot M, \quad w_0^k = 0 \tag{4.3.1b}$$
$$w_i^k \leq w_{i-1}^k + P_i \tag{4.3.1c}$$
$$w_i^k \geq w_{i-1}^k + x_i^k \cdot P_i + (x_i^k - 1) \cdot M \tag{4.3.1d}$$

Pihenéshossz kikötések:
$$r_i^k \leq (1 - x_i^k) \cdot M, \quad r_0^k = 0 \tag{4.3.1e}$$
$$r_i^k \leq r_{i-1}^k + P_i \tag{4.3.1f}$$
$$r_i^k \geq r_{i-1}^k + (1 - x_i^k) \cdot P_i - x_i^k \cdot M \tag{4.3.1g}$$

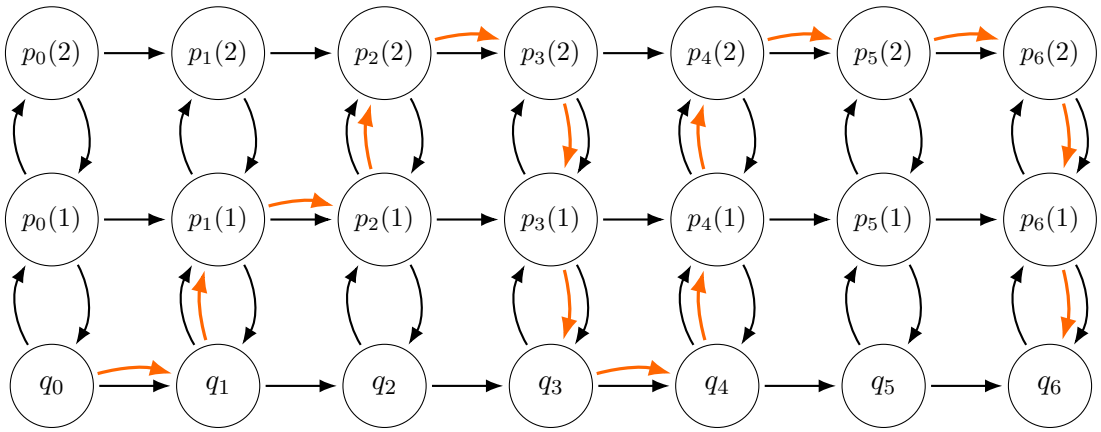
Aktivitást jelző kikötések:
$$d_i^k \leq (x_i^k + d_{i-1}^k), \quad d_0^k = 0 \tag{4.3.1h}$$
$$d_i^k \geq \frac{x_i^k + d_{i-1}^k}{2} \tag{4.3.1i}$$

Munkaerő szükséglet:
$$\sum_{k=1}^{K} x_i^k \geq n_i \tag{4.3.1j}$$

Legális műszakhossz:
$$w_i^k \leq U \tag{4.3.1k}$$
$$w_i^k \geq (x_i^k - x_{i+1}^k) \cdot L, \quad w_P^k \geq x_P^k \cdot L \tag{4.3.1l}$$
$$r_i^k \geq (x_i^k - x_{i+1}^k - x_i^k) \cdot R + (d_i^k - 1) \cdot M \tag{4.3.1m}$$

8 órás havi norma:
$$\sum_{i=1}^{P} x_i^k \cdot P_i \geq d_P^k \cdot (\frac{5}{7} \cdot 480 \cdot D - \delta) \tag{4.3.1n}$$
$$\sum_{i=1}^{P} x_i^k \cdot P_i \leq d_P^k \cdot (\frac{5}{7} \cdot 480 \cdot D + \delta) \tag{4.3.1o}$$

Többféle betöltendő pozició: Bővítsük az eddig tárgyalt alapfeladatot azáltal, hogy egyetlen pozíció helyett J darab különböző, különböző időszakaszokban különböző számú alkalmazottal betöltendő pozíció létezik. Az alapfeladatbeli x_i^k folyamváltozó helyett minden pozíciónak saját $x_i^k(j)$ bináris változója lesz aszerint, hogy k alkalmazott az i-edik időszakaszban a j pozícióban dolgozik-e, vagy sem. Ahogy az alapfeladatbeli hálózatban a felső szint tartozott a pozícióhoz, ezen probléma modellezéséhez épített G hálózatban minden különböző pozíciónak saját szintje lesz. Egy alkalmazottnak megfelelő termék pozitív folyamértékét szállító irányított útnak a vízszintes éleken való áthaladása továbbra is azt jelöli, attól függően, hogy hányadik sorban történik ez, hogy az adott alkalmazott az adott időszakaszban az adott pozícióban dolgozik vagy pihen. Egy alkalmazottnak megfelelő termék pozitív folyamértékét szállító irányított útnak a függőleges éleken való áthaladása tetszőleges hosszú függőleges irányított úton történhet egy lépésen belül; ez felel meg az adott alkalmazott az adott időszakaszban az adott út két végpontjának megfelelő pozíciók, a pihenést is ideértve, közti váltásának.



Példa egy alkalmazott beosztására a G hálózatban $P = 6$ és $J = 2$ esetén.

Fejlesztési lehetőség: Minden él-termék kombinációra bevezetett $c_i^k(j)$ költségfüggvények használatával további két hasznos ütemezési szempontot tudunk implementálni a modellbe. Az első ilyen szempont szerint egy elegendően nagy M költség esetén szabályozni tudjuk, ha egyes dolgozók alul- vagy felülkvalifikáltak egy pozícióra, és nem szeretnénk őket a számukra nem megfelelő pozícióban dolgoztatni. Például, ha k alkalmazott nem dolgozhat j pozícióban, akkor G hálózatban legyen $c(p_i(j), p_{i+1}(j))^k = M \quad \forall i \in \{1, 2, \ldots, P\}$, és $c(e)^k = 0$ különben. Ha esetleg egyes pozíciókban dolgozhat egy alkalmazott, de lehetőségekhez mérten minimalizálni szeretnénk az alkalmazott ezen pozíciókban dolgozott idejét; lehet az ennek megfelelő költségeket egy kisebb értékre, például 1-re állítani; ezzel teremtve egy prioritási sorrendet a megengedett megoldás keresésében nem akadályozó, de preferencia szerint elkerülendő esetek és szempontok között. Ugyanezen elegendően nagy M költséggel szabályozni tudjuk azt is, ha egyes periódusokban a dolgozó szabadságot akar kivenni; ekkor legyen $c(p_i(j), p_{i+1}(j))^k = M \quad \forall j \in \{1, 2, \ldots, J\}$, $c(e)^k = 0$ különben. A legtöbb adatot, például egyes alkalmazottak maximum és minimum műszakhosszát, havi normáját, stb.; alkalmazottanként változtathatjuk, szerződésüktől függően.

4.3.2. Többnapos újraütemezés

Tekintsünk egy olyan esetet, ahol minden napra előre meghatározott műszakokat kell emberekhez osztanunk. Ezek a műszakok különbözhetnek időpontjukban és hosszúságukban, illetve elvégzésükhöz szükséges kvalifikációban. Egy megengedett ütemezésre a következő kikötések állnak fenn:

- (i): minden alkalmazott egy nap egyetlen műszakban dolgozhat;
- (ii): minden műszakot egyidőben egyetlen munkás végezhet;
- (iii): egyes egymás utáni napokra definiált műszakpárok nem végezhetőek ugyanazon dolgozó által;
- (iv): egyes műszakokra (vagy napokra) egyes alkalmazottak nem elérhetőek (vagy alkalmatlanok);
- (v): minden dolgozó havi összmunkaszáma nem térhet el δ-nál többel a normától.

Újraütemezésre akkor van szükség, amikor egyes alkalmazottak a tárgyban forgó műszakjaik kezdete előtt jelzik, hogy hiányozni fognak egyes műszakokról. Ekkor jön létre a feladat, melyben az első hiányzás pillanatától kezdve egy új megengedett beosztást kell generálni, ami a lehető legkisebb mértékben tér el az eredetitől; ebből következik egy hatodik kikötés:

- (vi): hiányzó alkalmazottak nem dolgozhatnak olyan műszakon, amelyről hiányoznak.

Összesített többtermékes folyam modell

Legyen K darab alkalmazott és J féle pozíció, a pihenést vagy szabadnapot nem számítva pozíciónak; továbbá legyen a tervezési időintervallum hossza napban D. Legyen a bejelentett hiányzások elsőként érkező tárgynapja H. Legyen hálózatunk a következő $D-H+4$ oszlopos hálózat:

- az első oszlop K csúcsot tartalmaz, ezek a dolgozóknak megfelelő termékek forrásai;
- a következő $D-H+2$ oszlop mindegyike $J+1$ darab csúcspárt tartalmaz, ahol minden pár első eleme megfelel az egyik pozíciónak, a pihenést vagy szabadnapot is pozíciónak számítva; minden pár második eleme pedig a pozíció egy később részletezett fantomcsúcsa. Ezen oszlopok a $H-1$-edik naptól a D-edik napnak felelnek meg;
- az utolsó oszlop K csúcsot tartalmaz, ezek a dolgozóknak megfelelő termékek nyelői.

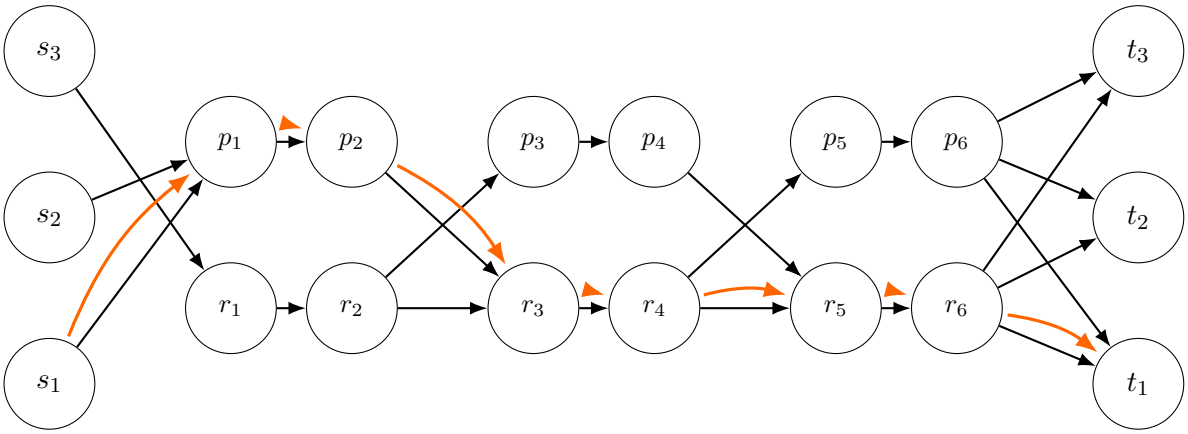
A források kínálatának értéke 1, a nyelők kínálatának értéke -1, minden más csúcs esetén pedig a kínálat értéke 0. A pozíciók csúcsaiból álló aloszlop és a hozzájuk tartozó fantomcsúcsokból álló aloszlop között csak $J+1$ darab él vezet, egyetlen él minden pozíció és a hozzá tartozó fantomcsúcs között. Ezen élek alsókorlátja biztosítja az adott napon szükséges minimális dolgozószámot az adott műszakban. Minden forrásból megy egy él abba a $H-1$-edik napi műszakba, amiben a hiányzást megelőző napon dolgozik az adott forrásnak megfelelő alkalmazott. Ezen felül minden fantomcsúcs össze van kötve azon hozzá képest következő napi műszakcsúcsokkal, amiknek megfelelő műszakokban dolgozhat egy munkás, aki a jelen napon épp az adott fantomcsúcsnak megfelelő műszakban dolgozott; egészen az utolsó oszlop fantomcsúcsaiig. A középső oszlopok között utolsónak tekinthető oszlop fantomcsúcsai minden olyan alkalmazottnak megfelelő termék nyelőjével össze vannak kötve, amelyre az adott terméknek megfelelő alkalmazottnak engedélyezett dolgoznia az adott műszakban.

A forrásokból induló- és nyelőkbe érkező-, illetve pozíciókat fantomcsúcsokkal összekötő élek költsége legyen 0 minden termék esetén. A különböző napoknak megfelelő oszlopok közti élek esetén minden termékre a következő költségfüggvényt használjuk:

$$c_e^k = \begin{cases} 0, & \text{ha } e \text{ él vége azon pozíciót jelöli a megfelelő napra, ahova } k \text{ eredetileg osztva volt,} \\ M, & \text{ha } e \text{ élnek megfelelő pozícióváltás nem megengedett } k \text{ esetében,} \\ 1, & \text{egyébként.} \end{cases}$$

A jelen részben definiált G hálózat és c költségfüggvény segítségével felírható minimális költségű többtermékes folyamfeladat megoldja a jelen újraütemezési problémát.

A következő ábra egy példát mutat a G hálózatra és egy alkalmazott megengedett beosztására abban az esetben, ha egyetlen pozíció van, és nem lehet egymás utáni napokon dolgozni; illetve s_1 alkalmazott hiányzik a tervezési idő utolsó 2 napján, így $c^1_{(r_4,p_5)} = M$.



Példa egy alkalmazott beosztására a G hálózatban $K = 3$, $H = D - 1$ és $J = 1$ esetén.

5. fejezet

Többutas folyamok

Ezen ötödik fejezet témája a többutas folyamfeladat lesz. Először bevezetjük a többutas folyam-, vagy adott k esetén k-folyam fogalmát és felírjuk a hozzá tartozó problémát; illetve definiáljuk és kiszámoljuk egy vágás k-méretét. Másodszor bizonyítani fogunk két tételt a klasszikus- és többutas folyamok kapcsolatáról, illetve a k-folyamok és k-vágások dualitásáról; továbbá vázoljuk a többutas folyamfeladat egy megoldási módszerét a bizonyított tételek alapján. Végül pedig egy példát adunk a többutas folyamok alkalmazására személyzetütemezésben.

Ezen fejezet többutas folyamok elméletét tárgyaló részei teljes mértékben *Amitabha Bagchi, Amitabh Chaudhary, Petr Kolman* és *Jiri Sgall* cikkje **[9]** alapján készültek. A többutas folyamok személyzetütemezésben történő alkalmazására hozott példa bemutatása *Charu C. Aggarwal* és *James B. Orlin* cikkje **[10]** alapján készült.

5.1. Bevezetés

A klasszikus értelemben vett egytermékes folyam elemi $s-t$ folyamok nemnegatív lineáris kombinációja; ahol *elemi $s-t$ folyamnak* nevezünk egy s forrás és egy t nyelő között, egyetlen irányított úton áthaladó, egységnyi méretű folyamot. Egy többutas folyam, amit ismert k érték mellett *k-folyamnak* nevezünk, a klasszikus folyamokhoz képest nem elemi $s-t$ folyamok-, hanem elemi $s-t$ k-folyamok nemnegatív lineáris kombinációja; ahol *elemi $s-t$ k-folyamnak* nevezünk egy s forrás és egy t nyelő között, k darab éldiszjunkt irányított úton, irányított útonként egységnyi folyamot szállító, k nagyságú folyamot. Nevezzük az előbb említett k darab éldiszjunkt irányított $s-t$ utat alkotó élek halmazát *k-rendszernek*.

Tekintsük az $G=(N,A)$ hálózatot $u: A \to \mathbb{R}_+$ kapacitásfüggvénnyel. Legyen $Q^2 = \{Q : Q$ k-rendszer G hálózatban$\}$. Jelölje továbbá $\delta_{ij}(Q)$ bináris változó azt, hogy egy $(i,j) \in A$ él eleme vagy sem Q k-rendszernek; és legyen $f(Q)$ az x folyam Q k-rendszeren keresztül szállított folyammennyisége.

A többutas folyam probléma kérdése az, hogy mennyi a G hálózaton maximálisan átszállítható k-folyam nagysága adott k-ra, a G hálózat élein értelmezett kapacitások betartásával:

Maximalizáljuk
$$\sum_{(s,j)\in A, s \text{ forrás}} x_{sj}; \tag{5.1.1a}$$

ahol
$$\sum_{Q\in Q^2} \delta_{ij}(Q) f(Q) \le u_{ij}, \quad \forall (i,j) \in A \tag{5.1.1b}$$
$$x_{ij} \ge 0, \quad \forall (i,j) \in A. \tag{5.1.1c}$$

Többutas folyamok esetén egy vágás *k-mérete* azon elemi k-folyamok maximális mennyisége, ami keresztül folyhat az adott vágásból kimenő, kapacitásokkal rendelkező éleken. Például, egy két élből álló vágás esetén, ahol az élek kapacitásai rendre az 1 és 100 értékeket veszik fel; a vágás klasszikus mérete 101-, míg 2-mérete 1-, és 3-mérete pedig 0 lesz.

5.1.1. Lemma a vágás k-méretének számításáról

Lemma: Legyen C egy l darab élből álló vágás, ahol az élek kapacitásai monoton csökkenő sorrendben $u_1 \ge u_2 \ge \cdots \ge u_l$; és legyenek $M_j = \frac{k}{k-j} \sum_{i=j+1}^{l} u_i \quad j \in \{0, 1, \ldots, k-1\}$ értékek. Ekkor C vágás k-mérete:

$$\gamma(u_1, u_2, \ldots, u_l) = \min\{M_0, M_1, \ldots, M_{k-1}\}. \tag{5.1.2a}$$

Továbbá, ha $M_j = \min\{M_0, M_1, \ldots, M_{k-1}\}$, akkor

$$u_i \ge \frac{1}{k} \cdot M_j \ge u_{i'} \quad \forall i \le j < i', \tag{5.1.2b}$$

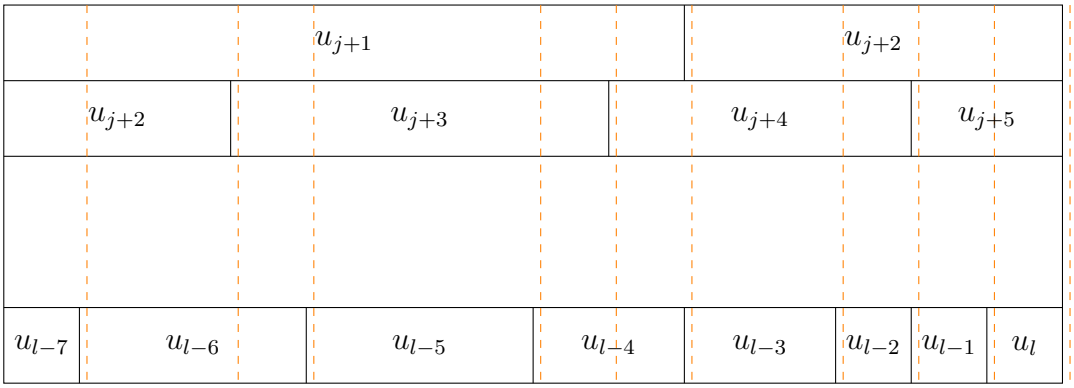
és
$$\gamma(u_1, u_2, \ldots, u_l) = \gamma(\frac{1}{k} \cdot M_j, \ldots, \frac{1}{k} \cdot M_j, u_{j+1}, \ldots u_l). \tag{5.1.2c}$$

Bizonyítás: Tegyük fel, hogy $M_j = \min\{M_0, M_1, \ldots, M_{k-1}\}$. Minden elemi k-folyam legalább $k-j$ darab élen átfolyik az $\{u_{j+1}, u_{j+2}, \ldots, u_l\}$ kapacitásokkal ellátott élek közül. Legyen $\bar{u}_j$ az $\{u_{j+1}, u_{j+2}, \ldots, u_l\}$ kapacitásokkal rendelkező élek közül, egy adott elemi k-folyam esetén pozitív folyamértékű élek száma. Ekkor minden C vágáson átfolyó k-folyam legfeljebb $\frac{1}{\bar{u}_j} \cdot \sum_{i=j+1}^{l} u_i \le \frac{1}{k-j} \cdot \sum_{i=j+1}^{l} u_i$ darab elemi folyam összegéből állhat; tehát minden k-folyam mérete legfeljebb $\frac{k}{k-j} \cdot \sum_{i=j+1}^{l} u_i = M_j$ lehet. Használva, hogy $M_j = \min\{M_0, M_1, \ldots, M_{k-1}\}$, minden $i \le j < i'$ esetén $\frac{M_j}{k} = \frac{1}{k-j} \cdot \sum_{i=j+1}^{l} u_i$-ra igaz, hogy $u_i \ge \frac{M_j}{k} \ge u_{i'}$, mert:

- Tegyük fel, hogy $u_i < \frac{M_j}{k}$. Ekkor $M_{i-1} = \frac{k}{k-i+1} \cdot \sum_{p=i}^{l} u_p < \frac{k}{k-i+1} \cdot \frac{M_j}{k} + \sum_{p=i+1}^{l} u_p = \frac{M_j}{k-i+1} + \frac{k-i}{k-i+1} \cdot M_i \le M_i \quad \forall i \le j$; tehát $i = j$-re $M_{j-1} < M_j$, ami ellentmondás.
- Tegyük fel, hogy $u_{i'} > \frac{M_j}{k}$. Ekkor $M_{i'} = \frac{k}{k-i'} \cdot \sum_{p=i'+1}^{l} u_p = \frac{k}{k-i'} \cdot \sum_{p=i'}^{l} u_p - \frac{k}{k-i'} \cdot u_{i'} < \frac{k-i'+1}{k-i'} \cdot M_{i'-1} - \frac{M_j}{k-i'} = M_{i'-1} + \frac{1}{k-i'} \cdot (M_{i'-1} - M_j) \quad \forall i' > j$; tehát $i' = j+1$-re $M_{j+1} < M_j$, ami ellentmondás.

A lemma utolsó része következik az előzőkből; mivel $u_1, u_2, \ldots, u_j$ megváltoztatása $\frac{M_j}{k}$-ra nem változtat M_j-n és nem csökkent egyetlen másik $\{M_i : i \le j\}$-beli értéket sem M_j alá, hiszen $M_i = \frac{k}{k-i} \cdot \sum_{p=i+1}^{l} u_p = \frac{k}{k-i} \cdot (j-i) \cdot \frac{M_j}{k} + \frac{k-j}{k-i} \cdot M_j = M_j \quad \forall i \le j$.

A bizonyítás befejezéhéz még meg kell mutatnunk, hogy M_j a C vágáson átférő k-folyamok méretének éles felsőbecslése. Ehhez építünk egy C vágáson átférő M_j méretű képzeletbeli k-folyamot. Vegyünk egy $(k-j)$ soros és $(\frac{M_j}{k})$ oszlopos téglalapot, amit ″töltsünk fel″, balról jobbra soron belül, fentről lefelé haladva a sorokon, rendre az $u_{j+1}, u_{j+2}, \ldots, u_l$ kapacitásokkal; ezek összege pontosan kitölti a téglalapot. Ezen feltöltés alapján a téglalap felosztható egységoszlopok összegéből álló cikkekre, amiken belül minden sort egyetlen él kapacitása tölt ki; és mivel $u_{i'} \leq \frac{M_j}{k} : \forall i' > j$, minden oszlopra igaz, hogy az oszlopot alkotó sortöredékek mindegyikét különböző él kapacitása tölti ki. Tehát, minden ilyen oszlopcikk egy-egy elemi $(k-j)$-folyam cikkszélességszeres többszörösének felel meg; így a teljes téglalap egy $\sum_{i=j+1}^{l} u_i = (k-j) \cdot \frac{M_j}{k}$ méretű $(k-j)$-folyamnak felel meg, ami nem használja a j legnagyobb kapacitású élet. Az első j él mindegyikén $\frac{M_j}{k}$ mennyiségű folyamegységet engedve át, és hozzátéve ezeket az eddig alkotott $(k-j)$-folyamhoz; megkapjuk a keresett $(k-j) \cdot \frac{M_j}{k} + j \cdot \frac{M_j}{k} = M_j$ nagyságú k-folyamot. □



Példa a $(k-j) \times \frac{F}{k}$-as téglalap kitöltésére és felosztására.

5.2. Dualitás

Egy F méretű $s-t$ folyamot akkor nevezünk *k-kiegyensúlyozottnak*, ha $x_{ij} \leq \frac{F}{k} \quad \forall (i,j) \in A$. Egy (i,j) élet akkor nevezünk *kritikusnak*, ha $x_{ij} = \frac{F}{k}$.

5.2.1. Lemma a k-kiegyensúlyozottságról és a kritikus élekről

Lemma: Minden k-kiegyensúlyozott körmentes x folyam esetén létezik olyan k-rendszer a G hálózatban, amelynek eleme G összes x szerint értelmezett kritikus éle.

Bizonyítás: Legyen $G' = (N', A')$ hálózat és u' kapacitásfüggvény azon F méretű x folyam szerinti hálózat-kapacitás pár, amiben $N' = N \cup \{s', t'\}$, ahol s' és t' egy új forrás és nyelő; és $A' = A_1 \cup A_2$, ahol $A_1 = \{(i,j) \in A : x_{ij} > 0\}, \quad u'_{ij} = x_{ij}, \forall (i,j) \in A_1$, illetve $A_2 = \{k$ darab (s', s) él és k darabb (t, t') él$\}, \quad u'_{ij} = \frac{F}{k}, \forall (i,j) \in A_2$. Feltéve, hogy x körmentes, $Q \subseteq A'$ akkor és csakis akkor egy $s' - t'$ közötti k-rendszer, ha minden (s, s') és (t', t) élt tartalmaz, illetve minden csúcs rendelkezik a folyam megmaradási tulajdonsággal.

Legyen Q a G' hálózat összes k-rendszere közül az, amelyik a legtöbb kritikus élt tartalmazza. Tegyük fel, hogy létezik egy $(u,v) \notin Q$ kritikus él. Legyen S azon csúcsok halmaza által alkotott részgráfja a G' hálózatnak, amelyek elérhetőek v-ből olyan javítóutakon, amik kétféle élekből állhatnak: Q által nem használt előreélekből és Q által használt nem kritikus élekből származtatott visszaélekből. Formailag $S = \{v$-ből elérhető $\bar{G}$ hálózatbeli csúcsok$\}$, ahol $\bar{G} = (N', A_+ \cup A_-), \quad A_+ = A' - Q, \quad A_- = \{(y,x) : (x,y) \in Q, (x,y)$ nem kritikus$\}$.

Tegyük fel, hogy $u \in S$. Tudjuk, hogy $(u,v) \in A_+$, így létezik S-ben egy (u,v)-t

tartalmazó K kör. Adjuk Q élhalmazhoz a $K \cap A_+ = K - Q$ éleket és töröljük Q élhalmazból az $\{(x,y) \in Q : (y,x) \in K\}$ éleket. Vegyük észre, hogy az új Q élhalmaz továbbra is k-rendszer, azonban legalább 1-gyel több kritikus élt tartalmaz, mint korábban. Ez ellentmond Q definíciójának; tehát, $u \notin S$.

Mivel $s', t' \notin S$, ezért Q ugyanannyiszor lép be S-be, mint amennyiszer kilép belőle. Mivel A_- definíció szerint a Q nem kritikus éleihez feleltet meg egy-egy S-beli visszaélt, Q csakis kritikus élek mentén kerülhet be az S részgráfba; minden találkozáskor S-be vezetve $\frac{F}{k}$ mennyiségű folyamot. Ezen kívül, S részgráfba vezetődik $\frac{F}{k}$ mennyiségű folyam $(u,v) \notin Q$ élen keresztül is. Mivel A_+ csak pozitív folyammennyiséggel rendelkező nem Q élhalmazbeli éleket tartalmaz, és minden folyamegység egy (t,t') él mentén folyik a nyelőbe; az S részgráfból csak Q élhalmazbeli élen folyhat ki folyam. A Q élhalmaz minden S részgráfba folyásakor maximális mennyiségű folyam folyt S-be, és pontosan annyiszor hagyja el a Q az S-et, mint amennyiszer belelép; tehát, az S-be bemenő folyamérték az (u,v) élen befolyó folyamérték miatt szigorúan nagyobb az S-ből kijövő folyamértéknél, ami ellentmond a G' hálózat folyammegmaradási kikötéseinek; vagyis a Q élhalmaz tartalmazza az összes kritikus élt. □

5.2.2. Tétel a k-folyam és k-kiegyensúlyozott folyam ekvivalenciájáról

Tétel: Egy körmentes folyam akkor és csakis akkor k-folyam, ha k-kiegyensúlyozott.

Bizonyítás: Minden k-folyam k-kiegyensúlyozott definíció szerint, ezen irány triviális. A fordított irány bizonyításához legyen x egy k-kiegyensúlyozott folyam. Az **5.2.1** lemma alapján keressük meg a G hálózatnak minden x folyam szerint kritikus élét tartalmazó k-rendszerét és jelöljük Q-val. Legyen $Q(x) = \min_{(i,j)\in Q} x_{ij}$ és $Q'(x) = \max_{(i,j)\in A-Q} x_{ij}$. Legyen $x'_{ij} = x_{ij} - \min\{Q(x), \frac{F}{k} - Q'(x)\} \quad \forall (i,j) \in Q \quad$ és $\quad x'_{ij} = x_{ij} \quad \forall (i,j) \in A - Q$.

Az x folyamról az x' folyamra való áttérés alatt 2 lehetséges eset következhet be: vagy keletkeznek 0 folyamot szállító élek a Q k-rendszerben, vagy a G hálózat minden kritikus élén szállított folyamérték leugrik a legnagyobb nem Q-beli él folyamértékére. Tehát, x' folyam F' méretére $F' = F - k \cdot \min\{Q(x), \frac{F}{k} - Q'(x)\}$, mert x' k darab éldiszjunkt $s-t$ úton csökkent x folyamhoz képest. Ebből következik, hogy $F' \geq k \cdot Q'(x)$, és $Q'(x)$ definíciójából $x'_{ij} \leq Q'(x) \quad \forall (i,j) \in A$; vagyis x' folyam is k-kiegyensúlyozott.

Az előző két lehetséges eset egyikének bekövetkezésétől függően x' folyamot tekintve a G hálózatban vagy kevesebb a pozitív folyammennyiséget szállító él, vagy több az x' folyam szerinti kritikus él. Tehát, az eddig bemutatott, x folyamról az x' folyamra való áttérést elegendően sokszor iterálva, elérhető egy olyan x' folyam, amely teljes egészében egy k-rendszeren folyik át. Ez azt jelenti, hogy az eredeti x folyam k-rendszereken átfolyó folyamok összege, vagyis elemi k-folyamok pozitív lineáris kombinációja, azaz egy k-folyam. Ha $I^{it} = |\{(i,j) \in A : x_{ij} \geq 0\}| + |\{(i,j) \in A : (i,j) \text{ nem kritikus él}\}|$, akkor I^{it} minden iterációban csökkenni fog, így $O(|A|)$ idő alatt megkapjuk a folyam k-rendszereken szállított folyamokra bontását. □

5.2.3. Tétel a k-folyam és k-vágás dualitásáról

Tétel: A G-beli maximális k-folyam mérete egyenlő a G-beli minimális k-vágás méretével.

Bizonyítás: A klasszikus folyamokra értelmezett minimális vágás - maximális folyam tétel, és az előző, klasszikus- és többutas folyamok kapcsolatáról szóló tétel ismeretében a jelen tétel nem triviális része keresni a G hálózatban egy, a maximális k-folyam méretével megegyező, k-vágást. Legyen x a maximális k-folyam G-ben, x mérete pedig legyen F. Legyen a G' segédgráf azonos a G hálózattal, annyi kivétellel, hogy G minden $\frac{F}{k}$-nál nagyobb élkapacitásának értéke G' hálózatban le van csökkentve $\frac{F}{k}$-ra. Vegyük észre, hogy a G' hálózatbeli maximális folyam F méretű. Legyen $C = \{e_1, e_2, \ldots, e_l\}$ minimális vágás G' hálózatban, ahol az élek kapacitásuk szerint monoton csökkenő sorrendben vannak indexelve, azaz $e_1 \geq e_2 \geq \cdots \geq e_l$. Legyen $j = \max\{i \in \{1, 2, \ldots, l\} : u_{e_i} > \frac{F}{k}, e_i \in G\}$. Ekkor $\forall e'_i \in G'$ esetén $\sum_{i=j+1}^{l} u_{e'_i} = F - j \cdot \frac{F}{k} = \frac{k-j}{k} \cdot F$; tehát $M_j = \frac{k}{k-j} \cdot \sum_{i=j+1}^{l} u_{e_i} = F$. A j definíciójából és az **5.1.1** lemmából következik, hogy C k-mérete $M_j = F$, így C egy minimális k-vágás G-ben. □

5.2.4. Megoldó algoritmus

Az **5.2.3** tétel következményeként, ha létezik F méretű k-folyam a G hálózatban, akkor egy klasszikus maximális folyamfeladatot megoldó, a maximális méretű folyam és a minimális kapacitású vágás dualitását kihasználó algoritmussal megkapható ez a folyam. Ahhoz, hogy a kapott F méretű folyam megoldás nem csupán klasszikus-, de k-folyam is legyen, a **5.2.2** tétel alapján biztosítanunk kell, hogy a G hálózat a kapott F méretű x folyam szerint k-kiegyensúlyozott legyen. Ezt a következő G hálózatbeli változtatott élkapacitásokkal érhetjük el: $u'_{ij} = \min\{u_{ij}, \frac{F}{k}\} \quad \forall (i,j) \in A$; és erre a módosított élkapacitású hálózatra alkalmazzuk a választott maximális folyam algoritmust.

Az **5.1.2** lemma alapján egészértékű élkapacitások mellett minden vágás mérete egy $\frac{y}{z} \leq kU$ racionális szám, ahol $z \leq k$ és U az élkapacitások maximuma. Ebből következik, hogy $y \leq zkU$ és $q \leq k \quad y, z \in \mathbb{N}$; tehát F legfeljebb $O(kU \sum_{z=1}^{k} z^2) = O(k^4 U)$ darab különböző értéket vehet fel. Innen látható, hogy bináris kereséssel legfeljebb $O(log(kU))$ darab F folyamértékre kell módosítsuk G hálózat élkapacitásait és megoldanunk ezen módosított hálózaton a maximális folyamfeladatot ahhoz, hogy ellenőrizzük, hogy létezik-e F méretű k-folyam a G hálózatban; és megkapjuk azon maximális F értékét, amire létezik ilyen k-folyam.

5.3. Ütemezés többutas folyamokkal

Legyen $\mathcal{K}$, $\mathcal{J}$ és $\mathcal{M}$ a dolgozók, a feladatok és a gépek halmaza. Minden feladat csupán egyetlen gépen és csupán adott $k \in \mathcal{K}_j$ személyek által végezhető el, ahol $\mathcal{K}_j$ a j feladat elvégzésére alkalmas személyek halmaza; és minden feladatnak van egy p_j elvégzési időtartama, ami azt jelöli, hogy j feladat m gépen p_j időegység alatt végezhető el. A feladat megkeresni a munkák azon gépekhez és személyekhez való ütemezését, amely esetén az alkalmazottak minimális idő alatt elvégzik az összes munkát.

A T összfeldolgozási időre ütemezéselméletből jól ismert triviális alsó becslés $\frac{1}{\min\{|\mathcal{M}|,|\mathcal{K}|\}} \cdot \sum_{j \in \mathcal{J}} p_j \leq T$. Ha egy ütemezésre $T = \frac{1}{\min\{|\mathcal{M}|,|\mathcal{K}|\}} \cdot \sum_{j \in \mathcal{J}} p_j$, vagy minden munkás vagy minden gép az $1, \ldots, T$ időpontok mindegyikében foglalt lenne, így az adott ütemezés összfeldolgozási ideje tovább nem csökkenthető-, ezáltal az adott ütemezés optimális lenne.

A probléma megoldásához konstruáljuk a következő $G = (N, A)$ hálózatot: legyen s a

forrás és t a nyelő; illetve legyen az $|\mathcal{M}|$ darab gépnek megfeleltetve $|\mathcal{M}|$ darab-, az $|\mathcal{J}|$ darab feladatnak megfeleltetve $|\mathcal{J}|$ darab-, és az $|\mathcal{K}|$ darab dolgozónak $|\mathcal{K}|$ darab csúcs; csoportokon belül egymás alá helyezve őket. Minden m gép esetén legyen egy végtelen kapacitású (s, m) irányított él, amire x_{sm} az m gép összaktivitását jelzi. Minden m gépnek és m-mel kompatibilis j feladatnak megfelelő csúcspár között legyen egy p_j kapacitású (m, j) irányított él, amire x_{mj} a j feladat m gépen töltött idejét jelzi. Minden k dolgozónak és k által elvégezhető j feladatnak megfelelő csúcspár között legyen egy végtelen kapacitású (j, k) irányított él, amire x_{jk} a k dolgozó j feladattal töltött idejét jelzi. Minden k dolgozó esetén legyen egy végtelen kapacitású (k, t) irányított él, amire x_{kt} folyamérték a k dolgozó aktív munkaidejét összegzi. Ezen hálózaton minden elemi l-folyam felfogható úgy, mint az ütemezés egy időegységének megfelelő része.

Tétel: Legyen $l = \min\{|\mathcal{M}|, |\mathcal{K}|\}$ és tegyük fel, hogy létezik egy $F^* = \sum_{j\in\mathcal{J}} p_j$ méretű $s - t$ l-folyam G hálózatban. Ekkor létezik egy olyan ütemezés, amiben vagy minden gépre, vagy minden alkalmazottra igaz, hogy minden időpontban mindannyian foglaltak.

Bizonyítás: A két eset szimmetrikussága miatt feltehető, hogy $|\mathcal{M}| \leq |\mathcal{K}|$. A feltétel szerint létezik G hálózatra egy megengedett F^* méretű $s - t$ $|\mathcal{M}|$-folyam, ami szétbontható elemi $|\mathcal{M}|$-folyamok összegére. Emellett egy-egy elemi $|\mathcal{M}|$-folyam $|\mathcal{M}|$ méretű, vagyis minden (s, m) élen egységnyi folyam megy át minden időpillanatnak megfelelő elemi $|\mathcal{M}|$-folyamban; tehát minden pillanatban minden gép foglalt. □

Következő lépésként találni akarunk G hálózatban egy F^* méretű klasszikus folyamot. Egy adott x folyam esetén legyen $r(x) = \max_{(i,j)\in A} x_{ij}$, és legyen x^* azon folyam, melyre $r(x^*)$ minimális. Vegyük észre, hogy minden x esetén $r(x)$ értéke egy (s, m) vagy (k, t) értéke lesz, mivel minden feladatokkal kapcsolatos élen szállított folyamérték az adott feladattal eltöltött időt jelöli, ami nem lesz nagyobb a feladattal kompatibilis gép- vagy a feladatot elvégző dolgozó összmunkaidejénél.

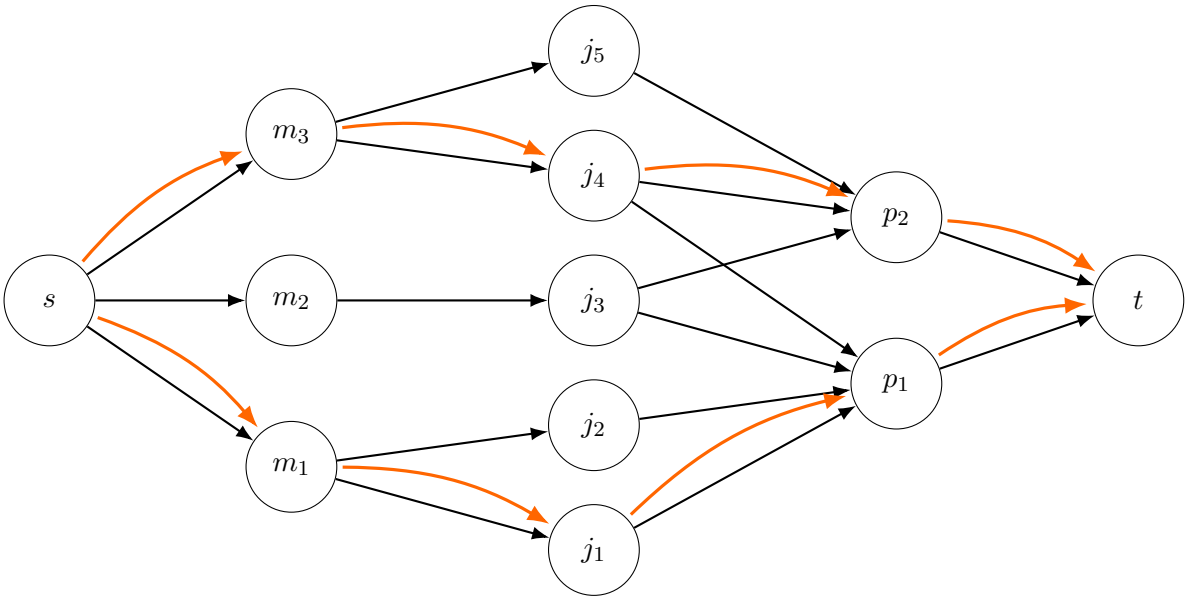
Tétel: Minden x^{opt} optimális ütemezésre $r(x^*) \leq T_{x^{opt}}$.

Bizonyítás: Az x^{opt} optimális ütemezés esetén $r(x^{opt})$ értéke egy (s, m) vagy (k, t) élen lesz felvéve, vagyis létezik olyan gép vagy személy, aki $r(x^{opt})$ időt aktív; tehát $T_{x^{opt}} \geq r(x^{opt})$, és definíció szerint $r(x^{opt}) \geq r(x^*)$. □

Tétel: Létezik olyan x^{opt} optimális ütemezés, amire $T_{x^{opt}} = r(x^*)$.

Bizonyítás: Legyen $l = \lfloor \frac{F^*}{r(x^*)} \rfloor$. Adjunk G hálózathoz egy (s, t) élet x_{st} folyamértékkel, amire a folyam összértéke $l \cdot r(x^*)$. Az l definíciója szerint $x_{st} \leq r(x^*)$; így $x_{ij} \leq r(x^*) \quad \forall (i, j) \in A$. Tehát, G hálózat x folyamra l-kiegyensúlyozott; vagyis x folyam l-folyam, ami szétbontható elemi l-folyamok összegére és $T_x \leq r(x^*)$. Mivel $\forall x^{opt} \quad r(x^*) \leq T_{x^{opt}}$, jelen x-re $r(x^*) \leq T_{x^{opt}} \leq T_x \leq r(x^*)$. □

Költségfüggvény bevezetése: Legyen minden j feladat, illetve egy adott j feladattal kompatibilis m gép és egy, a j feladat elvégzésére alkalmas k dolgozó esetén c_{mj} és c_{jk} a j feladat m gépen való, k dolgozó által történő elvégzésének időegységköltsége. A G hálózat éleihez rendelve c élköltségeket úgy, hogy ha c költségfüggvény nincs definiálva egy (i, j) élre, legyen $c_{ij} = 0$; G hálózaton megoldva a minimális költségű, F^* méretű többutas folyam problémát, megkaphatjuk a minimális költségű, optimális összmunkaidejű ütemezését a feladatoknak. Pontosabban, meg kell oldanunk a G hálózat és a c élköltségek által definiált minimális költségű problémát, F^* elvárt összfolyammértékkel és további, $x_{ij} \leq r(x^*) \quad \forall (i, j) \in A$ alakú lineáris kikötésekkel.



Példa egy időegység ütemezésére a G hálózatban $M = 3$, $J = 5$ és $P = 2$ esetén.

6. fejezet

Implementációs lehetőségek, alkalmazás

6.1. Elméleti kiegészítés - Megszorítások

A többtermékes folyamok **4.3** részben bemutatott személyzetütemezési alkalmazása során felmerült egy eddig még nem definiált fogalompár: egy optimalizációs feladat *erős* („hard") és *gyenge* („soft") megszorításai.

6.1.1. Erős megszorítások

Definíció: Az erős megszorítások olyan feltételek, amelyek definiálják a feladat megoldásainak megengedettségét. Egy megoldás akkor és csakis akkor megengedett, ha teljesíti a probléma összes erős megszorítását.

Minden probléma modellezésekor fontos megvizsgálni, hogy egy adott kikötést biztosan erős megszorításként szeretnénk-e modellezni, vagy sem. Abban az esetben, ha egy adott kikötést erős megszorításként modellezünk, és nem létezik olyan megengedett megoldása a problémának, ami teljesítene minden erős megszorítást; nem fogjuk tudni, hogy az adott kikötés törlése által létezne-e megengedett megoldás. Például, ha műszakos személyzetütemezésben nincs biztosítva egy, minden műszakot szükségesen lefedő megengedett megoldás létezése; ha a műszakok teljes lefedettségét erős kikötésként modellezzük, a lineáris program csupán azt közli, hogy nem létezik megengedett megoldás, ahelyett, hogy visszaadná a legnagyobb lehetséges lefedettséget biztosító részmegoldást. Gyakorlatban, megengedett megoldás hiánya esetén, érdemes lehet egyes erős megszorítások paraméterein változtatni, míg a változtatott paraméterekkel rendelkező problémára megengedett megoldást nem kapunk.

Egy lineáris program esetén az erős megszorítások használati módja természetes. Az erős megszorítások halmaza a lineáris program kikötéseinek részét képezi.

6.1.2. Gyenge megszorítások

Definíció: A gyenge megszorítások olyan feltételek, amelyek megszegése nem eredményez nem megengedett megoldást, ugyanakkor célunk, hogy ezek megszegésének mértékét a lehető legkisebbre csökkentsük az optimalizáció során.

A gyenge megszorításokat az adott optimalizációs problémában betöltött szerepük alapján két osztályba sorolhatjuk. Léteznek azok a gyenge megszorításokként modellezett kikötések, melyek jelentésükben erős megszorítások, viszont, mint az előbb bemutatott példában, szeretnénk elkerülhetetlen megszegésük esetén egyébként optimális megengedett részmegoldást kapni. Léteznek továbbá azok a gyenge megszorításokként modellezett kikötések, melyek jelentésükben is másodlagosak; vagyis nem befolyásolják egy megoldás gyakorlati

megengedettségének kérdését, viszont több megengedett megoldás közül egy olyat szeretnénk kapni, amely ezekben optimális. Például, egy tűzoltósági ügyelet személyzetütemezése esetén a legfontosabb, hogy folytonosan legyen elegendő szolgálatra kész tűzoltó; és az egyes tűzoltók szabadsági kérelmeinek engedélyezési aránya egy optimalizálandó, de a folytonos lefedettséghez képest másodlagos mérték. Azon gyakorlati problémák esetén, ahol személyek vagy embercsoportok sorsát befolyásolja egy-egy megoldás; objektíven nehezen metrizálható, jelentésben is gyenge megszorításra egy további fontos példa a személyek közti igazságosság.

Egy lineáris program esetén a gyenge megszorítások modellezése minden esetben egyes változókhoz rendelt költségfüggvények segítségével valósul meg. Mivel ezen megszorítások nem befolyásolhatják egy megoldás megengedettségének kérdését, ezért nem jelennek meg, mint a lineáris program kikötései. Ugyanakkor, mivel optimalizálni akarjuk ezek megszegésének mértékét, a lineáris program célfüggvényében megjelennek.

Mivel minden lineáris programnak egyetlen célfüggvénye, de változócsoportonként több gyenge megszorítása is lehet, a gyenge megszorítások célfüggvénybe foglalása nem triviális. A következőkben többfajta gyenge megszorítás egyetlen lineáris programon belül való kezelésére két, egyszerre is használható megközelítést ismertetünk.

Fontossági szinteken történő optimalizálás: Ez a módszer akkor alkalmazható, ha a feladat gyenge megszorításai egyértelműen fontossági sorrendbe tehetők. A módszer ötlete, hogy az optimalizáció folyamán elsősorban olyan megengedett megoldásokat keresünk, amelyek minél kisebb mértékben szegik meg a legfontosabb megszorításokat; az ezen szempont szerint optimális megengedett megoldások közül azokat keressük, amelyek minél kisebb mértékben szegik meg a második legfontosabb megszorításokat, és így haladva tovább az egyre kevésbé fontos gyenge megszorítások szerinti optimalizálási rétegeken. Ezen modellezési módszert igénylő gyenge megszorítások tipikusan az előbb bemutatott kategorizáció első csoportjába esnek.

Ezen módszer implementációjához olyan költségfüggvényre lesz szükség, amelynek a megfelelő változókhoz tartozó értékei a különböző fontosságú gyenge megszorítások szerint különböző nagyságrendűek. Ez azt jelenti, hogy a kevésbé fontos gyenge megszorítások lehetséges kereteken belül értelmezett sokszoros megszegése által keletkezett összköltség értéke kisebb egyetlen fontosabb kikötés megszegése által keletkezett költségnél. Ez a módszer számításilag gazdaságos, mivel egyetlen költségfüggvény elegendő több gyenge megszorítás modellezésére is.

Párhuzamos szinteken történő optimalizálás: Ez a megközelítés akkor alkalmazható, ha a feladat keretein belül több gyenge megszorítást is figyelembe kívánunk venni anélkül, hogy egyértelmű fontossági sorrendet állítanánk fel közöttük. A módszer lényege, hogy minden figyelembe vett gyenge megszorítás esetén egyidejűleg törekszünk azok minél kisebb mértékű megszegésére. Ezen modellezési módszert igénylő gyenge megszorítások tipikusan az előbb leírt kategorizáció második csoportjába esnek.

Ezen módszer implementációjához esetlegesen több költségfüggvényre lesz szükség; ezek segítségével hozzuk létre az egyes gyenge megszorítások optimalizálására szolgáló részcélfüggvényeket. A különböző célok súlyozásával egy közös, összegzett célfüggvény állítható elő, amely részcélfüggvények súlyozott összegeként működik, és lehetővé teszi, hogy a lineáris program párhuzamosan közelítsen több, egymással nem hierarchikus viszonyban álló szempont optimális teljesüléséhez.

6.2. Programcsomagok folyammodellek implementációjára

Mint azt korábban is láthattuk, az egyszerűbb, legrövidebb utak-, maximális folyam- vagy minimális költségű folyam problémák kombinatorikus módon, de lineáris programként is megoldhatóak. A dolgozat folyamán az is kiderült, hogy a kéttermékes folyam-, illetve a többutas folyam problémák is ebbe a kategóriába tartoznak. A többtermékes folyam problémák viszont, bár használjuk a többtermékes folyam hálózati tulajdonságait, elsősorban lineáris optimalizálási feladatokként oldhatóak meg.

Ezen észrevételek alapján nem meglepő, hogy általában folyammodellek implementációjához több, különböző funkciójú programcsomagra lesz szükségünk, programozási nyelvtől függetlenül. A következőkben ezen programcsomag-csoportok mindegyikéről külön is beszélünk. Először különböző gráfkönyvtárakról lesz szó, melyek gráfépítésre és gráfvizualizációs feladatok elvégzésére alkalmasak; illetve a legegyszerűbb folyam- és gráfoptimalizálási problémák megoldására rendelkeznek, nagy szinten optimalizált és könnyen használható, beépített függvényekkel. A következő két programcsomag-csoport egy-egy választottjára lesz szükségünk a bonyolultabb, lineáris programként felírt folyamfeladatok megoldásához: egy optimalizációs modellező környezet szükséges a feladat lineáris programkénti felírásához; és egy lineáris megoldóprogram szükséges az optimalizációs modellező környezet által létrehozott optimalizációs feladat megoldásához. Végül pedig olyan könyvtárakról és szoftverekről beszélünk, amik ütemezési problémák közvetlen megoldására alkalmasak.

A következőkben tárgyalt programcsomagok mindegyikéről a programcsomagok saját, hivatalos felhasználói dokumentációjuk **[11]** - **[22]** alapján beszélünk.

Gráfkönyvtárak

- *NetworkX*: Gráfvizualizációra és gráfoptimalizálásra alkalmas. Könnyű használni; leginkább kisméretű gráfok kezelésére, és tanításra alkalmas. *Python* alapú csomag. Rendelkezik beépített algoritmusokkal a következőkre: maximális folyam, minimális vágás, minimális költségű javítóút, minimális költségű folyam és hálózati szimplex módszer.
- *igraph*: Nagyobb gráfok hatékony kezelésére fejlesztett könyvtár. Főként *python* és *R* alapú csomag. Rendelkezik beépített algoritmusokkal a következőkre: maximális folyam, minimális vágás és maximális számú éldiszjunkt utak száma.
- *LEMON* (Library for Efficient Modeling and Optimization in Networks): *C++* alapú, rendkívül gyors és optimalizált gráfkönyvtár, amely ipari alkalmazásokhoz is alkalmas. Leginkább tapasztalt felhasználóknak ajánlott. Rendelkezik beépített algoritmusokkal a következőkre: minimális költségű utak, maximális folyam, minimális vágás, minimális költségű folyam és hálózati szimplex módszer. A könyvtár karbantartója az Eötvös Loránd Tudományegyetem Operációkutatási Tanszékén működő Egerváry Kutatócsoport.
- *Google OR-Tools*: Nyílt forráskódú optimalizációs könyvtár. Rendelkezik minta programkódokkal a következő algoritmusok implementációjára: minimális költségű utak, maximális folyam és minimális vágás. *Python, C++, Java* és más nyelveken is elérhető.

Optimalizációs modellező környezetek

- *PuLP*: Alapértelmezett *python* optimalizációs modellező környezet. Felhasználóbarát szintaxissal rendelkezik, de csak lineáris és egészértékű problémákat támogat. Kompatibilis a legnépszerűbb lineáris megoldóprogramokkal.
- *Pyomo*: Fejlettebb *python*-alapú modellező eszköz, amely támogatja a lineáris, nemline-

áris, vegyes-egészértékű és dinamikus optimalizációs problémák modellezését. Kompatibilis számos lineáris megoldóprogrammal.
- *AMPL*: Magas szintű modellező nyelv, amely széles körben használt ipari és kutatási környezetben is. Nagyon hatékony, de legtöbb funkciója nem ingyenes. Könnyen integrálható professzionális megoldókkal.
- *Google OR-Tools*: Széleskörű modellezési képességeit egy hatékony megszorítás programozó környezetbe ágyazza. A könyvtár minden beépített függvényének forráskódja elérhető, másolható és módosítható. Részletes dokumentációval rendelkezik-, valamint átfogó megoldásokat kínál a személyzetütemezési és a Job Shop ütemezési problémákra.

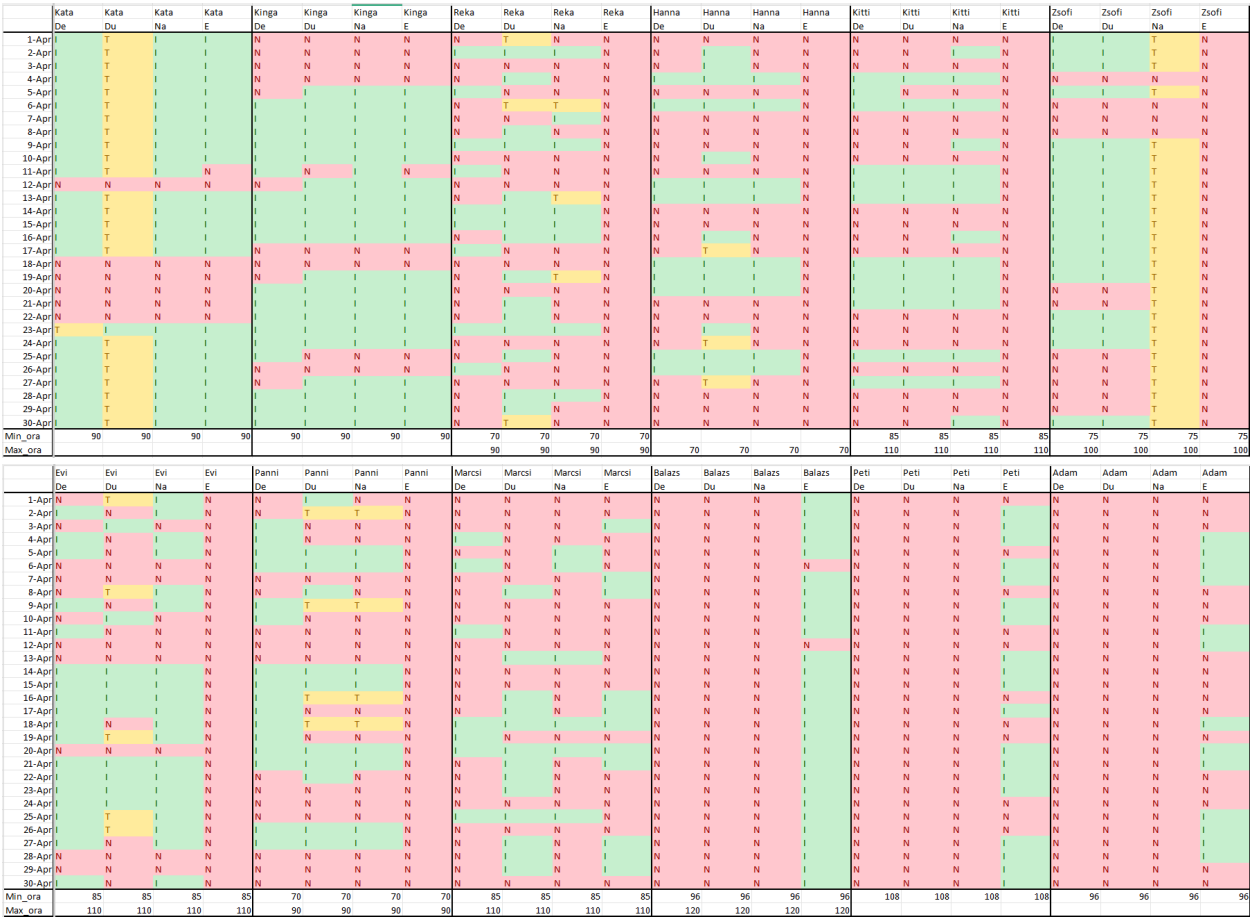
Lineáris megoldóprogramok

- *CBC* (Coin-or Branch and Cut): Alapértelmezett *python* lineáris és egészértékű megoldóprogram. Jól illeszkedik a *PuLP* és *Pyomo* környezetekhez. Kisebb problémákhoz megbízható választás.
- *Gurobi*: Egyik leggyorsabb és legmegbízhatóbb kereskedelmi lineáris programozási megoldó. Széleskörű támogatással rendelkezik *Python, C++* és *Java* nyelveken.
- *CPLEX*: Ipari standard megoldó az *IBM*-től, amely hatékonyan kezeli a lineáris, egészértékű és nemlineáris problémákat is. *AMPL*-lel és más keretrendszerekkel jól integrálható.
- *HiGHS*: Új, nyílt forráskódú, nagy teljesítményű lineáris és vegyes-egészértékű megoldóprogram, amely különösen gyors nagy, ritka mátrixú egyenlőtlenségrendszerekkel rendelkező problémák esetén.
- *SCIP*: Ingyenes és nyílt forráskódú lineáris megoldóprogram, amely lineáris, konvex nemlineáris és vegyes-egészértékű programokat képes megoldani. Az egyik legerősebb vegyes-egészértékű megoldóprogram.
- *Google OR-Tools*: Két megoldóprogrammal rendelkezik. A könyvtár eredeti CP (Constraint Programming) megoldója ütemezési és szállítási problémák megoldására alkalmas. Ha tisztán egyészértékű program megoldására van szükségünk, a könyvtár másik megoldója, a *CP-SAT* lesz a hatéknyabb választás; amely SAT (kielégíthetőségi) módszerek alapján működik. A *SAT módszerek* olyan algoritmusok, amelyek egy adott logikai kifejezéshez keresnek olyan megoldásokat, amelyek az adott kifejezést igazzá teszik.

6.3. Többtermékes folyamok alkalmazása valós adatkészleten

A dolgozat utolsó részeként tekintsünk meg egy gyakorlati példát a többtermékes folyamok személyzetütemezési alkalmazására. A jelen példa adatai egy diákszálló recepcióján dolgozó diákoknak a következő hónapra vonatkozó rendelkezésre állását tartalmazza. A beosztást készítő menedzsernek be kell osztania a diákokat ráérésük figyelembe vételével úgy, hogy a hostel recepcióján elvégzendő műszakok mindegyike be legyen töltve minden nap. A recepció 24 órában nyitva áll; minden nap 4 műszakban dolgozik egy-egy alkalmazott a recepción: délelőtti műszak 8-16 óráig, nappali műszak 12-20 óráig, délutáni műszak 16-22 óráig és éjszakai műszak 20-8 óráig. A tervezési periódusnak megfelelő hónap kezdete előtt a menedzser bekéri minden diák következő havi ráérését a következőképpen: minden alkalmazott a következő hónap minden műszakjáról elmondja, hogy ráér (I - igen), foglalt (N - nem), vagy ráér, de nem kényelmesen (T - talán). Továbbá, minden diák megadhat egy felső- és egy alsókorlátot a következő hónapban ledolgozni kívánt összmunkaóraszámára. A példában használt adatok eredeti alakjukban a **(6.3.1)** ábrán láthatóak.

A megoldandó feladat legfontosabb szempontja betölteni minden egyes műszakot úgy, hogy senki sem dolgozik olyan műszakban, amelyikben foglalt. A következő fontos szempontok a diákok összmunkaóráival kapcsolatos kérések betartása, illetve a kényelmetlen ráérések minimális használata. Ezen utóbbi két kikötéscsoportot gyenge megszorításokként kezeljük; törekedve egyfajta igazságosságra is a diákok között.



	Kata				Kinga				Reka				Hanna				Kitti				Zsofi			
	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E
1-Apr	I	T	I	I	N	N	N	N	N	T	N	N	N	N	N	N	N	N	N	N	I	I	T	N
2-Apr	I	T	I	I	N	N	N	N	I	I	I	N	N	I	N	N	N	N	I	N	I	I	T	N
3-Apr	I	T	I	I	N	N	N	N	N	N	N	N	N	I	N	N	N	N	N	N	I	I	T	N
4-Apr	I	T	I	I	N	N	N	N	N	I	N	N	I	I	I	N	I	I	I	N	N	N	N	N
5-Apr	I	T	I	I	N	I	I	I	I	N	N	N	N	N	N	N	I	N	N	N	I	I	T	N
6-Apr	I	T	I	I	I	I	I	I	N	T	T	N	I	I	I	N	I	I	I	N	N	N	N	N
7-Apr	I	T	I	I	I	I	I	I	N	N	I	N	N	N	N	N	N	N	N	N	N	N	N	N
8-Apr	I	T	I	I	I	I	I	I	N	I	N	N	N	N	N	N	N	N	N	N	N	N	N	N
9-Apr	I	T	I	I	I	I	I	I	I	I	I	N	N	N	N	N	N	N	I	N	I	I	T	N
10-Apr	I	T	I	I	I	I	I	I	N	N	N	N	N	I	N	N	N	N	N	N	I	I	T	N
11-Apr	I	T	I	N	I	N	I	N	I	N	N	N	N	N	N	N	I	I	I	N	I	I	T	N
12-Apr	N	N	N	N	N	I	I	I	N	N	N	N	I	I	I	N	I	I	I	N	I	I	T	N
13-Apr	I	T	I	I	I	I	I	I	N	I	T	N	I	I	I	N	I	I	I	N	I	I	T	N
14-Apr	I	T	I	I	I	I	I	I	I	I	I	N	N	N	N	N	N	N	N	N	I	I	T	N
15-Apr	I	T	I	I	I	I	I	I	I	I	I	N	N	N	N	N	N	N	N	N	I	I	T	N
16-Apr	I	T	I	I	I	I	I	I	N	I	I	N	N	I	N	N	N	N	I	N	I	I	T	N
17-Apr	I	T	I	I	N	N	N	N	I	N	N	N	N	T	N	N	N	N	N	N	I	I	T	N
18-Apr	N	N	N	N	N	N	N	N	N	N	N	N	I	I	I	N	I	I	I	N	I	I	T	N
19-Apr	N	N	N	N	N	I	I	I	N	I	T	N	I	I	I	N	I	I	I	N	I	I	T	N
20-Apr	N	N	N	N	I	I	I	I	N	N	N	N	I	I	I	N	I	I	I	N	N	N	T	N
21-Apr	N	N	N	N	I	I	I	I	N	I	N	N	N	N	N	N	I	I	I	N	N	N	T	N
22-Apr	N	N	N	N	I	I	I	I	N	I	N	N	N	N	N	N	N	N	N	N	I	I	T	N
23-Apr	T	I	I	I	I	I	I	I	I	I	I	N	N	I	N	N	N	N	N	N	I	I	T	N
24-Apr	I	T	I	I	I	I	I	I	N	N	N	N	N	T	N	N	N	N	N	N	I	I	T	N
25-Apr	I	T	I	I	I	N	N	N	N	I	N	N	I	I	I	N	I	I	I	N	N	N	T	N
26-Apr	I	T	I	I	N	N	N	N	I	N	N	N	I	I	I	N	N	N	N	N	N	N	T	N
27-Apr	I	T	I	I	N	I	I	I	N	N	N	N	N	T	N	N	I	I	I	N	N	N	T	N
28-Apr	I	T	I	I	I	I	I	I	N	I	I	N	N	N	N	N	N	N	N	N	N	N	T	N
29-Apr	I	T	I	I	I	I	I	I	N	I	N	N	N	N	N	N	N	N	I	N	I	I	T	N
30-Apr	I	T	I	I	I	I	I	I	N	T	N	N	N	N	N	N	N	N	I	N	I	I	T	N
Min_ora		90	90	90	90		90	90	90	90	70	70	70	70			85	85	85	85	75	75	75	75
Max_ora									90	90	90	90	70	70	70	70	110	110	110	110	100	100	100	100

	Evi				Panni				Marcsi				Balazs				Peti				Adam			
	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E
1-Apr	N	T	I	N	N	I	N	N	N	N	N	N	N	N	N	I	N	N	N	N	N	N	N	N
2-Apr	I	N	I	N	N	T	T	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	N
3-Apr	N	I	N	N	I	N	N	N	N	N	N	I	N	N	N	I	N	N	N	I	N	N	N	N
4-Apr	I	N	I	N	I	N	N	N	I	N	N	N	N	N	N	I	N	N	N	I	N	N	N	I
5-Apr	I	N	I	N	I	I	I	N	N	N	I	N	N	N	N	I	N	N	N	N	N	N	N	I
6-Apr	N	N	N	N	I	I	I	N	I	N	I	N	N	N	N	N	N	N	N	I	N	N	N	I
7-Apr	N	N	N	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	I	N	N	N	I
8-Apr	N	T	I	N	N	I	N	N	I	N	I	I	N	N	N	I	N	N	N	N	N	N	N	N
9-Apr	I	N	I	N	I	T	T	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	N
10-Apr	N	I	N	N	I	N	N	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	N
11-Apr	I	N	N	N	N	N	N	N	I	N	N	N	N	N	N	I	N	N	N	N	N	N	N	I
12-Apr	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	I
13-Apr	N	N	N	N	N	N	N	N	N	I	I	N	N	N	N	I	N	N	N	I	N	N	N	N
14-Apr	I	I	I	N	I	I	I	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	N
15-Apr	I	I	I	N	I	I	I	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	N
16-Apr	I	I	I	N	I	T	T	N	N	I	N	I	N	N	N	I	N	N	N	N	N	N	N	N
17-Apr	I	I	I	N	I	N	N	N	N	I	N	I	N	N	N	I	N	N	N	I	N	N	N	N
18-Apr	I	N	I	N	I	T	T	N	I	I	I	I	N	N	N	I	N	N	N	N	N	N	N	I
19-Apr	I	T	I	N	I	N	N	N	I	N	N	N	N	N	N	I	N	N	N	N	N	N	N	N
20-Apr	N	N	N	N	I	I	I	N	I	I	I	I	N	N	N	I	N	N	N	I	N	N	N	I
21-Apr	I	I	I	N	I	I	I	N	N	I	N	I	N	N	N	I	N	N	N	I	N	N	N	I
22-Apr	I	I	I	N	N	I	N	N	N	I	N	N	N	N	N	I	N	N	N	I	N	N	N	N
23-Apr	I	I	I	N	N	N	N	N	N	I	N	N	N	N	N	I	N	N	N	I	N	N	N	N
24-Apr	I	I	I	N	N	N	N	N	N	N	N	N	N	N	N	I	N	N	N	N	N	N	N	N
25-Apr	I	T	I	N	N	N	N	N	I	I	I	N	N	N	N	I	N	N	N	N	N	N	N	I
26-Apr	I	T	I	N	I	I	I	N	N	N	N	N	N	N	N	I	N	N	N	N	N	N	N	I
27-Apr	I	N	I	N	I	I	I	N	N	I	N	I	N	N	N	I	N	N	N	I	N	N	N	I
28-Apr	N	N	N	N	N	N	N	N	N	I	N	I	N	N	N	I	N	N	N	I	N	N	N	I
29-Apr	N	N	N	N	N	N	N	N	N	I	N	I	N	N	N	I	N	N	N	I	N	N	N	N
30-Apr	I	N	I	N	N	N	N	N	N	N	N	N	N	N	N	I	N	N	N	I	N	N	N	N
Min_ora		85	85	85		70	70	70		85	85	85		96	96	96		108	108	108		96	96	96
Max_ora		110	110	110		90	90	90		110	110	110		120	120	120								

A példában használt eredeti bemenet. (6.3.1)

A példa implementációjának első feladataként módosítjuk a bemenetet úgy, hogy minden diák összes ráérési adata egyetlen oszlopban legyen. Az I, T és N válaszokat helyettesítjük később használt költségekkel: I esetén 0, T esetén 1 és N esetén 1000 lesz az adott diák-műszak kombinációhoz tartozó változó költsége. Ha egy diák nem adott felső- vagy alsó korlátot a következő hónapban ledolgozott összmunkaóraszámára, kiegészítjük a hiányzó adatokat az alapértelmezett felső- és alsó korlátokkal; melyek esetünkben havi 56 és 240 óra. A módosított adathalmaz kis részét láthatjuk a **(6.3.2)** ábrán.

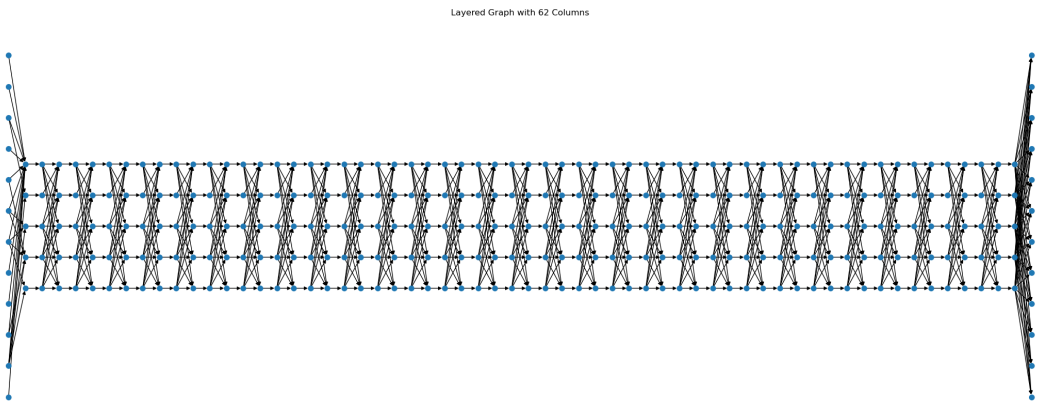
	Datum	Muszak	Kata	Kinga	Reka	Hanna	Kitti	Zsofi	Evi	Panni	Marcsi	Balazs	Peti	Adam
0	1-Apr	De	0	1000	1000	1000	1000	0	1000	1000	1000	1000	1000	1000
1	1-Apr	Du	1	1000	1	1000	1000	0	1	0	1000	1000	1000	1000
2	1-Apr	Na	0	1000	1000	1000	1000	1	0	1000	1000	1000	1000	1000
3	1-Apr	E	0	1000	1000	1000	1000	1000	1000	1000	1000	0	1000	1000
4	2-Apr	De	0	1000	0	1000	1000	0	0	1000	1000	1000	1000	1000
5	2-Apr	Du	1	1000	0	0	1000	0	1000	1	1000	1000	1000	1000
6	2-Apr	Na	0	1000	0	1000	0	1	0	1	1000	1000	1000	1000
7	2-Apr	E	0	1000	1000	1000	1000	1000	1000	1000	1000	0	0	1000

Módosított adathalmaz az eredeti bemenet alapján. (6.3.2)

Felmérve az adataink méreteit; azaz jelen esetben ismerve, hogy a tervezési periódus hossza 30 nap, a műszakok száma 4 és a diákok száma 12; az előbb tárgyalt *NetworkX* programcsomag segítségével létrehozzuk a feladat modellezéséhez használt $G = (N, A)$ hálózatot.

A többtermékes folyamok **4.3.2** alrészben tárgyalt személyzetújraütemezési alkalmazásában bevezetett hálózat mintájára alkotjuk meg a jelen példa megoldásához használt hálózatot. Legyenek a G hálózat csúcsai 62 oszlopba rendezve. Az első és utolsó oszlop minden diáknak megfeleltetve tartalmaz egy-egy csúcsot, mely az adott diáknak megfelelő termék forrása-, illetve nyelőjeként fog működni. A középső 60 oszlop 30 darab, egy-egy napnak megfelelő oszlop-párt tartalmaz. Minden ilyen oszlop $4 + 1 = 5$ csúcsot tartalmaz, melyek lentről felfelé rendre a délelőtti-, délutáni-, nappali-, éjszakai- és pihenőműszakokat modellezik.

Minden termék forrásából akkor megy él az első középső oszlop egy-egy csúcsába, ha az adott csúcspárnak megfelelő diák-műszak kombinációról kapott ráérési adat értéke I vagy T. Minden termék nyelőjébe akkor megy él az utolsó középső oszlop egy-egy csúcsából, ha az adott csúcspárnak megfelelő diák-műszak kombinációról kapott ráérési adat értéke I vagy T. Minden alkalmas középső oszlop-pár második oszlopának majdnem minden csúcsából megy a következő oszlop-pár első oszlopjának minden csúcsába él; kivéve az adott nap éjszakai műszakjának megfelelő csúcsot, amely csupán a következő nap éjszakai-, illetve pihenőműszakjának megfelelő csúcsokkal van összekötve. Ennek modellezési oka azon gyakorlati kikötés, hogy egy éjszakai műszakot végző diáknak a következő napon csupán újra éjszakáznia vagy pihennie szabad. Minden középső oszlop-pár két oszlopja között 5 él fut, 1-1 darab minden műszaknak megfelelő csúcs és az adott műszaknak megfelelő fantomcsúcs között. Az így kapott G hálózatot láthatjuk a **(6.3.3)** ábrán.



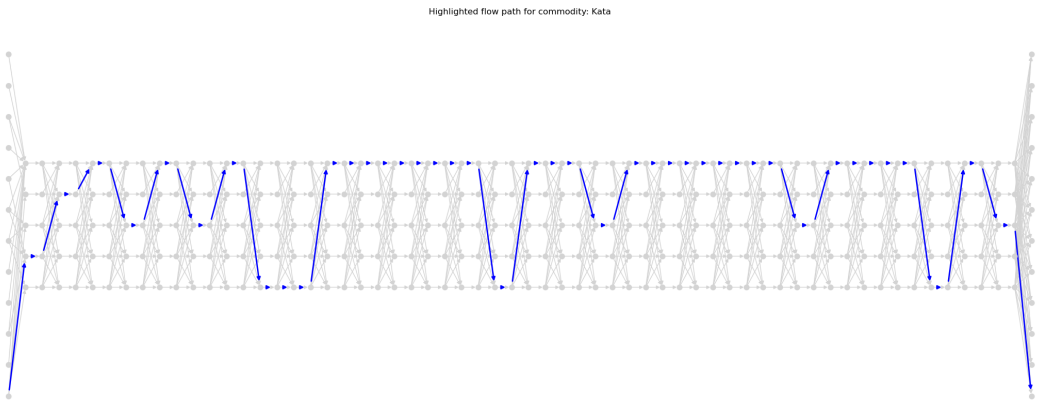
A példa modellezésére használt G hálózat. (6.3.3)

Azon erős megszorítást, hogy minden műszakban pontosan egyetlen diák dolgozzon; a legutoljára sorolt élek kapacitásának beállításával érjük el. Legyen minden valós műszaknak megfelelő oszlop-párbeli él kapacitása 1; és legyen minden pihenő műszaknak megfelelő oszlop-párbeli él kapacitása egyenlő azon értékkel, amit úgy kapunk, hogy a diákok számából kivonjuk a valós műszakok számát, ami jelen esetben $12 - 4 = 8$. A hálózat minden egyéb élének kapacitása maradjon korlátlan.

Bevezetjük a diákoknak megfelelő termékeket, melynek adott diák esetén a diáknak megfelelő termék forrásából a megfelelő termék nyelőjébe kell eljutnia egy, a két csúcs közti irányított úton, 1 folyamegységet szállítva. Definiáljuk diákonként a forrásokból az első középső oszlopba-, illetve minden középső oszlop-pár utolsó oszlopjából a következő oszlop-pár első oszlopába menő élek termék specifikus költségeit aszerint, hogy az adott terméknek megfelelő diák az adott következő napi műszakra való ráérésének módosított értéke $0,1$ vagy 1000. Minden nem meghatározott termék-él költség értéke legyen 0. Végül vegyük fel tényezőkként az adott műszakok hosszát óraszámban, jelen esetben $8, 6, 8$ és 12.
A következőkben az előző részben bemutatott *PuLP* optimalizációs modellező környezet segítségével írjuk fel a példa ütemezésére modellezett többtermékes folyam problémát lineáris programként. Ezen modellezést a következő lépések mentén tesszük:

- Létrehozunk egy üres lineáris programot a *prob* változóban; és paraméterként megadjuk neki, hogy egy minimalizáló feladatot szeretnénk implementálni.
- Létrehozunk egy-egy lineáris változót minden termék-él kombinációra; paraméterként jelezve ezen változók bináris voltát.
- Definiáljuk a minimalizálandó összköltséget, ahol minden termék-él kombinációnak megfelelő költség és folyammennyiség szorzatát összegezzük.
- Hozzáadjuk a lineáris programhoz a minden élre értelmezett kötegkorlát- vagy kapacitáskikötéseket a korábban definiált élkapacitások alapján.
- Hozzáadjuk a lineáris programhoz a minden termékre értelmezett kínálat-kikötéseket; melyek minden termék esetén a termék forrásának kínálatát 1-re-, nyelőjének kínálatát -1-re-, és G hálózat minden más csúcsának kínálatát 0-ra értékelik.
- Hozzáadjuk a lineáris programhoz azon erős megszorításokat, amelyek szabályozzák, hogy a tervezési periódus első 4 darab 7 napos időszakának mindegyikére teljesüljön, hogy minden diák egy adott hét napos perióduson belül legalább 2 napot pihen, és legalább 2 műszakot dolgozik, feltéve, hogy ráér egy héten 2 különböző nap egy-egy műszakjára. További erős megszorítás lesz az is, hogy 6 egymás utáni nap mindegyikén egy diák sem dolgozhat.
- Hozzáadjuk a lineáris programhoz azon erős megszorításokat, melyek szabályozzák, hogy a korábban definiált műszakhosszok segítégével kiszámolható, diákonként értelmezett összmunkaóra mennyisége a diák által megadott intervallum keretein belül helyezkedjen.
- Utolsó lépésként, az előző részben bemutatott *CBC* lineáris megoldóprogram segítségével megoldjuk a példa ütemezéséhez modellezett lineáris programot.

A kapott megoldás vizualizációjának céljából, a **(6.3.4)** ábrán az első diáknak megfelelő termék 1 folyamegységének átfolyását biztosító irányított út látható. Attól függően, hogy az adott termék által használt irányított út egyes éleinek költsége $0,1$ vagy 1000, az adott él kék, sárga vagy piros színű lehet.



Az első diák kapott beosztása G hálózaton. (6.3.4)

A példa megoldásának befejezéséhez a kapott megoldások alapján készítünk két táblázatot. Az egyik táblázat megmutatja minden műszak esetén, hogy az adott műszakot melyik diák fogja elvégezni. A második táblázat megmutatja minden diák esetén, hogy összesen hány órát fog dolgozni a következő hónapban a beosztás szerint. Az eredeti bemenet esetén kapott eredményül szolgáló táblázatokat a **(6.3.5)** ábrákon tekinthetjük meg.

Datum	De	Du	Na	E
1-Apr	Zsofi	Kata	Evi	Balazs
2-Apr	Zsofi	Panni	Evi	Kata
3-Apr	Reka	Hanna	Kitti	Peti
4-Apr	Panni	Zsofi	Kata	Peti
5-Apr	Marcsi	Reka	Kitti	Balazs
6-Apr	Kitti	Kinga	Kata	Adam
7-Apr	Marcsi	Kinga	Hanna	Adam
8-Apr	Kata	Kinga	Reka	Adam
9-Apr	Kata	Reka	Evi	Marcsi
10-Apr	Panni	Zsofi	Evi	Balazs
11-Apr	Panni	Evi	Kinga	Peti
12-Apr	Evi	Zsofi	Kitti	Balazs
13-Apr	Zsofi	Hanna	Kitti	Adam
14-Apr	Kinga	Marcsi	Hanna	Peti
15-Apr	Kata	Panni	Kinga	Peti
16-Apr	Panni	Evi	Reka	Peti
17-Apr	Panni	Zsofi	Kinga	Balazs
18-Apr	Zsofi	Evi	Kata	Marcsi
19-Apr	Hanna	Zsofi	Kitti	Adam
20-Apr	Panni	Reka	Kitti	Balazs
21-Apr	Marcsi	Hanna	Kitti	Adam
22-Apr	Kitti	Reka	Evi	Kinga
23-Apr	Zsofi	Panni	Evi	Peti
24-Apr	Evi	Marcsi	Kata	Balazs
25-Apr	Evi	Zsofi	Kinga	Balazs
26-Apr	Kitti	Hanna	Marcsi	Adam
27-Apr	Reka	Panni	Hanna	Adam
28-Apr	Kata	Kinga	Kitti	Peti
29-Apr	Kinga	Marcsi	Reka	Peti
30-Apr	Kinga	Reka	Kata	Marcsi

Kata	Kinga	Reka	Hanna	Kitti	Zsofi	Evi	Panni	Marcsi	Balazs	Peti	Adam
90	92	70	56	88	76	90	72	86	96	108	96

Eredeti bemenet szerinti eredmények. (6.3.5)

Az implementáció jelen állapotában, amennyiben létezik ezen erős megszorítások szerint értelmezett megengedett megoldás, a lineáris program eredménye egy olyan megengedett megoldás lesz, amely minimális számú kényelmetlen ráérést használ. A továbbiakban olyan változtatásokat végzünk az implementáción, amelyek a jelenlegi erős megszorítások szerint értelmezett megengedett megoldás hiányában egy egyébként optimális részmegoldás visszaadását biztosítják. A program használatának sebessége érdekében érdemes lehet először az eredeti implementációval próbálkozni; és csak akkor cserélni ezt le a következő verzióra, ha nem kaptunk megengedett megoldást.

A diákonként értelmezett összmunkaórára vonatkozó kikötéseket erős megszorításként tartjuk ezek után is, azzal a megjegyzéssel, hogy megengedett megoldás hiányában a menedzser saját meglátása szerint változtathatja az ezzel kapcsolatos adatokat a bemeneti táblázatban. Jelenleg a diákok egy-egy műszakra vonatkozó ráérése gyenge megszorításként van modellezve olyan módon, hogy az összköltség optimalizálása során, lehetőségekhez mérten, inkább fogunk olyan megengedett megoldást kapni, mely rengeteg kényelmetlen ráérést használ, minthogy egyetlen olyan ráérést is használjon, ami nem megfelelő egy adott diáknak.

A műszakok betöltéséről szóló erős megszorításokat, amelyeket eddig a kapacitáskikötések segítségével szabályoztunk, ezek után részben gyenge megszorításokként fogjuk kezelni. A G hálózatot ehhez úgy módosítjuk, hogy töröljük a korábban definiált, 8 értékű kapacitásokat minden pihenőműszaknak megfelelő élről. Ezáltal kötegkorlátokkal csupán azt biztosítjuk, hogy minden valós műszakban legfeljebb 1 diák dolgozik. A jelentésben erős-, de modellezésileg gyenge megszorítás azon részét, amely biztosítja, hogy minden valós műszakban pontosan 1 diák dolgozik; a következő célfüggvényhez adott részfüggvény segítségével biztosítjuk. Legyen egy adott napra a 8 darab megengedett pihenőműszakban levő diák feletti további diákok számának összege minden napra súlyozva 100-zal; jelezve azt, hogy ezen megszorítás megszegése kevésbé költséges a nem megfelelő ráérések használatánál, de költségesebb a többi gyenge megszorítás megszegésénél.

Végezetül igazságosságot szeretnénk bevezetni a diákok között a felhasznált kényelmetlen ráérések tekintetében. Kiszámítva minden diák esetén a felhasznált kényelmetlen ráéréseik és a kért kényelmetlen ráéréseik arányát, legyen az ezen gyenge megszorítás modellezéséhez használt igazságossági index a számolt arányok maximumának és minimumának különbsége. Ezt a legfeljebb 1 értékű indexet 1-es súllyal hozzáadva a célfüggvényhez jelezzük, hogy az igazságosság, mint optimalizációs szempont, a legkevésbé fontos az összes gyenge megszorítás közül.

Minden megengedett megoldás esetén megtekintve annak minimalizált célfüggvényértékét, pontosan, vagy szinte pontosan, láthatjuk, hogy egyes gyenge megszorításokból mennyit szegett meg az adott optimális megoldás.

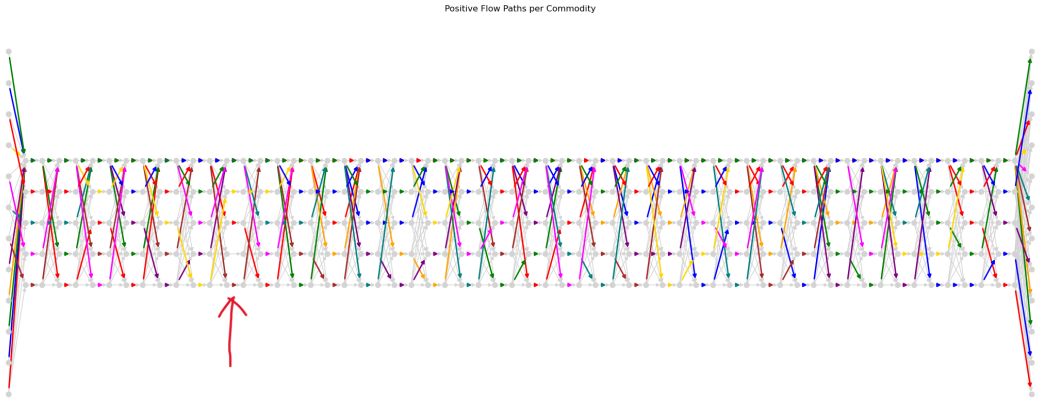
A következőkben két módosított bemenet esetén tekintjük meg az újonnan implementált funkciók hatását a megoldásra.

Először egy olyan esetet tesztelünk, ahol Kinga a hónap első 10 napjában szabadságon van; és április 7-én Réka sem elérhető. Ezen változtatásokkal azt tudjuk tesztelni, hogy, először is, a program jól kezeli egy diák egy hétnél hosszabb hiányzását; másodszor pedig, hogy, mint az a jelen esetben is lesz, a program visszaad egy egyébként optimális részmegoldást akkor is, ha minden műszak nem betölthető.

	Kata	Kata	Kata	Kata	Kinga	Kinga	Kinga	Kinga	Reka	Reka	Reka	Reka
	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E
1-Apr	I	T	I	I	N	N	N	N	N	T	N	N
2-Apr	I	T	I	I	N	N	N	N	I	I	I	N
3-Apr	I	T	I	I	N	N	N	N	N	N	N	N
4-Apr	I	T	I	I	N	N	N	N	N	I	N	N
5-Apr	I	T	I	I	N	N	N	N	I	N	N	N
6-Apr	I	T	I	I	N	N	N	N	N	T	T	N
7-Apr	I	T	I	I	N	N	N	N	N	N	N	N
8-Apr	I	T	I	I	N	N	N	N	N	I	N	N
9-Apr	I	T	I	I	N	N	N	N	I	I	I	N
10-Apr	I	T	I	I	N	N	N	N	N	N	N	N
11-Apr	I	T	I	N	I	N	I	N	I	N	N	N
12-Apr	N	N	N	N	N	I	I	I	N	N	N	N
13-Apr	I	T	I	I	I	I	I	I	N	I	T	N
14-Apr	I	T	I	I	I	I	I	I	I	I	I	N
15-Apr	I	T	I	I	I	I	I	I	I	I	I	N
16-Apr	I	T	I	I	I	I	I	I	N	I	I	N
17-Apr	I	T	I	I	N	N	N	N	I	N	N	N
18-Apr	N	N	N	N	N	N	N	N	N	N	N	N
19-Apr	N	N	N	N	N	I	I	I	N	I	T	N
20-Apr	N	N	N	N	I	I	I	I	N	N	N	N
21-Apr	N	N	N	N	I	I	I	I	N	I	N	N
22-Apr	N	N	N	N	I	I	I	I	N	I	N	N
23-Apr	T	I	I	I	I	I	I	I	I	I	I	N
24-Apr	I	T	I	I	I	I	I	I	N	N	N	N
25-Apr	I	T	I	I	I	N	N	N	N	I	N	N

Módosított bemeneti adatok limitált elérhetőség esetén. (6.3.6)

Ezen módosított bementi adatok esetén április 7-én Kata az ábrán látható módon-, Zsófi csak a délelőtti-, további diákok pedig csak az éjszakai műszakra elérhetők. A következő ábrán láthatjuk, ahogy ezen nap valós műszakjainak megfelelő élek egyikén egyetlen termék esetén sem folyik át pozitív mennyiségű folyam.



Az összes pozitív folyamértékkel rendelkező irányított út G hálózaton. (6.3.7)

A kapott beosztást tartalmazó táblázatnak a következő ábrán látható része alapján azt is észrevehetjük, hogy a kapott részmegoldás valóban egyébként optimális, hiszen olyan részmegoldást kaptunk, amiben Kata nem egy számára kényelmetlen ráérésben dolgozik.

Datum	De	Du	Na	E
1-Apr	Zsofi	Panni	Evi	Balazs
2-Apr	Kata	Reka	Evi	Balazs
3-Apr	Panni	Zsofi	Kata	Marcsi
4-Apr	Kata	Reka	Kitti	Peti
5-Apr	Kitti	Zsofi	Marcsi	Balazs
6-Apr	Marcsi	Kitti	Panni	Adam
7-Apr	Zsofi		Kata	Marcsi
8-Apr	Kata	Panni	Evi	Marcsi
9-Apr	Reka	Zsofi	Evi	Peti
10-Apr	Zsofi	Hanna	Kata	Peti
11-Apr	Evi	Kitti	Kinga	Adam
12-Apr	Kitti	Hanna	Kinga	Adam
13-Apr	Hanna	Kitti	Marcsi	Peti
14-Apr	Evi	Panni	Reka	Peti

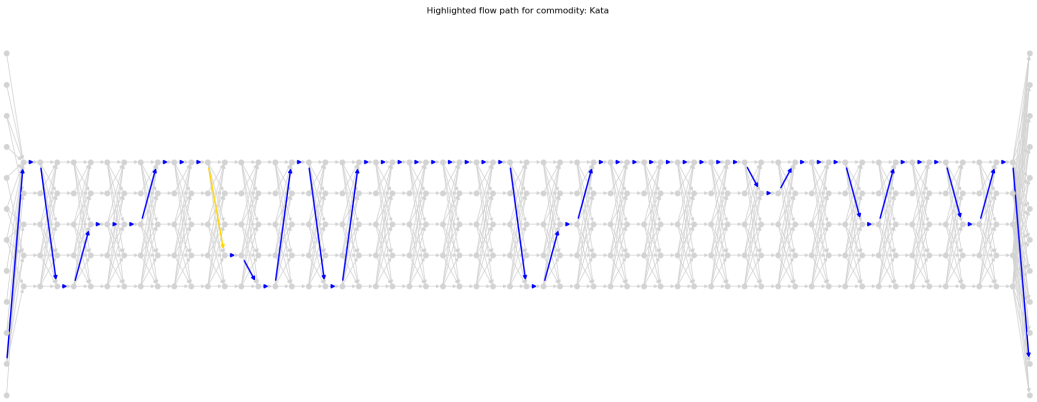
Kapott megoldás limitált elérhetőség esetén. (6.3.8)

Másodszor egy olyan esetet tesztelünk, ahol az eredeti ráéréshez viszonyítva Kata és Kinga ráérési adatai fordított sorrendben kerültek a bemeneti táblázatba, továbbá az április 7-ei délutáni műszakra mindketten csak kényelmetlenül érnek rá. A módosított bemenet **(6.3.9)** ábrán látható.

	Kinga	Kinga	Kinga	Kinga	Kata	Kata	Kata	Kata	Reka	Reka	Reka	Reka
	De	Du	Na	E	De	Du	Na	E	De	Du	Na	E
1-Apr	N	N	N	N	I	T	I	I	N	T	N	N
2-Apr	N	N	N	N	I	T	I	I	I	I	I	N
3-Apr	N	N	N	N	I	T	I	I	N	N	N	N
4-Apr	N	N	N	N	I	T	I	I	N	I	N	N
5-Apr	N	I	I	I	I	T	I	I	I	N	N	N
6-Apr	I	I	I	I	I	T	I	I	N	T	T	N
7-Apr	I	T	I	I	I	T	I	I	N	N	I	N
8-Apr	I	I	I	I	I	T	I	I	N	I	N	N
9-Apr	I	I	I	I	I	T	I	I	I	I	I	N
10-Apr	I	I	I	I	I	T	I	I	N	N	N	N
11-Apr	I	N	I	N	I	T	I	N	I	N	N	N
12-Apr	N	I	I	I	N	N	N	N	N	N	N	N
13-Apr	I	I	I	I	I	T	I	I	N	I	T	N
14-Apr	I	I	I	I	I	T	I	I	I	I	I	N

Módosított bemeneti adatok igazságosság tesztelésére. (6.3.9)

Az eredeti implementáció ezen módosított bemenetre egy 1 célfüggvényértékű optimális megoldást ad vissza, amelyben Kinga dolgozik kényelmetlen ráérésben. Ehhez képest a második implementáció, amely kezeli a diákok közti igazságosságot a kényelmetlen ráérések felhasználásának tekintetében, Katának adja a kényelmetlen műszakot. Ennek az az oka, hogy Kata több kényelmetlen ráérést kért; így abban az esetben, ahol az ő kényelmetlen ráéréseiből számolt arány az egyedüli pozitív tagja az arányok sorozatának, a korábban definiált igazságossági index értéke kisebb, mint abban az esetben, ahol Kinga kapja a kényelmetlen műszakot. A **(6.3.10)** ábrán Kata beosztása látható a G hálózaton.



Kata beosztása G hálózaton igazságosság figyelembevételével. (6.3.10)

7. fejezet

Programkód- és adatforrás

A **6.3** részben bemutatott gyakorlati alkalmazás implementációja *python* nyelven készült. A példához használt programkód, illetve a példa bemeneti adataiként és eredményeiként szolgáló `.csv` fájlok elérhetők nyilvánosan egy *Google Drive* mappában.

A programkód implementációja a *Visual Studio Code* szerkesztőben valósult meg, *python* 3.12.4-es verzióval, *Jupyter Notebook* formátumban. A felhasznált főbb csomagok a *pandas*, *NetworkX* és *PuLP* könyvtárak voltak. Az adatok tárolása és kezelése során egyszerű *.csv* formátum került használatra.

A *Google Drive* mappa elérhetősége:

`https://drive.google.com/drive/folders/1MWuFwewrNoK8cy9nXp4BfxhahAnO9XmN?usp=sharing`

A *Google Drive* mappa tartalmazza:

- a bemeneti adatkészleteket;
- a bemeneti adatok előzetes kezelését elvégző *python* programkódot;
- a lineáris programként implementált *python* nyelvű többtermékes folyammodelt;
- az eredményekként szolgáló adatkészleteket;
- a rövid *README* leírást a fájlok használatáról.

A programkódok implementációjában a *NetworkX* gráfkönyvtár dokumentációja **[11]** és a *PuLP* modellező környezet dokumentációja **[14]** nyújtott segítséget.

Hivatkozások

[1] Mieke Defraeye and Inneke Van Nieuwenhuyse; *Staffing and scheduling under nonstationary demand for service: A literature review*; Omega, Volume 58, January 2016, Pages 4-25.

[2] A.T. Ernst, H. Jiang, M. Krishnamoorthy, D. Sier; *Staff scheduling and rostering: A review of applications, methods and models*; European Journal of Operational Research 153 (2004) 3–27.

[3] Ravindra K. Ahuja, Thomas L. Magnanti, James B. Orlin; *Network Flows: Theory, Algorithms, and Applications*; Prentice-Hall, 1993, Upper Saddle River, New Jersey 07458.

[4] Peter Brucker, Rong Qu; *Network flow models for intraday personnel scheduling problems*; Annals of Operations Research (2014) 218:107–114.

[5] M. Segal; *The Operator-Scheduling Problem: A Network-Flow Approach*; Bell Telephone Laboratories, Holmdel, New Jersey (Received July 17, 1972).

[6] Alexander Schrijver; *A Course in Combinatorial Optimization*; 2017, CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands.

[7] Zsolt Ercsey, Zoltán Kovács; *Multicommodity network flow model of a human resource allocation problem considering time periods*; Central European Journal of Operations Research (2024) 32:1041–1059.

[8] Margarida Moz and Margarida Vaz Pato; *Solving the Problem of Rerostering Nurse Schedules with Hard Constraints: New Multicommodity Flow Models*; Annals of Operations Research 128, 179–197, 2004, Kluwer Academic Publishers.

[9] Amitabha Bagchi, Amitabh Chaudhary, Petr Kolman, Jiri Sgall; *A Simple Combinatorial Proof of Duality of Multiroute Flows and Cuts*; Charles Univ., 2004.

[10] Charu C. Aggarwal, James B. Orlin; *On Multiroute Maximum Flows in Networks*; Networks: An International Journal 39.1 (2002): 43-52.

[11] NetworkX Developers; *Flows - NetworkX 3.4.2 documentation*; `https://networkx.org/documentation/stable/reference/algorithms/flow.html`, Copyright 2004-2024.

[12] The igraph Core Team; *igraph Reference Manual For using the igraph C library*; `https://igraph.org/c/html/latest/igraph-Flows.html`, Copyright 2003-2025.

[13] Egerváry Research Group on Combinatorial Optimization at the Operations Research Department of the Eötvös Loránd University, Budapest, Hungary; *LEMON 1.3.1 Documentation*; `http://lemon.cs.elte.hu/pub/doc/1.3.1/modules.html`, Copyright 2004-2024.

[14] Pulp Documentation Team; *Optimization with PuLP - PuLP 3.0.2 documentation*; https://coin-or.github.io/pulp/main/includeme.html#more-examples, Copyright 2009-.

[15] Bynum, Michael L., Gabriel A. Hackebeil, William E. Hart, Carl D. Laird, Bethany L. Nicholson, John D. Siirola, Jean-Paul Watson, and David L. Woodruff; *Pyomo - Optimization Modeling in Python*; 3rd Edition. Springer, 2021; https://pyomo.readthedocs.io/en/stable/, Sandia National Laboratories, Copyright 2008-2025.

[16] AMPL Optimization Inc.; *Modeling guide for MP-based AMPL solvers*; https://mp.ampl.com/model-guide.html, Copyright 2015-2025.

[17] John Forrest, Robin Lougee-Heimer; *CBC User Guide*; https://www.coin-or.org/Cbc/cbcuserguide.html, Copyright 2005 IBM Corporation.

[18] Gurobi Optimization, LLC; *Gurobi Optimizer Reference Manual*; https://docs.gurobi.com/projects/optimizer/en/current/, Copyright 2025.

[19] IBM Corporation; *IBM(R) ILOG CPLEX Python API Reference Manual*; https://www.ibm.com/docs/en/icos/22.1.2?topic=optimizers-cplex-python-api-reference-manual, Copyright 1987-2024 IBM Corporation.

[20] Q. Huangfu and J. A. J. Hall; *Parallelizing the dual revised simplex method*; Mathematical Programming Computation, 10 (1), 119-142, 2018. DOI: 10.1007/s12532-017-0130-5 https://ergo-code.github.io/HiGHS/dev/.

[21] Zuse Institute Berlin (ZIB); *SCIP Doxygen Documentation*; https://scipopt.org/scip/doc/html/index.php, Copyright 2025.

[22] Google OR-Tools Developers; *About OR-Tools - Google for Developers*; https://developers.google.com/optimization/introduction, Copyright 2024.

Mesterséges intelligencia alapú eszközök használatáról szóló nyilatkozat

Alulírott *Jánó Róbert* nyilatkozom, hogy szakdolgozatom elkészítése során az alább felsorolt feladatok elvégzésére a megadott mesterséges intelligencia alapú eszközöket alkalmaztam:

Feladat	**Felhasznált eszköz**	**Felhasználás helye**	**Megjegyzés**
LaTeX szintaktika	GPT-4o	Teljes dolgozat	Lineáris programok felírása, formai követelmények teljesítése
Ábra készítés	GPT-4o	2. és 3. fejezet ábrái, 4. fejezet első ábrája	—
Adatfeldolgozás és lineáris modellezés	GPT-4o	6.3 fejezet	Python kód vázlat
Fogalmazáshelyesség ellenőrzése	GPT-4o	Bevezetés, 6.1 fejezet, 7. fejezetek	—
Nyelvhelyesség ellenőrzése	Writefull	Teljes dolgozat	—

A felsoroltakon túl más mesterséges intelligencia alapú eszközt nem használtam a dolgozat elkészítése során.

Dátum: 2025. május 24. Aláírás: Jánó Róbert