Eötvös Loránd Tudományegyetem
Természettudományi Kar

Marosvári Ágnes

Robusztus optimalizálás

Diplomamunka
Alkalmazott matematikus MSc

Témavezető:

Kis Tamás

Operációkutatási Tanszék

Budapest, 2021

NYILATKOZAT

Név: Marosvári Ágnes

ELTE Természettudományi Kar, szak: Alkalmazott matematikus MSc

NEPTUN azonosító: T6HL0V

Diplomamunka címe:
Robusztus optimalizálás

A **diplomamunka** szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló szellemi alkotásom, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.

Budapest, 2021.12.15.

Marosvári Ágnes

a hallgató aláírása

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, Kis Tamásnak, hogy megismertette velem ezt a témát, hogy mindig tudott időt szakítani rám vagy arra, hogy segítsen kibogarászni, miért is nem úgy működik a programom, mint szeretném, továbbá a remek pedagógiai érzékéért, amivel mindig tudott motiválni. Köszönöm a rengeteg segítséget!

Tartalomjegyzék

1. Bevezetés **4**

2. Linearizálható robusztus modellek **6**

2.1. A Bertsimas-Sim modell 6

2.2. Poliéderes bizonytalanság 9

2.3. Egy másik megközelítés: vágásgenerálás 10

3. Többszintű lineáris programozás **13**

3.1. A feladat 13

3.2. A polinomiális hierarchia 16

3.3. A feladat nehézsége 18

4. Kétfázisú optimalizálás **24**

4.1. A feladat 24

4.2. A feladat számítási bonyolultsága 25

4.3. Oszlop- és vágásgeneráló algoritmus 27

4.4. Egy robusztus p-median modell 31

1. fejezet

Bevezetés

Amikor lineáris programozási feladatokkal foglalkoztam a tanulmányaim során, természetesnek vettem, hogy költségfüggvény vagy a feltételeket leíró mátrix és korlátozóvektor elemei pontosan adottak. A való életbeli optimalizálási problémák adatai azonban gyakran tartalmazhatnak bizonytalanságot. Ez a bizonytalanság számos különböző okból eredhet, ez lehet mérési hiba, vagy egyszerűen az, hogy azt az adatot nem is lehet pontosan felvenni (például egy hozzzárendelési feladat esetén nem feltétlenül lehet előre tudni, hogy egyes termékekre mekkora lesz a kereslet), de az is előfordulhat, hogy a megoldás megengedettsége függ a jövőtől is (ismét a hozzárendelési feladatot példaként használva, egy sztrájk vagy természeti katasztrófa megbéníthat egy kiszolgáló pontot). A bizonytalanság kezelésére alapvetően kétféle megközelítés létezik: a sztochasztikus és a robusztus optimalizálás. Az előbbi módszer a valószínűségszámításon alapul, várható értékben optimalizálunk, és a megoldástól azt várjuk, hogy elég nagy valószínűséggel ne sértse meg az előírt feltételeket. A robusztus optimalizálásnál viszont olyan megoldást keresünk, amely megengedett marad, bármely bizonytalan eset is következzen be (a lehetséges szcenáriók halmazát ismerjük előre), és ezek közül a lehető legjobbat szeretnénk kiszámítani. Azonban a két szempont, azaz a megengedettség és az optimalitás, egymás ellen hathat. Egy olyan megoldás, amely a legvalószínűtlenebb esetekben is megengedett marad, nagyon messzire eshet attól a megoldástól, ami a bizonytalan értékek kiderülése után optimális lenne, hiszen sok korlátot "feleslegesen" teljesít.
Szakdolgozatom célja bemutatni az operációkutatásnak ezt a viszonylag fiatal területét. A 2. fejezet bevezető jellegű, a legegyszerűbb robusztus modellekről lesz itt szó, ahol a korlátokat leíró mátrix elemeinek intervallumos bizonytalanságát fogjuk tekinteni. Megismerünk egy módszert arra, hogyan lehet a fentebb említett jelenséget (amikor is a megoldás túlságosan konzervatív) elkerülni, illetve belátjuk, hogy az így kapott modell átírható lineáris programozási feladattá. Ezek után az átalakítás ötletét felhasználva bebizonyítom, hogy intervallumos bizonytalanság helyett poliéder bizonytalanságot is tekinthetünk, és a feladat linearizálható marad. Az

alfejezetet egy vágásgeneráló algoritmussal zárom, majd az alfejezetben megismert módszereket implementálva összehasonlítom a különféle megközelítések hatékonyságát.
A 3. fejezet a többszintű lineáris programozási modellt mutatja be, amelynél teljesen természetesen merül fel a robusztusság kérdése. Az ilyen modellek a hierarchikus döntéshozást kívánják modellezni, a változók partícionálva vannak, és ezekről a játékosok a köztük lévő hierarchia szerinti sorrendben hoznak döntést a saját célfüggvényüket szem előtt tartva. Ennek a fejezetnek alapvetően az a célja, hogy demonstrálja a robusztus modellek hasznosságát a mindennapi életben, ezt egy kétszintű modell ismertetésével fogom megtenni. Ezen kívül ez a modell jó alkalmat ad arra, hogy bemutathassam, általában a robusztus megoldás kiszámítása nehéz feladat. A nehézségi bizonyítás egy nagyon érdekes konstrukciót használ, ehhez azonban szükség lesz a polinomiális hierarchia ismertetésére, a fejezetben erről is lesz szó röviden.
A 4. fejezetben visszatérünk a robusztus megoldás konzervatív tulajdonságára, és ezt kívánjuk javítani egy kétfázisú modell segítségével. Ebben a változók halmazát kettéválasztjuk, vannak változók amelyekről a bizonytalanság kiderülése előtt hozunk döntést, és vannak, amelyekről utána, így tudunk újratervezni. Az így kapott modell nem meglepő módon ismét egy nehéz feladathoz vezet, ez be is látom egy saját bizonyítással. A megoldást így csak egy iterációs algoritmus segítségével tudjuk keresni, az alfejezet további részében ezt mutatom be, és egészítem ki azon kérdések megválaszolásával, amelyek az implementálás folyamán felmerülhetnek. Ezután bemutatok egy robusztus p-median modellt, és ezt felhasználva alkalmazom az algoritmust egy valóéletbeli problémára.

2. fejezet

Linearizálható robusztus modellek

Ebben a fejezetben kettő olyan modell fog szerepelni, amely könnyen kezelhető, azaz átírható lineáris programozási feladattá. Mindkét esetben soronkénti bizonytalanságot fogunk tekinteni. Az első esetben [1] alapján egy olyan feladatot mutatok be, ahol azt tudjuk, hogy a feladathoz tartozó A mátrix elemei egy bizonyos előre meghatározott intervallumba esnek. Belátom azt is, hogy miért elegendő csak az A mátrix bizonytalanságát feltételezni. A második az első eset egy általánosítása lesz, ahol a bizonytalan értékek nem egy intervallumból, hanem egy poliéderből kerülnek ki, itt megadom a modell linearizált alakját. Ezek után [2] alapján egy olyan algoritmust is bemutatok, amelyhez nem szükséges ezen modellek linearizálása, és összehasonlítom a kétféle megközelítés hatékonyságát.

2.1. A Bertsimas-Sim modell

Tekintsük az alábbi lineáris programozási feladatot, ahol $A \in \mathbb{R}^{n\times m}, b \in \mathbb{R}^n$ és $c \in \mathbb{R}^m$ a bemeneti adatok nominális értéke, amelyek azonban bizonytalanságot is tartalmazhatnak, ezeket $\tilde{A}, \tilde{b}$ és $\tilde{c}$ jelöli.

$$\max \tilde{c}x \tag{2.1}$$

$$\tilde{A}x \leq \tilde{b} \tag{2.2}$$

Valójában feltehetjük, hogy a b és c vektorok pontosan ismertek, és ez nem jelent megszorítást a feladatra nézve.

2.1.1. Állítás. *Feltehető, hogy csak az A mátrix elemei tartalmaznak bizonytalanságot.*

Bizonyítás. Tegyük fel, hogy a b vektor tartalmaz bizonytalan koordinátát is, legyen ez $\tilde{b}_i$. Egy új $z \in \mathbb{R}$ változót vezetünk be, és ezzel már olyan alakra hozhatjuk az i-edik sort, hogy csak a baloldalon szerepelnek bizonytalan adatok. Legyen az új

feltétel $\tilde{a}_i x - \tilde{b}_i z \leq 0$, és rögzítsük z-t 1-nek.
Ha a c vektorban szerepelnek bizonytalan együtthatók, akkor ismét vegyünk fel egy $z \in \mathbb{R}$ változót, és azt a feltételt, hogy $z - \tilde{c}x \leq 0$, a célfüggvényt pedig cseréljük le $\max z$-re. ■

Ezek után a következőképpen modellezzük az A mátrixban lévő bizonytalanságot. Vegyük a mátrix i-edik sorát, és legyen J_i azon a_{ij} elemeknek a halmaza, amelyek bizonytalanságot tartalmaznak. Ezeket $\tilde{a}_{ij}$ jelöli, és azt tudjuk, hogy $\tilde{a}_{ij}$ az $[a_{ij} - \hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$ intervallumból vesz fel értéket, ahol tehát a_{ij} a nominális érték és $\hat{a}_{ij}$ előre ismert. A cél az, hogy olyan megoldást találjunk, amely tetszőleges $\tilde{a}_{ij}$ adatok esetén megoldás marad. Tekintsük a következő modellt.

$$\max cx \tag{2.3}$$

$$\sum_j a_{ij}x_j + \sum_{j \in J_i} \hat{a}_{ij} y_j \leq b_i \quad \forall i \tag{2.4}$$

$$-y_j \leq x_j \leq y_j \tag{2.5}$$

$$y \geq 0 \tag{2.6}$$

Legyen (x^*, y^*) optimális megoldása (2.3)-(2-6)-nak, ekkor $y_j^* = |x_j^*|$, és az alábbi feltétel teljesül:

$$\sum_j a_{ij}x_j^* + \sum_{j \in J_i} \hat{a}_{ij}|x_j^*| \leq b_i \quad \forall i.$$

2.1.2. Állítás. *Tetszőleges $\tilde{a}_{ij}$ adatok esetén x^* megengedett megoldása (2.1)-(2.2)-nek.*

Bizonyítás. Legyen $\eta_{ij} = \frac{\tilde{a}_{ij} - a_{ij}}{\hat{a}_{ij}}$, ekkor $\eta_{ij} \in [-1, 1]$, továbbá

$$\begin{aligned}\sum_j \tilde{a}_{ij}x_j^* &= \sum_j a_{ij}x_j^* + \sum_{j \in J_i} \eta_{ij}\hat{a}_{ij}x_j^* \\ &\leq \sum_j a_{ij}x_j^* + \sum_{j \in J_i} \hat{a}_{ij}|x_j^*| \leq b_i \quad \forall i.\end{aligned}$$

■

Tehát az ilyen módon felírt modellből kapott megoldás valóban robusztus, azonban egyben nagyon konzervatív is, azaz sokkal rosszabb lehet, mint a nominális adatok mellett számolt optimum. Nem biztos ugyanis, hogy az összes bizonytalanságot tartalmazó adat megváltozik, és hogy a legrosszabb értékét veszi fel. Ahhoz, hogy a modell konzervatívsága szabályozható legyen, bevezetünk $\Gamma_i \in \mathbb{R}$ $(i = 1, ..., n)$ paramétereket, amely a $[0, |J_i|]$ intervallumból vesz fel értéket, és azt mondja meg, hogy egy sorban hány adat megváltozására számítunk. A cél az, hogy olyan megoldást

találjunk, ami az összes olyan esetben megoldás marad, amikor legfeljebb Γ_i adat változik meg. Tekinstük a következő modellt.

$$\max cx \tag{2.7}$$

$$\sum_j a_{ij}x_j + \max_{\Omega_i}\{\sum_{j\in S_i} \hat{a}_{ij}y_j + (\Gamma_i - \lfloor\Gamma_i\rfloor)\hat{a}_{it_i}y_{t_i}\} \leq b_i \quad \forall i \tag{2.8}$$

$$-y_j \leq x_j \leq y_j \quad \forall j \tag{2.9}$$

$$y \geq 0, \tag{2.10}$$

ahol $\Omega_i = \{S_i \cup \{t_i\} : S_i \subseteq J_i, |S_i| = \lfloor\Gamma_i\rfloor, t_i \in J_i \setminus S_i\}$. A modell linearizálásához a következő definíció és segédállítás szükséges.

2.1.3. Definíció. *Legyen x^* tetszőleges vektor, ekkor az i-edik feltétel teljesülését biztosító tagot jelölje a $\beta_i(x^*, \Gamma_i)$ függvény, azaz*

$$\beta_i(x^*, \Gamma_i) = \max_{\Omega_i}\{\sum_{j\in S_i} \hat{a}_{ij}|x_j^*| + (\Gamma_i - \lfloor\Gamma_i\rfloor)\hat{a}_{it_i}|x_{t_i}^*|\}.$$

2.1.4. Állítás. *Tetszőleges x^* vektor esetén*

$$\beta_i(x^*, \Gamma_i) = \max_z \sum_{j\in J_i} \hat{a}_{ij}|x_j^*|z_{ij} \tag{2.11}$$

$$\sum_{j\in J_i} z_{ij} \leq \Gamma_i \tag{2.12}$$

$$0 \leq z_{ij} \leq 1 \quad \forall j \in J_i \tag{2.13}$$

Bizonyítás. Az optimális megoldásban $\lfloor\Gamma_i\rfloor$ darab változónak 1 az értéke, és egy változónak pedig $\Gamma_i - \lfloor\Gamma_i\rfloor$. Ez pedig megfelel egy Ω_i-beli halmaznak $\sum_{j\in S_i} \hat{a}_{ij}|x_j^*| + (\Gamma_i - \lfloor\Gamma_i\rfloor)\hat{a}_{it_i}|x_{t_i}^*|$ célfüggvényértékkel. ■

2.1.5. Állítás. *A (2.7)-(2.10) modellnek létezik ekvivalens LP felírása.*

Bizonyítás. Vegyük a (2.11)-(2.13) feladat duálisát, legyen p_i a (2.12) feltételhez tartozó és q_{ij} ($j \in J_i$) a (2.13) feltételekhez tartozó duál változó.

$$\min_{p,q} p_i\Gamma_i + \sum_{j\in J_i} q_{ij} \tag{2.14}$$

$$p_i + q_{ij} \geq \hat{a}_{ij}|x_j^*| \quad \forall j \in J_i \tag{2.15}$$

$$p_i \geq 0 \tag{2.16}$$

$$q_{ij} \geq 0 \quad \forall j \in J_i \tag{2.17}$$

Mivel (2.11)-(2.13) megengedett és az optimum véges, így a dualitás tételből következik, hogy a duális is megengedett és az optimuma véges, ami megegyezik $\beta_i(x^*, \Gamma_i)$ értékével. Ezt behelyettesítve az eredeti modellbe megkapjuk a keresett felírást.

$$\max cx \tag{2.18}$$

$$\sum_j a_{ij}x_j + p_i\Gamma_i + \sum_{j\in J_i} q_{ij} \leq b_i \quad \forall i \tag{2.19}$$

$$p_i + q_{ij} \geq \hat{a}_{ij}y_j \quad \forall i, j \in J_i \tag{2.20}$$

$$-y_j \leq x_j \leq y_j \quad \forall j \tag{2.21}$$

$$p_i \geq 0 \quad \forall i \tag{2.22}$$

$$q_{ij} \geq 0 \quad \forall i, j \in J_i \tag{2.23}$$

$$y_j \geq 0 \quad \forall j \tag{2.24}$$

■

2.2. Poliéderes bizonytalanság

Ebben a modellben olyan típusú korlátok szerepelnek, amelyek azt írják elő, hogy egy P_i poliédernek bármely a_i vektorát véve teljesüljön, hogy $a_i x \leq b_i$ $\forall i = 1, ..., n$. Az ilyen korlátok is linearizálhatók, és ezt elegendő $i = 1$ esetén belátni. Világos, hogy az $ax \leq b \quad \forall a \in P$ feltétel ekvivalens azzal, hogy $\max\limits_{a\in P} ax \leq b$. Ezek után tekintsük a következő modellt, ahol $c \in \mathbb{R}^m, a \in \mathbb{R}^m, b \in \mathbb{R}, D \in \mathbb{R}^{k\times m}$ és $d \in \mathbb{R}^k$.

$$\max cx \tag{2.25}$$

$$0 \leq x \leq 1 \tag{2.26}$$

$$\max_{a\in P} ax \leq b \tag{2.27}$$

$$P = \{a : Da \leq d\} \tag{2.28}$$

Az alábbi példa azt szemlélteti, hogy egy egészértékű feladat esetén a bizonytalan feltételek miatt előfodulhat, hogy csak az azonosan nulla vektor megengedett megoldás, ha azonban relaxáljuk az egészértékű feltételt, a bizonytalanság ellenére is lesz nem nulla tört megoldás.

2.2.1. Példa.

$$\max c_1x_1 + c_2x_2 \tag{2.29}$$

$$\max_{a\in P} a_1x_1 - a_2x_2 \leq 1 \tag{2.30}$$

$$x_i \in \{0,1\} \quad i = 1,2 \tag{2.31}$$

$$P = [1,2] \times [-2,0] \tag{2.32}$$

Látható, hogy csak az $x = 0$ vektor esetén garantálható, hogy a (2.30) feltétel nem sérül. Ha azonban a (2.31) feltételt relaxáljuk, és csak azt írjuk elő, hogy $0 \leq x_i \leq 1$, akkor például az $x_i = \frac{1}{4}$ $(i = 1,2)$ is megengedett megoldás már.

2.2.2. Állítás. *A (2.25)-(2.28) modellnek létezik ekvivalens LP felírása.*

Bizonyítás. Tekintsük a bizonytalanságot adó poliéderre vonatkozó maximalizálási problémát (azaz a (2.27) és (2.28) feltételeket), és írjuk fel ennek a duálisát y duál változókkal.

$$\min yd \tag{2.33}$$

$$yD = x \tag{2.34}$$

$$y \geq 0 \tag{2.35}$$

A dualitás tétel alapján $\max ax = \min yd$, így ezt behelyettesítve megkapjuk a keresett felírást.

$$\max cx \tag{2.36}$$

$$0 \leq x \leq 1 \tag{2.37}$$

$$yd \leq b \tag{2.38}$$

$$yD = x \tag{2.39}$$

$$y \geq 0 \tag{2.40}$$

■

2.3. Egy másik megközelítés: vágásgenerálás

Az eddig látott modellek nemlineáris tulajdonsága azonban máshogyan is kezelhető. Tekintsük az alábbi felírást, ahol $U_i \subseteq \mathbb{R}^m$ tetszőleges bizonytalan halmaz, így ez

magába foglalja a 2.1 és a 2.2 alfejezetben szereplő feladatokat.

$$\max cx \tag{2.41}$$

$$\tilde{a}_i x \leq b_i \quad \forall \tilde{a}_i \in U_i,\ i = 1, ..., n \tag{2.42}$$

Az eddigi jelölést megtartva legyen J_i az i-edik sor bizonytalan elemeinek halmaza. A linearizálás helyett egy vágásgeneráló algoritmus segítségével is kereshető robusztus megoldás. Az algoritmus ötletét az adja, hogy bár a bizonytalan halmazok miatt nagyon sok feltétel van, ezeknek csak egy kis részhalmaza lesz végül az, amelyik korlátozza a megoldást. Így elegendő az ezek közül szükségeseket a megoldás keresése közben előállítani, amivel az alábbi eljáráshoz jutunk.

Vágásgeneráló algoritmus

1: Inicializáljuk a mesterproblémát (MP) a nominális a_{ij} adatokkal.
2: Oldjuk meg MP-t, az optimális megoldás legyen x^*.
3: **for** $i \leftarrow 1$ **to** n ahol $J_i \neq \emptyset$ **do**
4: Legyen $\bar{a} = \arg\max_{\tilde{a} \in U_i} \tilde{a}x$
5: Ha $\bar{a}x^* > b_i$, akkor adjuk az $\bar{a}x \leq b_i$ feltételt MP-hez.
6: **end for**
7: Ha nem keletkezett új feltétel, akkor x^* optimális, STOP.
8: Lépjünk a 2. lépéshez.

Az algoritmus hatékonyságát egy olyan hátizsák feladaton teszteltem, ahol n darab hátizsák van egyenként b_i $(i = 1, ..., n)$ kapacitással és m darab tárgy egyenként c_j $(j = 1, ..., m)$ értékkel. A bizonytalanság a tárgyak súlyában van, a j-edik tárgy súlya az i-edik hátizsákban legyen $\tilde{w}_{ij}$, amely a $[(1-p)w_{ij}, (1+p)w_{ij}]$ intervallumból vesz fel értéket, ha a nominális értéke w_{ij} és $p \geq 0$ a megváltozás mértékét jelölő paraméter. A hátizsák feladatot az alábbi korlátok írják le.

$$\max \sum_{j=1}^{m} c_j (\sum_{i=1}^{n} x_{ij}) \tag{2.43}$$

$$\sum_{j=1}^{m} \tilde{w}_{ij} x_{ij} \leq b_i \quad \forall i = 1, ..., n \tag{2.44}$$

$$\sum_{i=1}^{n} x_{ij} \leq 1 \quad \forall j = 1, ..., m \tag{2.45}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \tag{2.46}$$

A Bertsimas-Sim modellt használtam, azaz a (2.44) feltételeknél minden i esetén rögzítettem egy Γ_i értéket, hogy legfeljebb hány adat változik meg az adott sorban. Az implementálást FICO Xpress + Mosel környezetben végeztem, és összehasonlításképpen elkészítettem még két további programot is, az egyikben a 2.1.5 állításban

szereplő linearizált alakot használtam, míg a másikban a w_{ij}-ket *uncertain* típusú változókként adtam meg a megfelelő korlátokkkal, így az Xpressre bíztam, hogy megkeresse a legrosszabb esetet. Tapasztalataim szerint az *uncertain* változókat használó megoldás és a linearizálás hasonlóan jól teljesített, a vágásgenerálás ezektől pedig jelentősen lemaradt a vizsgált tesztfeladatokon. A használt paraméterek és az így kapott futásidők a 2.1 táblázatban láthatóak. Ahol a program nem találta meg az optimumot legfeljebb 120 másodperc alatt, ott T szerepel.

$n \times m$	p	Γ_i	Linearizált	Uncertain	Vágásgenerálás
10×30	1	1	0.075	0.057	0.083
		5	0.105	0.056	0.253
		15	0.049	0.043	0.223
	5	1	0.059	0.039	0.592
		5	0.088	0.041	4.637
		15	0.072	0.043	2.352
	10	1	0.044	0.053	2.317
		5	0.664	0.041	23.775
		15	0.085	0.033	0.432
50×150	1	1	0.568	1.933	1.791
		5	0.81	1.725	12.161
		75	0.872	1.837	1.313
	5	1	0.897	0.326	20.156
		5	2.198	0.308	T
		75	1.593	0.384	T
	10	1	1.314	0.208	35.05
		5	2.61	0.205	T
		75	1.183	0.198	T
100×300	1	1	2.162	11.75	13.385
		5	3.537	7.518	T
		150	4.063	8.683	5.484
	5	1	2.522	4.275	T
		5	7.692	4.292	T
		150	6.58	3.293	T
	10	1	2.098	1.027	T
		5	66.89	0.846	T
		150	4.51	0.871	T

2.1. táblázat. Futásidők (másodpercben)

3. fejezet

Többszintű lineáris programozás

A továbbiakban kettő olyan modellt fogok bemutatni, amelyek sokkal nehezebben kezelhetők, mint a 2. fejezetben szereplő linearizálható modellek. Ebben a fejezetben a többszintű lineáris modellekről lesz szó, ezen belül egy kétszintű vagy bilevel optimalizálási feladat fog részletesen szerepelni, illetve [3] alapján látni fogjuk, mely nehézségi osztályban is helyezkedik el a feladat.

3.1. A feladat

A valóéletben gyakran előfordulhat, hogy egy optimalizálási feladat hierarchikus szerkezetű. Gondolhatunk itt állami döntéshozásra, egy tetszőleges cég menedzselésére vagy éppen közlekedéstervezésre, a közös bennük, hogy mindegyik esetén egy központi, magasabb szinten meghozott döntést és az arra adott lentebbi szintek reakcióját szeretnénk vizsgálni. Ezt egy többszintű lineáris modell felállításával tehetjük meg. A lineáris programozástól eltérő módon itt nem egy személy dönt minden változóról, hanem ezek L részre lesznek partícionálva, és L különböző játékos dönt egymás után sorban a saját partíciójába eső változókról. Először az L-edik játékos választ, és előírjuk, hogy minden játékos olyan módon döntsön, hogy az utána következő szinteknek maradjon megengedett megoldása. Minden játékosnak van egy lineáris célfüggvénye is, amelyet maximalizálni szeretne, és ez a függvény függhet előtte és utána következő játékosok döntéseitől is. Az eddig leírtak precíz definíciója a következő.

3.1.1. Definíció. *Legyen $L \in \mathbb{N}$ a játékosok száma, $x = (x_1, x_2, ..., x_L) \in \mathbb{R}^n$ a változók egy partíciója, ahol $x_k \in \mathbb{R}^{n_k}$ valamely $n_k \in \mathbb{N}$ számokra, amelyekre $\sum_{k=1}^{L} n_k = n$. Az x_i változókról dönt az i-edik játékos. Legyen továbbá az i-edik játékos költségfüggvénye $f_i : \mathbb{R}^n \to \mathbb{R}$ $(i = 1, ..., L)$ és megengedett megoldásainak halmaza $S_i \subseteq \mathbb{R}^n$.*

Ekkor az L-szintű programozási feladatot az alábbi módon definiáljuk:

$$\max_{x_L} f_L(x) \tag{3.1}$$

$$\text{s.t. } x \in S_L \tag{3.2}$$

$$\max_{x_{L-1}} \{f_{L-1}(x) : x_L \text{ rögzített}\} \tag{3.3}$$

$$\text{s.t. } x \in S_{L-1} \tag{3.4}$$

$$\max_{x_{L-2}} \{f_{L-2}(x) : x_L, x_{L-1} \text{ rögzített}\} \tag{3.5}$$

$$\ldots$$

$$\max_{x_1} \{f_1(x) : x_L, ..., x_2 \text{ rögzített}\} \tag{3.6}$$

$$\text{s.t. } x \in S_1 \tag{3.7}$$

A definíciót egy kétszintű modellel fogom illusztrálni, amelyet [4]-ből vettem, és egy hálózat-árazási problémát ír le. A hálózatot egy irányított gráf modellezi, V csúcshalmazzal és $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ élhalmazzal. Minden $a \in \mathcal{A}$ élen adott egy $c_a \geq 0$ alapköltség, továbbá a felső szint (vagy vezető) az $\mathcal{A}_1$-be eső a élek tulajdonosa, és ezeken T_a használati díjat kíván bevezetni az alsó szint számára úgy, hogy a lehető legnagyobb legyen a profitja. Az alsó szintet (vagy követőket) egy K halmaz definiálja, $k \in K$ egy η^k számú csoportot fog jelölni, amely az o^k csúcsból akar a d^k csúcsba eljutni legrövidebb úton. Feltesszük, hogy minden $k \in K$ esetén létezik olyan megengedett útvonal, amely csak $\mathcal{A}_2$-beli éleket használ, különben a vezető tetszőlegesen nagy használati díjat vezethetne be. A vezető döntési változója T_a ($a \in \mathcal{A}_1$) lesz, a követőké $k \in K$ esetén x_a^k, ha $a \in \mathcal{A}_1$, és y_a^k, ha $a \in \mathcal{A}_2$. Ez utóbbi változók értéke 1, ha k használja az a élt, különben 0.

$$\max_{T \geq 0} \sum_{a \in \mathcal{A}_1} T_a (\sum_{k \in K} \eta^k x_a^k) \tag{3.8}$$

$$\text{s.t.} (x, y) \in \arg\min_{x,y} \sum_{k \in K} \left(\sum_{a \in \mathcal{A}_1} (c_a + T_a) x_a^k + \sum_{a \in \mathcal{A}_2} c_a y_a^k \right) \tag{3.9}$$

$$\sum_{a \in \varrho(v)} (x_a^k + y_a^k) - \sum_{a \in \delta(v)} (x_a^k + y_a^k) = b_v^k \quad \forall k \in K, \ \forall v \in V \tag{3.10}$$

$$x_a^k \geq 0 \quad \forall k \in K, \ \forall a \in \mathcal{A}_1 \tag{3.11}$$

$$y_a^k \geq 0 \quad \forall k \in K, \ \forall a \in \mathcal{A}_2, \tag{3.12}$$

ahol $b_v^k = -1$, ha $v = o^k$, $b_v^k = 1$, ha $v = d^k$ és 0 különben. Mivel az alsó szint egy legrövidebb út probléma, így teljesül rá a teljesen unimoduláris tulajdonság, és nem kell x-re és y-ra egészértékűségi feltételt előírni.

Visszatérve a modell általános alakjához a robusztusság kérdése természetes módon merül fel. Egy játékosnak több optimális megoldás is lehet, az alábbi kétszintű példa azt szemlélteti, hogy mennyire eltérő megoldásokat kaphatunk, ha nem teszünk fel semmit a játékosok viselkedéséről.

3.1.2. Példa. *Három változó van, a kettes számú játékos dönt először x_2-ről, ezután pedig az egyes számú az x_1 és az x_3-ról.*

$$\max_{x_2} 2x_1 + x_2 \tag{3.13}$$

$$\text{s.t. } x_2 \geq 1 \tag{3.14}$$

$$\max_{x_1, x_3} x_2 + x_3 \tag{3.15}$$

$$\text{s.t. } x_3 \leq 3 \tag{3.16}$$

$$x_2 - x_1 \leq 1 \tag{3.17}$$

$$x_1 + x_2 \leq 3 \tag{3.18}$$

$$x_1, x_3 \geq 0 \tag{3.19}$$

A (3.17) és (3.18) korlátok miatt $x_2 \leq 2$. Ha a második játékos az $x_2 = 2$ választást teszi, akkor az első szinten az optimális megoldás egyértelmű, $x_3 = 3$ és $x_1 = 1$, tehát az optimum $2 \cdot 1 + 2 = 4$. Ha azonban az $x_2 = 1$ esetet vesszük, akkor $x_3 = 3$ ismét, de az egyes számú játékos x_1-et tetszőlegesen választhatja a $[0, 2]$ intervallumból, hiszen nincs hatással a saját célfüggvényére, így a kettes számú játékos célfüggvénye 1 és 5 között bármi lehet az egyes játékos választásától függően.

Alapvetően kétféle feltételezést tehetünk, egy optimistát és egy pesszimistát. Az optimista megközelítés együttműködő játékosokat feltételez, azaz ha az i-edik játékosnak több optimális megoldása is van, akkor azt választja, amely az $i + 1$-edik számára a legkedvezőbb. A pesszimista megközelítéssel kapjuk a modell robusztus változatát, itt az i-edik játékos nem együttműködő, így akár a legrosszabb esetre is fel kell készülnie az $i + 1$-edik játékosnak. A 3.1.2 példában az optimista feltételezés mellett a kettes számú játékos nyugodtan választhatja az $x_2 = 2$ megoldást, és a feladat optimuma 5 lesz, míg a pesszimista feltételezés mellett az $x_2 = 1$ a robusztus megoldás, és az optimum pedig 4.
A feladat bonyolultságát azonban nem befolyásolják ezek a feltevések, ahogy látni fogjuk, még akkor is nehéz feladatot kapunk, ha minden szinten az optimális megoldás egyértelmű. Ahhoz, hogy a többszintű lineáris programozási feladat nehézségét megvizsgálhassuk, először kell egy rövid kitérőt tennünk a különböző bonyolultsági osztályok világába. A polinomiális hierarchiát [5] alapján mutatom be.

3.2. A polinomiális hierarchia

3.2.1. Definíció. *Legyenek L_1 és L_2 tetszőleges nyelvek, ekkor L_1 Turing-visszavezethető L_2-re, ha létezik olyan determinisztikus Turing-gép, amely kiegészítve egy L_2-re vonatkozó orákulummal felismeri L_1-et polinomiális időben. Jelölés: $L_1 \propto_T L_2$.*

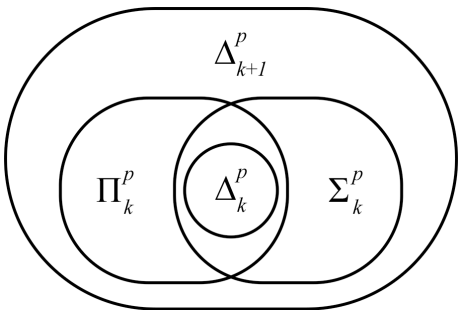
3.2.2. Definíció. *Legyenek L_1 és L_2 tetszőleges nyelvek, ekkor L_1 nemdeterminisztikusan Turing-visszavezethető L_2-re, ha létezik olyan nemdeterminisztikus Turing-gép, amely kiegészítve egy L_2-re vonatkozó orákulummal felismeri L_1-et polinomiális időben. Jelölés: $L_1 \propto_T^{NP} L_2$.*

3.2.3. Definíció. *Legyen Y egy tetszőleges nyelvosztály, ekkor*
$P^Y = \{L : \exists\, L' \in Y, \text{amelyre } L \propto_T L'\}$
$NP^Y = \{L : \exists\, L' \in Y, \text{amelyre } L \propto_T^{NP} L'\}$.

3.2.4. Definíció. *Legyen $\Delta_0^p = \Sigma_0^p = \Pi_0^p = P$.*
$k \geq 1$ esetén rekurzívan definiáljuk az osztályokat:

$$\Delta_k^p = P^{\Sigma_{k-1}^p}$$
$$\Sigma_k^p = NP^{\Sigma_{k-1}^p}$$
$$\Pi_k^p = co - \Sigma_k^p$$

3.2.5. Megjegyzés. *$k = 1$ esetén a jól ismert osztályokat kapjuk: $\Delta_1^p = P, \Sigma_1^p = NP$ és $\Pi_1^p = co - NP$, továbbá a következő tartalmazás áll fenn az osztályok között: $\Sigma_k^p \cup \Pi_k^p \subseteq \Delta_{k+1}^p \subseteq \Sigma_{k+1}^p \cap \Pi_{k+1}^p$.*



3.1. ábra. PH-beli tartalmazások

3.2.6. Definíció. *Jelölje PH$=\bigcup_k \Sigma_k^p = \bigcup_k \Pi_k^p$ a polinomiális hierarchia osztályait.*

Amikor egy nyelvről be akarjuk látni, hogy NP-ben van, akkor leggyakrabban nem az osztály eredeti definícióját használjuk, hanem egy polinomiális tanút adunk meg. Ez a karakterizálás általánosítható a Σ_k^p és Π_k^p osztályokra $k > 1$ esetén is.

3.2.7. Definíció. *Legyen Γ egy ábécé, ekkor $\langle z_1, ..., z_k \rangle$ k-asok egy halmazát ($z_i \in \Gamma^*$, $\forall i = 1, ..., k$) egy Γ^* feletti k-dimneziós relációnak nevezzük. Egy R reláció polinomiális időben felismerhető, ha létezik olyan polinomiális idejű Turing-gép, amely pontosan azokat a $\langle z_1, ..., z_k \rangle$ k-asokat ismeri fel, amelyek R-ben vannak.*

3.2.8. Tétel. *Legyen $L \subseteq \Gamma^*$ egy nyelv, $|\Gamma| \geq 2$. Tetszőleges $k \geq 1$ esetén L pontosan akkor eleme Σ_k^p-nak, ha léteznek $p_1, p_2, ..., p_k$ polinomok és egy polinomiális időben felismerhető $k+1$ dimenziós R reláció Γ^* felett, amelyre tetszőleges $x \in \Gamma^*$ esetén*

$$\begin{aligned} x \in L \Leftrightarrow & \exists\, y_1 \in \Gamma^*,\ ahol\ |y_1| \leq p_1(|x|), \\ & \forall\, y_2 \in \Gamma^*,\ ahol\ |y_2| \leq p_2(|x|), \\ & ... \\ & \mathcal{Q}\, y_k \in \Gamma^*,\ ahol\ |y_k| \leq p_k(|x|), \\ & amelyre\ \langle x, y_1, y_2, ..., y_k \rangle \in R. \end{aligned}$$

Ahol a minden és a létezik kvantorok váltakoznak, és így $\mathcal{Q} = \begin{cases} \forall, & ha\ k\ páros \\ \exists, & ha\ k\ páratlan \end{cases}$.

Hasonló tétel mondható ki a Π_k^p osztályba tartozásról a kvantorok felcserélésével.

3.2.9. Tétel. *Legyen $L \subseteq \Gamma^*$ egy nyelv, $|\Gamma| \geq 2$. Tetszőleges $k \geq 1$ esetén L pontosan akkor eleme Π_k^p-nak, ha léteznek $p_1, p_2, ..., p_k$ polinomok és egy polinomiális időben felismerhető $k+1$ dimenziós R reláció Γ^* felett, amelyre tetszőleges $x \in \Gamma^*$ esetén*

$$\begin{aligned} x \in L \Leftrightarrow & \forall\, y_1 \in \Gamma^*,\ ahol\ |y_1| \leq p_1(|x|), \\ & \exists\, y_2 \in \Gamma^*,\ ahol\ |y_2| \leq p_2(|x|), \\ & ... \\ & \mathcal{Q}'\, y_k \in \Gamma^*,\ ahol\ |y_k| \leq p_k(|x|), \\ & amelyre\ \langle x, y_1, y_2, ..., y_k \rangle \in R. \end{aligned}$$

Ahol a minden és a létezik kvantorok váltakoznak, és így $\mathcal{Q}' = \begin{cases} \exists, & ha\ k\ páros \\ \forall, & ha\ k\ páratlan \end{cases}$.

Ezzel a kitérő végéhez értünk, és folytathatjuk a többszintű modell számítási bonyolultságának vizsgálatát.

3.3. A feladat nehézsége

3.3.1. Tétel. *Egy $L+1$-szintű program Σ_L^p-nehéz.*

Bizonyítás. A bizonyítás menete a következő lesz. Először tekintünk egy Σ_L^p-teljes problémát, ez lesz az L-szintű SAT-játék, és belátjuk a 3.3.4 lemma és a 3.3.6 állítás segítségével, hogy ez átalakítható egy L-szintű hátizsák-játékká. Ezután pedig a 3.3.10 tételben az L-szintű hátizsák-játékot vezetjük vissza egy $L+1$-szintű programozási feladatra. ■

3.3.2. Definíció. *L-szintű SAT-játék*
Legyen $\varphi = D_1 \wedge D_2 \wedge ... \wedge D_N$ egy logikai formula, ahol $D_n \equiv x_{n_1} \vee x_{n_2} \vee x_{n_3}$ és minden x_{n_i} egy logikai változó vagy annak egy tagadása. Vegyük a változók egy L elemű partícióját, az i-edik részhalmazbeli változóknak az i-edik játékos ad igaz vagy hamis értéket közvetlenül a $i+1$-edik játékos választása után ($1 \leq i \leq L-1$). A páratlan sorszámú játékosok célja, hogy φ igaz legyen, a párosoké, hogy hamis. Ki nyer?

3.3.3. Tétel. *Az L-szintű SAT-játék Σ_L^p-teljes.*

3.3.4. Lemma. *Legyen φ egy olyan logikai formula, mint 3.3.2-ben, amely x_m ($m \in M$) változókból áll. Ekkor léteznek olyan u, a_m ($m \in M$), b_h ($h \in S$) természetes számok, hogy tetszőleges $v : M \to \{T, F\}$ kiértékelés esetén φ pontosan akkor igaz, ha létezik $S_v \subset S$, amelyre*

$$\sum_{m:v(m)=T} a_m + \sum_{h \in S_v} b_h = u.$$

Továbbá $\max\{a_m, b_h, u\} \leq 22^{N+1}$ és $|S| \leq 4N$, ahol N a logikai formula mérete.

Bizonyítás. Először definiáljuk a c_{nm} számokat. Minden $1 \leq n \leq N, m \in M$ esetén legyen $c_{nm} = 0$, ha sem x_m, sem x_m tagadása nem szerepel D_n-ben. A φ formula definíciója miatt minden n esetén pontosan három olyan m van, hogy x_m benne van D_n-ben, az ezekhez tartozó c_{nm} értékekhez tetszőlegesen rendeljük hozzá az 1, 2 és 4 számokat. Legyen

$$a_m \equiv \sum_{n=1}^{N} 22^n c_{nm} \mod 8$$

$$u \equiv \sum_{n=1}^{N} 22^n \mod 8.$$

Továbbá minden $0 \leq j \leq 7$ esetén léteznek olyan s_{jt} $(1 \leq t \leq 4)$ számok, hogy tetszőleges 1 és 8 közötti szám, kivéve $8-j$-t felírható az s_{jt} számok egy részhalmazának összegeként. Válasszuk ezeket a számokat úgy, hogy j-t rögzítve $\sum_t s_{jt} \leq 14$ teljesüljön. Például $j = 1$ esetén válasszuk az $s_{11} = 1, s_{12} = 2, s_{13} = 3, s_{14} = 8$ számokat.
Minden n esetén legyen $j(n)$ az olyan c_{nm}-ek összege, amelyekre x_m tagadása benne van D_n-ben. Így D_n-nek a v kiértékelés igaz értéket ad, kivéve ha

$$\sum_{m:v(m)=T} c_{nm} = j(n).$$

Ahhoz, hogy a megfelelő b_h értékeket is megkapjuk, legyen $S = \{(n,t) : 1 \leq n \leq N, 1 \leq t \leq 4\}$ és

$$b_{n,t} \equiv 22^n s_{j(n)t} \mod 8 \quad 1 \leq t \leq 4.$$

Pontosan akkor létezik a megfelelő S_v halmaz, ha minden n esetén létezik $W \subset \{1,2,3,4\}$, amelyre

$$\sum_{m:v(m)=T} c_{nm} + \sum_{k \in W} s_{j(n)k} = 8.$$

Az s_{jt} definíciója miatt ilyen W létezik, kivéve ha $\sum_m c_{nm} = j(n)$. Ez pedig azzal ekvivalens, hogy a v kiértékelés mellett D_n hamis. ■

3.3.5. Definíció. *L-szintű hátizsák-játék*
Adottak u és a_i $(i \in I)$ természetes számok. Vegyük I-nek egy $\cup I_j$ partícióját, a j-edik játékos kiválasztja az I_j halmaz egy P_j részhalmazát, amelyet a hátizsákba tesz $j = L, ..., 1$ sorrendben. A páratlan sorszámú játékosok célja, hogy végül $\sum_{i \in \cup P_j} a_i = u$ legyen, a páros sorszámúak pedig ezt akarják megakadályozni. Ki nyer?

3.3.6. Állítás. *Az L-szintű hátizsák-játék segítségével megolható az L-szintű SAT-játék.*

Bizonyítás. Adott φ-hez tartozó SAT-játék a 3.3.4 lemma segítségével alakítható át, vegyük a megfelelő u, a_m $(m \in M)$ és b_h $(h \in S)$ természetes számokat. Legyen $I = M \cup S$. M-et úgy partícionáljuk, ahogyan a játékosok az x_m logikai változókról döntenek, továbbá az első játékos dönt az S halmazról is. A két játéknak pontosan ugyanaz a csapat a győztese. ■

A továbbiakban egy L-szintű hátizsák-játékot alakítunk át egy $L+1$-szintű feladattá. Ahhoz, hogy ne kelljen a változókról egészértékűséget feltenni, legyen egy nulladik játékos, aki utoljára választ. Tekintsük a következő modellt, ahol a nulladik

játékos dönt a Q, A, B és D_i $(i \in I)$ változókról, és $j \geq 1$-re pedig a j-edik játékos az x_i $(i \in I_j)$ változók értékét határozza meg.

$$A - B - \sum_{i \in I} a_i x_i = u \tag{3.20}$$

$$Q \leq A + B \tag{3.21}$$

$$Q \leq 0.6 \tag{3.22}$$

$$A, B \geq 0, \tag{3.23}$$

ahol a nulladik játékos célfüggvénye a következő:

$$\max_{Q,A,B,D_i} Q - A - B + \sum_{i \in I} D_i. \tag{3.24}$$

Továbbá legyen $c = \max_{i \in I} a_i$, ekkor a j-edik játékos $(j \geq 1)$ célfüggvénye a következő:

$$\max_{x_i : i \in I_j} (-1)^j Q - 2^{j+2} c \sum_{i \in I_j} D_i - \sum_{i \in I_j} \epsilon_i x_i \tag{3.25}$$

3.3.7. Definíció. *Tetszőleges $t \in \mathbb{R}$ esetén jelölje $\delta(t)$ a t-nek a hozzá legközelebb eső egész számtól vett távolságát.*

Látható, hogy a nulladik játékos megoldásában $D_i = \delta(x_i)$ lesz, és Q pedig a $\min\{0.6, |u - \sum a_i x_i|\}$ értékkel fog megegyezni. $j \geq 1$ esetén ha tekintjük a (3.20)-as célfüggvényt, akkor látszik, hogy a D_i értékek azt büntetik, ha a j-edik játékos nem egész x_i-ket választ. Ha az x_i változókat a játékosok közel egészeknek választják, akkor Q értéke 0.6 lesz, ha $\sum a_i x_i \neq u$ (mivel az a_i-k is egészek), különben pedig 0-hoz közeli. Így a páros sorszámú játékosokat a $(-1)^j Q$ tag arra motiválja, hogy $\sum a_i x_i$ ne legyen u-val egyenlő, a páratlan sorszámúakat pedig arra, hogy ezek megegyezzenek.
Az ϵ_i $(i \in I)$ számok azt fogják biztosítani, hogy az optimális megoldás egyértelmű, hiszen azt is be akarjuk látni, hogy a játékosok viselkedésére tett feltevések nem befolyásolják a feladat nehézségét. Ehhez elegendő lesz olyan $\epsilon_i > 0$ számokat venni, amelyekre $\sum \epsilon_i < 0.3$ és tetszőleges $H, H' \subset I_j$ esetén

$$\sum_{i \in H} \epsilon_i \neq \sum_{i \in H'} \epsilon_i \quad \text{ha } H \neq H'. \tag{3.26}$$

Innentől a cél az, hogy belássuk, az optimális megoldásban minden x_i egész. Ezt a játékosok száma szerinti indukcióval fogjuk megtenni, ugyanis ha az L-edik játékos választott, akkor u-t $u - \sum_{i \in I_L} a_i x_i$-vel helyettesítve egy eggyel kevesebb szintű feladatot kapunk. Mivel a változók folytonosak, így azt az esetet is figyelembe kell venni,

hogy ez az érték esetleg nem egész. Először következzen két segédállítás.

3.3.8. Állítás. *Legyen x_i $(i \in I)$ és $\bar{x}_i$ $(i \in I)$ egészek, Q és $\bar{Q}$ az ezekhez tartozó, nulladik játékos által meghatározott értékek. Ekkor ha $\delta(u) \leq 0.25$ és $Q \neq \bar{Q}$, akkor $|Q - \bar{Q}| \geq 0.35$. Továbbá*

$$(-1)^L Q > (-1)^L \bar{Q} \Leftrightarrow (-1)^L Q - \sum_{i \in I_L} \epsilon_i x_i > (-1)^L \bar{Q} - \sum_{i \in I_L} \epsilon_i \bar{x}_i. \tag{3.27}$$

Bizonyítás. Mivel az a_i számok is egészek, így a $\sum a_i x_i$ összegnek csak egyetlen olyan értéke van, amelyre $Q \neq 0.6$, és ilyenkor $Q = \delta(u) \leq 0.25$. Ha $\bar{Q} \neq Q$, akkor $\bar{Q} = 0.6$, és $|Q - \bar{Q}| \geq 0.35$.
Ha $(-1)^L(Q - \bar{Q}) > 0$, akkor $(-1)^L(Q - \bar{Q}) \geq 0.35 > \sum \epsilon_i \geq \sum_{i \in I_L} \epsilon_i \geq \sum_{i \in I_L} \epsilon_i(x_i - \bar{x}_i)$, hiszen $\sum \epsilon_i < 0.3$ és $0 \leq x_i, \bar{x}_i \leq 1$.
A másik irányhoz pedig $(-1)^L(Q - \bar{Q}) > \sum_{i \in I_L} \epsilon_i(x_i - \bar{x}_i) \geq -\sum \epsilon_i > -0.3$, és felhasználva, hogy $|Q - \bar{Q}| \geq 0.35$ megkapjuk, hogy $(-1)^L(Q - \bar{Q}) > 0$. ■

A következő állítás az L-edik játékos stratégiájára vonatkozik.

3.3.9. Állítás. *Az L-edik játékos választása után*

$$\sum_{i \in I_L} \delta(x_i) \leq \frac{2^{-(L+1)}}{c}. \tag{3.28}$$

Bizonyítás. Ha L olyan stratégiát választ, ahol

$$\sum_{i \in I_L} \delta(x_i) > \frac{2^{-(L+1)}}{c}, \tag{3.29}$$

akkor ez rosszabb célfüggvényértéket ad számára, mint ha az $x_i = 0$ $(i \in I_L)$ döntést hozná, ugyanis (3.29) fennállása esetén

$$\max(-1)^L Q - 2^{L+2} c \sum_{i \in I_L} D_i - \sum_{i \in I_L} \epsilon_i x_i \leq \max(-1)^L Q - 2 \leq -1.4,$$

míg ha az $x_i = 0$ választást tekintjük, akkor az L-edik játékos célfüggvénye ennél jobb lesz, hiszen $\max(-1)^L Q \geq -0.6$. ■

3.3.10. Tétel. *Legyen u' az u-hoz legközelebb eső egész. Ha $|u - u'| \leq 2^{-(L+1)}$, az egyértelmű optimális megoldásban minden x_i egész. Ráadásul ugyanezeket az x_i-ket kapjuk megoldásként, ha azt feladatot tekintjük, ahol u-t lecseréljük u'-re.*

3.3.11. Megjegyzés. *Az L-edik játékos választásakor $|u - u'| \leq 2^{-(L+1)}$ fennáll, hiszen $u = u'$, az indukciós lépéshez szükséges a tételt ilyen formában kimondani.*

Bizonyítás. A bizonyítási módszer tehát egy L szerinti indukció lesz, ugyanis az L-edik játékos választása után egy eggyel kevesebb szintű feladatot kapunk, és az $L = 1$ esetet menet közben látjuk be. A 3.3.9 állítás miatt csak olyan x_i ($i \in I_L$) választást kell figyelembe venni, ahol (3.28) fennáll. Legyen x_i' az x_i-hez legközelebb eső egész, ekkor

$$|(u - \sum_{i \in I_L} a_i x_i) - (u' - \sum_{i \in I_L} a_i x_i')| \leq |u - u'| + c \sum_{i \in I_L} \delta(x_i) \leq 2^{-(L+1)} + 2^{-(L+1)} = 2^{-L}.$$

Ez azt jelenti, hogy miután az L-edik játékos meghozta döntését, az eggyel kevesebb szintű feladatban, ahol u-t $u - \sum_{i \in I_L} a_i x_i$-re cseréltük le, teljesülni fog a tétel feltétele. Tehát ennek az optimális megoldásában minden x_i ($i \in I \setminus I_L$) biztosan egész, ráadásul ugyanezeket az x_i-ket kapjuk, ha kerekítünk, azaz u-t $u' - \sum_{i \in I_L} x_i'$-re cseréljük. Így ha az L-edik játékos a saját változóit x_i-ről x_i'-re cseréli le, a többi játékos nem változtat a saját stratégiáján.

3.3.12. Megjegyzés. *Az innentől a bizonyítás végéig tartó érvelés az $L = 1$ esetben is igaz, így bizonyítja az indukció alapesetét.*

Továbbá

$$-\sum_{i \in I_L} a_i \delta(x_i) + 2^{L+2} c \sum_{i \in I_L} \delta(x_i) - 0.3 \sum_{i \in I_L} \delta(x_i) > 0,$$

kivéve ha minden x_i egész. Így az L-edik játékos többet nyer a D_i-ken, mint amennyit a célfüggvénye Q-s és ϵ_i-s részén veszít, azaz megéri kerekíteni a változóit, tehát létezik olyan optimális megoldás, ahol minden x_i ($i \in I$) egész.
Az egyértelműség a következőképpen látható be (az L-edik játékos döntését és célfüggvényét tekintjük). Ha két különböző egész megoldás esetén különböző Q értékeket kapunk, akkor a 3.3.8 állítás szerint a célfüggvényértékek sem fognak megegyezni a (3.27)-es egyenlőtlenség miatt. Ha azonos Q értékeket kapunk, akkor pedig az ϵ_i-k definíciójában szereplő (3.26)-os feltétel miatt nem lehetnek egyenlőek a célfüggvények.
Végül már csak az u lecserélésére vonatkozó állítást kell bizonyítani. Legyen x_i^* ($i \in I_L$) az egyértelmű optimális egész megoldása az eredeti problémának, $\bar{x}_i$ pedig az L-edik játékos egy tetszőleges egész választása. Az ezekhez kapott Q értékek legyenek Q^* és $\bar{Q}$. Amikor u-t lecseréljük u'-re, akkor ezek helyett $Q^{*\prime}$ és $\bar{Q}'$ értékeket kapunk az x_i^*, illetve $\bar{x}_i$ vektorok mellett. Q^* és $\bar{Q}$ külön-külön vagy 0.6-tal egyenlő vagy legfeljebb 0.25, esetszétválasztással látjuk be, hogy minden esetben az L-edik játékos az eredeti x_i^* választáshoz fog ragaszkodni u' mellett is.
$Q^* = \bar{Q} = 0.6$ esetén $Q^{*\prime} = \bar{Q}' = 0.6$ is teljesül. Ha $Q^*, \bar{Q} \leq 0.25$, akkor $Q^* = \bar{Q}'$ és

$Q^{*\prime} = \bar{Q}' = 0$. Ebben a két esetben a célfüggvény ϵ_i-s részétől függ, hogy az L-edik játékos x^* és $\bar{x}$ közül melyiket választja, az ϵ_i-ket azonban nem befolyásolja, ha u-t lecseréljük u'-re.
Ha $Q^* = 0.6$ és $\bar{Q} \leq 0.25$, akkor $Q^{*\prime} = 0.6$ és $\bar{Q}' = 0$. Ekkor az L-edik játékos a Q-t maximalizálni szeretné, így a $Q^{*\prime} = 0.6$ eset jobb neki, mint a $\bar{Q}' = 0$.
Ha $Q^* \leq 0.25$ és $\bar{Q} = 0.6$, akkor $Q^{*\prime} = 0$ és $\bar{Q}' = 0.6$. Ekkor az L-edik játékos a Q-t minimalizálni szeretné, így a $Q^{*\prime} = 0$ eset jobb neki, mint a $\bar{Q}' = 0.6$. ■

Összességében tehát a (3.15)-(3.20) által leírt $L+1$-szintű lineáris programozási feladat megoldásából kiolvasható, mely csapat nyeri meg az L-szintű hátizsák-játékot, tehát a feladat legalább olyan nehéz, mint a polinomiális hierarchia L-edik szintjén elhelyezkedő problémák.

4. fejezet

Kétfázisú optimalizálás

A 2.1 alfejezetben már szerepelt egy megoldás arra, hogy hogyan lehet elkerülni, hogy egy robusztus modell túlságosan konzervatív legyen. Azonban másfajta megközelítések is léteznek, ilyen például a kétfázisú modell. Ebben a változóknak csak egy részéről kell döntést hoznunk úgy, hogy az adatokban bizonytalanság van, és miután ezeket pontosan megismertük, utána döntünk a maradék változókról, ez egyfajta újratervezési lehetőség. Ebben a fejezetben részletesen ismertetem a kétfázisú optimalizálási feladatot, majd kimondok, és bizonyítok két állítást a feladat számítási bonyolultságával kapcsolatban (ez saját eredmény), ezután [6] alapján bemutatok egy oszlop- és vágásgeneráló algoritmust a feladat megoldására, és ezt az implementáció folyamán felmerülő kérdések megválaszolásával egészítem ki. Végül [7] alapján alkalmazom az algoritmust egy p-median modellre.

4.1. A feladat

Tekintsük az alábbi robusztus kétfázisú modellt.

$$\min_{y \in S_y} cy + \max_{u \in U} \min_{x \in F(y,u)} bx \tag{4.1}$$

$$Ay \geq d \tag{4.2}$$

$$F(y,u) = \{x \in S_x : Gx \geq h - Ey - Mu\} \tag{4.3}$$

$$S_y \subseteq \mathbb{R}^n_+,\ S_x \subseteq \mathbb{R}^m_+ \tag{4.4}$$

Az első fázis döntési változója $y \in \mathbb{R}^n$, amely az S_y halmazból választható, illetve vonatkoznak rá kiindulási feltételek is, ezeket (4.2) írja le, ahol $A \in \mathbb{R}^{k \times n}$ és $d \in \mathbb{R}^k$. Miután y értéke rögzítésre kerül, kiderül a bizonytalan vektor, u értéke, amely az $U \subseteq \mathbb{R}^l$ halmazból származik. Ennek és az első fázisban választott y-nak a függvényében döntünk a második fázis döntési változójáról, x-ről ($x \in \mathbb{R}^m$), amely az S_x halmazban van. Azt, hogy x értéke y-tól és u-tól is függ a (4.3)-as feltétel

fejezi ki, ahol $G \in \mathbb{R}^{p\times m}, h \in \mathbb{R}^p, E \in \mathbb{R}^{p\times n}$ és $M \in \mathbb{R}^{p\times l}$. Összességében a $cy + bx$ költséget szeretnénk minimalizálni, ahol $c \in \mathbb{R}^n_+$ és $b \in \mathbb{R}^m_+$. Mivel u értékét kezdetben nem ismerjük, így a robusztus megoldás keresése azt jelenti, hogy azt a minimumot szeretnénk kiszámolni, amelyet tetszőleges u esetén elérhetünk, ezt fejezi ki a költségfüggvényben a bizonytalan vektorokat tartalmazó U halmazra vett maximum.

4.2. A feladat számítási bonyolultsága

Felmerül a kérdés, hogy az optimum kiszámolása vajon mennyire nehéz probléma, ezt én is megvizsgáltam, és a továbbiakban a feladat nehézségére vonatkozó saját állításokat mondok ki, és bizonyítok.

4.2.1. Definíció. *Kétfázisú optimalizálás döntési verziója*
Adott egy (4.1)-(4.4) típusú minimalizálási probléma és egy K szám. A feladat eldönteni, hogy létezik-e megoldás, amelynek az értéke kisebb-egyenlő-e mint K.

Tekintsük azt a speciális esetet, ahol c, A, d és E mindegyike csupa 0, tehát y tetszőlegesen választható, és nincsen hatással a második fázisra. Ekkor a következő feladatot kapjuk.

$$\max_u \min bx \tag{4.5}$$

$$Gx \geq h - Mu \tag{4.6}$$

$$x \in S_x \subseteq \mathbb{R}^m_+ \tag{4.7}$$

4.2.2. Definíció. *3-CNF-SAT*
Adott $\varphi(x, y)$ konjuktív normálformájú logikai formula klózonként pontosan három literállal. Igaz-e, hogy minden x esetén létezik y, hogy $\varphi(x, y)$ igaz?

4.2.3. Tétel. *A 3-CNF-SAT feladat Π^p_2-teljes.*

4.2.4. Állítás. *A (4.5)-(4.7) feladat Π^p_2-nehéz.*

Bizonyítás. A 3-CNF-SAT feladatot vezetjük vissza, legyen a logikai formula $\varphi(u, x)$. Minden literálhoz vegyünk fel egy bináris változót, a célfüggvény pedig legyen $\max_u \min_x \neg\varphi(u, x)$. Ha ugyanis az optimum 0, akkor az azzal ekvivalens, hogy az u változók tetszőleges kiértékelése mellett létezik olyan kiértékelése x-nek, amelyre $\neg\varphi$ hamis, azaz φ igaz, azaz a 3-CNF-SAT feladatra a válasz igen. Pozitív optimum (a (4.8)-as célfüggvényből majd látszik, hogy kaphatunk 1-nél nagyobb számot is) esetén viszont tudunk az u-nak olyan igaz-hamis hozzárendelését megadni, hogy az x változók tetszőleges kiértékelése mellett φ hamis legyen, azaz a 3-CNF-SAT feladatra a válasz nem.

A logikai formulából a következő módon kaphatunk egy lineáris célfüggvényt. Először is tegyük fel, hogy $\varphi(u,x) = \varphi_1(u,x) \wedge \varphi_2(u,x) \wedge ... \wedge \varphi_k(u,x)$, ahol minden φ_i pontosan három literált tartalmaz, azaz $\varphi_i = l_1^i \vee l_2^i \vee l_3^i$, ahol l_1^i, l_2^i és l_3^i egy-egy u_j vagy x_j változóval vagy ezek tagadásával egyezik meg. Ezután alkalmazzuk a De-Morgan azonosságot $\neg\varphi(u,x)$-re, így a továbbiakban a célfüggvényünk $\neg\varphi_1(u,x) \vee \neg\varphi_2(u,x) \vee ... \vee \neg\varphi_k(u,x)$ lesz. A $\neg\varphi_i$ formulákra ismét a De-Morgan azonosságot alkalmazva, majd a *vagy* műveletet összeadással, az *és* műveletet pedig szorzással helyettesítve a következő célfüggvényt kapjuk.

$$\max_u \min_x \sum_{i=1}^{k} l_1^i \cdot l_2^i \cdot l_3^i \tag{4.8}$$

Ezt szeretnénk linearizálni. Ha l_r^i egy u_j (vagy x_j) változó tagadása, akkor a célfüggvényben szerepeljen $1 - u_j$ (illetve 1-x_j). Így egy olyan összeget kapunk, amelynek tagjai között bináris változók szorzatai is lesznek. Általában egy p és q bináris változó szorzatát egy új $v = pq$ változó bevezetésével linearizálhatjuk, illetve a következő feltételek felírásával.

$$v \leq p \tag{4.9}$$

$$v \leq q \tag{4.10}$$

$$v \geq p + q - 1 \tag{4.11}$$

$$p, q, v \in \{0, 1\} \tag{4.12}$$

A (4.8)-ban szereplő összeg tagjai között olyanok is lehetnek, amelyek kettő vagy három bináris változó szorzataként álltak elő, utóbbi esetben a (4.9)-(4.12) által leírt módszert kétszer is alkalmazva és a célfüggvényben a megfelelő tagokat az új változókra lecserélve végül egy lineáris célfüggvényt kapunk. Összességében pedig a kétfázisú optimalizálás döntési verziójára a linearizált (4.8)-as feladat és $K = 0$ esetén pontosan akkor lesz igen a válasz, ha a 3-CNF-SAT feladatra a válasz igen. ■

4.2.5. Következmény. *A kétfázisú optimalizálási feladat Π_2^p-nehéz.*

4.2.6. Állítás. *A kétfázisú optimalizálási feladat Σ_3^p-ben van.*

Bizonyítás. Tekintsük a feladat döntési verzióját, és alkalmazzuk rá a 3.2.8 tételt. Legyen (Π, K) egy feladatpéldány, az R egy 4-dimenziós reláció, amelyre $\langle(\Pi, K), y, u, x\rangle \in R$ pontosan akkor, ha Π a (4.1)-(4.4) feltételekből áll, amelyet y, u és x együttesen kielégít, és a célfüggvényérték legfeljebb K. ■

4.2.7. Megjegyzés. *A kétfázisú optimalizálási feladat tehát a polinomiális hierarchia második vagy harmadik szintjén helyezkedik el. A 4.2.4 állításban a feladatnak csak egy speciális esetét tekintettem, így a sejtésem az, hogy a feladat általános esetben még nehezebb, azaz Σ_3^p-teljes.*

4.3. Oszlop- és vágásgeneráló algoritmus

Beláttuk, hogy egy kétfázisú modell optimumának kiszámítása NP-nehéz, így nem várható polinomiális algoritmus, amely ezt megoldja. A számítási nehézségek leküzdésére több megközelítés is létezik, ezek közül most egy mesterprobléma-szubprobléma szerkezetű algoritmus következik, a C&CG-algoritmus, amely minden iterációban egyre jobb alsó, illetve felső korlátot állít elő az optimumra. Ezt a primál oldalon generált oszlopokkal és vágásokkal teszi, amelyről a nevét is kapta (column and constraint generation-C&CG).
Az algoritmus alapötlete a következő: először tegyük fel, hogy U egy véges halmaz, elemei $u_1, ..., u_r$, és az ezekhez tartozó változók a második fázisban $x_1, ..., x_r$. Ekkor a feladatunk az alábbi alakot ölti:

$$\min_y cy + \eta \tag{4.13}$$

$$Ay \geq d \tag{4.14}$$

$$\eta \geq bx_l \quad \forall l = 1, ..., r \tag{4.15}$$

$$Gx_l \geq h - Ey - Mu_l \quad \forall l = 1, ..., r \tag{4.16}$$

$$y \in S_y,\ x_l \in S_x \quad \forall l = 1, ..., r,\ \eta \in \mathbb{R},\ u_l \in U \quad \forall l = 1, ..., r \tag{4.17}$$

Ha U számossága nem túl nagy, akkor ezt az LP-t könnyen megoldhatjuk. Ha viszont nem így van, például U egy poliéder, akkor a benne lévő vektorok számbavétele, és így az összes feltétel felsorolása a gyakorlatban nem megoldható. Azonban ha csak a feltételek egy részét soroljuk fel, akkor az eredeti feladatunk egy relaxációját és így optimumának egy alsó korlátját kapjuk. A cél tehát az, hogy megtaláljuk a szignifikáns $u \in U$ vektorokat, és az erre felvett feltételek miatt egyre erősebb alsó korlátokat kapjunk az optimumra. Felső korlátokat pedig onnan kapunk, ha a mesterprobléma optimumaként kapott y-t rögzítve kiszámoljuk a teljes feladat optimumát, hiszen ilyenkor a minimalizálási problémának egy rögzített változó melletti célfüggvényértékét kapjuk. Az algoritmus lépései a következők.

C&CG

1: Legyen $LB = -\infty, UB = \infty, k = 1$ és $O = \emptyset$.

2: Oldjuk meg a mesterproblémát (MP).

$$\min_{y,\eta} cy + \eta$$
$$Ay \geq d$$
$$\eta \geq bx_l \ \ \forall l \in O$$
$$Ey + Gx_l \geq h - Mu_l^* \quad \forall l = 1, ..., k-1$$
$$y \in S_y, \ x_l \in S_x \quad \forall l = 1, ..., k-1, \ \eta \in \mathbb{R}_+, \ u_l^* \in U \quad \forall l = 1, ..., k-1$$

Legyen az optimális megoldás $(y_k^*, \eta_k^*, x_1^*, ..., x_{k-1}^*)$.

3: Definiáljuk az SP részproblémát a következőképpen.

$$\mathcal{Q}(y_k^*) = \max_{u \in U} \min_{x \in S_x} \{bx : Gx \geq h - Ey_k^* - Mu\}$$

Legyen $UB = \min\{UB, cy_k^* + \mathcal{Q}(y_k^*)\}$.

4: Ha $UB - LB < \epsilon$, akkor STOP, a megoldás optimális.

5: Ha $\mathcal{Q}(y_k^*) < \infty$, akkor legyen (x^*, u_k^*) az optimális megoldása SP-nek, vegyünk fel egy új x_k változót, és adjuk a következő feltételeket MP-hez:

$$\eta \geq bx_k$$
$$Ey + Gx_k \geq h - Mu_k^*$$

Legyen $k = k + 1$, $O = O \cup \{k\}$, és lépjünk a 2. lépéshez.

6: Ha $\mathcal{Q}(y_k^*) = \infty$, akkor legyen u_k^* az a vektor, amelyre SP nem megengedett. Vegyünk fel egy új x_k változót, és adjuk a következő feltételt MP-hez:

$$Ey + Gx_k \geq h - Mu_k^*$$

Legyen $k = k + 1$, és lépjünk a 2. lépéshez.

4.3.1. Definíció. *Ha rögzített y és u esetén mindig létezik x megengedett megoldása a második fázisnak, akkor ezt RCRA tulajdonságnak fogjuk nevezni (relatively complete recourse assumption).*

4.3.2. Megjegyzés. *Ha a kétfázisú feladatra úgy tekintünk, mint egy háromszintű feladatra (a három játékos az y, u és x változókat választja ebben a sorrendben), akkor ez a feltételezés megegyezik azzal, hogy egy többszintű modell esetén a játékosok úgy választanak, hogy az utánuk következő játékosnak legyen megengedett megoldása.*

4.3.3. Tétel. *Tegyük fel, hogy teljesül az RCRA tulajdonság. Legyen p az U poliéder extrém pontjainak száma, vagy ha U véges halmaz, akkor a számossága, ekkor az algoritmus az optimális megoldást találja meg $\mathcal{O}(p)$ lépésben.*

Bizonyítás. Ebben az esetben $\mathcal{Q}(y_k^*) < \infty$ áll fenn mindig, így a legrosszabb esetben is az algoritmus legenerálja vagy a poliéder csúcsait, vagy a véges halmaz minden elemét. ■

Amennyiben az algoritmust implementálni szeretnénk, kettő kérdés merül fel. Az első a szubprobléma alakja, hiszen a célfüggvényben szerepel egy max min alakú kifejezés. A második kérdés pedig az optimális u vektor kiolvasásához kapcsolódik.

4.3.4. Állítás. *SP átírható egy egészértékű programozási feladattá.*

Bizonyítás. Ahhoz, hogy SP ne egy max min probléma legyen, egy vele ekvivalens alakra írjuk át. A belső $\min bx$-et úgy kényszeríthetjük arra, hogy az optimális értéket vegye fel adott u mellett, hogy a primál feltétel mellé felvesszük a duális, illetve a komplementaritási feltételeket.

$$\max bx \tag{4.18}$$

$$Gx \geq h - Ey - Mu \tag{4.19}$$

$$\pi G \leq b \tag{4.20}$$

$$(Gx - h + Ey + Mu)_i \pi_i = 0 \quad \forall i \tag{4.21}$$

$$(b - \pi G)_j x_j = 0 \quad \forall j \tag{4.22}$$

Ezután a (4.21) és (4.22) komplementaritási feltételeket például a nagy M-módszer segítségével írhatjuk át, ahol $\mathbb{M}$ egy megfelelően nagy pozitív szám, χ^1, χ^2 pedig megfelelő dimenziós bináris vektorváltozók.

$$(Gx - h + Ey + Mu)_i \leq \mathbb{M}\chi_i^1 \quad \forall i \tag{4.23}$$

$$\pi_i \leq \mathbb{M}(1 - \chi_i^1) \quad \forall i \tag{4.24}$$

$$(\pi G)_j \geq b_j - \mathbb{M}\chi_j^2 \quad \forall j \tag{4.25}$$

$$x_j \leq \mathbb{M}(1 - \chi_j^2) \quad \forall j \tag{4.26}$$

■

Ha nem tesszük fel az RCRA tulajdonságot, akkor a k-adik iterációban el kell tudni döntenünk, hogy $Q(y_k^*)$ esetleg $+\infty$-t vesz-e fel, azaz hogy SP nem megengedett. Ehhez elegendő (4.18)-(4.22)-nek csak egy részét megnézni, és eldönteni, hogy lehet-e olyan u-t mondani, hogy már ez a rész sem oldható meg. Elég, ha a (4.19)-es primál feltételről döntjük el, hogy megoldható-e. Ha ugyanis meg tudjuk oldani, akkor a dualitás tétel alapján tudjuk, hogy a primál és a duál optimum megegyezik. Ebben a feladatban egy minimalizálási probléma van elrejtve, tehát az még előfordulhatna, hogy a primál probléma megengedett, de az optimuma $-\infty$, és így a duális feltétel, $\pi G \leq b$ nem elégíthető ki. Ez azoban mégsem történhet meg, hiszen x-ről és b-ről is feltettük, hogy nemnegatív, tehát a 0 mindenképpen alsó korlát. Így tehát ha a primál feltételre van megoldás, akkor véges optimumot kapunk, és van olyan x, π primál-duál pár, amely kielégíti a komplementaritási feltételeket, és a teljes részproblémának megoldása.

A $Gx \geq h - Ey - Mu$ feltétel megolhatóságának tesztelését a következőképpen írhatjuk fel.

$$\max_{u \in U} \min \sum_i s_i \tag{4.27}$$

$$(Gx)_i + s_i \geq (h - Ey - Mu)_i \quad \forall i \tag{4.28}$$

$$x \in S_x, \; s \geq 0 \tag{4.29}$$

Amennyiben az optimum értéke 0, akkor a $(Gx)_i + \geq (h - Ey - Mu)_i \;\; \forall i$ feltétel minden u esetén kielégíthető, és továbblépünk a teljes SP megoldásának keresésére. Ha viszont pozitív optimumot kapunk, akkor létezik egy olyan u, ahol legalább az egyik s_i pozitív értéket vesz fel, és így olyan u-t találtunk, amelyre SP nem megoldható.
Ahhoz viszont, hogy ez szintén ne egy maxmin feladat legyen, át kell írnunk. Hasonlóan járunk el, mint amikor SP-t alakítottuk át.

$$\max_u \sum_i s_i \tag{4.30}$$

$$(Gx)_i + s_i \geq (h - Ey - Mu)_i \quad \forall i \tag{4.31}$$

$$(\alpha G)_j \leq 0 \quad \forall j \tag{4.32}$$

$$\alpha_j \leq 1 \quad \forall j \tag{4.33}$$

Továbbá a komplementaritási feltételek:

$$x_i > 0 \implies (\alpha G)_i = 0 \quad \forall i \tag{4.34}$$

$$s_i > 0 \implies \alpha_i = 1 \quad \forall i \tag{4.35}$$

$$\alpha_i > 0 \implies (Gx)_i + s_i = (h - Ey - Mu)_i \quad \forall i \tag{4.36}$$

Végül ezt a három feltételt ismét a nagy M-módszer segítségével alakíthatjuk át.

$$x_j \leq \mathbb{M} \cdot (1 - \chi_j^1) \quad \forall j \tag{4.37}$$

$$(\alpha G)_j \geq -\mathbb{M} \cdot \chi_j^1 \quad \forall j \tag{4.38}$$

$$s_i \leq \mathbb{M} \cdot (1 - \chi_i^2) \quad \forall i \tag{4.39}$$

$$\alpha_i \geq 1 - \mathbb{M} \cdot \chi_i^2 \quad \forall i \tag{4.40}$$

$$\alpha_i \leq \mathbb{M} \cdot (1 - \chi_i^3) \quad \forall i \tag{4.41}$$

$$(Gx)_i + s_i \leq (h - Ey - Mu)_i + \mathbb{M} \cdot \chi_i^3 \quad \forall i \tag{4.42}$$

4.4. Egy robusztus p-median modell

A most következő modell egy p-median problémának a kétfázisú robusztus változata. Ebben egy hozzárendelési feladatot szeretnénk megoldani az I-vel jelölt kliens- és a J-vel jelölt gyárhalmaz között ($J \subseteq I$, sőt először még tegyük fel, hogy $I = J$). A $|J|$ darab gyár közül pontosan $p \geq 1$ darabot szeretnénk megnyitni, ezeknek nincsen megnyitási költsége. Ezután a megnyitott gyárakhoz rendeljük hozzá az I kliens halmazt. Innentől viszont eltér a modell a jól ismert p-median problémától, ugyanis azt a jelenséget is szeretnénk modellezni vele, ahol előre nem látható okoból néhány gyár leáll, ekkor pedig a megmaradt gyárakkal kell újratervezni a hozzárendeléseket. A gyárakhoz tartozó bináris döntési változó $y \in \{0,1\}^{|J|}$ lesz, ahol $y_j = 1$ pontosan akkor, ha megnyitjuk a j-edik gyárat. Minden i klienshez tartozik egy d_i igény, amelyből ha egy egységet a j-edik gyár szolgál ki, akkor ennek a költsége c_{ij}. Előírjuk, hogy a költségfüggvény olyan legyen, hogy $c_{ii} = 0 \quad \forall i$. Az első fázisnak van egy másik döntési változója is, $x \in [0,1]^{|I|\times|J|}$, ahol x_{ij} azt írja le, hogy i-edik kliens igényének hányad részét szolgálja ki a j-edik gyár (amelynek nyitottnak kell lennie). Ezek után térjünk rá a modellben szereplő bizonytalanságra, amelynek értéke azután derül ki, hogy y és x változókat rögzítettük. Ehhez bevezetünk egy $z \in \{0,1\}^{|J|}$ bináris változót, ahol $z_j = 1$ pontosan akkor, ha a j-edik gyár kiesik. A kieső gyárak számát korlátozzuk egy előre rögzített K egész számmal. A kieső gyárak miatt több kliens igénye kiszolgáló nélkül marad. Ezeket a klienseket a megmaradt gyárakhoz kell hozzárendelnünk, ezért bevezetjük a második fázis döntési változóit, $w \in [0,1]^{|I|\times|J|}$-t és $q \in [0,1]^{|I|}$-t. w_{ij} fogja kifejezni azt, hogy a második fázisban az i-edik kliens igényének hányad részét szolgálja ki a j-edik gyár, míg q_i azt, hogy az i-edik kliens igényének mekkora része marad kiszolgálatlanul. Egy i klienst csak egy megnyitott és épen maradt gyárhoz rendelhetjük hozzá, amelynek az igény $1 - q_i$ részét kell kiszolgálnia. Az eddig leírtakat az alábbi korlátok fejezik ki.

$$(1-\varrho)\min \sum_{i\in I}\sum_{j\in J} c_{ij}d_jx_{ij} +$$

$$+\varrho \max_{z} \min_{(w,q)\in S(y,z)} \left(\sum_{i\in I}\sum_{j\in J} c_{ij}(1-\theta z_i)d_iw_{ij} + \sum_{i\in I} M(1-\theta z_i)d_iq_i \right) \tag{4.43}$$

$$x_{ij} \leq y_j \quad \forall i,j \tag{4.44}$$

$$\sum_i x_{ij} = 1 \quad \forall i \tag{4.45}$$

$$\sum_j y_j = p \tag{4.46}$$

$$y_j \in \{0,1\} \quad \forall j, \quad x_{ij} \geq 0 \quad \forall i,j \tag{4.47}$$

$$\sum_j z_j \leq K \tag{4.48}$$

$$z_j \in \{0,1\} \quad \forall j \tag{4.49}$$

Ahol $S(y,z)$ azokból a (w,q) vektorokból áll, amelyek teljesítik a következő korlátokat:

$$w_{ij} \leq 1 - z_j \quad \forall i,j \tag{4.50}$$

$$w_{ij} \leq y_j \quad \forall i,j \tag{4.51}$$

$$\sum_j w_{ij} + q_i = 1 \quad \forall i \tag{4.52}$$

$$w_{ij} \geq 0 \quad \forall i,j, \quad q_i \geq 0 \quad \forall i \tag{4.53}$$

Ebben (4.43)-(4.47) fejezi ki a célfüggvényt és a p-median probléma feltételeit, (4.48)-(4.49) a bizonytalan halmazt, míg (4.50)-(4.53) az újrakonfigurálást.
A célfüggvény két tagból tevődik össze: a normál körülmények közötti működés költségének és a legrosszabb esetbeli működés költségének összege. A $\varrho \in [0,1]$ paraméterrel állíthatjuk be, hogy a tervezés során melyik esetet milyen súllyal szeretnénk figyelembe venni. Minél nagyobb a ϱ, annál konzervatívabb a modell. Az első tagban az első fázisbeli hozzárendeléssel kapott szállítási költség minimalizálása szerepel, a második tagban a lehetséges z vektorokra vett maximum fejezi ki, hogy a legrosszabb esetet keressük, és ezen belül minimalizáljuk a költséget, amely két dologból tevődik össze. A kiszolgált igények költségéből és a nem kiszolgált igényekből, amelyet egységenként egy M konstanssal büntetünk. Az itt szereplő θ paraméter azt a jelenséget kívánja modellezni, hogy amennyiben az i-edik telephelyen leáll egy gyár, ott a kliens igénye is megváltozhat. Például egy természeti katasztrófa esetén egy luxustermék iránti kereslet lecsökkenhet, míg a gyógyszerek iránti kereslet megnövekedhet. Ehhez a θ paramétert egy $(-\infty,1]$-beli valós számnak rögzítjük, és így látható, hogy ha a $(0,1]$ intervallumba esik, akkor az az igénycsökkenést fejezi ki, míg negatív számként az igény növekedését.

4.4.1. Megjegyzés. *A célfüggvény felírásánál kihasználtuk, hogy a I és J halmazok megegyeznek, azaz minden kliens egyben potenciális telephely is. Ha azonban a $J \subset I$ esetet szeretnénk vizsgálni, akkor azon klienseknél, amelyek az $I - J$ halmazba esnek, az $(1-\theta z_i)$ kifejezés nem értelmes, így itt például csinálhatjuk azt, hogy kettébontjuk*

a célfüggvényben szereplő összegzést a következőképpen.

$$(1-\varrho)\min ... + \varrho \max_{z} \min_{(w,q)\in S(y,z)} \left(\sum_{i\in J}\sum_{j\in J} c_{ij}(1-\theta z_i)d_i w_{ij} + \right.$$
$$\left. + \sum_{i\in I-J}\sum_{j\in J} c_{ij} d_i w_{ij} + \sum_{i\in J} M(1-\theta z_i) d_i q_i + \sum_{i\in I-J} M d_i q_i \right)$$

Nem okoz tehát problémát, ha a modellt a $J \subset I$ esetre szeretnénk alkamazni, de azért, hogy a továbbiakban is tömören leírható legyen a célfüggvény, maradjon meg a feltevés, hogy $I = J$.

Az előző alfejezetben részletesen foglalkoztam az algoritmus elméleti hátterével. Röviden összefoglalva tehát az ilyen kétfázisú modellek esetén a megoldás keresése nehéz, ezért nem a teljes problémát szeretnénk rögtön megoldani, hanem újabb és újabb iterációkban megkeressük a szignifikáns szcenáriókat, amelyek a (4.48)-(4.49) feltételek által leírt poliéder extrém pontjai közül kerülnek ki.

4.4.2. Állítás. *A (4.43)-(4.53) feltételek által leírt modellre teljesül az RCRA tulajdonság.*

Bizonyítás. Mivel $p \geq 1$, így legalább egy gyár meg lesz nyitva, és mivel ezeknek nincsen kapacitása, így tetszőlegesen sok kliens hozzárendelhető egy gyárhoz a második fázisban, vagy akár kiszolgálatlanul maradt igények is lehetnek. ■

Így tehát a C&CG-algoritmusnak a szubprobléma megoldása után csak egyetlen ága lesz, mivel mindig véges optimumot fogunk kapni. Az algoritmust a modellre alkalmazva a következő eljárást kapjuk.

1. Legyen az alsó korlát $LB = -\infty$, a felső korlát $UB = +\infty, k = 1$.

2. Oldjuk meg a mester problémát (MP):

$$\min\ (1-\varrho)\sum_i \sum_j c_{ij} d_i x_{ij} + \varrho\eta \tag{4.54}$$

$$x_{ij} \leq y_j \quad \forall i,j \tag{4.55}$$

$$\sum_i x_{ij} = 1 \quad \forall i \tag{4.56}$$

$$\sum_j y_j = p \tag{4.57}$$

$$x_{ij} \geq 0 \quad \forall i,j, \quad y_j \in \{0,1\} \quad \forall j \tag{4.58}$$

$$\eta \geq \sum_i \sum_j c_{ij}(1-\theta z_i^l)d_i w_{ij}^l + \sum_i M(1-\theta z_i^l)d_i q_i^l \quad \forall l = 1,..,k-1 \tag{4.59}$$

$$\sum_j w_{ij}^l + q_i^l = 1 \quad \forall i,\ \forall l = 1,..,k-1 \tag{4.60}$$

$$w_{ij}^l \leq 1 - z_j^l \quad \forall i,j,\ \forall l = 1,..,k-1 \tag{4.61}$$

$$w_{ij}^l \leq y_j \quad \forall i,j,\ \forall l = 1,..,k-1 \tag{4.62}$$

$$w_{ij}^l \geq 0 \quad \forall i,j,\ \forall l = 1,..,k-1,\ \ q_i^l \geq 0 \quad \forall i,\ \forall l = 1,..,k-1,\ \ \eta \geq 0 \tag{4.63}$$

3. Állítsuk be LB-t a kapott optimumra, és legyen (y^k, x^k) MP optimális megoldása, ezt használva oldjuk meg a következő részproblémát (SP):

$$\max_z \min_{w,q} \sum_i \sum_j c_{ij}(1-\theta z_i)d_i w_{ij} + \sum_i M(1-\theta z_i)d_i q_i \tag{4.64}$$

$$w_{ij} \leq 1 - z_j \quad \forall i,j \tag{4.65}$$

$$w_{ij} \leq y_j^k \quad \forall i,j \tag{4.66}$$

$$\sum_j w_{ij} + q_i = 1 \tag{4.67}$$

$$\sum_j z_j \leq K \tag{4.68}$$

$$w_{ij} \geq 0 \quad \forall i,j,\ \ q_i \geq 0 \quad \forall i,\ \ z_j \in \{0,1\} \quad \forall j \tag{4.69}$$

4. Legyen SP optimális megoldásának értéke $\mathcal{Q}(y^k)$, és az optimális z értéke pedig z^k. Legyen $UB = \min\{UB,\ (1-\varrho)\sum_i \sum_j c_{ij} d_i x_{ij}^k + \varrho \mathcal{Q}(y^k)\}$.

5. Ha $UB - LB \leq \epsilon$, akkor álljunk meg. Különben vegyünk MP-hez új (w^k, q^k) változókat a rájuk vonatkozó korlátokkal együtt z^k érték mellett. Legyen $k = k+1$, és menjünk a 2. lépéshez.

4.4.3. Megjegyzés. *A 4.4.2 állítás egyben az algoritmus végességét is bizonyítja, hiszen legfeljebb annyi iteráció lesz, amennyi a* $\{z \in \{0,1\}^{|J|} : \sum_j z_j \leq K\}$ *halmaz elemeinek száma.*

Az algoritmus implementálhatóságához SP-t itt is át kell alakítani, a 4.3.4 állításhoz hasonlóan ismét a duális feltételeket vesszük fel u_{ij}, v_{ij} és s_i duál változókkal, de a komplementaritási feltételek helyett a célfüggvényt írjuk át, és összeolvasztjuk a z-re vett maximumot és a duális problémában szereplő maximalizálást.

$$\max_{z,u,v,s} \sum_i \sum_j (1 - z_j) u_{ij} + \sum_i \sum_j y_j^k v_{ij} + \sum_i s_i \tag{4.70}$$

$$u_{ij} + v_{ij} + s_i \leq c_{ij} d_i (1 - \theta z_i) \quad \forall i, j \tag{4.71}$$

$$s_i \leq M d_i (1 - \theta z_i) \quad \forall i \tag{4.72}$$

$$\sum_j z_j \leq K \tag{4.73}$$

$$u_{ij} \leq 0 \quad \forall i, j, \quad v_{ij} \leq 0 \quad \forall i, j, \quad z_j \in \{0, 1\} \quad \forall j \tag{4.74}$$

Az így kapott célfüggvény nemlineáris, az $u_{ij} z_j$ tag okozza a problémát (y_j^k értéke rögzített, így a második tag is lineáris). Új $U_{ij} = u_{ij} z_j$ változókat bevezetve és kihasználva, hogy z bináris, ezt a következőképpen küszöbölhetjük ki a nagy M-módszer segítségével. Itt M a kiszolgálatlanul maradt igények költsége, és $\mathbb{M}$ a megfelelően nagy konstans a nagy M-módszerhez.

$$\max_{z,u,v,s} \sum_i \sum_j (u_{ij} - U_{ij} + y_j^k v_{ij}) + \sum_i s_i \tag{4.75}$$

$$u_{ij} + v_{ij} + s_i \leq c_{ij} d_i (1 - \theta z_i) \quad \forall i, j \tag{4.76}$$

$$s_i \leq M d_i (1 - \theta z_i) \quad \forall i \tag{4.77}$$

$$U_{ij} \geq u_{ij} \quad \forall i, j \tag{4.78}$$

$$U_{ij} \geq -\mathbb{M} z_j \quad \forall i, j \tag{4.79}$$

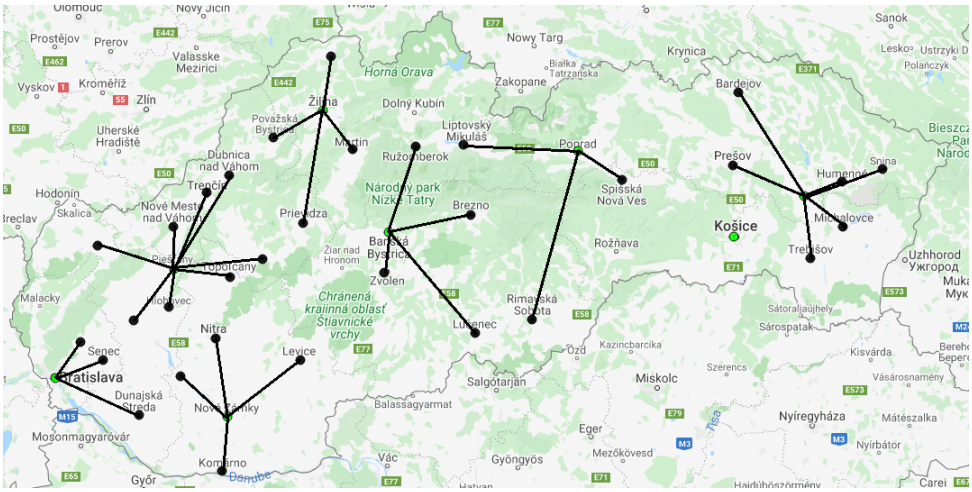
$$U_{ij} \leq u_{ij} + \mathbb{M}(1 - z_j) \quad \forall i, j \tag{4.80}$$

$$\sum_j z_j \leq K \tag{4.81}$$

$$u_{ij} \leq 0 \quad \forall i, j, \quad U_{ij} \leq 0 \quad \forall i, j, \quad v_{ij} \leq 0 \quad \forall i, j, \quad z_j \in \{0, 1\} \quad \forall j \tag{4.82}$$

4.4.4. Példa. *Az algoritmust kipróbáltam egy teljes egészében valós adatokon alapuló feladaton is. Ehhez a c_{ij} értékeket tartalmazó mátrixra volt szükségem, illetve a d_i igényvektorra. A távolságmátrixot a [8] linkről vettem, ez egy 2928 szlovák települést és távolságaikat tartalmazó fájlcsomag. Ez olyan nagy feladat lenne, amely előreláthatólag nagyon lassan futna le, és a kapott eredményeket nehéz lenne ábrázolni is. Emiatt a feladat méretét jelentősen lecsökkentettem, és csak Szlovákia 40 legnagyobb városát vettem, határként a 20 ezres lélekszámot húztam meg. Az igényeket pedig a lakosságszám adta.*

A következő paramétereket választottam: $|I| = |J| = 40$, $p = 8$, $\varrho = 0.25$, $K = 2$ és $\theta = 1$.



4.1. ábra. Normál működés a megadott paraméterek mellett

4.2. ábra. K=2 gyár leállása után

Az 4.1-es ábrán láthatóak az első fázisbeli normál működés melletti hozzárendelések, a térképeken zöld pöttyel jelöltem a megnyitott gyárak városait. Két érdekességet vehetünk észre, amelyek az ország szerkezetéből adódnak. Mivel a lakosságszámot használtam igényként, így nyilvánvaló volt, hogy Pozsonyban (Bratislava) fog nyitni gyárat az algoritmus. De mivel az ország délnyugati csücskében helyezkedik el, így kevés várost rendelt hozzá. A másik ehhez hasonló eset Kassa (Košice) város esete, amelyet szintén úgy nyitott meg az algoritmus, hogy kevés más várost szolgál ki, egész pontosan egyet sem.

A 4.2-es ábrán látható, hogy a legrosszabb eset az, amikor az a két gyár áll le, amelyhez a legtöbb város volt rendelve, nyugaton Pöstyén (Piešťany) és keleten Varannó (Vranov nad Topľou). Ezeket az ábrán piros pötty jelzi, az újrakonfigurálást pedig a kék élek.

Összefoglalás

Szakdolgozatomban a robusztus optimalizálás, ezen belül pedig linearizálható, többszintű és kétfázisú modellek bemutatása volt a cél. A bizonytalanságot modellező halmaz sokféle lehet, láthattunk példát feltételenkénti bizonytalanságra (ezen belül intervallummal, illetve poliéderrel reprezentált bizonytalanságra is) a 2. fejezetben, az optimális megoldás megválasztása függhet más döntéshozók viselkedésétől, erről szólt a 3. fejezet, továbbá láthattunk a jövőtől függő és véges bizonytalan halmazt is a 4. fejezetben. Az operációkutatásnak ez a területe még nem teljesen kiforrott, kevésbé ismert is, így a megfelelő modell vagy algoritmus megválasztása gyakran történhet próbálkozással és erősen függ az adott probléma szerkezetétől, nincsen általános recept. A célom az volt, hogy ízelítőt adjak a robusztus optimalizálásból, amennyiben a téma felkeltette az olvasó figyelmét, [9] egy összefoglaló cikk, amely könnyen érthető módon áttekinti a területet, a legfontosabb fogalmakat és modelleket.

Irodalomjegyzék

[1] D. Bertsimas and M. Sim, „The price of robustness," *Operations research*, vol. 52, pp. 35–53, 2004.

[2] D. Bertsimas, I. Dunning, and M. Lubin, „Reformulation versus cutting-planes for robust optimization," *Computational Management Science*, vol. 13, pp. 195–217, 2016.

[3] C. Blair, „The computational complexity of multi-level linear programs," *Annals of Operations Research*, vol. 34, pp. 13–19, 1992.

[4] M. Labbé and A. Violin, „Bilevel programming and price setting problems," *Annals of Operations Research*, vol. 240, pp. 141–169, 2016.

[5] M. Garey and D. Johnson, *Computers and intractability-A guide to the theory of NP-completeness*, pp. 161–167. W. H. Freeman and Company, 1979.

[6] B. Zeng and L. Zhao, „Solving two-stage robust optimization problems using a column-and-constraint generation method," *Operations Research letters*, vol. 41, pp. 457–461, 2013.

[7] Y. An, B. Zeng, Y. Zhang, and L. Zhao, „Reliable p-median facility location problem: two-stage robust models and algorithms," *Transportation Research Part B*, vol. 64, pp. 54–72, 2014.

[8] `http://frdsa.uniza.sk/~buzna/page5/supplement4/benchmarks.html` (Letöltés: 2021. december 31.)

[9] J. García and A. Peña, *Robust Optimization: Concepts and Applications*. IntechOpen, 2018.