NYILATKOZAT

Név: Lekli Henrietta

ELTE Természettudományi Kar, szak: Biztosítási és Pénzügyi Matematika MSc

NEPTUN azonosító: BHERQE

Szakdolgozat címe:
Iskolaválasztás Európában

A **szakdolgozat** 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, 2020.12.21.

Lekli Henrietta

a hallgató aláírása

Eötvös Loránd Tudományegyetem

Budapesti Corvinus Egyetem

Lekli Henrietta

Iskolaválasztás Európában

Iteratív eljárások vizsgálata utánajárási költségek esetén

Biztosítási és Pénzügyi Matematika MSc szakdolgozat

Témavezető:

Dr. Biró Péter
Operációkutatás és Aktuáriustudományok Tanszék

Budapest, 2020

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani elsősorban témavezetőmnek, Dr. Biró Péternek, aki rengeteg időt és energiát fordított a konzultációkra, hasznos tanácsaival és szakértelmével nagyon sokat segített abban, hogy ez a dolgozat elkészüljön. Külön köszönöm a sok szakirodalmat neki, amelyek segítségével jobban megismerkedhettem a témával. Szeretnék még köszönetet mondani a családomnak és a barátaimnak, hogy támogattak az egyetemi tanulmányaim során.

Tartalomjegyzék

1. Bevezetés **6**

2. Alapfogalmak **7**

2.1. Matematikai és közgazdasági alapfogalmak 7

2.2. Iskolaválasztás alapfogalmai 9

3. Mechanizmusok **12**

3.1. Boston mechanizmus 12

3.2. Columbus mechanizmus 14

3.3. Gale-Shapley algoritmus 15

3.4. Legjobb csere körök módszere 19

3.5. Melyik mechanizmust érdemes választani? 22

4. Európában alkalmazott felvételi eljárások **26**

4.1. Magyar egyetemi felvételi 26

4.2. Francia egyetemi felvételi 30

4.3. Egyéb felvételi eljárások 32

5. Kapcsolat a biztosítással **33**

6. Szimulációs modell leírása **36**

6.1. Ötletek és modellezési lehetőségek 37

6.2. Költséges GS-függvény 41

7. Szimulációs eredmények **44**

7.1. Utánajárási költség nélküli eset 44

7.2. Utánajárási költség figyelembevételével a direkt és az iteratív eljárás összehasonlítása 50

7.3. Következtetés 54

8. Összefoglalás **56**

Irodalomjegyzék **57**

9. Függelék **59**

9.1. Költséges GS-algoritmushoz tartozó kód 59

9.2. Utánajárási költség nélküli eset modellezésének kódja 64

9.3. Utánajárási költség figyelembevételével direkt és iteratív eljárás összehasonlításához tartozó kód 66

9.4. Ábrákhoz tartozó táblázatok 68

1. fejezet

Bevezetés

Minden diák életében az egyik legfontosabb kérdések közé tartozik, hogy melyik iskolába jelentkezzen továbbtanulni, legyen szó akár általános iskoláról, középiskoláról, vagy egyetemről. Szakdolgozatom célja bemutatni a legismertebb iskolaválasztási modelleket, köztük a direkt Gale-Shapley mechanizmust, majd ezeket különböző szempontok szerint összehasonlítani. Ezenkívül részletesebben szó lesz a magyar egyetemi felvételinek a rendszeréről, különös tekintettel arra, hogy miket kellett módosítani az alapmodellen és ezekre a módosításokra miért volt szükség. Majd francia egyetemi felvételin alkalmazott, egyedülálló, úgynevezett iteratív Gale-Shapley mechanizmust szeretném ismertetni. Ezt követően pár oldalon arról lesz szó, hogy milyen kapcsolat fedezhető fel a hazai felsőoktatás és a biztosítás között. Ezt követően a francia iteratív mechanizmus direkt mechanizmushoz viszonyított egyik lehetséges előnyét fogom szimulációval szemléltetni.

2. fejezet

Alapfogalmak

Mielőtt rátérünk az iskolaválasztás problémájára, szükséges bevezetnünk néhány matematikai és közgazdasági alapfogalmat.

2.1. Matematikai és közgazdasági alapfogalmak

Az iskolaválasztás voltaképpen egy párosítási feladat, hiszen az iskolákhoz diákokat kell rendelnünk. Így ebben az alfejezetben az ehhez szükséges fogalmakat definiáljuk.

2.1.1. Definíció. Egy G gráfot páros gráfnak nevezünk, ha csúcsainak halmaza felosztható A és B halmazra úgy, hogy G összes élére teljesül, hogy egyik végpontja A-ban, a másik végpontja B-ben van.

2.1.2. Definíció. Egy G gráf élhalmazának egy M részhalmazát párosításnak nevezzük, ha egyik csúcsára sem illeszkedik egynél több M-beli él. Ha minden csúcsra pontosan egy él illeszkedik, teljes párosításról beszélünk.

2.1.3. Definíció. Ha a G gráf minden u csúcsára adott egy $c(u) \in \mathbb{Z}$ kapacitásszám, akkor legfeljebb $c(u)$ darab él illeszkedhet az u csúcsra. Ezen élek ezen halmazát c-párosításnak nevezzük. Ezt másként több-a-többhöz párosításnak is szoktuk nevezni.

2.1.4. Definíció. Hozzárendelési feladatnak (angol nevén assignment problem) nevezzük az olyan feladatot, amely során páros gráfban kell párosítást keresnünk.

A következőkben nézzünk néhány hozzárendelési feladatot:

1. Házassági modell: Rendelkezésünkre áll nők és férfiak nem feltétlenül azonos méretű halmaza, és a feladatunk a lehető legmegfelelőbb házastársat megtalálni mindenki számára a megadott preferenciák alapján.

2. Menekültek allokációja: A menekültek befogadásakor érdemes lehet figyelembe venni a menekültek preferenciáit, igényeit, illetve a befogadó közösségek szempontjait is.

3. Örökbefogadási modell: Az előzőekhez hasonlóan itt is figyelembe kell venni a szülők preferenciáit, illetve a gyerekek érdekeit és igényeit is.

4. Felvételi rendszerek modellje (angol nevén collage admission): Felvételiző diákokat szeretnénk egyetemekre elhelyezni úgy, hogy a diákok és iskolák preferencia sorrendjeit is figyelembe vesszük. Hasonló a helyzet, amikor orvostan hallgatókat szeretnénk kórházakba elhelyezni a gyakornoki idejükre.

5. Iskolaválasztási rendszerek modellje (angol nevén school choice): Ez a modell csak abban különbözik az előző modelltől, hogy itt az iskolák preferencia-sorrend helyett prioritássorrenddel rendelkeznek. Ez azt jelenti, hogy az iskolák nem tekintendők döntéshozó játékosoknak, a szubjektív módon meghatározott preferencia-sorrend helyett objektív módon kialakított prioritássorrendjük van. Azaz a diákokhoz egy pontszámot rendelnek figyelembe véve, hogy a diák körzetben lakik-e, oda jár-e a testvére, illetve néhány alkalmazásban a korábbi teljesítményt, esetleg egy központosított írásbeli felvételit vesznek leginkább figyelembe. Tehát ez általában a helyi törvények által külsőleg szabályozott pontszámítás. Érdemes megemlíteni, hogy ezek a modellek főként az amerikai gyakorlatra építenek, a hazai gyakorlat pont fordított, a középiskolák tetszőleges módon határozhatják meg a rangsoraikat, szóbeli vizsgákat is tartva, míg az egyetemek döntő része csak a jegyek és a központi írásbeli felvételi pontszámai alapján határozza meg a rangsorát. Mivel a preferencia-sorrend, és a prioritássorrend is egy rangsort határoz meg, így az ebből fakadó tulajdonságokat ugyanúgy értelmezzük mindkét fogalomra.

Ahhoz, hogy pontosan meg tudjuk fogalmazni, mit értünk preferencia rendezés alatt, szükségünk van a preferencia relációk definiálására.

2.1.5. Definíció. Tegyük fel, hogy f fogyasztó preferenciái között szerepel a és b jászág. Ekkor

(*i*) az f fogyasztó gyengén preferálja az a jószágot b-vel szemben, ha a-t legalább annyira szereti, mint b-t. Jelölés: $a \succeq_f b$;

(*ii*) az f fogyasztó szigorúan preferálja az a jószágot b-vel szemben, ha a-t jobban szereti, mint b-t. Jelölés: $a \succ_f b$;

(*iii*) az f fogyasztó közömbös a és b jószágokkal szemben, ha $a \succeq_f b$ és $b \succeq_f a$ egyszerre teljesül, azaz egyformán szereti őket. Jelölés: $a =_f b$.

Feltehető, hogy ezen relációk teljesek, reflexívek és tranzitívek és így egy preferenciarendezést definiálnak. A prioritásrendezés is hasonló módon definiálható.

2.1.6. Definíció. A hasznosságfüggvény egy olyan függvény, melynek segítségével a fogyasztó különböző jószágokra vett preferenciáit matematikailag tudjuk modellezni. Ez azt jelenti, hogy ha $a >_f b \Leftrightarrow u(a) > u(b)$. A kardinális hasznosság elmélet szerint a fogyasztó számszerűen meg tudja határozni, mekkora hasznosságot jelent számára egy jószág. Az ordinális hasznosság elmélet szerint a fogyasztó csak rangsorolja a jószágokat hasznosságuk alapján, tehát ebben az esetben a konkrét függvényértékek nem bírnak jelentéssel.

2.2. Iskolaválasztás alapfogalmai

Ennek az alfejezetnek a megírásához főként az [1] és [3] forrásokat használtam. Először fogalmazzuk meg a feladatot általánosan. Az iskolaválasztás egyfajta párosítási feladat, mely során diákokhoz iskolai helyeket kell rendelnünk. Minden iskola rendelkezik egy maximális kapacitással, aminél több diákot nem képes befogadni. Tehát iskolaválasztás esetén több-az-egyhez párosításról van szó. A diákok szigorú preferencia-sorrenddel rendelkeznek az iskolákra nézve. Az iskolák diákokra vett prioritássorrendjét az állam vagy a helyi törvények határoznak meg. Az olyan diákok között, akik az iskola szempontjából azonosnak tekinthetők, általában sorsolással döntenek, így egy szigorú prioritássorrendet kapunk. Az iskolaválasztás során minden diákhoz pontosan egy iskolát rendelünk úgy, hogy egyetlen iskolának sem haladjuk meg a maximális kapacitását. Most írjuk le a feladatot matematikai jelölésekkel. Az iskolák halmazát jelöljük $S = (s_1, s_2, .., s_n)$-sel, a diákok halmazát jelöljük $D = (d_1, d_2, \ldots, d_k)$-vel. Ezen halmazok együttese egy páros gráfnak feleltethető meg. Az iskolák kapacitását jelöljük $c(s)$-sel. Az iskolák rangsorolják a diákokat, a diákok is az iskolákat. Ez a rangsor a diákok részéről egy preferencia-sorrendet, míg az iskolák részéről egy prioritássorrendet határoz meg, amit az egyszerűség kedvéért felsorolással jelölünk: $P(d) = s_1, \ldots s_n$, illetve $P(s) = d_1, \ldots, d_k$. Jelölje P a preferencia-, illetve prioritássorrendek halmazát: $P = (P(s_1), \ldots, P(s_n), P(d_1), \ldots, P(d_k))$. Tehát az iskolaválasztási feladatot (S, D, P)-vel tudjuk megadni. A diákokra vonatkozó hozzárendelési mechanizmus (angolul student assignment problem) egy szisztematikus eljárás, ami egy (S, D, P) feladatra egy konkrét párosítást ad vissza. Mivel valós életben nem lehetséges minden hallgatóhoz a legjobban preferált iskolát rendelni, ezért nagyon fontos a diákokra vonatkozó hozzárendelési mechanizmus megtervezése. A párosítást függvényként is meg lehet fogalmazni.

2.2.1. Definíció. Az m párosítás egy olyan halmazfüggvény, amely $S \cup D \to S \cup D$ úgy, hogy a következő feltételeket teljesíti:

(*i*) $\forall d \in D$-re $m(d) \subseteq D \cup S$ és $\mid m(d) \mid = 1$, azaz egy diák párja vagy egy iskola, vagy önmaga lehet. Az utóbbi eset akkor fordul elő, ha a diákot egy iskolába sem veszik fel, ilyenkor izolált pontként tekintünk rá.

(*ii*) $\forall s \in S$-re $m(s) \subseteq S \cup D$ és $\mid m(s) \mid = c(s)$, azaz ha egy iskola nem tölti fel az összes helyét diákokkal, akkor a maradék helyekkel önmagát párosítja.

(*iii*) $\forall d \in D$ és $\forall s \in S$-re: $d \in m(s) \Leftrightarrow s \in m(d)$, azaz a párosítás kölcsönös.

A diákok az iskolákat rangsorolják, az iskoláknak viszont tulajdonképpen a diákok egy csoportját (részhalmazát) kell rangsorolnia, hiszen a diákok csoportjával párosítjuk. Az s iskola diákcsoportokra vonatkozó prioritássorrendjét jelöljük $P'(s)$-sel. Ezen a halmazon értelmezett preferencia relációk jelölésére használjuk a $\succ_s$ jelölést.

2.2.2. Definíció. Azt mondjuk, hogy a diákok részhalmazain értelmezett $P'(s)$ prioritásrendszer fogékony (angolul responsive) a diákokon egyénenként értelmezett $P(s)$ prioritásrendszerre, ha $d \in m(s)$ és $d' \notin m(s)$ feltételek esetén teljesül, hogy

$$m(s) \cup d' \setminus d \succ_s m(s) \Leftrightarrow d' >_s d. \tag{2.1}$$

Hasonlóan értelmezhető a preferencia-rendszer fogékonysága, de iskolaválasztás esetén ennek nincs jelentősége.

2.2.3. Definíció. Az m párosítás tartalmaz blokkoló élt, ha létezik olyan párosítatlan diák-iskola pár, azaz (d, s) pár, ahol

(*i*) $s >_d m(d)$, azaz d preferencia-sorrendjében s iskola előrébb szerepel, mint a jelenlegi párja és

(*ii*) $s \in m(s)$ vagy $\exists d' \in m(s) : d >_s d'$, azaz az iskolában még van szabad hely (azaz nem pazarlásmentes), vagy van olyan diák, akivel össze van párosítva, de a prioritássorrendjében hátrébb van, mint d (azaz nem mentes az indokolt irigységtől) .

2.2.4. Definíció. Adott preferencia-sorrend és fogékony prioritássorrend esetén egy párosítás stabil, ha nem tartalmaz blokkoló éleket.

Azaz stabil párosítás esetén egy diákot csak abban az esetben utasíthat vissza egy iskola, ha az összes férőhelye megtelt számára jobb diákokkal.

2.2.5. Definíció. Egy diákokra vonatkozó hozzárendelési mechanizmus stabil, ha mindig stabil párosítást ad vissza.

2.2.6. Definíció. Egy m párosítás Pareto-hatékony, ha nem lehet rajta Pareto-javítást végezni, azaz nem lehet senkihez sem jobb párt találni anélkül, hogy valaki emiatt rosszabb párt kapjon.

Érdemes lehet megkülönböztetni a diákok számára Pareto-hatékony párosítást és az iskolák számára Pareto-hatékony párosítást.

2.2.7. Definíció. Egy m párosítás Pareto-hatékony a diákok számára, ha nem lehet rajta Pareto-javítást végezni, azaz nem lehet egyik diákhoz sem lehet jobb párt találni anélkül, hogy egy másik diák emiatt rosszabb párt kapjon.

Hasonlóan definiálható az iskolák számára Pareto-hatékony párosítás, ezért ezt nem részletezem.

2.2.8. Definíció. Egy diákokra vonatkozó hozzárendelési mechanizmus Pareto-hatékony, ha mindig Pareto-hatékony párosítást ad vissza.

2.2.9. Definíció. Egy diákokra vonatkozó hozzárendelési mechanizmus stratégiabiztos vagy mentes a taktikázástól, ha egyetlen hallgató sem részesülhet előnyben nem valós preferencia megadása esetén.

3. fejezet

Mechanizmusok

Ebben a fejezetben néhány valóságban alkalmazott mechanizmust mutatunk be. A fejezet megírásához a [1], [2], [3], és [4] forrásokat használtam.

3.1. Boston mechanizmus

Ezt a mechanizmust azonnal elfogadó (angolul immediate acceptance) mechanizmusnak, vagy angol nevéből adódóan IA-algoritmusnak is szokás nevezni. Néhány éve Bostonban és még sok amerikai városban használták, innen ered a neve is, azóta azonban a kritikák miatt egyre kevesebbé népszerű. Nézzük meg az algoritmus lépéseit:

(*i*) 0. lépés: Minden diák benyújtja a preferencia-sorrendjét. Minden iskola a prioritási sorrendjét a következők határozzák meg: első helyen szerepelnek azok a diákok, akiknek a testvérük abba az iskolába jár és a körzetben laknak, utánuk következnek azok, akiknek csak a testvérük jár oda, ezt követően jönnek, akik csak körzetben laknak, végül azok a diákok, akikre egyik sem teljesül. Ezenfelül a sorrendet sorsolással döntik el.

(*ii*) 1. lépés: Először a diákok első döntése számít. Minden iskola csak ezen diákoknak osztja ki helyeit a prioritási sorrend alapján úgy, hogy nem hagyja figyelmen kívül az iskola kapacitását sem.

(*iii*) k. lépés: Csak a megmaradt diákokat és a preferencia-sorrendben következőnek megjelölt iskolákat veszik figyelembe. Ezenkívül ugyanúgy folytatódik az algoritmus, mint az 1. lépésben.

(*iv*) Az algoritmus addig folytatódik, amíg van olyan diák, aki nem vettek fel sehova, de még nem értek végig a preferencia-listáján. Ha már minden ilyen diák elfogyott, az algoritmus

leáll.

3.1.1. Példa. Legyen három iskolánk: s_1, s_2, s_3 és hét diákunk: $d_1, \ldots, d_7$. Az s_1-ben 2 szabad hely, az s_2-ben és az s_3-ban is 3 szabad hely elérhető a diákok számára. Az iskolák prioritása legyen a következő:

$$P(s_1) = d_7, d_1, d_2, d_3, d_4, d_5, d_6;$$
$$P(s_2) = d_5, d_6, d_1, d_4, d_7, d_3, d_2;$$
$$P(s_3) = d_4, d_6, d_5, d_1, d_3, d_2, d_7.$$

A diákok preferenciáinak sorrendjei a következőféleképpen alakulnak:

$$P(d_1) = s_2, s_3, s_1;$$
$$P(d_2) = s_1, s_2, s_3;$$
$$P(d_3) = s_1, s_3, s_2;$$
$$P(d_4) = s_1, s_2, s_3;$$
$$P(d_5) = s_3, s_1, s_2;$$
$$P(d_6) = s_1, s_2, s_3;$$
$$P(d_7) = s_2, s_1, s_3.$$

Nézzük meg, milyen párosítást ad erre a feladatra a Boston mechanizmus. Első lépésben nézzük a diákok első helyén megjelölt iskoláit. Az s_1 iskolát a d_2, d_3, d_4, illetve a d_6 diák is megjelölte. Az s_1 iskolának a kapacitása 2, és a prioritássorrendjében ezek közül a diákok közül a d_2 és a d_3 diákok szerepelnek a legjobb helyen, tehát őket veszik fel, és d_4 és d_6 diákot elutasítják. Az s_2 iskolát a d_1 és d_7 diák jelölte meg első helyen és mivel az iskola kapacitása 3, ezért mindketten felvételt nyertek. Az s_3 iskolát csak a d_5 diák jelölte meg elsőnek és őt fel is vették. Az 1. táblázat mutatja 1. lépés utáni eredményét az algoritmusnak.

s_1	d_2, d_3
s_2	d_1, d_7
s_3	d_5

3.1. táblázat. 1. lépés után

Mivel d_4 és d_6 diák nem nyertek felvételt a kedvenc iskolájukba, folytatjuk az algoritmust. Mindkettőjüknek az s_2 iskola szerepel a második helyen. Az s_2 iskolában már csak egy sza-

bad hely van, és a prioritássorrendjében hamarabb szerepel d_6 diák, ezért őt veszik fel. A 2. táblázatban láthatjuk a 2. lépés után kapott eredményt.

s_1	d_2, d_3
s_2	d_1, d_7, d_6
s_3	d_5

3.2. táblázat. 2. lépés után

A d_4 diák marad már csak egyedül iskola nélkül, neki harmadik helyen megjelölt iskolája az s_3, ahol még van is szabad hely, ezért oda nyert felvételt.

s_1	d_2, d_3
s_2	d_1, d_7, d_6
s_3	d_5, d_4

3.3. táblázat. 3. lépés után

Az algoritmus tulajdonságai:

(i) Nem stratégia biztos, ezt a tulajdonságot tartják sokan az algoritmus legnagyobb hibájának. Hiszen egy diáknak hiába van magas prioritása egy iskolában, ha nem azt jelölte meg elsőnek, előfordulhat, hogy nem veszik oda fel, hiszen elveszti a prioritását azokkal a diákokkal szemben, akik első helyen jelölték meg az iskolát. Ez a mechanizmus arra ösztönzi diákokat, hogy ne a valódi preferenciáikat adják meg, hanem azokat az iskolákat vegyék előre, ahol magas a prioritásuk.

(ii) Igazmondás esetén a diákok számára Pareto-hatékony, hiszen egy diák sem kerülhet be jobb iskolába anélkül, hogy egy másik diáknak ne ártana ezzel. Mivel azonban sokan taktikáznak, egyáltalán nem biztos a Pareto-hetékonyság.

3.2. Columbus mechanizmus

A Columbus mechanizmus azon mechanizmusok közé tartozik, amely a diákok preferenciasorrendjét nem veszi figyelembe. Az algoritmus lépési a következőek:

1. lépés: Minden diák legfeljebb 3 különböző iskolába jelentkezhet.

2. lépés: Bizonyos iskolákban garantálva vannak a helyek az iskola körzetében lakó diákok számára. A fennmaradó jelentkezők prioritását sorsolással határozzák meg.

3. lépés: Minden iskola a szabad helyeit ajánlja fel a jelentkezőknek a prioritássorrend alapján, a többi jelentkezőt várólistára teszik. Az ajánlat megérkezése után 3 nap áll rendelkezésre a diákoknak, hogy elfogadják vagy elutasítsák azt. Ha egy diák elfogadja az ajánlatot, akkor felvették az adott iskolába. Ezt követően el kell utasítania a többi iskola ajánlatát, illetve a várólistákról is törölni kell. Amint ismét rendelkezésre állnak szabad helyek az iskolákban az elutasított ajánlatok miatt, a várólistán szereplő hallgatók számára tesznek ajánlatot.

Tulajdonságok:

(*i*) Nem egyértelmű az optimális stratégia. A diákok, függetlenül attól, hogy a mechanizmus nem veszi figyelembe, rendelkeznek preferencia-sorrenddel. Amikor egy diák ajánlatot kap a számára 2. vagy a 3. helyen preferált iskolától, akkor nem világos, hogy az optimális stratégia elfogadni, vagy elutasítani azt. Hiszen a diáknak van esélye bekerülni abba az iskolába, amelybe legjobban szeretne, ha elutasítja őket, de az is lehet, hogy így iskola nélkül marad. Az iskolák prioritássorrendje is nagy részben a szerencsén múlik, ezért a diákok nagyon nehéz helyzetben vannak.

(*ii*) Ezenkívül az sem biztos, hogy a diákok számára Pareto-hatékony megoldást kapunk. A legegyszerűbb példa erre, hogy van két iskola: s_1 és s_2 és két diák: d_1 és d_2, és d_1 az s_1 iskolát preferálja jobban, d_2 az s_2-t. Tegyük fel, hogy d_1 az s_2 iskolától kapott ajánlatot, d_2 az s_1-től, és mindkét diák elfogadja, tehát nem Pareto-hatékony megoldást kaptunk.

3.3. Gale-Shapley algoritmus

Ezt az algoritmust szokták röviden GS-algoritmusnak nevezni, illetve halasztott elfogadási (deferred acceptance) algoritmusnak is, amelyet az angol nevéből adódóan röviden DA-algoritmusként is emlegetünk. Ez az egyik legnépszerűbb és nagyon sok helyen alkalmazott algoritmus. Diákok szemszögéből lefuttatva az algoritmus lépései a következők:

(*i*) 1. lépés: Minden diák a legjobban preferált iskolájába jelentkezik. Ez alapján az iskolák ideiglenesen kiosztják a helyeiket a prioritássorrendjüknek megfelelően, a fennmaradó diákokat elutasítják.

(*ii*) k. lépés: Az előző lépésben elutasított diákok a preferencia-sorrendjükben a következő iskolába jelentkeznek. Az iskolák így újra kiosztják a helyeiket a prioritássorrend alapján.

(*iii*) Az algoritmus addig fut, amíg van olyan diák, akit még nem vettek fel sehova, de még nem ért végig a preferencia-listáján. Ha már minden ilyen diák elfogyott, az algoritmus leáll.

Iskolák szemszögéből lefuttatva az algoritmus lépései a következők:

(*i*) 1. lépés: Minden iskola prioritássorrendjének elejétől kezdve a kapacitásszámának megfelelő diáknak tesz ajánlatot. A diákok a preferencia-sorrendjük alapján a számukra legkedvezőbb ajánlatot elfogadják, a többit elutasítják.

(*ii*) k. lépés: Minden iskola a prioritássorrendjében következő diákok közül annyinak tesz ajánlatot, ahány szabad helye van. A diákok a preferencia-sorrendjük alapján a számukra legkedvezőbb ajánlatot elfogadják, a többit elutasítják. A diák természetesen a korábban elfogadott ajánlatát is visszautasíthatja.

(*iii*) Az algoritmus addig fut, amíg van olyan iskola, akinek van szabad helye, de még nem ért végig a prioritáslistáján.

Nyilván a 2 algoritmus nem feltétlenül ugyanazt az eredményt adja.

3.3.1. Példa. Talán a legegyszerűbb példa az, amikor 2 iskolánk és 2 diákunk van és mindkét iskola kapacitása 1. Az d_1 diák az s_1-t iskolát, a d_2 diák az s_2-t preferálja jobban. Azonban az s_1 iskola a d_2-es diákot, az s_2 iskola a d_1 szeretné jobban felvenni. Ekkor diákok felől futtatva az algoritmust a d_1-es diák az s_1 iskolába, a d_2-es diák az s_2-es iskolába kerül be. Iskolák felől futtatva azonban a d_1-es diák az s_2 iskolába, míg a d_2-es diák az s_1 iskolába nyer felvételt.

Nézzük meg, hogy egy kicsit nagyobb feladatra hogy fut le diákok felől az algoritmus. Vegyük például a 3.1.1 példában megadott feladatot.

3.3.2. Példa. Első lépésben az s_1 iskola helyeit ideiglenesen megkapja a d_2 és d_3 diák, az s_2 iskolába ideiglenesen felvételt nyert d_1 és d_7 diák, az s_3 iskolába pedig d_5. Tehát első lépésben ugyanazt kaptuk, mint a Boston mechanizmus első lépése után.

s_1	d_2,d_3
s_2	d_1,d_7
s_3	d_5

3.4. táblázat. 1. lépés után

A d_4 és a d_6 diák maradt iskola nélkül. Mindkettőjüknek s_2 iskola szerepel a második helyen. Az s_2 iskola így ideiglenesen felveszi még a d_6 és d_4 diákot is, azonban kénytelen volt elutasítani a d_7 diákot, így ő iskola nélkül maradt.

s_1	d_2, d_3
s_2	d_6, d_1, d_4
s_3	d_5

3.5. táblázat. 2. lépés után

A d_7 diák preferencia-listáján szereplő következő iskola az s_1. Az s_1 iskola prioritáslistáján a d_7 megelőzi mindkét ideiglenesen felvett diákot, és mivel az s_1 kapacitása csak 2, így kénytelen elutasítani d_3-at, aki így iskola nélkül maradt.

s_1	d_7, d_2
s_2	d_6, d_1, d_4
s_3	d_5

3.6. táblázat. 3. lépés után

A d_3 preferencia-listáján az s_3 iskola következik. Ide fel is tudják venni, mivel van üres hely.

s_1	d_7, d_2
s_2	d_6, d_1, d_4
s_3	d_5, d_3

3.7. táblázat. 4. lépés után

Mivel minden diáknak jutott hely, ezért az ideiglenes felvételeket véglegesíteni lehet.

Nézzük meg diákok felől futtatott GS-algoritmus a tulajdonságait.

(i) Stratégia biztos algoritmus, tehát a diákoknak megéri megadni a valós preferenciáikat. A diákok a preferencia-sorrendjüknek megfelelően haladtak, és úgy kerültek az i. iskolába, hogy jobban preferált i-1 iskola már elutasította őket, mert találtak náluk számukra jobb diákokat. Tehát ezekbe az iskolákba semmiképpen sem tud bekerülni a diák, függetlenül attól, hogy a valódi preferenciáját adta meg vagy sem. Viszont ha hamis preferenciát adott

meg, és az i. iskola helyére egy olyan iskola került, amit kevésbé preferál, elképzelhető, hogy bekerül oda, és akkor valójában rosszabbul jár.

(*ii*) Diák-optimális stabil párosítást eredményez. Ha a diákok közül valaki jobban szeretne egy másik iskolába járni, az csak úgy lehetséges, hogy már jelentkezett oda, de visszautasították, hiszen diákok a preferencia-sorrendjüknek megfelelően haladtak. Ennek következtében a diákok a lehető legjobb helyre kerülnek be. Az elképzelhető viszont, hogy az iskolák szívesebben vennének fel más diákokat, de ezek a diákok nem jelentkeztek oda, mert találtak számukra preferáltabb iskolát. Tehát ez a stabil párosítás a diákok számára lesz a legmegfelelőbb. Ha az iskolák részéről futtatnánk az algoritmust, nyilván iskola-optimális stabil párosítást kapnánk.

(*iii*) Nem eredményez Pareto-hatékony megoldást. Tekintsük a következő példát.

3.3.3. Példa. Legyen három iskolánk és három diákunk. Az iskolák prioritássorrendje a következő:

$$P(s_1) = d_1, d_3, d_2;$$
$$P(s_2) = d_2, d_1, d_3;$$
$$P(s_3) = d_2, d_1, d_3.$$

A diákok preferencia-sorrendje pedig:

$$P(d_1) = s_2, s_1, s_3;$$
$$P(d_2) = s_1, s_2, s_3;$$
$$P(d_3) = s_1, s_2, s_3.$$

A Gale-Shapley algoritmus a következő párosítást eredményezi.

s_1	d_1
s_2	d_2
s_3	d_3

3.8. táblázat. nem Pareto-hatékony párosítás

Ennek azonban a diákok számára Pareto-javítása a következő párosítás:

s_1	d_2
s_2	d_1
s_3	d_3

3.9. táblázat. Pareto-hatékony párosítás

3.4. Legjobb csere körök módszere

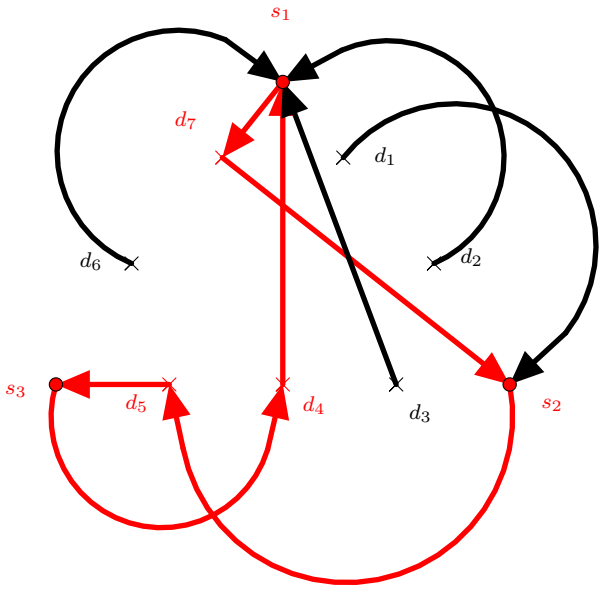
Ez a mechanizmus angol nevén Top Trading Cycles Mechanism-ként ismert, ebből adódóan TTC-mechanizmusként szokták leggyakrabban emlegetni itthon is ezt a módszert. Az a lényege, hogy az iskolák által legjobbnak tartott diákok elcserélik a nekik szánt helyeket egymással. Nézzük meg az algoritmus lépéseit!

(*i*) 1. lépés: Minden iskolához rendeljünk egy számlálót, ami nyomon követi, hogy hány ülőhely áll rendelkezésre az iskolákban. Ez most az iskolák kapacitását jelenti. Az iskolák és a diákok feleljenek meg egy gráf csúcsainak. Minden diákból mutasson egy irányított él a legjobban preferált iskolára. Minden iskolából mutasson egy irányított él arra a diákra, akinek a legmagasabb a prioritása az adott iskolában. Mivel az iskolák és a diákok száma is véges, ezért van legalább egy irányított kör a gráfban, melyben az iskolák és a diákok váltakozva szerepelnek. Minden iskola és minden diák legfeljebb egy körnek lehet része. A kör minden diákja helyet kap abban az iskolában, akire mutat. Ezeket a diákokat kiszedjük a gráfból. A kör minden iskolájának a számlálója 1-gyel csökken, és ha 0-ra csökken, akkor azt az iskolát is ki kell venni. A többi iskola számlálója nem változik.

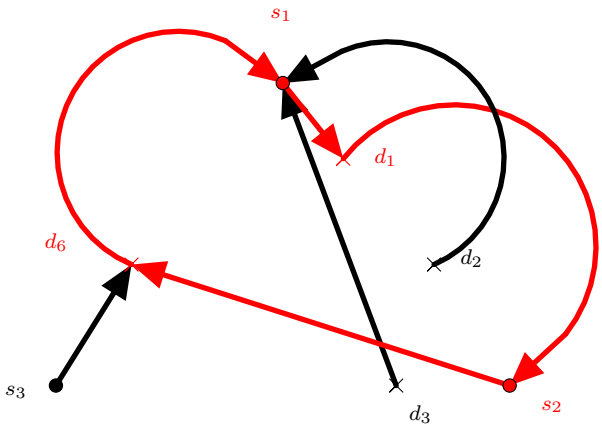
(*ii*) k. lépés: Minden megmaradt diákból mutasson egy irányított él a legjobban preferált iskolára a megmaradt iskolák közül és minden megmaradt iskolából mutasson egy irányított él a legmagasabb prioritású diákra. Ezután ugyanúgy folytatódik, mint az első lépés.

(*iii*) Addig ismételjük a k. lépést, amíg van olyan diák, akit nem vettek fel sehova, de még nem értek végig a preferencia-listáján. Ha már minden ilyen diák elfogyott, az algoritmus leáll.

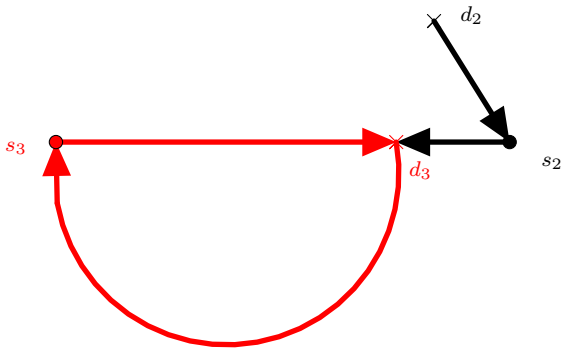
3.4.1. Példa. Nézzük meg, hogy mit ad a TTC-algoritmus a 3.1.1 példában megadott feladatra eredményül. Ekkor a kapacitásoknak megfelelő számlálók: $c_{s_1}(1) = 2$, $c_{s_2}(1) = 3$, $c_{s_3}(1) = 3$. Rajzoljuk fel az irányított gráfot!



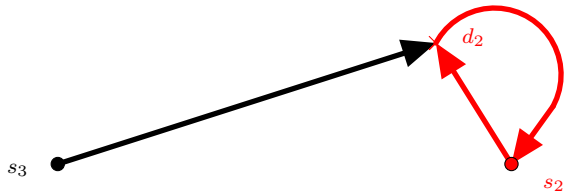
Az 1. lépésben látható, hogy egy irányított körünk van: $(s_1, d_7, s_2, d_5, s_3, d_4)$. Ez azt jelenti, hogy a d_4 diák felvételt nyert az s_1 iskolába, a d_7 diák az s_2-be, illetve a d_5 az s_3-ba. Ezáltal a 2. lépésre mindegyik iskola számlálója 1-gyel csökkent, így $c_{s_1}(2) = 1$, $c_{s_2}(2) = 2$, illetve $c_{s_3}(2) = 2$.



Itt is egy irányított kör látható: (s_1, d_1, s_2, d_6), azaz a d_6 diák felvételt nyert s_1-be, d_1 diák s_2-be. Emiatt a felvett diákok és s_1 kikerült a rendszerből a kör végén. A 3. lépés elejére két diák maradt már csak: d_2 és d_3, és két iskola, melyeknek számlálója a következő: $c_{s_2} = 1$ és $c_{s_3} = 2$.



A (s_3, d_3) irányított körből látható, hogy s_3 és d_3 kölcsönösen egymásra mutattak, tehát d_3 bekerült az s_3 iskolába. A 4. kör elejére $c_{s_2} = 1$ és $c_{s_3} = 1$, és csak d_2 diák maradt.



Mivel itt már igaziból csak d_2 döntése számított, ezért d_2 bekerült az s_2 iskolába. Az algoritmus végére érve a következő párosítást kapjuk:

s_1	d_4, d_6
s_2	d_7, d_1, d_2
s_3	d_5, d_3

3.10. táblázat. TTC eredmény

Az algoritmus tulajdonságai:

(*i*) Stratégiabiztos, azaz a diákoknak megéri a valós preferenciájukat megadni. Tegyük fel, hogy a diák a k. lépésben hagyja el az algoritmust, ha a valós preferenciáját adja meg. Az algoritmus minden lépésében a számára legmegfelelőbb helyre mutatott, de ezek a helyek már elhagyták az algoritmust a k. lépés előtt. Tehát ezekbe a jobb iskolákba semmiképpen sem tudott bekerülni, akár az igazi, akár hamis preferenciát ad meg, hiszen ez nem befolyásolja azt. Hamis preferencia viszont árthat, mert elképzelhető, hogy a diák így bekerül egy olyan iskolába, ami számára valójában rosszabb, mint ahova bekerült volna a k. lépésben.

(*ii*) A diákok számára Pareto-hatékony. Minden diák, aki elhagyta az algoritmust az 1. lépésben, a számára legjobb iskolába került. Minden diák, aki a 2. lépésben hagyta el az algoritmust, a megmaradt iskolák közül a számára legjobba került, ezért nem fordulhat elő olyan, hogy jobb iskolába kerül, anélkül, hogy bántana valakit, aki az 1. lépésben hagyta el az algoritmust. Ebből következik, hogy senki sem kerülhet jobb iskolába anélkül, hogy bántana valakit, aki az előző lépések valamelyikében hagyta el az algoritmust.

(*iii*) Nem stabil, ld. 3.1.1 példa.

3.5. Melyik mechanizmust érdemes választani?

Ebben az alfejezetben a Boston algoritmust, a GS-algoritmust és a TTC-algoritmust szeretnénk összehasonlítani, megvizsgálni azokat az eshetőségeket, amikor érdemesebb valamelyiket használni.

Először hasonlítsuk össze a Boston mechanizmust a GS-algoritmussal. Az algoritmusok leírásánál említettük, hogy a Boston mechanizmus nem stabil, nem stratégiabiztos és mivel ez taktikázásra kényszeríti a diákokat, ezért valószínűleg nem Pareto-hatékony párosítást kapunk. A GS-algoritmus ugyan nem Pareto-hatékony, de stratégiabiztos, illetve kiküszöböli az indokolt irigységet és a pazarlást, tehát stabil is. Ez arra enged következtetni, hogy a GS-algoritmus sokkal jobb, mint a Boston mechanizmus. Sok amerikai nagyvárosban emiatt 2005 körül lecserélték az addig alkalmazott Boston mechanizmust GS-mechanizmusra. Szerintem azt, hogy melyik a jobb mechanizmus, azt nem lehet ennyire egyértelműen eldönteni. Tekintsük a következő példát:

3.5.1. Példa. Legyen három diákunk, d_1, d_2, d_3 és három iskolánk s_1, s_2, s_3 és minden iskola egy szabad férőhellyel rendelkezik. Az iskoláknak nincsen prioritássorrendjük, minden diákot

egyenlő esélyekkel induló jelentkezőnek tekint. A diákok preferencia-sorrendje megegyezik, legyen például a következő: $P(d_1) = P(d_2) = P(d_3) = s_1, s_2, s_3$. Ekkor bármely párosítás stabil. A diákok kardinális hasznosságainak értékei az alábbi táblázatban található.

	d_1	d_2	d_3
s_1	0,8	0,8	0,6
s_2	0,2	0,2	0,4
s_3	0	0	0

3.11. táblázat. kardinális hasznosság

Az iskolák prioritássorrendjét sorsolással határozzák meg. A GS-mechanizmus esetén minden diák a valódi preferenciáját adja meg, emiatt minden diák $\frac{1}{3}$ valószínűséggel kerül be s_1-be, $\frac{1}{3}$ valószínűséggel s_2-be és $\frac{1}{3}$ valószínűséggel s_3-ba. Ezért a várható hasznosságok a következőképpen alakulnak: $EU_{d_1}^{GS} = EU_{d_2}^{GS} = \frac{1}{3}(0,8 + 0,2) = \frac{1}{3}$ és $EU_{d_3}^{GS} = \frac{1}{3}(0,6 + 0,4)$. A Boston mechanizmus esetén d_3 diák s_2 iskolát jelöli meg az első helyen. Mivel ide biztos bekerül, ezért $EU_{d_3}^{B} = 0,4$. A másik két diák $\frac{1}{2}$ valószínűséggel kerül be s_1-be és s_3-ba szintén. Ennek következtében a várható hasznosságuk $EU_{d_1}^{B} = EU_{d_2}^{B} = \frac{1}{2}(0,8 + 0) = 0,4$. Tehát mindegyik diák a várható hasznossága javult. A Boston mechanizmusnak a kardinális hasznosságot nem kell megadni. Ez egy természetes szelekció, hiszen csak azok a diákok fogják bejelölni a jobban preferált, de rizikósabb iskolát, akiknek sokkal nagyobb hasznosságot jelent. Azok a diákok, akik számára kevesebb hasznosságot jelent ez a jobb iskola, hajlamosabbak bejelölni elsőnek a kevésbé preferált iskolát, ha úgy sejtik, hogy oda biztosan felvételt nyernek.

Ennek ellenére sok helyen, köztük Bostonban is, lecserélték a Boston mechanizmust a GS-mechanizmusra, amely mellett az egyik legfontosabb érv a stabilitás. Ez egyfajta védelem a döntéshozók számára, hiszen egy diák visszautasítását könnyen meg lehet indokolni.

A következőkben hasonlítsuk össze a két stratégiai szempontból biztos mechanizmust, azaz a GS-mechanizmust és a TTC-mechanizmust. A GS-mechanizmus esetén nincs blokkoló pár, azaz kiküszöböli az indokolt irigységet, de nem hatékony.

3.5.2. Tétel (Roth, 1982)**.** *Nincs olyan diákokra vonatkozó hozzárendelési mechanizmus, amely egyszerre hatékony és az indokolt irigységet is kiküszöböli.*

Néhány országban elsődleges szempont az indokolt irigység megszüntetése, ezért ott GS-mechanizmust alkalmaznak, máshol a hatékonyság a fontosabb, ezért ott TTC-algoritmust szeretik alkalmazni. A GS-algoritmus a főiskolai felvételi rendszerekben nagyon népszerű, hiszen meg lehet adni egy ponthatárt, ami alatt nem veszik fel a diákokat. Ez egy mindenki

számára elfogadható bizonyíték az elutasításra. A TTC olyan Pareto-hatékony stratégiabiztos mechanizmus, amely minimalizálja az indokolt irigységet.

Tegyük fel, hogy elszeparálják a kerületeket, azaz a diákokat nem engedik más kerületbe jelentkezni, csak ahol laknak. Minden egyes kerületben GS-algoritmust alkalmaznak, így minden kerületben kijön egy diák optimális stabil megoldás. Ezután egyesítjük a kerületeket, de a kerületi prioritásokat megtartjuk. Azaz minden diák jelentkezhet bármelyik városon belüli iskolába, de minden diáknak nagyobb prioritása van a saját kerületén belüli iskolákba, mint egy másik kerületből jövő diáknak. Szeretnénk megvizsgálni, hogy az egyesítés előnyös-e vagy sem. Tegyük fel, hogy minden kerületben annyi diák, mint amennyi férőhely, tehát az iskolák telítettek. A kerületenkénti stabil megoldás stabil megoldás marad az egyesítés után, legfeljebb nem diák optimális. Nézzük meg a következő példát.

3.5.3. Példa. Legyen most két diákunk: d_1, d_2 és két iskolánk: s_1, s_2, melyek kapacitása 1. Az s_1 iskola abban a kerületben van, ahol d_1 lakik, így d_1 csak s_1-be jelentkezhet. Hasonlóképpen d_2 is csak s_2-be jelentkezhet. Ha egyesítjük a kerületeket, és bármelyik iskolába lehet jelentkezni, akkor kiderül, hogy valójában d_1 az s_2 iskolát preferálja, míg d_2 az s_1-et. Az eredeti párosítás az egyesítés után is stabil marad, hiszen az új élek nem lehetnek blokkoló élek, mivel a kerületi diákok magasabb prioritással rendelkeznek. Ez azonban nem zárja ki azt, hogy ne lehetne a diákok számára kedvezőbb stabil párosítást találni. Ha a diákok cserélnek, mindketten jobban járnak. Ha az iskolák telítettek, akkor biztos, hogy mindegyik diák legalább olyan jó helyre jut be, mint az egyesítés előtt. Tehát a kerületenkénti diák optimális stabil párosítás az egyesítés után is stabil marad, de ennél csak jobb lehet a városra vetített diák optimális stabil párosítás.

Fontos megemlíteni, hogy miszerint a diákok egyesítés után jobban járnak akkor is igaz, ha az iskolák nem telítettek, azaz vannak szabad férőhelyek még az iskolákban. Ezt úgy a legegyszerűbb végiggondolni, hogy feltesszük, hogy az iskolák telítettek, és utána egyenként nyitjuk meg az üres helyeket. Így is javulhat a diákok helyzete, hiszen ha megnyitunk egy üres helyet, akkor felvételre kerül egy új diák, és egy másik helyen támad üresedés, ott is felvételre kerül egy új diák, és így folytatva végül egy népszerűtlen iskolában marad egy üres hely. Tehát a diákok helyzete az egyesítés hatására csak javulhat.

Ez a tulajdonság a TTC-algoritmus esetén nem teljesül.

3.5.4. Példa. Legyen két kerületünk, az egyik kerületben két diák: d_{11} és d_{12}, és két iskola: s_{11} és s_{12}, a másik kerületben egy diák: d_{21} és egy iskola: s_{21}. Az iskolák prioritássorrendje a

következő:

$$P(s_{11}) = d_{11}, d_{12}, d_{21};$$
$$P(s_{12}) = d_{12}, d_{11}, d_{21};$$
$$P(s_{21}) = d_{21}, d_{11}, d_{12}.$$

A diákok preferencia-sorrendje pedig:

$$P(d_{11}) = s_{12}, s_{11}, s_{21};$$
$$P(d_{12}) = s_{21}, s_{11}, s_{12};$$
$$P(d_{21}) = s_{12}, s_{21}, s_{11}.$$

Ha a TTC-algoritmust kerületenként külön futtatjuk, azt kapjuk, hogy d_{11} diák az s_{12}, d_{12} diák az s_{11}-be és d_{21} diák az s_{21} iskolába nyert felvételt. Ugyanakkor ha egyesítjük a kerületeket, d_{12} az s_{21}-be, d_{21} az s_{12}-be nyer felvételt, így d_{11} viszont csak s_{11}-be, így ő az egyesítés után rosszabb helyzetbe került, mint az egyesítés előtt. Tehát nem feltétlenül jár jól minden diák az egyesítéssel.

Más alkalmazások esetén a TTC-algoritmus lehet kedveltebb. Nézzük a Columbusi iskolaválasztást. Mint már említettük, a kerületben lakó diákoknak prioritásuk van a kerületi iskolákba, a maradék helyekre a többi diák prioritását sorsolással határozzák meg. Mivel itt az iskolák prioritása nagy részben a véletlentől függ, ezért ebben az esetben a hatékonyság fontosabb szempont lehet, mint a stabilitás.

4. fejezet

Európában alkalmazott felvételi eljárások

4.1. Magyar egyetemi felvételi

Ezt az alfejezetet a [5], [6], [10], [13] és [14] források alapján írtam. Magyarországon a központosított felsőoktatási felvételi eljárásban alapképzésre, mesterképzésre, osztatlan mesterképzésre, illetve felsőoktatási szakképzésre van lehetőségük jelentkezni a diákoknak. Keresztféléves eljárásban a februárban induló képzésekre, általános eljárásban a szeptemberben induló képzésekre tudnak jelentkezni. A februárban induló képzések jelentkezési határideje az azt megelőző év november 15-e, a szeptemberben induló képzések jelentkezési határideje ugyanaz az év február 15-e. Ezenkívül pótfelvételi eljárást is meghirdetnek, amelynek jelentkezési határideje augusztus 15. Itt már csak 1 helyre tudnak jelentkezni a diákok. Azok a diákok, akik a pótfelvételi eljárással nyernek felvételt, szintén el tudják kezdeni a tanulmányaikat szeptemberben. A képzési forma finanszírozás szerint lehet költségtérítéses vagy állami ösztöndíjas. Erről kicsit részletesebben később lesz szó. A keresztféléves illetve az általános felvételi eljárás során ingyenesen három, összesen hat szakra lehet jelentkezni, az ingyenes helyeken felül megjelölt további helyekért egyenként 2000 Ft-os díjat kell fizetni. A diákoknak lehetőségük van mindegyik kiválasztott szaknak, mind az állami ösztöndíjas, mind a költségtérítéses finanszírozási formáját is megjelölni, ezek nem vesznek igénybe újabb költségeket, azonban a prioritássorrendet ezeknek a figyelembevételével kell összeállítani. A felvételi algoritmus a diákok felől futtatott Gale-Shapley módszeren alapszik, azonban néhány dologgal szükséges volt kiegészíteni. A következőkben a módosítások okát fogom elsősorban feltárni, a módosított algoritmus lépéseit nem fogom részletesen ismertetni.

Minden szakra szükséges volt bevezetni a diákok számára vonatkozó maximális korlát mellett egy minimális korlátot is. Ezek a korlátok egy adott szakon belül is finanszírozási formákat tekintve eltérőek. Alapképzés, osztatlan mesterképzés és felsőoktatási szakképzés esetén ezeket a korlátokat kapacitásoknak, mesterképzés esetén keretszámoknak nevezzük. Az alsó korlát bevezetése azért volt szükséges, mert ha a hallgatók létszáma egy bizonyos szintet nem üt meg, akkor az intézmény már nem tudja fedezni a képzés költségeit, tehát nem éri meg elindítania az adott képzést. Az alsó korlát a felső korláthoz hasonlóan képzésenként eltérő. Jelöljük az s iskolához tartozó alsó korlátot $a(s)$-sel, a felső korlátot pedig $c(s)$-sel, mint ahogyan azt az 2. fejezetben is jelöltük. Tehát minden s iskolához tartozik egy $a(s)$ alsó, illetve egy $c(s)$ felső korlát, melyekre igaz, hogy $a(s) \leq c(s)$. Az alsó korlát bevezetése sok mindent megváltoztat. Elsősorban a párosítás definíciója kiegészül egy új feltétellel.

4.1.1. Definíció. Kiegészítés a párosítás definíciójához:

(*iv*) $\forall s \in S$-re $m(s) \setminus \{s\} = \emptyset$, vagy $\mid m(s) \setminus \{s\} \mid \geq a(s)$, azaz vagy nem vesznek fel senkit (bezárt iskola), vagy a felvett diákok száma biztosan eléri az alsó korlátot (nyitott iskola).

Emiatt a stabilitás definícióját is szükséges módosítanunk.

4.1.2. Definíció. Egy párosítást zárt-stabil párosításnak nevezünk, ha nincsenek blokkoló élek, és nem létezik olyan s bezárt iskola, amelyre $\exists D' \subseteq D$, $\mid D' \mid \geq a(s)$ úgy, hogy $\forall d \in D'$-re $s >_d m(d)$.

Azaz annyival bővül a stabilitás definíciója, hogy alsó korlát létezése esetén akkor is indokolt a diákok elutasítása, ha az elutasított diákok száma nem éri el a szak alsó korlátját. Az alsó korlát feltételezése mellett annak eldöntése, hogy létezik-e stabil párosítás az NP-teljes problémák közé tartozik, azaz pillanatnyilag nem ismerünk olyan algoritmust, amely polinomiális futási időn belül erre a kérdésre választ tudna adni. Lásd [6]. A magyar felsőoktatásban erre egy heurisztikát alkalmaznak, azaz olyan algoritmust, ami gyorsan talál megoldást, de nem biztos, hogy stabil megoldást kapunk, akkor sem, ha létezik ilyen megoldás. Az algoritmusnak a [6] és [10] forrásokban lehet utánaolvasni.

Minden egyetemi felvételi képzésre, azaz (szak, finanszírozási forma) párra vonatkoznak felső korlátok. Ezenkívül a kari közös korlát ugyanazon szak államilag finanszírozott és költségtérítéses diákjainak a halmazára értendő. Azokra a diákokra, akik államilag finanszírozott formában jelentkeznek egy adott szakra, vonatkozik egy úgynevezett országos közös korlát is, tehát például legfeljebb 3000 informatikus hallgatót támogat az állam. A végső felvételi eredménynek eleget kell tennie minden korlátnak. Tehát ha az előbb említett informatikus hallgatókat nézzük, akkor

a gyengébb egyetemeknél inkább ez az országos korlát lehet korlátozó tényező, mint az egyetem képzésére vonatkozó korlátja. Matematikailag megfogalmazva a közös korlát azt jelenti, hogy az iskolák egy-egy részhalmazához is adott egy felső korlát. A következőkben jelöljük $Z \subseteq 2^S$-vel az iskolák azon részhalmazainak a családját, melyekhez tartozik egy felső korlát. Legyen $c(z)$ az iskolák $z \subseteq S$ halmazához tartozó felső korlát. Ezek a közös korlátok. Ennek értelmében $c(s)$ jelentése nem változik. Szükségünk van a párosítás definíciójának módosítására, hogy a közös korlátok feltételeinek is eleget tegyen.

4.1.3. Definíció. Módosítás a párosítás definíciójának (ii) részében:

(ii) $\forall z \in Z$-re $\mid m(z) \mid = \mid \bigcup_{s \in z} m(s) \mid = \sum_{s \in z} \mid m(s) \mid = c(z)$.

Legyen P_z a $z \in Z$ iskolák közös preferencia-listája. Ekkor tegyük fel, hogyha $s \in z$, akkor $d >_s d' \Leftrightarrow d >_z d'$. Ez azt jelenti, hogy azoknak az intézményeknek, amelyeknek van közös korlátjuk, ugyanúgy kell értékelniük a közös jelentkezőket. A blokkoló él és a stabilitás definíciója a következőképpen módosul.

4.1.4. Definíció. Az m párosítás tartalmaz blokkoló élt, ha létezik olyan párosítatlan (d, s) pár, ahol $s >_d m(d)$ és vagy $s \in m(s)$ vagy $\exists d' \in m(z) : d >_z d'$ $\forall z \ni s$-re. Egy párosítás stabil, ha nem tartalmaz blokkoló éleket.

Azaz ha egy diákot elutasítottak egy iskolából, akkor vagy az iskola korlátja lett feltöltve jobb diákokkal, vagy az iskola benne van egy olyan közös korlátba, ami fel lett töltve jobb diákokkal. A közös korlát feltételezése mellett annak eldöntése, hogy létezik-e stabil párosítás, szintén az NP-teljes problémák közé tartozik, lásd [6]. Azonban érdemes megemlíteni a halmazrendszereknek egy speciális fajtáját.

4.1.5. Definíció. Egy $Z \subseteq 2^S$ halmazrendszer egymásba ágyazott, ha $\forall z, z' \in Z$-re, ahol $z \bigcap z' \neq \emptyset$ fennáll, hogy vagy $z \subseteq z'$ vagy $z' \subseteq z$.

Az egymásba ágyazott korlátok esetén mindig létezik stabil megoldás, sőt van olyan algoritmus, amellyel a diák-optimális stabil párosítást kapjuk, lásd [6].

Most nézzük meg, hogyan rangsorolják az egyetemek a diákokat. Alapképzés, osztatlan mesterképzés, illetve felsőoktatási szakképzés esetén a diákok összpontszámát 400+100 pontos rendszerben számítják. A 400 pont a tanulmányi pontok és az érettségi pontok összeadásával, vagy az érettségi pontok duplázásával jön létre, attól függően, melyik pontszámítással jár jobban a diák. A 100 pontot a többletpontok adják, mely emelt szintű érettségi vizsgaeredményért, nyelvvizsgáért, illetve esélyegyenlőség miatt kapható jogszabály alapján. Képzési területenként

tanulmányi és művészeti versenyeredmények, szakképesítés vagy sporteredmény alapján is adhatóak többletpontok. Mivel többletpontokból 100-nál is több pont összejöhet, ilyenkor 100 pontnak veszik a többletpontot. Mesterképzés esetén a diákok összpontszámát 100 pontos rendszerben számítják. Ebben az esetben a felsőoktatási intézményeknek néhány kötelező megkötést figyelembe kell venni, de nagyobb szabadságuk van a pontszámítást illetően, mint az előbb leírt esetben. Ebből az okból kifolyólag sok helyre írásbeli és szóbeli felvételi is szükséges. Minden képzés esetén van egy jogszabályban meghatározott minimumpontszám, melyet a diákoknak el kell érniük ahhoz, hogy részt vehessenek a felvételi eljárásban. 2021. februárban induló képzésekre a minimumpontszám a következő: alapképzés és osztatlan mesterképzés esetén 280 pont, felsőoktatási szakképzés esetén 240 pont, mesterképzés esetén 50 pont. A felvételi eredmény a ponthatárok megállapításával dől el. Minden képzéshez, azaz (szak, finanszírozási forma) párhoz tartozik egy ponthatár. Ezzel rögtön egyértelmű lesz, ki hova került be, mivel a diákok a megadott preferencia-sorrendjük szerint az első olyan helyre nyernek felvételt, melynek elérték a ponthatárát. Felmerül a kérdés, hogy mi történik holtverseny esetén, hiszen feltehetően van olyan alapképzés, melyre 221-nél többen jelentkeztek, illetve olyan mesterképzés is, melyre 51-nél többen jelentkeztek, azaz biztos hogy lesz legalább két olyan diák, akik azonos pontszámot értek el a felvételin. Ez azért probléma, mert azonos pontszámú diákok esetében az intézmények elveszítik a szigorú preferenciarendezésüket. Az a megállapodás itthon, hogy ugyanannyi ponttal rendelkező diákok esetében vagy mindegyik diákot felveszi az adott intézmény, vagy egyiket sem. Azaz a ponthatár alatt lévő összes diák elutasításra kerül, akik viszont elérik a ponthatárt, azok közül mindenki felvételt nyer. Ennek figyelembevételével kell meghúzni a ponthatárt. Ez azért nehéz döntés, mert ha például egy szak esetében 400 pontnál húzzák meg a ponthatárt, és így 100 diák nyer felvételt, akkor belekalkulálva azokat a hallgatókat, akik valamilyen okból kifolyólag abbahagyják a képzést, elképzelhető, hogy a későbbiekben nem lesz elég diák a szak fenntartásához. Azonban ha 399-re állítják a ponthatárt, és 30 olyan diák van, aki erre a szakra szeretne menni legjobban, és mindegyiküknek 399 pontja van, akkor 130 diák nyer felvételt az adott képzésre. Elképzelhető, hogy így már túl sokan lesznek, például több gyakorlati csoportot kell indítani, amihez nem áll rendelkezésre elég oktató, vagy tanterem. Vezessünk be két definíciót.

4.1.6. Definíció. A ponthatárokat H-lehetségesnek nevezzük, ha azok mellett egyetlen felső korlátot sem lépünk túl.

4.1.7. Definíció. A ponthatárokat L-lehetségesnek nevezzük, ha csak azok az iskolák lépik túl a felső korlátjukat, ha egyel növelve a ponthatárukat nem érnék el a felső korlátjukat.

Ezek alapján általánosítsuk a stabilitás fogalmát.

4.1.8. Definíció. A ponthatárok H-stabilak, ha H-lehetségesek és minden ponthatár vagy 0, vagy olyan, hogy 1-gyel csökkentve már nem H-lehetséges.

4.1.9. Definíció. A ponthatárok L-stabilak, ha L-lehetségesek és minden ponthatár vagy 0, vagy olyan, hogy 1-gyel csökkentve már nem L-lehetséges.

Mind iskolák, mind diákok felől futtatva létezik olyan algoritmus, mely H-stabil eredményt ad, és olyan is, mely L-stabil eredményt ad, lásd [5]. A diákok felől futtatott L-stabil párosítás optimális a diákok számára, lásd [5]. Azaz általános értelemben vett stabil és diák-optimális megoldás holtverseny esetén is adható. Azonban a diákok tudják úgy manipulálni a preferencialistájukat, hogy az eredmény számukra kedvezően alakuljon, azaz az algoritmus nem mentes a taktikázástól, lásd [5]. Holtversenyek feloldására lehetne alkalmazni a skálafinomítást például úgy, hogy alapképzésnél az érettségik százalékos eredményét több tizedesjegy pontossággal veszik figyelembe, mesterképzés esetén a szóbeli felvételin nem feltétlenül egész pontszámokat kapnak a diákok, vagy valamilyen egyéb tulajdonság, például a diákok neve vagy születési dátuma alapján állítják fel a sorrendet, és így szigorú prioritássorrendet kapnának az egyetemek.

4.2. Francia egyetemi felvételi

Ezt az alfejezetet a [7], [8], [9] és [11] források alapján írtam. A francia egyetemi felvételi eljárásban egy korábbi újításnak köszönhetően pár évig randomizált eljárást használtak egy szoftver hibája miatt. Ez a hiba 2016-ig észrevétlen maradt, azonban 2016-ban nagyon sok diákot érintett. Az érintett diákoknak lehetőséget adtak, hogy újra felvételizzenek, ennek ellenére a diákszervezetek egész nyáron tüntettek. Ezenkívül a szelekció is jelen volt a rendszerben. A diákok lakóhelyének elhelyezkedést is figyelembe vették a felvételi eljárás során a tanulmányi eredmények mellett, és az egyetemek sok esetben a közelebb lakó diákokat részesítették előnyben. Ez a szelekció rengeteg diákot érintett és ezt sokan nem tartották igazságosnak, hiszen ebből az okból kifolyólag egy kevésbé jó képességű diák előnyben részesülhetett egy jó egyetemen egy nagyon okos diákkal szemben, pusztán azért mert közelebb lakott. Emiatt a diákok tanulmányi teljesítménye is romlott, mert nem a tanulmányi eredmény volt a legmeghatározóbb a felvételinél. Ráadásul nemcsak így szelektálták a diákokat, hanem a diákok a jelentkezési sorrendjük szerint is prioritást kaptak. Vagyis azok a diákok, akik 1. helyen jelöltek meg egy szakot, ők több plusz-pontot kaptak, mint akik 2. helyre tették. Vagyis Boston szerű volt az eljárás, tehát nem volt stratégiailag biztos. Szükségessé vált egy új rendszer létrehozása. Ezért létrehozták a

Parcoursup digitális platformot, melynek kódját mindenki számára elérhetővé tették, így bárki ellenőrizni tudja, hogy a platform működése megfelel a törvényeknek. Ez a platform lehetőséget ad a felsőoktatásban továbbtanulni kívánó diákoknak, hogy megjelöljék azokat az egyetemek által indított képzéseket, amelyek érdeklik őket, és válaszolhassanak a képzést indító egyetemek ajánlataira. Ez az eljárás 2018. január 22. és 2018. szeptember 21. között került először megrendezésre. A diákoknak előzetesen regisztrálniuk kell a platform felületén, majd itt tudják benyújtani a felvételi igényeiket. A diákoknak 10 képzést kell kiválasztaniuk, és nem kell rangsorolniuk őket. (Korábban 24 képzést kellett rangsorolva kiválasztaniuk.) Az egyetemek felállítják a prioritássorrendjüket úgy, hogy a diákok korábbi tanulmányi eredményeit, érettségi eredményeit, egyéb eredményeit veszik figyelembe. Ez jobban inspirálja a diákokat tanulásra, mint a régebben alkalmazott eljárás. Ezután iskolák felőli Gale-Shapley mechanizmus fut, de mivel az egyetemek nem ismerik a diákok valódi preferenciáit, ezért ezen a platformon ajánlatokat tesznek a diákoknak. Tehát első körben az egyetemek ajánlatot tesznek annyi diáknak, amennyi a kapacitásuk. Ha egy diák több egyetemtől is kapott ajánlatot, most kell döntenie, melyiket preferálja inkább. A diákoknak lehetőségük van egy ajánlatot véglegesen, vagy feltételesen elfogadni, azonban a többi ajánlatot vissza kell utasítaniuk. Ha véglegesen elfogad egy diák egy ajánlatot, akkor kikerül a rendszerből. Ha egy diák feltételesen fogad el egy ajánlatot, akkor nem kerül ki a rendszerből, továbbra is kap ajánlatokat, és ha egy jobb ajánlatot kap, akkor ezt a feltételesen elfogadott ajánlatot visszautasítja és a jobb ajánlatot fogadja el. Tehát a diák fenntartja annak a lehetőségét, hogy esetleg egy jobb iskolába kerüljön be. Fontos, hogy minden kör végén a diák csak egy ajánlatot tarthat meg. Természetesen a diák az összes kapott ajánlatát is visszautasíthatja, de ezt nem érdemes tennie, hiszen elképzelhető, hogy a továbbiakban egyik iskolától sem kap ajánlatot. A diákok válaszait rögzítik a platformon, majd sor kerül a következő körre. Az egyetemek annyi diáknak tesznek ajánlatot, ahány rendelkezésre álló szabad helyük maradt. Utána ismét a diákok válaszolnak az ajánlatokra, és így tovább. Mivel azonban nagyon sokáig elhúzódik az algoritmus, a diáknak törvényrendeletben meghatározott idő alatt kell döntést hozniuk. Ez első kör esetén 5 nap, majd egyre kevesebb, a végén már 1 nap alatt kell dönteniük. Ezzel gyorsítják az algoritmust, hogy minél több kör menjen le, de mivel a folyamat szeptember végéig tart, ezért valószínűleg nem fut végig az algoritmus, és így nem jutnak el az iskolák számára optimális megoldásig. Felmerülhet a kérdés, hogy miért nem a diákok felől fut az algoritmus, hiszen akkor diák-optimális megoldáshoz közeli eredményhez jutnának. Ha a diákok tennének ajánlatokat, és az egyetemek döntenének, hogy véglegesen elfogadják, feltételesen elfogadják, vagy elutasítják, akkor elképzelhető, hogy egy diák jelentkezését egy egyetem még az algoritmus elején feltételesen elfogadja, majd az algoritmus végén kap egy

másik diáktól egy jobb ajánlatot, az övét elfogadja, azt a diákot viszont, akitől az algoritmus elején kapott ajánlatot, visszautasítja, mert betelt a kapacitása. Az algoritmus így véget ér, ennek a diáknak már nincs lehetősége új iskolába jelentkezni, tehát nem kerül be egyetemre. Mivel nem fut végig idő hiányában az algoritmus, ezért nem járnának jobban a diákok ezzel a megoldással.

4.3. Egyéb felvételi eljárások

Említsünk meg még néhány példát. Spanyolországban az általános iskolai és a középiskolai felvételinél is Boston mechanizmust alkalmaznak. Az iskolák prioritása a területi elhelyezkedést veszi alapul, illetve előnyben részesítik azokat a diákokat, akiknek odajár a testvérük. A magyar középiskolai felvételi eljárás a felsőoktatási felvételi eljáráshoz hasonlóan szintén a diákok felől futtatott Gale-Shapley mechanizmuson alapszik, itt azonban szigorú sorrendet kell felállítaniuk az iskoláknak a diákok között. A finn felsőoktatásban az iskolák felől futtatott Gale-Shapley mechanizmust veszik alapul. Az ukrán felsőoktatásban egy félig központosított eljárás van, mely részben megegyezik az iskolák felől futtatott Gale-Shapley mechanizmussal. Lásd [12].

5. fejezet

Kapcsolat a biztosítással

Ennek a fejezetnek az ötletét a [11] forrásból vettem és fejtettem ki kicsit részletesebben. Az íráshoz felhasználtam a [13], [14], [15], [16], [17] és [18] forrásokat. A magyar felsőoktatás esetében a hallgatókat a teljesítményük alapján átsorolhatják állami ösztöndíjas képzésből önköltségesre, illetve visszasorolhatják önköltséges képzésből állami ösztöndíjasba. Az átsorolás törvényi hátterét a Nftv 48. § (2) bekezdése, valamint minden egyetemnél a Hallgatói követelményrendszer adja. Ez a 2015/16-os tanévben, illetve régebbi tanévekben felvett hallgatók számára azt jelenti, hogy ha egy állami ösztöndíjas hallgató legutóbbi 2 félévben teljesített krediteinek száma kevesebb, mint 27, akkor őt átsorolják önköltségesre. A 2016/17-ben vagy azután felvett hallgatók számára azt jelenti, hogyha a legutóbbi 2 félévben teljesített kreditek száma 36-nál kevesebb, vagy az átlag nem éri el az átlaglimitet, akkor kerül átsorolásra az állami ösztöndíjas hallgató. Az átlaglimit képzési területenként egy meghatározott érték, 2020 nyarán az ELTE-n 3,00 vagy 3,25 volt ez a limit. Minden egyes átsorolt hallgató helyére megpróbálnak visszasorolni egy hallgatót, azaz költségtérítéses hallgató is bekerülhet államilag finanszírozott helyre. Ehhez a hallgatónak a Neptunon keresztül nyilatkozatot kell tennie, hogy át szeretne kerülni állami finanszírozású képzésbe. Ennek az a feltétele, hogy az adott képzés ne legyen kizárva a visszasorolásból, illetve a diák teljesítse a következő feltételeket: a 2015/16-os félévben, vagy régebbi tanévekben felvett hallgató esetén a legutóbbi 2 félévben teljesített krediteinek száma legalább 27 legyen, 2016/17-ben vagy azután felvett hallgatók legutóbbi 2 félévben a teljesített krediteinek száma legalább 36 legyen, és az átlaga elérje az átlaglimitet. A nyilatkozatot leadó és a feltételeket teljesítő hallgatók között az egyetemek rangsort állítanak fel, és a legjobb tanulmányi eredményt nyújtó önköltséges hallgatók átkerülhetnek államilag finanszírozott helyekre. 2012 óta a diákoknak lehetőségük van állami részösztöndíjas képzésre is jelentkezni, azaz a hallgatók finanszírozási formája akár 3 féle lehet: állami ösztöndíjas, állami

részösztöndíjas, illetve önköltséges. A részösztöndíjas finanszírozási formát nem minden képzési területen lehet választani. A következő területeken van lehetőség ezt a képzési formát választani: műszaki, természettudományi, informatikai képzésekhez tartozó alapképzéseken, földmérő, földrendező mérnöki, mezőgazdasági és élelmiszeripari és gépészmérnöki, tájrendező és kertépítő mérnöki, az orvosi laboratóriumi és képalkotó diagnosztikai analitikus alapképzési szakon. Az átsorolás és visszasorolás itt is érvényes marad. Az állami ösztöndíjas hallgatók rosszabb teljesítmény nyújtása esetén átkerülhetnek részösztöndíjas, vagy önköltséges helyekre, illetve a legjobb teljesítményt elérő önköltséges hallgatók átkerülhetnek a "megüresedett" részösztöndíjas, vagy teljes ösztöndíjas hallgatók helyeire. Részösztöndíjas hallgatóként szintén van lehetőség bekerülni államilag finanszírozott helyre, illetve átkerülni önköltségesre. Ez az átsorolás, illetve visszasorolás, ami a magyar felsőoktatásban jelen van, párhuzamba állítható a kötelező gépjármű-felelősségbiztosítási szerződések bonus-malus besorolásával. Ebben a rendszerben a pozitív fokozatok a bonus osztályok. Ez egyfajta díjkedvezményt jelent, a biztosító ebbe az osztályba sorolással jutalmazza a biztosítottat a kármentes vezetéséért. A negatív fokozatok a malus osztályok, vagyis ha a biztosított kárt okoz, azaz igénybe veszi a biztosítási szolgáltatást, a biztosító a kedvezmény megvonásával, vagy pótdíjjal büntet. A rendszer 1 alap (A00), 10 bunus (B) és 4 malus (M) osztályból áll. A legjobb besorolás a B10, a legrosszabb az M04. Új szerződés esetén a biztosító az üzembentartót az A00 osztályba teszi, kivéve ha az üzembentartó kárelőzményi igazolással rendelkezik. A következő évi besorolás a megfigyelési időszak alatt történt károkozások alapján kerül meghatározásra. A megfigyelési időszak az évfordulóváltástól számított 1 év (minimum 270 nap). Ha ezen időszak alatt az üzembentartó nem okozott kárt, akkor a besorolás 1 osztályt emelkedik, ha 1 kárt okozott, akkor 2 osztályt süllyed, ha 2 kárt okozott, akkor 4 osztályt süllyed, ha 3 kárt okozott, 6 osztályt süllyed, ha 4 kárt okozott, automatikusan M04-be, azaz a legrosszabb osztályba kerül. Így a biztosítottnak érdekévé válik, hogy ne okozzon kárt, hasonlóan a magyar felsőoktatásban a hallgatóknak érdekükké válik, hogy jól tanuljanak, hiszen ha egy jobb képességű hallgató elrontotta az érettségijét, és emiatt nem tudott bekerülni államilag finanszírozott szakra, akkor ha jól tanul, van lehetősége átkerülni. Ha azonban egy hallgató állami ösztöndíjasként nem veszi elég komolyan a tanulást, akkor ő kicsúszhat a támogatásból. Azaz azokon a szakokon, ahol csak állami ösztöndíjas illetve önköltséges finanszírozás van, az hasonló egy 2 osztályos bonus-malus rendszerhez, ahol megjelenik a részösztöndíjas finanszírozás, az egy 3 osztályos bonus-malus rendszerhez hasonlít. Azért nem teljesen feleltethető meg a kettő egymásnak, hiszen vannak olyan hallgatók, akik kimondottan önköltséges képzésre jelentkeztek, mert nem akarták vállalni a hallgatói szerződés feltételeit. Ezenkívül előfordulhat, hogy a jól tanuló önköltséges finanszírozásban résztvevő hallgatók nem

tudnak átkerülni ösztöndíjas képzésbe, mert például nem szabadul fel hely abban az esetben, ha az összes állami finanszírozásban részt vevő hallgató is nagyon jól tanul. Tehát a jól tanulás nem feltétlenül vezet jutalomhoz. Általában azonban szoktak kiesni hallgatók az állami ösztöndíjas képzésből. Mindenesetre a gépjármű-biztosításoknál alkalmazott bonus-malus rendszer mindenkit a kármentességre ösztönöz, a magyar felsőoktatásban alkalmazott átsorolás illetve visszasorolás szinte mindenkit a jó tanulásra ösztönöz (kivéve azokat a hallgatókat, akik egyéb okok miatt nem kívánnak állami ösztöndíjas hallgatók lenni). A biztosítók megfigyelték, hogy akik korábban kevesebb kárt okoztak, azok a későbbiekben is általában kevesebb kárt okoznak. Ez hasonlóan elmondható azokról a hallgatókról, aki kimagaslóan jól teljesítették az első évet az egyetemen, mert ők általában a következő években is hasonlóan jó eredményt fognak produkálni.

Érdemes említést tenni a Corvinus Ösztöndíjról is, mely egy teljesítményalapú ösztöndíj. 2019-ben a Budapesti Corvinus Egyetem alapítványi fenntartásba került, így az ösztöndíjrendszer is átalakult. Ez az ösztöndíj viszonylag magas pontszámmal bekerült diákok számára érhető el, de a magas bekerülési pontszám csupán az első 2 félévre garantálja az ösztöndíjat. Alapképzés esetén körülbelül 430-450 ponttól, mesterképzés esetén körülbelül 70-75 ponttól nyerhetik el a diákok ezt az ösztöndíjat. Az ösztöndíjra való jogosultságot évente megvizsgálják, így be is lehet kerülni az ösztöndíjprogramba, de akár ki is lehet kerülni belőle. Azoknak a diákoknak van esélyük erre az ösztöndíjra, akik az előző tanévben 46 kreditet teljesítettek, és legalább 3,8-as volt az átlaguk (ez az átlag függ az évfolyamátlagtól és a csoportátlagtól is, tehát lehet hogy jóval magasabb). Azokat a diákokat, akik ezeknek a szempontoknak megfelelnek, rangsorolják, és így csak a legjobb hallgatók nyernek támogatást a következő 2 félévre. Ebben a rendszerben sokkal tisztábban jelenik meg a versengés, mit az állami ösztöndíjas rendszernél, és emiatt a diákok sokkal jobban ösztönözve vannak arra, hogy jól tanuljanak.

6. fejezet

Szimulációs modell leírása

Szimuláció segítségével szeretném összehasonlítani a francia egyetemi felvételin alkalmazott iteratív Gale-Shapley mechanizmust az iskolák illetve a diákok felől futtatott direkt Gale-Shapley mechanizmussal olyan módon, hogy figyelembe veszem a diákok költségeit. (Direkt mechanizmus esetében a diákok előre rögzítik a preferencia-sorrendjüket, iteratív esetben nem. A "Mechanizmus fajtája" résznél részletesebben lesz erről szó.) Először nézzük meg kicsit alaposabban, hogyan alakul a diákok hasznossága. Ha Magyarországot vesszük példának, akkor általános, illetve középiskola választás esetén a diákok hasznosságának alakulásában szerepet játszhat hogy az adott iskolában milyen tagozatú osztályok indulnak: humán tagozat, reál tagozat, művészeti tagozat, zene tagozat, tánc tagozat, nyelvtanulási lehetőségek, lakóhelytől való távolság, valamilyen rangsorpontszám vagy statisztika alapján milyen erősségű az adott iskola. Egyetemi szakok esetében a diákok hasznosságának értékét sokkal jobban befolyásolja az érdeklődési kör, hiszen rengeteg szak közül van lehetőségük választani, ezenkívül befolyásoló tényező, hogy mennyi volt tavaly a ponthatár, mennyire becsüli az érettségi előtt álló diák a saját pontszámát. Ezenkívül szerepet játszhat a szakok vagy egyetemek hasznosságának becslésében valamilyen rangsorpontszám vagy statisztikai adat is. A diákok sok esetben ezek alapján becsülik meg, hogy mekkora hasznosságot jelent számukra egy adott iskola, és így kialakul a preferencia-sorrendjük. Azonban sok esetben ezek a hasznosságok módosulhatnak egy alaposabb utánajárással. Ahhoz, hogy a diákok közelebb kerüljenek ahhoz a hasznossághoz, amit az iskolák valóban jelentenek számukra, bizonyos értelemben jobban utána kell járniuk az iskoláknak, egyetemeknek. Általános iskolák esetében a szülőknek és a diákoknak érdemes elmenni nyíltnapra, beszélni a lehetséges osztályfőnökkel, tanárokkal. Érdemes lehet megnézni, hogy milyen arányban veszik fel a végzős diákokat gimnáziumba. Középiskolák esetében érdemes figyelembe venni, hogy a végzős évfolyamoknak hogy sikerült az érettségi az előző években, milyen arányban szereznek a diákok

nyelvvizsgát a középiskola befejezésére, illetve hány diák nyer felvételt a végzős diákok közül különböző egyetemekre. Itt sem utolsó szempont az osztályfőnök, illetve a tanárok, ezért itt is érdemes elmenni nyíltnapra. Egyetemek esetében az előző példákhoz hasonlóan valamelyest pontosabb képet lehet kapni a szakról egy nyíltnap keretében. Esetleg már az adott szakot elvégzett hallgatóktól lehet információt kapni arra vonatkozóan, hogy mennyire nehéz vagy könnyű elvégezni az adott szakot, a szak elvégzése után milyen munkalehetőségek vannak. A lakhatás sem utolsó szempont, érdemes megnézni, hogy az egyetemeknek milyen kollégiumi lehetőségei vannak, vagy az adott városban milyenek az albérlet árak. Ez az alaposabb utánajárás időbeli költséget biztosan igényel, de sok esetben anyagi jellegű költséget is. Tehát ki kell fizetni egy bizonyos költséget, hogy csökkentsük a bizonytalanságot, és ezáltal közelebb jussunk a valódi hasznossághoz. Viszont, ha minden szóba kerülő iskolára ráköltjük ezt a költséget, akkor a költségek nagyon magasak is lehetnek. Tehát a költség kifizetése felfogható úgy is, mint egyfajta hasznosságveszteség. Ezért nem biztos, hogy érdemes mindegyik iskolára költeni, hogy megkapjuk a pontos hasznosságot.

Tehát a célom az, hogy egyfajta utánajárási költség figyelembevételével a diákok szemszögéből hasonlítsam össze az iteratív mechanizmust a direkt mechanizmusokkal úgy, hogy az utánajárási költséget egyfajta hasznosság csökkenésként értelmezem.

6.1. Ötletek és modellezési lehetőségek

A szimuláció előtt a következő kérdések és modellezési lehetőségek merültek fel:

(*i*) Hasznosság-generálás: A hasznosságokról nem volt elérhető adathalmaz, ezért a generálás mellett döntöttem. Felmerült, hogy milyen eloszlással érdemes generálni a hasznosságokat. Első körben mind az iskolák, mind a diákok hasznosságát normális eloszlással generáltam. De miután jobban utána olvastam ennek, szinte a legtöbb esetben a diákok pontszámokat kapnak, ezek a pontszámok nem minden iskolában azonosak, de egy jobb képességű diáknak szinte mindegyik iskolában magasabb pontszáma van, mint egy kevésbé jó képességű diáknak. Így ezek a pontszámok feleltethetőek meg az iskolák hasznosságának. A [19] cikkre támaszkodva a diákok pontszáma olyan béta-binomiális eloszlást követ, amelyben a béta eloszlás paraméterei: $(3, 1)$.

6.1.1. Definíció. Az (n, α, β) paraméterű béta-binomiális eloszlás az a binomiális eloszlás, amelyben az egyes kísérleteknél a siker valószínűsége rögzítve van, de véletlenszerűen merül fel az (α, β) paraméterű béta eloszlásból az n Bernoulli-kísérlet előtt.

Ezért ha béta eloszlással generálunk valószínűségeket, akkor az így kapott valószínűséget felfoghatjuk úgy is, hogy egy diák mekkora valószínűséggel válaszol helyesen 1 kérdésre a felvételin, azaz mekkora valószínűséggel kap 1 pontot. A diákokhoz tartozó generált valószínűségeket jelöljük a továbbiakban p-vel. (Természetesen p minden diák esetében más.) Ha az n értékét 100-nak vesszük, azt úgy is értelmezhetjük, hogy legfeljebb 100 pontot lehet elérni a felvételin, ahol mindegyik kérdés helyes megválaszolása p paraméterű Bernoulli-eloszlást követ, és a különböző kérdések helyes megválaszolása egymástól független. A modellezés keretein belül feltehető, hogy minden iskolában más felvételit írnak a diákok, így az (n, p) paraméterű binomiális eloszlással mindegyik diákhoz annyi pontszámot generálunk, ahány iskolába jelentkezik. Erre a gondolatmenetre alapozva a diákok hasznosságát is felfoghatjuk úgy, mintha ők is pontoznák az iskolákat különböző szempontok szerint. Az előző részben levezetett gondolatmenet miatt a diákok számára kétféle hasznosságot is szükséges definiálnunk.

6.1.2. Definíció. A d diák s iskolára vonatkozó becsült hasznosságán azt a kardinális hasznosságot értjük, amekkora hasznosságot az "első körös" utánajárás után vár a d diák az s iskolától, ha oda nyer felvételt.

6.1.3. Definíció. A d diák s iskolára vonatkozó valódi vagy valós hasznosságán azt a kardinális hasznosságot értjük, amekkora hasznosságot valóban jelent d diáknak az s iskolába való felvétel. Ezt a "második körös" utánajárás után kapja meg a diák.

Feltettem, hogy a diákok becsült hasznossága $(99, 3, 1)$ paraméterű béta-binomiális eloszlást követ. Ez azt jelenti, hogy mindegyik iskolához generálunk egy valószínűséget $(3, 1)$ paraméterű béta eloszlással. Ebben az esetben $n = 99$ felfogható úgy, hogy 99 szempont van, és mindegyik szempont pontozása egymástól független q paraméterű Bernoulli eloszlást követ, ahol q az adott iskolához tartozó generált valószínűség. (Az n-et azért nem 100-ra állítottam, mert tökéletes iskola valószínűleg nincs. Ez úgy is értelmezhető, hogy van olyan szempont, amelyre egyik iskola sem kap pontot.) Ez egy elég erős feltevés, hiszen a valóságban például az iskola lakóhelytől való távolsága és az, hogy milyen erős az iskola szinte biztos hogy nem ugyanolyan eloszlást követ. Ha viszont azt a példát vesszük, hogy milyen erős a középiskola és hogy hány diák került be abból a középiskolából egyetemre, akkor elképzelhető, hogy ezek a valóságban is hasonló eloszlást követnek. Mivel diákok hasznosságának alakulása eléggé összetett, mint azt korábban is említettem, ezért a modell egyszerűsége miatt azt tettem fel, hogy a szempontok egymástól függetlenek, mindegyik q paraméterű Bernoulli eloszlást követ. A diákok valódi hasznosságát ezekből

a becsült hasznosságokból generáltam $(99, r)$ paraméterű binomiális eloszlással, ahol r az adott diák megfelelő iskolára vett becsült hasznossága osztva 100-zal. (Azért osztottam 100-zal, hogyha esetleg a diák becsült hasznossága 99, akkor ne legyen biztos, hogy a valós hasznossága is 99. Tökéletes iskola létezését itt is kizárhatjuk, ezért itt is feltehető, hogy egy szempontra egyik iskola sem kap pontot.) Ha így generáljuk a valós hasznosságokat, akkor az azt jelenti, hogyha egy diák alaposabban utánajár egy iskolának, akkor mindegyik szempontra akkora valószínűséggel ad pontot, mint a becsült hasznosság osztva 100-zal. Ez is egy elég erős feltételezés, csak akkor állja meg a helyét, ha a becsült hasznosság elég közel van a valós hasznossághoz, azaz ha a valós hasznosságot a becsült hasznosság pontosításaként értelmezzük. Tehát összefoglalva:

- (a) iskolák hasznossága: $(100, 3, 1)$-béta-binomiális eloszlás;
- (b) diákok hasznossága:
  - $(b1)$ diákok becsült hasznossága: $(99, 3, 1)$-béta-binomiális eloszlás,
  - $(b2)$ diákok valódi hasznossága: $(99, r)$-binomiális eloszlás, ahol r a diák megfelelő iskolára vonatkozó becsült hasznossága 100-zal osztva.

(ii) Mechanizmus fajtája: Ezen belül kétféle lehetőséget érdemes számításba venni:

- (a) direkt: A diákoknak előre meg kell adni a preferenciáikat, ezért az algoritmus futása előtt van lehetőségük utánajárni a valódi hasznosságuknak, azaz költeni az iskolákra. A direkt mechanizmus a legtöbb felvételi eljárásban alkalmazott mechanizmus.
- (b) iteratív: A diákoknak csak azokat iskolákat kell megadni, ahova szeretnének jelentkezni, de a sorrendet nem (például a francia egyetemi felvételi). Ebben az esetben az algoritmus futása közben érdemes utánajárni az iskoláknak, de csak azoknak, akiktől kap ajánlatot. A modellezés miatt feltehető, hogy a diákoknak mindig van elég idejük utánajárni az iskoláknak, és az algoritmus is végigfut.

(iii) GS-algoritmus: Milyen irányból futtassuk az algoritmust:

- (a) iskolák felől;
- (b) diákok felől.

(iv) Megfigyelés jellege: A költség fizetése esetén

- (a) rögtön kiderül a diákok valódi hasznossága;

(*b*) nem rögtön derül ki a diákok valódi hasznossága. Elképzelhető, hogy a diákok egy újabb megfigyelés által csak közelebb kerülnek a valódi hasznosságukhoz, de a valódi hasznosságuk pontos értékének kiderítéséhez még több megfigyelés, alaposabb utánajárás szükséges. Ezzel az esettel a modellezésben most nem foglalkozunk.

(*v*) Döntési heurisztika: A diákok dönthetnek arra vonatkozóan, hogy mi alapján választják ki azokat az iskolákat, amikre költenek:

(*a*) A diákok a preferencia-sorrendjükben szereplő legjobb k iskolára költenek.

(*b*) A diákok különbség szerint döntenek, azaz akkor költenek az iskolákra, ha a gondolt hasznosságok közötti eltérés kicsi.

(*c*) A diákok valószínűségi alapon döntenek, azaz akkor költenek az iskolákra, ha elég nagy az esély rá, hogy változik a preferencia-sorrend.

Ezen ötletek alapján arra jutottam, hogy a modellben az iskolák és a diákok hasznosságait generáljuk a fentebb leírt módon. A direkt mechanizmust az iskolák felől és diákok felől is érdemes futtatni, esetleg őket is össze lehet hasonlítani, az iteratív mechanizmust csak az iskolák felől a francia példát alapul véve. A diákok valódi hasznossága utánajárást követően rögtön derüljön ki. A döntési heurisztikák közül az elsőt választottam ki a modellezéshez. Ebben a modellben a diákok a preferencia-sorrendjükben szereplő legjobb k iskolára költenek. Ez direkt mechanizmus esetén azt jelenti, hogy vesszük a diákok becsült hasznosságai alapján felállított preferencia-sorrendet, és a diákok a legjobb k iskolára kifizetik az utánajárási költséget, azaz ezeknek az iskoláknak megkapják a valódi hasznosságukat, a többi iskolánál csak a becsült hasznosságot tudják, és így keletkezik egy új preferencia-sorrend, és ezzel fog lefutni az algoritmus. Iteratív algoritmus esetében ez kicsit mást jelent, hiszen csak akkor érdemes utánajárást végezni egy diáknak, ha legalább 2 iskolától kap ajánlatot. Én úgy fogalmaztam meg, hogy akkor történik itt utánajárás, ha mindkét iskola a preferencia-sorrendet nézve az első k iskola között van. Ha történik utánajárás, természetesen változik a preferencia-sorrend, így ha kap a diák új ajánlatot, akkor már ezt az új preferencia-sorrendet nézzük. Érdemes megjegyezni, hogyha egy iskolára sem, vagy minden iskolára költenek a diákok, akkor természetesen az iskolák felől futtatott direkt és iteratív GS-algoritmus ugyanazt az eredményt adják. De ha például az első 3 iskolára költenek a diákok, elképzelhető, hogy direkt mechanizmusnál a 3-as és a 4-es helyen lévő iskola valamelyik diák esetében helyet cserél (nem biztos, hogy jogosan, mert a diák a 3-as helyen lévő iskolának utánajár, a 4-es helyen lévőnek pedig nem), iteratív mechanizmusnál pedig ha ettől a 2 iskolától kap ajánlatot a diák, akkor nincs utánajárás, tehát a 3-as helyen

szereplő iskolát választja a diák. A modell megvalósításához szükséges volt írni egy költséges Gale-Shapley függvényt. A következő alfejezetben ezt a kódot fogom ismertetni.

6.2. Költséges GS-függvény

A költséges GS-függvényt az R programozási nyelvben készítettem el. A teljes kód a Függelékekben tekinthető meg. Ebben az alfejezetben a kód részleteit ismertetem. Még a függvény megírása előtt importáltam `"matchingR"` package-t és betettem a könyvtárba. Direkt mechanizmus esetén ezzel a package-vel könnyen meg tudjuk oldani a párosítási problémát, és csak a költségeket szükséges majd kiszámolnunk. A függvényünk neve legyen `koltseges_gs`, melynek 8 bemeneti változója van. Azért, hogy a kód ne legyen túl hosszú, a változók neveire sok esetben rövidítéseket használtam. Elsőként meg kell adni a diákok becsült hasznosságának a mátrixát, azaz a `dhb`-t. Ez egy olyan mátrix, mely soraink száma megegyezik a diákok számával, oszlopainak száma megegyezik az iskolák számával, és az i-edik sor j-edik eleme annak a hasznosságnak az értéke, amekkora a gondolt hasznossága az i-edik diáknak, ha bekerül a j-edik iskolába. Ezt követően meg kell adni a diákok valódi hasznosságának a mátrixát, azaz `dhv`-t, melynek mérete megegyezik a `dhb` méretével, és az i-edik sorának a j-edig eleme az a hasznosság, amelyet a diáknak valójában jelent, ha bekerül a j-edik iskolába. A következő változó a `ktsg`, azaz az utánajárási költség, ennyivel csökken a diákoknak a hasznossága, ha utánajárnak egy iskolának. Feltettük, hogy diákok bármelyik iskola valódi hasznosságát ugyanakkora költségért tudhatják meg, tehát ez egy konstans változó. Ha a `d_vagy_i` bemeneti változó értéke "d", akkor direkt mechanizmusként fog futni, ha az értéke "i", akkor iteratívként. A `hany` változónál lehet megadni, hogy hány legjobb iskolára költenek a diákok. Ezt követően `ih` változó következik, mely az iskolák hasznosságát jelöli. Ez egy olyan mátrix, mely i-edik sorának a j-edik eleme azt jelenti, hogy mekkora hasznosságot jelent az i-edik iskolának, ha felveszi a j-edik diákot. Az utolsó bemeneti változó a `kap`, mely alatt az iskolák kapacitását értjük. A modellezés keretein belül feltehető, hogy mindegyik iskolának ugyanakkora a kapacitása. Ezeknek a bemeneti változóknak a segítségével fut le a függvény.

A bemenő mátrixok méretéből kiolvasható a diákok száma `dsz` és az iskolák száma `isz`. Ez azért hasznos, mert amikor létrehozunk mátrixokat, egyszerűbb így megadni a méreteket. Ezután az `order` függvény segítségével kialakítjuk az iskolák prioritássorrendjét, `ip`-t. Így holtverseny esetén az a diák élvez előnyt, akinek a sorszáma kisebb. A diákok hasznosságát, `dh`-t egyenlőre a diákok becsült hasznosságával tesszük egyenlővé, a diákok preferencia-sorrendjét, `dp`-t az iskolák prioritássorrendjéhez hasonlóan számoljuk. Azaz holtverseny esetén itt is a

kisebb sorszámú iskola élvez majd előnyt. Ezután definiálunk néhány új változót, a diákok végső hasznosságát, `dvh`-t, azaz mekkora hasznossága lesz az algoritmus végén a diáknak; a felvett diákok mátrixát, `fd`-t, melynek i-edik sora az i-edik iskola által felvett diákokat tartalmazza; egy költségmátrixot, mk-t, mely i-edik sorának j-edik eleme az a költség, amelyet az i-edik diák a j-edik iskolára költ; a `dk` a diákok költségeit tartalmazza. Ezt követően ha az algoritmus direkt, akkor utánajár pontosan annyi iskolának a diák, amennyit bemeneti változóként megadtunk. Ez követően módosul a diákok hasznossága, a preferencia-sorrend és a költséghez tartozó változók is. Így ezekkel az adatokkal annak függvényében hívjuk meg a beépített `galeShapley.collegeAdmissions` függvényt, hogy diákok vagy iskolák felől fut a GS-algoritmus. Ennek eredményéből kinyerhető a felvett diákok, és a végső iskolák, azaz `vi`-k. (A `vi`-ket nem volt szükséges az elején definiálnunk, ugyanis mind direkt, mind iteratív változatban egy másik változó eredményeként jön ki). Ha a mechanizmus iteratív, akkor definiáljuk a kapacitás vektort, azaz `kap_vektor`-t, mely azt jelöli, hogy melyik iskolának mennyi még a szabad kapacitása. Az `elf_isk` i-edik eleme jelölje az i-edik diák feltételesen elfogadott iskoláját. Az `elozo_isk` jelölje a diákok előzőleg feltételesen elfogadott iskoláját, melyet a későbbiekben mégis visszautasított a diák. A `sorsz` egy segédvektor, i-edik eleme azt jelöli, hogy az i-edik iskola a prioritássorrendben szereplő hányadik diáktól fogja tenni az ajánlatokat. Ezt követően kezdődik az iteratív algoritmus. Az algoritmust addig ismételjük, amíg a 2 állítás közül valamelyik nem lesz igaz mindegyik iskolára: elfogynak a szabad helyek, azaz a `kap` vektor megfelelő eleme 0 lesz, vagy az adott iskola már az összes diáknak ajánlatot tett, azaz a `sorsz` a diákok számánál 1-gyel nagyobb lesz. Az algoritmus a következő. Az `aj` mátrix i-edik sora jelöli, hogy az i-edik diák az adott körben melyik iskolától kapott ajánlatot. Mindegyik elemét 0-ára állítjuk először. A `diak_sorsz` vektor segítségével tartjuk számon, hogy hányadik ajánlat következik a diákoknál. Mindegyik elemét 1-re állítjuk először. Ezt követően az iskolák megteszik az ajánlatukat. Az `eddig` változó értéke `sorsz[j]+kap_vektor[j]-1`. Ha értéke meghaladja a diákok számát, akkor a diákok számára állítjuk be. A j-edik iskola a `sorsz[j]` és az `eddig` változó között lévő prioritássorrendbeli diákoknak megteszi az ajánlatait, ha a kapacitásvektor értéke nem 0. Ennek megfelelően módosul az `aj` mátrix és a `diak_sorsz` vektor megfelelő elemeinek az értéke. Végül a `sorsz[j]`-t egyenlővé tesszük `eddig+1`-gyel. Ezután végigmegyünk a diákok kapott ajánlatain, ezeket preferencia-sorrendbe tesszük. Erre azért van szükség, hogy rendezve legyenek, és minél kevesebb összehasonlításra legyen szükség. Ez úgy csináljuk, hogy a `hely_p`-be beletesszük, hogy a kapott ajánlat hol helyezkedik el a preferencia-sorrendben, így ezt a vektort felhasználva minimum helykereséssel preferencia-sorrend szerint tudjuk rendezni az ajánlatokat, ez lesz a `aji_p` vektor. Ezt követően nézzük, váltott-e iskolát az adott

körben a diák. Először az ezt jelző `valt` változó értékét 0-ra állítjuk. Ezután végigmegyünk a preferenciák alapján rendezett ajánlatokon. A legjobb ajánlatnak azaz `legjobb_aj` a preferencia sorrendben következő ajánlatot értjük. Ha még nem volt elfogadott iskolája a diáknak, akkor ez a legjobb ajánlat lesz a diák elfogadott iskolája. Ekkor változás is történt, tehát a `valt` értéke is változott. Ha már fogadott el ajánlatot, akkor lehet hogy történik utánajárás annak függvényében, hogy a mindkét kapott ajánlat szerepel-e a legjobb `hany` iskola között. Ha történik utánajárás, akkor a költséges változók megfelelő elemei is módosulnak, megváltozik a diákok hasznossága és preferenciája is. Miután szükség esetén utánajárást végzett a diák, összehasonlítja a diák a legjobb ajánlatát a már elfogadott iskoláéval. Ha cserél, és ebben a körben most cserél először, akkor az előző iskola vektor megfelelő elemében rögzítjük a korábban elfogadott iskolát. Erre azért van szükség, hogy ennek az iskolának a kapacitását vissza tudjuk majd növelni. A `valt` értéke is megnő, hogy tudjuk, hogy csak az első cserénél lévő iskola kapacitását kell majd csak megnövelnünk. Így végigmegyünk a diák összes ajánlatán. Miután az ajánlatok végére értünk, az elfogadott iskola kapacitását csökkentjük, az előző iskola kapacitását megnöveljük, feltéve hogy volt előző iskola. Ekkor már látjuk, menyit költött összesen a diák, így ennek a változónak az értékét is beállítjuk. A kilépés feltételeit úgy ellenőrizzük, hogy veszünk egy `v` változót, melyet kezdetben 0-ra állítunk és végigmegyünk az iskolákon, és ha az iskola kapacitása 0 vagy már minden diáknak tett ajánlatot az iskola, akkor növeljük `v` értékét. Akkor lépünk ki, ha `v` eléri az iskolák számát. Ezután az elfogadott iskolák lesznek a végleges iskolák, és a felvett diákokat ez alapján be tudjuk állítani. (A felvett diákokat nem rendeztem az iskolák prioritássorrendje szerint, erre nem volt szükségem az elemzéshez.) Eddig tartott az iteratív algoritmus. Ezt követően tudjuk beállítani a `dvh`-t, a diákok végső hasznosságát, azaz azt a hasznosságot, amelyet azaz iskola jelent a diák számára, ahova végül felvették, illetve a `dnh`-t is, a diákok nettó hasznosságát, ami végső hasznosság minusz a költség. Végül az `oh`-t, azaz az összhasznosságot, az `ok`-t, azaz az összköltséget, illetve a `netto_oh`-t, azaz a nettó összhasznosságot egyszerű összegzéssel kiszámoljuk. Végső soron pedig kilistázzuk a végső iskolákat (`vegleges_iskola`), a felvett diákokat (`felvett_diakok`), diákok végső hasznosságát (`diakok_haszn`), diákok költségét (`diakok_ktsg`), diákok nettó hasznosságát (`diakok_netto_haszn`), az összhasznosságot (`osszhaszn`), az összköltséget (`osszktsg`), és végül a nettó összhasznosságot (`netto_osszhaszn`).

7. fejezet

Szimulációs eredmények

Ebben a fejezetben először a generált adatokat vizsgáljuk, majd az utánajárási költség nélkül kapott eredményeket, végül utánajárási költség figyelembevételével hasonlítjuk össze a direkt és az iteratív eljárást.

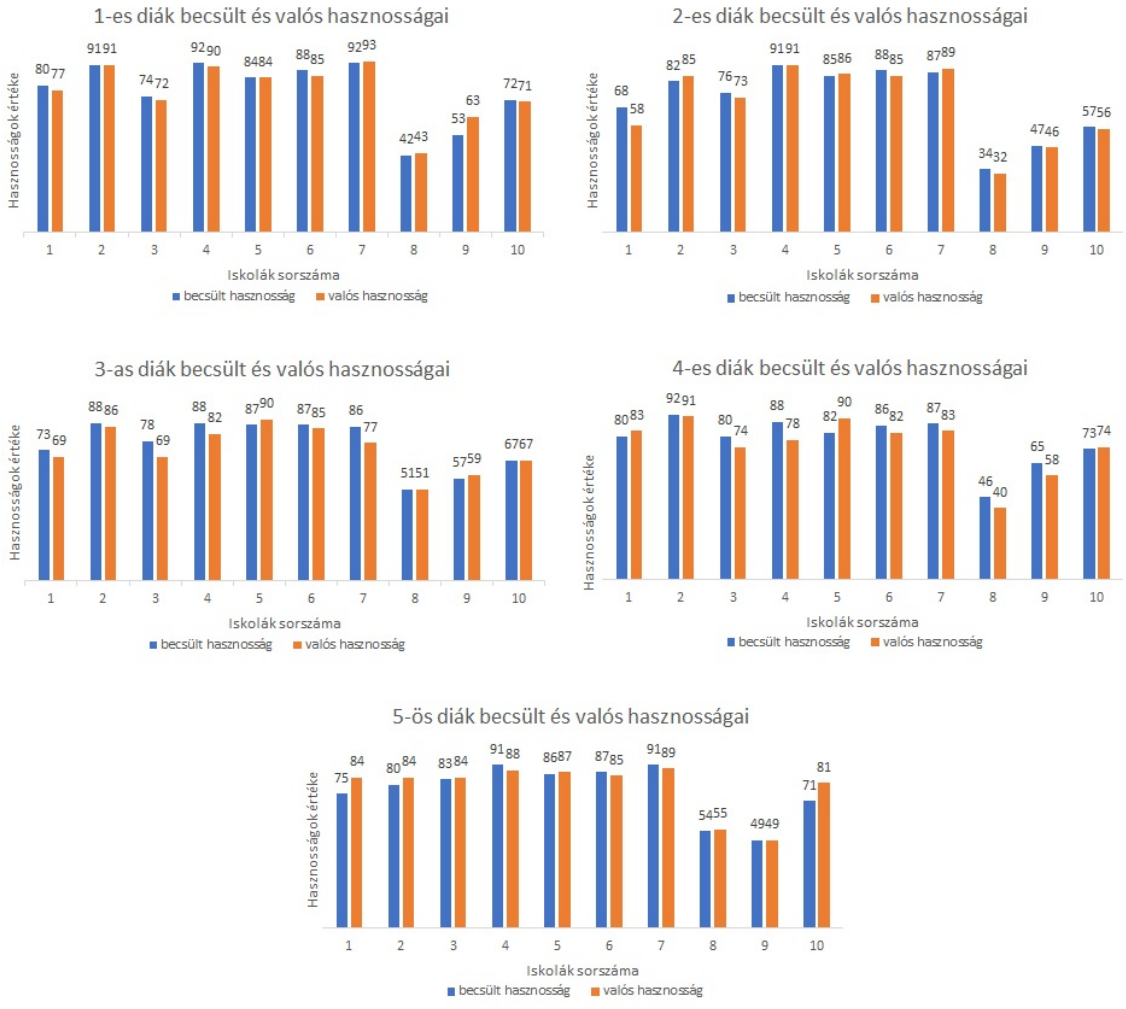
7.1. Utánajárási költség nélküli eset

Ehhez a részhez tartozó kód a Függelékekben tekinthető meg, a változók elnevezése megfeleltethető a `koltseges_gs` kódban használt elnevezéseknek. Ahhoz, hogy futtatni tudjuk a költséges Gale-Shapley függvényünket, mindenekelőtt szükségünk van az iskolák hasznosságának generálásához, majd a diákok becsült, illetve valós hasznosságainak generálásához. Ezekhez a diákok számát, az iskolák számát, és az iskolák kapacitását kell elsősorban rögzítenünk. A francia példát alapul véve az iskolák számát állítsuk 10-re. Mivel főleg a szemlétetés a célja a szakdogozatomnak, ezért az átláthatóság kedvéért az iskolák kapacitását állítsuk 12-re. Ha ezt a két adatot rögzítettnek tekintjük, akkor a diákok számát be lehet állítani úgy, hogy több hely legyen, mint diák, azaz ha például 118 diák van. Ha a diákok számát 120-ra állítjuk, akkor ugyanannyi hely van, mint diák. Ha a diákok számát 122-nek vesszük, akkor több diák van, mit hely. Részletesebben a 118 diákkal kapott eredményeket fogjuk nézni, a végeredményeket illetve az aggregált eredményeket a többi esetben is megnézzük. Tehát egyenlőre állítsuk a diákok számát 118-ra. Feltettem, hogy mindegyik diák mind a 10 iskolába jelentkezik. (Ezt a költséges GS-függvény kódjában is feltételeztem, hiszen a diákok becsült és valós hasznosságainak a mátrixait ilyen méretűnek vettem.) Ahhoz, hogy mindig ugyanonnan induljon az adatok generálása, azaz újrafuttatva a kódot ugyanazokat az értékeket kapjuk eredményül, a `set.seed(1)` parancsot használtam. A $(3, 1)$-paraméterű béta eloszlással először a diákokhoz generáltam valószínűségeket, ezeket a `diakok_kepessege` vektorba tettem, majd binomiális

eloszlással az előző fejezetben ismertetett módon generáltam az iskolák hasznosságát. Ezt követően szintén $(3, 1)$-paraméterű béta eloszlással az iskolákhoz generáltam valószínűségeket, melyeket az `iskolak_nepszerusege` vektorba tettem. Ezek a következők lettek:

```
> iskolak_nepszerusege
[1] 0.7351948 0.8840955 0.7714396 0.9047095 0.8369755 0.8787568 0.9003905
0.4507880 0.5861513 0.7173205
```

Az így kapott generált valószínűségekből látható, hogy a 2-es, 4-es, 6-os és 7-es iskolák nagyon erős és népszerű iskolák, az 1-es, a 3-as, az 5-ös és a 10-es iskolák jó iskolának számítanak, a 8-as és a 9-es iskola viszont nem túl erős. Ezeknek a népszerűségeknek a segítségével generáljuk a diákok gondolt hasznosságát az előző fejezetben leírt módon, majd ezeknek a becsült hasznosságoknak a segítségével generáljuk a valós hasznosságokat szintén az előző fejezetben ismertetett módon. A következő ábra mutatja, hogy ezekkel a generált adatokkal hogyan alakul az 1-es, 2-es, 3-as, 4-es és 5-ös sorszámú diák becsült és valós hasznossága.



Nézzük meg, hogy hogyan változik az első 5 diák preferencia-sorrendje, hogyha a fenti oszlopdiagramon látható módon változik a diákok hasznossága utánajárással.

		Preferenciasorrend változása									
1-es diák	becsült	4	7	2	6	5	1	3	10	9	8
	valós	7	2	4	6	5	1	3	10	9	8
2-es diák	becsült	4	6	7	5	2	3	1	10	9	8
	valós	4	7	5	2	6	3	1	10	9	8
3-as diák	becsült	2	4	5	6	7	3	1	10	9	8
	valós	5	2	6	4	7	1	3	10	9	8
4-es diák	becsült	2	4	7	6	5	1	3	10	9	8
	valós	2	5	1	7	6	4	3	10	9	8
5-ös diák	becsült	4	7	6	5	3	2	1	10	8	9
	valós	7	4	5	6	1	2	3	10	8	9

Látható, hogy bár a hasznosságok nem változtak túl sokat, mindegyik diák prioritássorrendjében történt változás. Az 1-es diák esetében a 4-es iskola az első helyről a 3. helyre csúszott, a 2-es diák esetében a 6-os iskola a 2. helyről az 5. helyre, a 3-as diáknál a becsült hasznosságok alapján 3. helyen lévő 5-ös iskola került az első helyre, a 4-es diák esetében az eredetileg 2. helyen lévő 4-es iskola a 6. helyre csúszott, míg az eredetileg 5. helyen lévő 5-ös iskola a 2. helyre jött fel. Az 5-ös diák esetében az 1. és 2. helyen lévő iskolák, illetve a 3. és 4. helyen lévő iskolák is helyet cseréltek egymással.

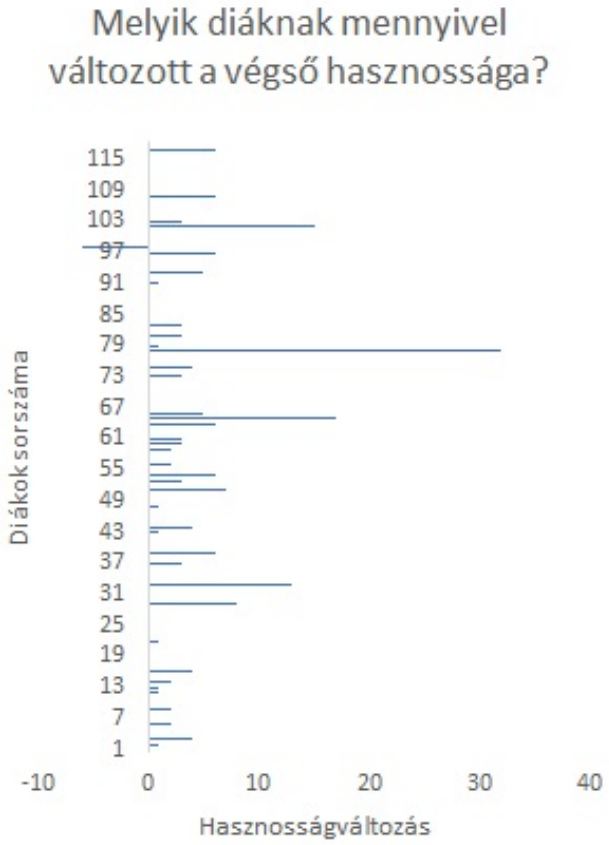
Most futtassuk a becsült hasznosságokkal, illetve a valós hasznosságokkal is a `koltseges_gs` függvényünket mindhárom esetben, azaz direkt mechanizmus esetében az iskolák és diákok felől is, iteratív mechanizmus esetében csak iskolák felől, hiszen a függvényünk csak ezt tudja. Amikor a becsült hasznosságokkal kapott eredményre vagyunk kíváncsiak, akkor a diákok nem végeznek utánajárást, azaz a `hany` változónak az értékét 0-ra állítjuk. Amikor a valós hasznosságokkal kapott eredményre vagyunk kíváncsiak, akkor az olyan, mint amikor a diákok az összes iskolának utánajárnak, azaz a `hany` változónak az értékét 10-ra állítjuk. Itt a költségeknek még nincs jelentősége, ezért a `ktsg` változó értékét egyenlőre 0-nak vegyük. Elsőként becsült hasznosságokkal futtassuk az algoritmus mindhárom esetben. Ekkor azt kapjuk, hogy a diákok által elért összhasznosság mindhárom esetben ugyanannyi:

```
> eredmeny_df_d_0$osszhaszn
[1] 9108
> eredmeny_if_d_0$osszhaszn
[1] 9108
> eredmeny_if_i_0$osszhaszn
[1] 9108
```

Iskolák felől futtatva direkt és iteratív mechanizmus esetén is ugyanabba az iskolába kerülnek be a diákok, ha a becsült hasznosságokkal számolunk, hiszen nincs utánajárás az algoritmus közben, és feltételeztük, hogy iteratív mechanizmusnál is végigfut az algoritmus, tehát ezen egyáltalán nem lepődünk meg. Viszont a direkt algoritmus diákok felől és iskolák felől futtatva ugyanazt adná eredményül, vagy csak az összhasznosság egyezik meg véletlenül? Ha összehasonlítjuk, hogy melyik iskolába vették fel a diákokat, látszik, valóban mindkét esetben ugyanabba az iskolába kerültek be a diákok. Ez vajon véletlen, vagy sokszor előfordul? Most futtassuk le az algoritmust valós hasznosságokkal, azaz minden iskolának járjanak utána a diákok. Ekkor a következő összhasznosságokat kapjuk:

```
> eredmeny_df_d_10$osszhaszn
[1] 9298
> eredmeny_if_d_10$osszhaszn
[1] 9298
> eredmeny_if_i_10$osszhaszn
[1] 9298
```

Mivel itt is ugyazazt az eredményt kaptam direkt algoritmus esetén mindkét oldalról futtatva, és a diákok is ugyanabba az iskolába kerültek be, ezért jobban utánaolvastam a témának. Először Roth és Peranson figyelték meg az amerikai rezidensek allokációjánál, hogy nagyon sok esetben megegyezik a diák-optimális, illetve az iskola-optimális stabil párosítás, lásd [20]. Tehát voltaképpen elég az egyik oldalról futtatott direkt mechanizmust összehasonlítani az iteratív mechanizmussal. Mivel Magyarországon diák felől futtatják az algoritmust, ezért a következő alfejezetben a diákok felől futtatott direkt mechanizmust és az iskolák felől futtatott iteratív mechanizmust fogjuk összehasonlítani. A kapott összhasznosságokból kiszámolható, hogy 190-nel növekedett az összhasznosság, ha a diákok végeztek utánajárást. Vajon elképzelhető, hogy van olyan diák, akinek csökkent a hasznossága utánajárást követően? A következő ábrán látható, melyik diáknak mennyivel változott a végső hasznossága (azaz az a hasznosság, amelyet azaz iskola nyújt számára, ahova felvételt nyert) utánajárást követően.



Az ábrán látható, hogy egy diák, pontosan a 98-as sorszámú diák hasznossága csökkent csak, a többiek hasznossága viszont nőtt. Miért csökkenhetett ennek a diáknak a végső hasznossága? Becsült hasznosságokkal futtatott algoritmussal a 9-es iskolába került, ami számára valójában 50-es hasznosságot ér, valós hasznosságok esetén a 8-as iskolába került, ami csak 44-es hasznosságot jelent számára. Nézzük meg, hogy mely diákokat vették fel a 9-es iskolába becsült, illetve valós hasznosságokkal való futtatás esetén.

```
> eredmeny_df_d_0$fd[9,]
[1]  25  85  98  95  90 112  84  69  86  40  78  57
> eredmeny_df_d_10$fd[9,]
[1]  75  85 112  95  90  69  86  40  84  57  39  25
```

Valós adatokkal futtatva a 39-es és 75-ös diák került be a 9-es iskolába a 78-as, és 98-as diákok helyett. A 39-es diák becsült és valós hasznosságai a következők:

```
> dhg[39,]
[1] 67 89 74 91 79 84 90 54 54 67
> dhv[39,]
[1] 70 88 72 91 83 83 89 51 57 75
```

A becsült hasznosságokat nézve a 8-as és 9-es iskola holtversenyben szerepelnek, 54-es hasznossággal. Mint a GS függvény leírásában is említettem, holtversenyek esetén a diák a hamarabbi sorszámú iskolát részesíti előnyben, tehát itt a 8-as iskola van előnyben. Nézzük meg mekkora hasznosságot jelent a 9-es iskolának a 39-es, illetve a 98-as diák.

```
> ih[9,39]
[1] 58
> ih[9,98]
[1] 50
```

Látszik, hogy a 9-es iskola számára nagyobb hasznosságot jelent a 39-es diák, mint a 98-as, ezért őt veszi fel, és emiatt kiszorul a 98-as diák, és ezért csökken a hasznossága. Tehát előfordulhat, hogy hiába jár utána egy diák a valós hasznosságának, csökken a végső hasznossága.

Nézzük meg, mi történik, ha a diákok számát 120-ra állítjuk, azaz ugyanannyi diák van, mint ahány hely. (Itt is diákok illetve iskolák felől futtatva ugyanaz jött ki eredménynek.)

```
> sum(haszn_valt_df_d)
[1] 217
```

Itt is összességében ezekkel az adatokkal hasznosságnövekedést látunk. Ha a diákok számát 122-re növeljük, azaz több diák lesz, mint férőhely, akkor is hasznosságnövekedés figyelhető meg (diákok és iskolák felől futtatva ugyanakkora):

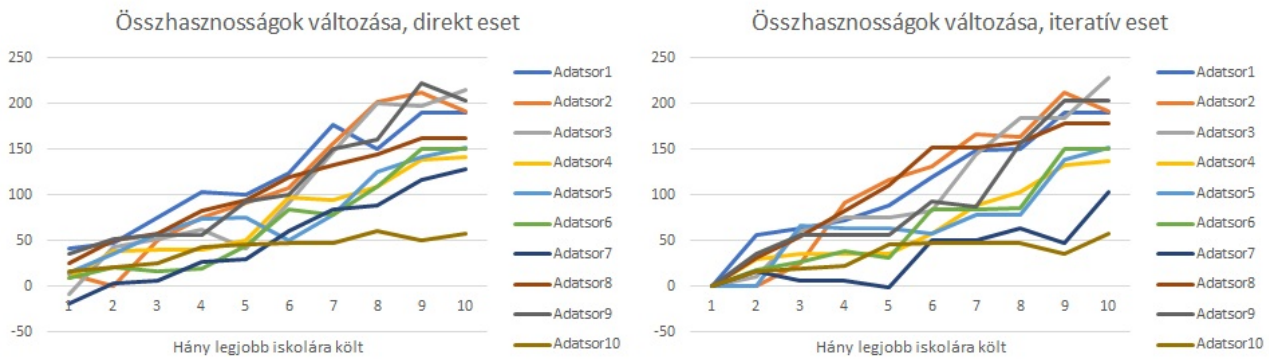
```
> sum(haszn_valt_df_d)
[1] 184
```

7.2. Utánajárási költség figyelembevételével a direkt és az iteratív eljárás összehasonlítása

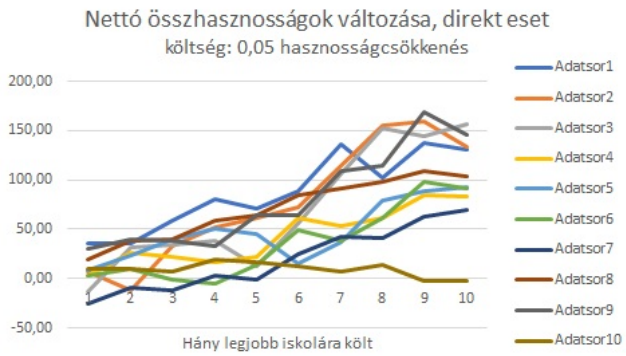
Ebben az alfejezetben a modellünket fogjuk részletesebben megnézni. Mint az előző rész végén is említettem, az iskolák felől futtatott iteratív algoritmust fogjuk összehasonlítani a diákok felől futtatott direkt algoritmussal. Mint ahogy modell leírásánál is megfogalmaztam, ebben a modellben a diákok a preferencia-sorrendjükben szereplő legjobb k iskolára költenek. Ez a modell azért állja meg a valóságban is a helyét, mert a diákok hajlamosabbak az első néhány iskolának jobban utánajárni, hiszen ezek számukra fontosabbak, mivel remélik, hogy ide nyernek felvételt. Az ide tartozó kódrészlet a Függelékekben tekinthető meg, most ezt fogom ismertetni röviden. A kód elején lehetőség van megadni a diákok számát, az iskolák, számát, a kapacitást. Állítsuk ezeket be ugyanúgy, mint az előző fejezetben, tehát a `dsz` legyen 118, az `isz` legyen 10, a `kap` legyen 12. Hogy átfogóbb eredményt kapjunk, többszöri adatgenerálásra van szükség, így állítsuk a `futasszam`-ot 10-re, azaz 10 adatsorunk lesz. A `ktsg`-et állítsuk egyenlőre 0.05-re, azaz ha egy diák utánajár egy iskolának, az 0.05 hasznosságcsökkenést eredményez neki. Létre kell hozni azokat a mátrixokat, melyekben az összhasznosságokat tároljuk, azaz `oh_mtx_d`-t és `oh_mtx_i`-t, illetve a nettó összhasznosságok tárolására is létre kell hoznunk mátrixokat, azaz `noh_mtx_d`-t és `noh_mtx_i`-t. Ezen mátrixok sorainak a száma `futasszam`, oszlopainak a száma `isz+1` . Ezután, hogy mindig ugyanonnan induljon az adatgenerálás, szintén a `set.seed(1)` parancsot használjuk. Ezután belépünk a ciklusba, ami 1-től a `futasszam`-ig megy. A cikluson belül generáljuk az iskolák hasznosságát, a diákok hasznosságát, majd bemegyünk egy másik ciklusba, ami 0-tól 10-ig megy. Ez adja meg, hogy hány iskolának jár a diák utána. A direkt mechanizmus esetén a diákok felől, az iteratív mechanizmus esetén az iskolák felől futtatjuk a GS-függvényünket, és a direkt mechanizmus eredményét `eredmeny_d`-be, az iteratív mechanizmus eredményét az `eredmeny_i`-be rögzítjük. Ezekből az eredményekből kinyert összhasznosságokkal és nettó összhasznosságokkat töltjük ki a ciklus előtt létrehozott mátrix elemeit értelemszerűen. Az így létrejött mátrixokkal lehetőségünk van kiszámolni, hogy minden egyes kör után mennyit változott az összhasznosság, illetve nettó összhasznosság ahhoz képest, ahhoz az összhasznossághoz képest, amikor egy iskolának sem jártunk utána. (A nettó összhasznosság megegyezik abban az esetben az összhasznossággal, amikor egy iskolának sem járunk utána, azaz amikor nincs költségkifizetés.) Tehát az `oh_valt_d`, és az `oh_valt_i` mátrixok i-edik sorának j-edik eleme azt jelenti, hogy az i-edik futtatás során generált adatokkal ha minden diák a preferencia-sorrendjében szereplő első j iskolának jár utána, akkor mennyit változik az összhasznosság ahhoz képest, amikor nincs utánajárás. Hasonlóan értelmezhetőek az `noh_valt_d`

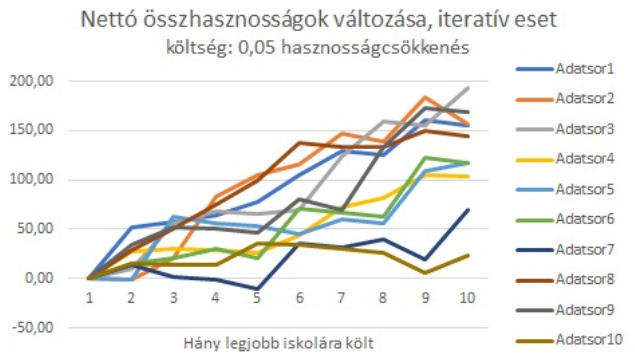
és az `noh_valt_i` mátrixok elemei is, csak itt a nettó összhasznosság változásokat rögzítjük.

Először nézzük meg, hogyan változnak az összhasznosságok. Ezek a mátrixok táblázat formájában a Függelékekben tekinthetők meg, a következő ábrákat a táblázatok felhasználásával Excelben készítettem.

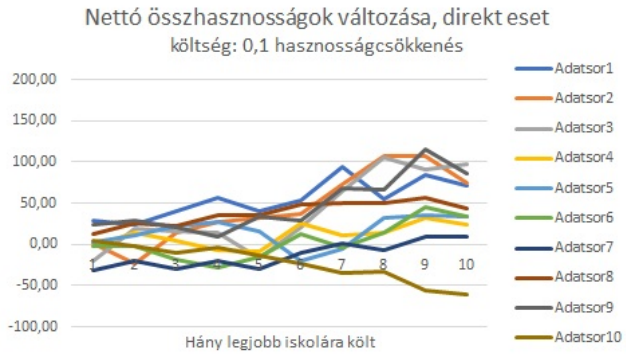


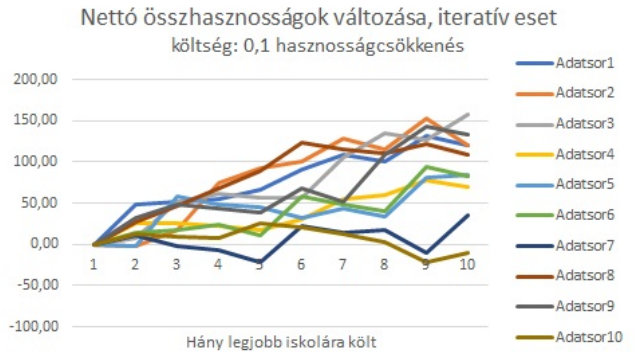
Látszik, hogy direkt mechanizmusnál ha az 1. legjobb iskolának járnak utána a diákok, az valamelyik adatsornál összhasznosság növekedéshez, valamelyik adatsornál viszont összhasznosság csökkenéshez vezet. Ez azért lehet, mert ez jelentősen torzíthat a prioritáson. Iteratív mechanizmusnál úgy értelmeztük ezt a modellt, hogy akkor van utánajárás, ha több olyan iskolától kap ajánlatot, akik a legjobb k iskola között szerepelnek, így az első legjobb iskolánál iteratív mechanizmus esetén nincs utánajárás. Ha mind a 10 iskolának utánajárnak a diákok, akkor látszik, hogy több helyen ugyanazt a megoldást kapjuk. A néhány helyen látható kis eltérés amiatt van, mert az iteratív algoritmust iskolák felől futtattuk, a direkt algoritmust pedig a diákok felől, és itt más-más eredményt ad a két oldalról való futtatás, például az *Adatsor*3 esetében. Ha a diák a legjobb 2, 3, ..., 9 iskolának jár utána, akkor a direkt és iteratív mechanizmusok között lévő eredmények eltérése azzal magyarázható, hogy kicsit másképp definiáltuk a modellt iteratív és direkt mechanizmus esetén. Az azonban látható, hogy az iteratív és a direkt mechanizmusnál is az összhasznosság növekvő tendenciát mutat, azaz ha több iskolának járnak utána a diákok, akkor általában nő az összhasznosság. Most nézzük hogyan alakulnak abban az esetben a nettó összhasznosságok, amikor egy utánajárás 0.05 hasznosságcsökkenést eredményez.



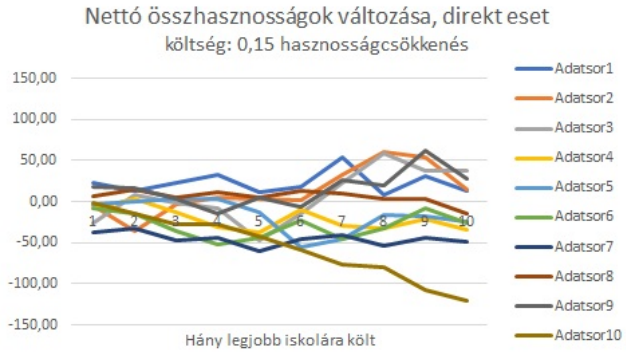


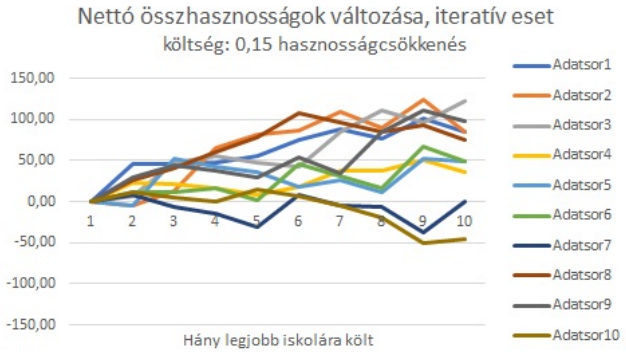
Az látszik, hogy ilyen kis hasznosságcsökkenést eredményező költség esetén is direkt mechanizmusnál már nem minden adatsornál éri meg utánajárni az összes iskolának. Az ábrán látható *Adatsor*10, azaz a 10. generált adatsor esetében a nettó összhasznosság negatív értéket vesz fel, ha mind a 10 iskolának utánajárnak a diákok, míg iteratív mechanizmusnál ha mindegyik iskolának utánajárnak a hallgatók, akkor hasznosságnövekedés figyelhető meg. Nézzük meg, mi történik, ha egy utánajárás 0,1 hasznosságcsökkenést eredményez.



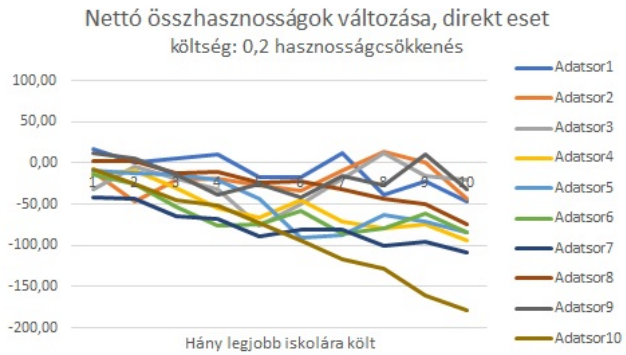


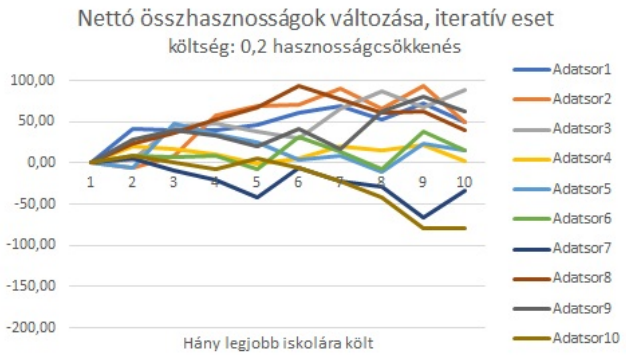
Ekkora költség esetén a 10. adatsor esetében már iteratív mechanizmus esetén sem éri meg mind a 10 iskolának utánajárni, de 5 iskolának viszont igen, direkt mechanizmusnál azonban csak az első iskolának éri meg utánajárni. A többi adatsornál megéri a diákoknak, hogy több iskolának járjanak utána, de direkt mechanizmus esetén sokkal magasabbak lesznek a költségek, így sokkal kevesebb lesz a nettó hasznosság, minél több iskolára költenek a diákok. Növeljük tovább a költséget, jelentsen egy utánajárás 0,15 hasznosságcsökkenést.



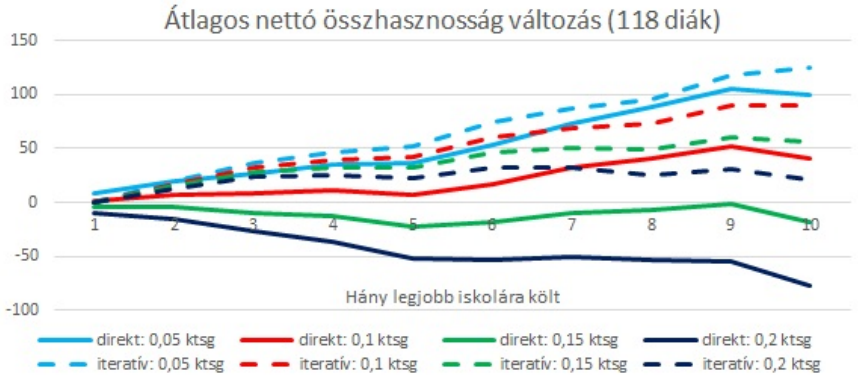


Ekkora költség esetén a 6., 7., és 10. kerültekben direkt mechanizmus esetén egyáltalán nem éri meg utánajárást végezni, iteratív mechanizmus esetén a 6. adatsornál viszont abszolút jó döntés az utánajárás, sőt a legjobban akkor járnak a diákok, ha a 9 legjobb iskolának járnak utána, feltéve, hogy kapnak onnan ajánlatot. Sőt a 7. adatsornál a legjobb 2 iskolának, a 10. adatsornál a legjobb 5 iskolának való utánajárás is növeli a nettó hasznosságot. Növeljük még egy kicsit tovább a költségeket, egy utánajárás költsége feleljen meg 0,2 hasznosságcsökkenésnek.

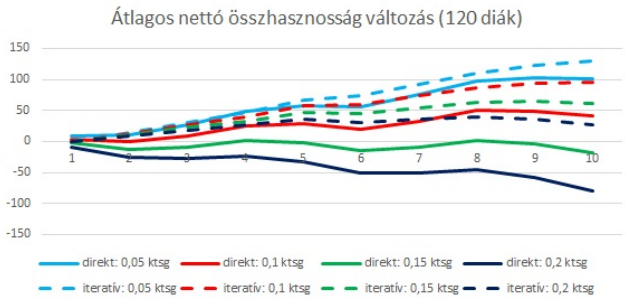


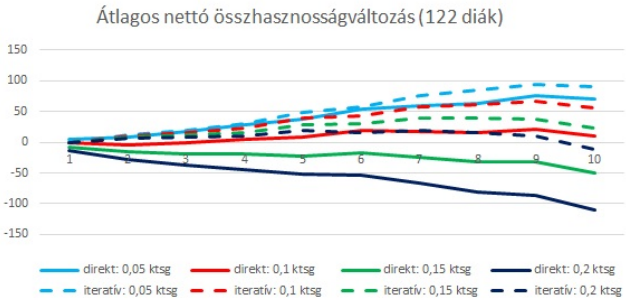


Ekkora költség esetén direkt mechanizmusnál szinte alig éri meg néhány helyen utánajárást végezni. Iteratív mechanizmusnál viszont az látható, hogy a 7. és 10. adatsor kivételével megéri elég sok iskolának utánajárni. Általában a legjobb 7, 8, vagy 9 iskolának éri meg legjobban utánajárni, ha kapnak a diákok onnan több ajánlatot, legjobb 10 iskolának való utánajárás, azaz ha minden iskolának utánajár a diák, ahonnan kap ajánlatot, már inkább nettó hasznosságbeli visszaesést eredményez. A következő ábrán látható, hogyan változott az átlagos nettó összhasznosság, mind direkt, mind iteratív mechanizmusnál különböző költségek esetén.



Ha ezeket az átlagos eredményeket vesszük, látható, hogy nagyon magas költségek esetén már a legjobb 2 iskolának való utánajárásnál egyértelműen az iteratív mechanizmussal sokkal nagyobb hasznosságot eredményez, a legjobb 3, vagy annál több iskolának való utánajárás esetén már alacsonyabb költségek esetén is az iteratív mechanizmus ad jobb megoldást. Nézzük, meg, hasonló eredményt kapunk-e akkor is, ha a diákok számát megemeljük 120-ra, azaz ugyanannyi diák van, mint férőhely, vagy 122-re, azaz több diák van, mint férőhely, és a diákok versenyeznek a helyekért.





Hasonló eredményt kapunk így is, magas költségek esetén direkt mechanizmus esetében egyáltalán nem éri meg utánajárást végezni, míg iteratív mechanizmus esetében utánajárással hasznosságnövekedés érhető el, alacsonyabb költségek esetén mindkét esetben hasznosságnövekedés érhető el, azonban ha elég sok iskolának járnak utána a diákok, iteratív mechanizmus esetén nagyobb hasznosságnövekedés figyelhető meg.

7.3. Következtetés

A modell erős feltételezéseket tett az adatok generálására, a diákok költségeire és döntéseire is, melynek realitását érdemes valódi adatokon tesztelni. Az mindenesetre elmondható, hogy az adott feltételezések mellett a szimulációk világos tanulsággal szolgálnak, az iteratív eljárás jóval

költséghatékonyabb, mint a direkt mechanizmus. Ha a költségek nagy hasznosságcsökkenést eredményeznek, akkor előfordulhat, hogy direkt mechanizmus esetében a diáknak nem éri meg utánajárást végezni, iteratív mechanizmus esetében viszont még igen. Ez természetesen nem azt jelenti, hogy az iteratív eljárás mindenképpen jobb, mint a direkt mechanizmus, hiszen ebben a szimulációban pusztán egy tényezőt, az utánajárási költséget vizsgáltuk meg. Mint láttuk a Franciaországban alkalmazott iteratív mechanizmus leírásánál, hogy időben nagyon sokáig elhúzódik, és az idő előrehaladtával a diákoknak egyre stresszesebb, ezért sok esetben a diákok hamarabb elfogadják véglegesen a kapott ajánlat egyikét, hogy tudjanak kollégiumba jelentkezni, albérletet keresni, és nem várják meg a végső eredményt, arról nem is beszélve, hogy az idő előrehaladtával a diákoknak egyre kevesebb idejük van dönteni, és a végső eredmény sem az iskola-optimális megoldást adja sok esetben, hiszen nincs elég ideje az algoritmusnak végigfutni.

8. fejezet

Összefoglalás

A szakdolgozat elején bevezettem néhány alapfogalmat, majd leírtam az alapmechanizmusokat, és egy példával szemléltettem őket, majd elméleti szinten összehasonlítottam néhány mechanizmust. Ezt követően bemutattam a hazai, illetve a francia egyetemi felvételi rendszert. Ezután ismertettem azt a párhuzamot, ami a magyar felsőoktatás és a kötelező gépjármű felelősségbiztosítás között fedezhető fel. Ezután a szimuláció következett. Célom az volt, hogy rávilágítsak arra, hogy az iteratív mechanizmusnak, amit Franciaországban alkalmaznak, és elsőre kicsit furcsának és idegennek tűnhet, létezhet előnye is. A modell úgy épült fel, hogy a diákok 1 mérés alapján megbecslik, hogy mekkora hasznosságot jelet számukra az iskola, és aztán ahhoz, hogy tegyenek egy második mérést, már költséget kell kifizetniük, és ha ezt a költséget egyfajta hasznosságcsökkenésként értelmezzük, akkor az iteratív mechanizmussal nagyobb nettó hasznossághoz juthatunk. (A nettó hasznosság az a hasznosság, amit a tényleges hasznosságból kapunk meg, ha levonjuk a kifizetett költséget.) Ezenkívül érdemes megjegyezni, hogy az iteratív mechanizmus elég nyilvánvalóan ösztönöz az igazmondásra, míg a direkt mechanizmusnál a diákoknak hiába érdekük a valós preferenciáik megadása, a diákok számára ez sokkal kevésbé nyilvánvaló. Azonban ez az iteratív mechanizmus időben nagyon sokáig elhúzódat, ezért Franciaországban úgy alkalmazzák, hogy a diákoknak egyre kevesebb idejük van dönteni, és így egyre stresszesebb a diákok számára, és sok esetben még így sem fut le végig az algoritmus. Mindenesetre ez az iteratív mechanizmus egy nagyon érdekes téma, és sok lehetőséget rejthet magában. További kutatásban érdemes lehet megvizsgálni ezt a modellt nem generált, hanem valós adatokkal, illetve azt a modellt is, amikor a diákok hasznosságbeli különbség alapján döntenek az utánajárásról.

Irodalomjegyzék

[1] Atila Abdulkadiroğlu, Tayfun Sönmez: *School Choice: A Mechanism Design Approach*, Columbia University, Department of Economics Discussion Paper Series (2003)

[2] Atila Abdulkadiroğlu, Yeon-Koo Che, Yosuke Yasuda: *Resolving Conflicting Preferences in School Choice: The "Boston Mechanism" Reconsidered*, American Economic Review 101 (February, 2011)

[3] Kóczy Á. László: *Központi felvételi rendszerek. Taktikázás és stabilitás*, Közgazdasági Szemle, LVI. évf. (2009. május)

[4] Atila Abdulkadiroğlu, Yeon-Koo Che, Parag A. Pathak, Alvin E. Roth, Olivier Tercieux: *Minimalizing Justified Envy in School Choice: The Design of New Orleans' OneApp*, NBER Working Paper Series (March, 2017)

[5] Biró Péter, Sofya Kiselgof: *College admissions with stable score-limits*, Central European Journal of Operations Research 23 (2015)

[6] Biró Péter, Fleiner Tamás, Robert W. Irving, David F. Manlove: *The College Admissions problem with lower and common quotas*, Theoretical Computer Science Volume 411, Issues 34–36 (17 July 2010)

[7] Internetes weblap: Parcoursup, `https://www.parcoursup.fr/index.php?desc=cest_quoi`. Utolsó letöltés: 2020.12.16. 18:00

[8] MINISTERE DE L'ENSEIGNEMENT SUPÉRIOR, DE LA RECHERCHE ET DE L'INNOVATION *Document de présentation des algorithmes de Parcoursup* (March, 2018)

[9] Stephanie Mignot-Gerard: *The Reform of French University Admissions* (13 March 2018), Internetes weblap: `https://www.insidehighered.com/blogs/world-view/reform-french-university-admissions`. Utolsó letöltés: 2020.12.16. 18:00

[10] Czotter Eszter: A magyar felsőoktatási felvételi algoritmus. *Szakdolgozat* (2014)

[11] Toma Zsófia: Egyetemi felvételik és iskolaválasztás Európában. *Szakdolgozat* (2019)

[12] Internetes weblap: Matching, `http://www.matching-in-practice.eu/`. Utolsó letöltés: 2020.12.15. 16:00

[13] Internetes weblap: Felvi, `http://www.felvi.hu/`. Utolsó letöltés: 2020.12.15. 16:00

[14] Internetes weblap: Eduline, `http://www.eduline.hu/`. Utolsó letöltés: 2020.12.15. 16:00

[15] Internetes weblap: ELTE Átsorolás, `https://qter.elte.hu/Statikus.aspx/Dokumentumok-Atsorolas`. Utolsó letöltés: 2020.12.15. 16:00

[16] Internetes weblap: Bonus-Malus1,

`https://www.mnb.hu/fogyasztovedelem/biztositasok/gepjarmu-biztositas/a-bonus-malus-besorolasi-rendszer`. Utolsó letöltés: 2020.12.15. 16:00

[17] Internetes weblap: Bonus-Malus2, `https://bank360.hu/blog/bonus-malus-rendszer`. Utolsó letöltés: 2020.12.15. 16:00

[18] Internet: Budapesti Corvinus Egyetem, `https://www.uni-corvinus.hu/fooldal/hello-corvinus/corvinus-osztondijprogram/`. Utolsó letöltés: 2020.12.15. 16:00

[19] Andrew D. Ho, Carol C. Yu: *Descriptive Statistics for Modern Test Score Distributions: Skewness, Kurtosis, Discreteness, and Ceiling Effects*, Educ Psychol Meas (Jun, 2015)

[20] Alvin E. Roth, Elliott Peranson: *The Redesign of the Matching Market for American Physicians: Some Engineering Aspects of Economic Design*, American Economic Review, 89(4), 748-780 (1999)

9. fejezet

Függelék

9.1. Költséges GS-algoritmushoz tartozó kód

```
install.packages("matchingR")
library(matchingR)
koltseges_gs<-function(dhb, dhv, ktsg, d_vagy_i, merrolfut, hany, ih, kap)
{
dsz<-nrow(dhb)
isz<-ncol(dhv)

#iskolak preferencia sorrendje
ip<-matrix(nrow=isz,ncol = dsz)
for (i in 1:isz){
ip[i,]<-order(ih[i,], decreasing=TRUE)
}

#diakok hasznossaga:
dh<-dhb

#diakok preferencia sorendje
dp<-matrix(nrow = dsz,ncol = isz)
for (i in 1:dsz){
dp[i,]<-order(dh[i,], decreasing=TRUE)
}

#outputhoz szükséges változók definiálása
```

```
dvh<-matrix(rep(0, dsz), nrow=dsz, ncol=1)
fd<-matrix(nrow=isz, ncol=dsz)
mk<-matrix(rep(0, dsz*isz), nrow=dsz, ncol=isz)
dk<-matrix(rep(0, dsz), nrow=dsz, ncol=1)
dnh<-matrix(rep(0, dsz), nrow=dsz, ncol=1)

if(d_vagy_i=="d"){
if(hany!=0){
for(i in 1:dsz){
for(j in 1:hany){
dh[i,dp[i,j]]<-dhv[i,dp[i,j]]
mk[i,dp[i,j]]<-ktsg
}
dp[i,]<-order(dh[i,], decreasing=TRUE)
dk[i]<-sum(mk[i,])
}
}
if(merrolfut=="df"){
parositas=galeShapley.collegeAdmissions(studentPref=t(dp),
collegePref=t(ip),
slots=kap, #iskolák kapacitása
studentOptimal=TRUE)
}
if(merrolfut=="if"){
parositas=galeShapley.collegeAdmissions(studentPref=t(dp),
collegePref=t(ip),
slots=kap, #iskolák kapacitása
studentOptimal=FALSE)
}

vi<-parositas[["matched.students"]]
fd<-parositas[["matched.colleges"]]
}

if(d_vagy_i=="i"&&merrolfut=="if"){ #iteratív eset
#algoritmushoz szükséges változók:
```

```
kap_vektor<-rep(kap, isz)
elf_isk<-matrix(nrow=dsz, ncol=1)#elfogadott iskola
elozo_isk<-matrix(rep(0, dsz), nrow=dsz, ncol=1)#előző iskola
sorsz<-rep(1, isz)
repeat{
aj<-matrix(rep(0, dsz*isz), nrow = dsz, ncol=isz)
diak_sorsz<-matrix(rep(1, dsz), nrow=dsz, ncol=1)

#az iskolák megteszik az ajánlatukat:
for(j in 1:isz){
if(sorsz[j]<=dsz){
eddig<-sorsz[j]+kap_vektor[j]-1
if(eddig>dsz){
eddig<-dsz
}
for(p in sorsz[j]:eddig){
if(kap_vektor[j]!=0){
aj[ip[j, p],diak_sorsz[ip[j, p],1]]<-j
diak_sorsz[ip[j, p],1]<-diak_sorsz[ip[j, p],1]+1
}
}
sorsz[j]<-eddig+1
}
}

#végigmegyünk az ajánlatok mátrixon, a diákok döntenek:
for(i in 1:dsz){
if(diak_sorsz[i]!=1){ #ha a diák ebben a körben kapott ajánlatot
#ajokat prioritassorrendbe tesszük
aji_p<-matrix(rep(0, diak_sorsz[i]-1), nrow = 1, ncol=diak_sorsz[i]-1)
hely_p<-matrix(rep(0, diak_sorsz[i]-1), nrow = 1, ncol=diak_sorsz[i]-1)
for(k in 1:(diak_sorsz[i]-1)){
hely_p[k]<-which(dp[i,]==aj[i,k])
}
for(k in 1:(diak_sorsz[i]-1)){
aji_p[k]<-aj[i, which.min(hely_p)]
```

```
hely_p[which.min(hely_p)]<-NA
}

#váltott-e a diák iskolát ebben a körben
valt<-0
for(j in 1:(diak_sorsz[i]-1)){
legjobb_aj<-aji_p[j]
if(is.na(elf_isk[i])==TRUE){
elf_isk[i]<-legjobb_aj
valt<-valt+1
}
else{
if(hany!=0){
if(which(dp[i,]==elf_isk[i])<=hany&&which(dp[i,]==legjobb_aj)<=hany){
if(mk[i,elf_isk[i]]==0){
mk[i,elf_isk[i]]<-ktsg
dh[i, elf_isk[i]]<-dhv[i,elf_isk[i]]
}
if(mk[i,legjobb_aj]==0){
mk[i,legjobb_aj]<-ktsg
dh[i, legjobb_aj]<-dhv[i,legjobb_aj]
}
dp[i,]<-order(dh[i,], decreasing=TRUE)
}
}
if(which(dp[i,]==elf_isk[i])>which(dp[i,]==legjobb_aj)){
if(valt==0){
elozo_isk[i]<-elf_isk[i]
}
elf_isk[i]<-legjobb_aj
valt<-valt+1
}
}
}
if(valt!=0){
if(elozo_isk[i]!=0){
```

```
kap_vektor[elozo_isk[i]]<-kap_vektor[elozo_isk[i]]+1
}
kap_vektor[elf_isk[i]]<-kap_vektor[elf_isk[i]]-1
}
}
dk[i]<-sum(mk[i,])
}

#kilépés
v<-0
for(j in 1:isz){
if(kap_vektor[j]==0|sorsz[j]==dsz+1){
v<-v+1
}
}
if(v==isz){
break
}

}

#végleges iskola
vi<-elf_isk

#felvett diákok
szaml<-matrix(c(rep(1, isz), nrow=isz, ncol=1))
for(i in 1:dsz){
vi[i]<-elf_isk[i]
if(is.na(vi[i])==FALSE){
fd[vi[i],szaml[vi[i]]]<-i
szaml[vi[i]]<-szaml[vi[i]]+1
}
}
}#iteratív algoritmus vége

#diákok végső hasznosság, nettó hasznosság
```

```
for(i in 1:dsz){
if(is.na(vi[i])==FALSE){
dvh[i]<-dhv[i,vi[i]]
dnh[i]<-dvh[i]-dk[i]
}
}
oh<-sum(dvh[])
ok<-sum(dk[])
netto_oh<-sum(dnh[])

return(
list(
vegleges_iskola=vi,
felvett_diakok=fd,
diakok_haszn=dvh,
diakok_ktsg=dk,
diakok_netto_haszn=dnh,
osszhaszn=oh,
osszktsg=ok,
netto_osszhaszn=netto_oh
)
)
}
```

9.2. Utánajárási költség nélküli eset modellezésének kódja

```
dsz<-118 #120 vagy 122 diák
isz<-10
kap<-12
ktsg<-0
alfa<-3
beta<-1
set.seed(1)
```

```
#iskolák hasznossága generálás
diakok_kepessege<-rbeta(dsz, alfa, beta)
ih<-matrix(rep(0,dsz*isz), nrow = isz, ncol = dsz)
for (i in 1:dsz){
ih[,i]<-rbinom(isz, 100, diakok_kepessege[i])
}

#diákok hasznosságának generálása
iskolak_nepszerusege<-rbeta(isz,alfa,beta)
#gondolt hasznosság
dhb<-matrix(rep(0,dsz*isz), nrow = dsz, ncol = isz)
for (i in 1:isz){
dhb[,i]<-rbinom(dsz, 99, iskolak_nepszerusege[i])
}
#valós hasznosság
dhv<-matrix(rep(0,dsz*isz), nrow = dsz, ncol = isz)
for (i in 1:isz){
dhv[,i]<-rbinom(dsz, 99, dhb[,i]/100)
}
#########
#diákok preferenciája
dpg<-matrix(nrow = dsz,ncol = isz)
for (i in 1:dsz){
dpg[i,]<-order(dhb[i,], decreasing=TRUE)
}
dpv<-matrix(nrow = dsz,ncol = isz)
for (i in 1:dsz){
dpv[i,]<-order(dhv[i,], decreasing=TRUE)
}
#########
eredmeny_df_d_0<-koltseges_gs(dhb, dhv, ktsg, 'd', "df", 0, ih, kap)
eredmeny_if_d_0<-koltseges_gs(dhb, dhv, ktsg, 'd', "if", 0, ih, kap)
eredmeny_if_i_0<-koltseges_gs(dhb, dhv, ktsg, 'i', "if", 0, ih, kap)
eredmeny_df_d_10<-koltseges_gs(dhb, dhv, ktsg, 'd', "df", 10, ih, kap)
eredmeny_if_d_10<-koltseges_gs(dhb, dhv, ktsg, 'd', "if", 10, ih, kap)
eredmeny_if_i_10<-koltseges_gs(dhb, dhv, ktsg, 'i', "if", 10, ih, kap)
```

```
#melyik diáknak mennyit változik a hasznossága, ha valós adatokat adnak meg?
haszn_valt_df_d<-eredmeny_df_d_10$diakok_haszn-eredmeny_df_d_0$diakok_haszn
sum(haszn_valt_df_d)

haszn_valt_if_d<-eredmeny_if_d_10$diakok_haszn-eredmeny_if_d_0$diakok_haszn
sum(haszn_valt_if_d)

haszn_valt_if_i<-eredmeny_if_i_10$diakok_haszn-eredmeny_if_i_0$diakok_haszn
sum(haszn_valt_if_i)
```

9.3. Utánajárási költség figyelembevételével direkt és iteratív eljárás összehasonlításához tartozó kód

```
dsz<-118 #120, 122
isz<-10
kap<-12
ktsg<-0.05 #0.1, 0.15, 0.2
alfa<-3
beta<-1
set.seed(1)

futasszam<-10
oh_mtx_d<-matrix(nrow=futasszam, ncol=isz+1)
oh_mtx_i<-matrix(nrow=futasszam, ncol=isz+1)
noh_mtx_d<-matrix(nrow=futasszam, ncol=isz+1)
noh_mtx_i<-matrix(nrow=futasszam, ncol=isz+1)
for(k in 1:futasszam){
diakok_kepessege<-rbeta(dsz, alfa, beta)
ih<-matrix(rep(0,dsz*isz), nrow = isz, ncol = dsz)
for (i in 1:dsz){
ih[,i]<-rbinom(isz, 100, diakok_kepessege[i])
}
iskolak_nepszerusege<-rbeta(isz,alfa,beta)
```

```
dhg<-matrix(rep(0,dsz*isz), nrow = dsz, ncol = isz)
for (i in 1:isz){
dhg[,i]<-rbinom(dsz, 99, iskolak_nepszerusege[i])
}
dhv<-matrix(rep(0,dsz*isz), nrow = dsz, ncol = isz)
for (i in 1:isz){
dhv[,i]<-rbinom(dsz, 99, dhg[,i]/100)
}

for(j in 0:isz){
eredmeny_d<-koltseges_gs(dhg, dhv, ktsg, 'd', "df", j, ih, kap)
eredmeny_i<-koltseges_gs(dhg, dhv, ktsg, 'i', "if", j, ih, kap)

oh_mtx_d[k, j+1]<-eredmeny_d$osszhaszn
noh_mtx_d[k, j+1]<-eredmeny_d$netto_osszhaszn
oh_mtx_i[k, j+1]<-eredmeny_i$osszhaszn
noh_mtx_i[k, j+1]<-eredmeny_i$netto_osszhaszn
}
}

oh_valt_d<-matrix(nrow=futasszam, ncol=isz)
noh_valt_d<-matrix(nrow=futasszam, ncol=isz)
oh_valt_i<-matrix(nrow=futasszam, ncol=isz)
noh_valt_i<-matrix(nrow=futasszam, ncol=isz)

for(i in 1:futasszam){
for(j in 1:isz){
oh_valt_d[i,j]<-oh_mtx_d[i, j+1]-oh_mtx_d[i, 1]
noh_valt_d[i,j]<-noh_mtx_d[i, j+1]-noh_mtx_d[i, 1]
oh_valt_i[i,j]<-oh_mtx_i[i, j+1]-oh_mtx_i[i, 1]
noh_valt_i[i,j]<-noh_mtx_i[i, j+1]-noh_mtx_i[i, 1]
}
}
```

9.4. Ábrákhoz tartozó táblázatok

		Összhasznosság változás, direkt eset - diákok felől futtatva (118 diák)									
		Hány iskolának jár utána									
		1	2	3	4	5	6	7	8	9	10
Hanyadik generált adatsor	1	41	48	76	104	100	124	177	150	190	190
	2	13	0	50	75	91	108	156	202	213	192
	3	-8	43	52	62	42	91	147	200	197	215
	4	9	38	40	40	51	97	94	109	138	142
	5	15	35	57	74	75	51	78	126	142	152
	6	9	21	17	19	44	84	79	109	151	151
	7	-19	3	6	27	29	60	84	88	116	128
	8	25	50	58	83	94	120	133	145	162	162
	9	36	52	56	56	93	100	150	161	222	204
	10	16	21	25	43	46	48	48	61	51	57

		Összhasznosság változás, iteratív eset - iskolák felől futtatva (118 diák)									
		Hány iskolának jár utána									
		1	2	3	4	5	6	7	8	9	10
Hanyadik generált adasor	1	0	56	63	72	88	120	149	150	190	190
	2	0	0	25	92	116	131	167	163	213	192
	3	0	11	60	75	75	83	144	184	185	228
	4	0	29	35	35	35	57	89	104	133	137
	5	0	0	67	63	63	58	78	78	138	152
	6	0	18	26	38	31	84	84	86	151	151
	7	0	16	6	6	-1	50	50	63	48	104
	8	0	31	55	82	111	152	152	158	179	179
	9	0	36	56	56	56	93	87	157	204	204
	10	0	17	19	22	46	48	48	48	35	57

		Nettó összhasznosság változás, direkt eset - diákok felől (118 diák - költség: 0,05)									
		Hány iskolának jár utána									
		1	2	3	4	5	6	7	8	9	10
Hanyadik generált adatsor	1	35,10	36,20	58,30	80,40	70,50	88,60	135,70	102,80	136,90	131,00
	2	7,10	-11,80	32,30	51,40	61,50	72,60	114,70	154,80	159,90	133,00
	3	-13,90	31,20	34,30	38,40	12,50	55,60	105,70	152,80	143,90	156,00
	4	3,10	26,20	22,30	16,40	21,50	61,60	52,70	61,80	84,90	83,00
	5	9,10	23,20	39,30	50,40	45,50	15,60	36,70	78,80	88,90	93,00
	6	3,10	9,20	-0,70	-4,60	14,50	48,60	37,70	61,80	97,90	92,00
	7	-24,90	-8,80	-11,70	3,40	-0,50	24,60	42,70	40,80	62,90	69,00
	8	19,10	38,20	40,30	59,40	64,50	84,60	91,70	97,80	108,90	103,00
	9	30,10	40,20	38,30	32,40	63,50	64,60	108,70	113,80	168,90	145,00
	10	10,10	9,20	7,30	19,40	16,50	12,60	6,70	13,80	-2,10	-2,00
	Átlag	7,80	19,3	26	34,7	37	52,9	73,3	87,9	105,1	100,3

		Nettó összhasznosság változás, iteratív eset - iskolák felől (118 diák - költség: 0,05)									
		Hány iskolának jár utána									
		1	2	3	4	5	6	7	8	9	10
Hanyadik generált adatsor	1	0,00	52,30	57,00	63,85	77,35	105,15	128,90	125,50	160,60	154,70
	2	0,00	-1,35	21,00	83,25	104,25	116,00	147,65	138,80	183,15	156,45
	3	0,00	9,30	55,95	68,30	65,60	69,55	124,45	159,75	155,45	193,00
	4	0,00	26,90	30,35	28,95	25,85	44,00	71,85	82,00	105,35	103,45
	5	0,00	-1,40	62,20	55,90	53,70	44,65	60,65	55,70	109,50	117,75
	6	0,00	15,80	21,35	30,85	21,35	71,00	66,25	62,80	122,70	116,95
	7	0,00	13,30	2,05	-0,65	-11,20	35,90	31,90	40,05	19,25	69,50
	8	0,00	29,25	50,50	74,70	100,15	137,45	133,35	133,85	150,05	144,25
	9	0,00	33,90	52,05	50,10	47,10	80,10	69,60	133,20	173,10	168,60
	10	0,00	14,90	14,50	14,50	35,75	34,35	30,50	25,55	6,35	23,05
	Átlag	0,00	19,29	36,70	46,98	51,99	73,82	86,51	95,72	118,55	124,77

Nettó összhasznosság változás, direkt eset - diákok felől (118 diák - költség: 0,1)										
	Hány iskolának jár utána									
Hanyadik generált adatsor	1	2	3	4	5	6	7	8	9	10
1	29,20	24,40	40,60	56,80	41,00	53,20	94,40	55,60	83,80	72,00
2	1,20	-23,60	14,60	27,80	32,00	37,20	73,40	107,60	106,80	74,00
3	-19,80	19,40	16,60	14,80	-17,00	20,20	64,40	105,60	90,80	97,00
4	-2,80	14,40	4,60	-7,20	-8,00	26,20	11,40	14,60	31,80	24,00
5	3,20	11,40	21,60	26,80	16,00	-19,80	-4,60	31,60	35,80	34,00
6	-2,80	-2,60	-18,40	-28,20	-15,00	13,20	-3,60	14,60	44,80	33,00
7	-30,80	-20,60	-29,40	-20,20	-30,00	-10,80	1,40	-6,40	9,80	10,00
8	13,20	26,40	22,60	35,80	35,00	49,20	50,40	50,60	55,80	44,00
9	24,20	28,40	20,60	8,80	34,00	29,20	67,40	66,60	115,80	86,00
10	4,20	-2,60	-10,40	-4,20	-13,00	-22,80	-34,60	-33,40	-55,20	-61,00
Átlag	1,9	7,5	8,3	11,1	7,5	17,5	32	40,7	52	41,3

Nettó összhasznosság változás, iteratív eset - iskolák felől (118 diák - költség: 0,1)										
	Hány iskolának jár utána									
Hanyadik generált adatsor	1	2	3	4	5	6	7	8	9	10
1	0,00	48,60	51,00	55,70	66,70	90,30	108,80	101,00	131,20	119,40
2	0,00	-2,70	17,00	74,50	92,50	101,00	128,30	114,60	153,30	120,90
3	0,00	7,60	51,90	61,60	56,20	56,10	104,90	135,50	125,90	158,00
4	0,00	24,80	25,70	22,90	16,70	31,00	54,70	60,00	77,70	69,90
5	0,00	-2,80	57,40	48,80	44,40	31,30	43,30	33,40	81,00	83,50
6	0,00	13,60	16,70	23,70	11,70	58,00	48,50	39,60	94,40	82,90
7	0,00	10,60	-1,90	-7,30	-21,40	21,80	13,80	17,10	-9,50	35,00
8	0,00	27,50	46,00	67,40	89,30	122,90	114,70	109,70	121,10	109,50
9	0,00	31,80	48,10	44,20	38,20	67,20	52,20	109,40	142,20	133,20
10	0,00	12,80	10,00	7,00	25,50	20,70	13,00	3,10	-22,30	-10,90
Átlag	0,00	17,18	32,19	39,85	41,98	60,03	68,22	72,34	89,50	90,14

Nettó összhasznosság változás, direkt eset - diákok felől (118 diák - költség: 0,15)										
	Hány iskolának jár utána									
Hanyadik generált adatsor	1	2	3	4	5	6	7	8	9	10
1	23,30	12,60	22,90	33,20	11,50	17,80	53,10	8,40	30,70	13,00
2	-4,70	-35,40	-3,10	4,20	2,50	1,80	32,10	60,40	53,70	15,00
3	-25,70	7,60	-1,10	-8,80	-46,50	-15,20	23,10	58,40	37,70	38,00
4	-8,70	2,60	-13,10	-30,80	-37,50	-9,20	-29,90	-32,60	-21,30	-35,00
5	-2,70	-0,40	3,90	3,20	-13,50	-55,20	-45,90	-15,60	-17,30	-25,00
6	-8,70	-14,40	-36,10	-51,80	-44,50	-22,20	-44,90	-32,60	-8,30	-26,00
7	-36,70	-32,40	-47,10	-43,80	-59,50	-46,20	-39,90	-53,60	-43,30	-49,00
8	7,30	14,60	4,90	12,20	5,50	13,80	9,10	3,40	2,70	-15,00
9	18,30	16,60	2,90	-14,80	4,50	-6,20	26,10	19,40	62,70	27,00
10	-1,70	-14,40	-28,10	-27,80	-42,50	-58,20	-75,90	-80,60	-108,30	-120,00
Átlag	-4	-4,3	-9,4	-12,5	-22	-17,9	-9,3	-6,5	-1,1	-17,7

Nettó összhasznosság változás, iteratív eset - iskolák felől (118 diák - költség: 0,15)										
	Hány iskolának jár utána									
Hanyadik generált adatsor	1	2	3	4	5	6	7	8	9	10
1	0,00	44,90	45,00	47,55	56,05	75,45	88,70	76,50	101,80	84,10
2	0,00	-4,05	13,00	65,75	80,75	86,00	108,95	90,40	123,45	85,35
3	0,00	5,90	47,85	54,90	46,80	42,65	85,35	111,25	96,35	123,00
4	0,00	22,70	21,05	16,85	7,55	18,00	37,55	38,00	50,05	36,35
5	0,00	-4,20	52,60	41,70	35,10	17,95	25,95	11,10	52,50	49,25
6	0,00	11,40	12,05	16,55	2,05	45,00	30,75	16,40	66,10	48,85
7	0,00	7,90	-5,85	-13,95	-31,60	7,70	-4,30	-5,85	-38,25	0,50
8	0,00	25,75	41,50	60,10	78,45	108,35	96,05	85,55	92,15	74,75
9	0,00	29,70	44,15	38,30	29,30	54,30	34,80	85,60	111,30	97,80
10	0,00	10,70	5,50	-0,50	15,25	7,05	-4,50	-19,35	-50,95	-44,85
Átlag	0,00	15,07	27,69	32,73	31,97	46,25	49,93	48,96	60,45	55,51

		Nettó összhasznosság változás, direkt eset - diákok felől (118 diák - költség: 0,2)									
		Hány iskolának jár utána									
		1	2	3	4	5	6	7	8	9	10
H anyadik generált adatsor	1	17,40	0,80	5,20	9,60	-18,00	-17,60	11,80	-38,80	-22,40	-46,00
	2	-10,60	-47,20	-20,80	-19,40	-27,00	-33,60	-9,20	13,20	0,60	-44,00
	3	-31,60	-4,20	-18,80	-32,40	-76,00	-50,60	-18,20	11,20	-15,40	-21,00
	4	-14,60	-9,20	-30,80	-54,40	-67,00	-44,60	-71,20	-79,80	-74,40	-94,00
	5	-8,60	-12,20	-13,80	-20,40	-43,00	-90,60	-87,20	-62,80	-70,40	-84,00
	6	-14,60	-26,20	-53,80	-75,40	-74,00	-57,60	-86,20	-79,80	-61,40	-85,00
	7	-42,60	-44,20	-64,80	-67,40	-89,00	-81,60	-81,20	-100,80	-96,40	-108,00
	8	1,40	2,80	-12,80	-11,40	-24,00	-21,60	-32,20	-43,80	-50,40	-74,00
	9	12,40	4,80	-14,80	-38,40	-25,00	-41,60	-15,20	-27,80	9,60	-32,00
	10	-7,60	-26,20	-45,80	-51,40	-72,00	-93,60	-117,20	-127,80	-161,40	-179,00
	Átlag	-9,9	-16,1	-27,1	-36,1	-51,5	-53,3	-50,6	-53,7	-54,2	-76,7

		Nettó összhasznosság változás, iteratív eset - iskolák felől (118 diák - költség: 0,2)									
		Hány iskolának jár utána									
		1	2	3	4	5	6	7	8	9	10
H anyadik generált adatsor	1	0,00	41,20	39,00	39,40	45,40	60,60	68,60	52,00	72,40	48,80
	2	0,00	-5,40	9,00	57,00	69,00	71,00	89,60	66,20	93,60	49,80
	3	0,00	4,20	43,80	48,20	37,40	29,20	65,80	87,00	66,80	88,00
	4	0,00	20,60	16,40	10,80	-1,60	5,00	20,40	16,00	22,40	2,80
	5	0,00	-5,60	47,80	34,60	25,80	4,60	8,60	-11,20	24,00	15,00
	6	0,00	9,20	7,40	9,40	-7,60	32,00	13,00	-6,80	37,80	14,80
	7	0,00	5,20	-9,80	-20,60	-41,80	-6,40	-22,40	-28,80	-67,00	-34,00
	8	0,00	24,00	37,00	52,80	67,60	93,80	77,40	61,40	63,20	40,00
	9	0,00	27,60	40,20	32,40	20,40	41,40	17,40	61,80	80,40	62,40
	10	0,00	8,60	1,00	-8,00	5,00	-6,60	-22,00	-41,80	-79,60	-78,80
	Átlag	0,00	12,96	23,18	25,60	21,96	32,46	31,64	25,58	31,40	20,88

		Átlagos nettó összhasznosság változás (118 diák)									
		Hány iskolának járnak utána a diákok									
	Költségek	1	2	3	4	5	6	7	8	9	10
Direkt eset	0,05	7,8	19,3	26	34,7	37	52,9	73,3	87,9	105,1	100,3
	0,1	1,9	7,5	8,3	11,1	7,5	17,5	32	40,7	52	41,3
	0,15	-4	-4,3	-9,4	-12,5	-22	-17,9	-9,3	-6,5	-1,1	-17,7
	0,2	-9,9	-16,1	-27,1	-36,1	-51,5	-53,3	-50,6	-53,7	-54,2	-76,7
Iteratív eset	0,05	0	19,29	36,695	46,975	51,99	73,815	86,51	95,72	118,55	124,77
	0,1	0	17,18	32,19	39,85	41,98	60,03	68,22	72,34	89,5	90,14
	0,15	0	15,07	27,685	32,725	31,97	46,245	49,93	48,96	60,45	55,51
	0,2	0	12,96	23,18	25,6	21,96	32,46	31,64	25,58	31,4	20,88

		Átlagos nettó összhasznosság változás (120 diák)									
		Hány iskolának járnak utána a diákok									
	Költségek	1	2	3	4	5	6	7	8	9	10
Direkt eset	0,05	9,2	11,3	26,7	48,9	57,9	56,5	75	98	103,3	101
	0,1	3,2	-0,7	8,7	24,9	27,9	20,5	33	50	49,3	41
	0,15	-2,8	-12,7	-9,3	0,9	-2,1	-15,5	-9	2	-4,7	-19
	0,2	-8,8	-24,7	-27,3	-23,1	-32,1	-51,5	-51	-46	-58,7	-79
Iteratív eset	0,05	0	14,215	31,22	47,04	67,575	73,25	91,84	109,41	122,675	130,47
	0,1	0	12,13	26,94	40,08	57,25	59,1	73,28	86,02	93,95	95,94
	0,15	0	10,045	22,66	33,12	46,925	44,95	54,72	62,63	65,225	61,41
	0,2	0	7,96	18,38	26,16	36,6	30,8	36,16	39,24	36,5	26,88

		Átlagos nettó összhasznosság változás (122 diák)									
		Hány iskolának járnak utána a diákok									
	Költségek	1	2	3	4	5	6	7	8	9	10
Direkt eset	0,05	4,7	8,3	17,7	28,3	38,1	54,6	59,2	63,7	76,1	70,4
	0,1	-1,3	-3,7	-0,3	4,3	8,1	18,6	17,2	15,7	22,1	10,4
	0,15	-7,3	-15,7	-18,3	-19,7	-21,9	-17,4	-24,8	-32,3	-31,9	-49,6
	0,2	-13,3	-27,7	-36,3	-43,7	-51,9	-53,4	-66,8	-80,3	-85,9	-109,6
Iteratív eset	0,05	0	12,955	19,73	29,77	49,235	57,54	75,555	85,415	94,32	90,575
	0,1	0	11,11	15,76	23,04	39,27	43,68	57,11	62,13	66,04	56,55
	0,15	0	9,265	11,79	16,31	29,305	29,82	38,665	38,845	37,76	22,525
	0,2	0	7,42	7,82	9,58	19,34	15,96	20,22	15,56	9,48	-11,5