BUDAPESTI CORVINUS EGYETEM

Közgazdasági és
Gazdaságinformatikai
Doktori Iskola

Tézisgyűjtemény

Kondor Gábor

Egyoldali párosítási piacok egyenletes klaszterezési megközelítésben

című Ph.D. értekezéséhez

Témavezető:

Dr. Vidovics-Dancs Ágnes

Egyetemi docens

Budapest, 2022

Pénzügy Tanszék

Tézisgyűjtemény

Kondor Gábor

Egyoldali párosítási piacok egyenletes klaszterezési megközelítésben

című Ph.D. értekezéséhez

Témavezető:

Dr. Vidovics-Dancs Ágnes

Egyetemi docens

Tartalomjegyzék

1. fejezet

Kutatási előzmények és a téma indoklása

A közgazdaságtan egyik fontos kérdése, hogy az egyes piacok hogyan allokálják az erőforrásokat. A párosítási piacokon nincs, vagy csak részben van olyan árrendszer, amely meghatározza az allokációkat, és a létrejövő párosításokat elsősorban a piacot szabályozó mechanizmusok adják meg. (Nobel Prize, 2012b)

A párosítási piacok elméletének alapjait Gale és Shapley (1962) fektették le[1], akik a párosítási feladat megoldására a *stabilitás* koncepciót javasolták. Ennek megfelelően kétfős párok esetében olyan párosítás kialakítása a cél, amelyben nincs két olyan különböző párokban lévő szereplő, akik jobban preferálják egymást, mint saját párjukat. Az egyoldali párosítási piacok egyik alapvető modellje a Gale és Shapley (1962) által meghatározott *stabil szobatársak probléma*, amelyben a szereplőknek szigorú preferencia-rendezése van a többiekre nézve, és a feladat egy stabil párosítás létrehozása, amennyiben az létezik.

Az egyoldali párosítások egyik fontos alkalmazási területe a vesecsere-programok (lásd pl. Biró (2006)). Ezek esetén olyan beteg-donor párok a piaci szereplők, amelyek egyik tagjának új vesére van szüksége, a másik tagja pedig hajlandó lenne donorként odaadni az egyik veséjét, ugyanakkor egymással, a transzplantáció szempontjából, nem kompatibilisek. A feladat olyan párosítás kialakítása, amelyben az egymáshoz társított két különböző beteg-donor pár között keresztben valószínűsíthetően megvalósulhat a transzplantáció -

[1] Az elmélet később számos gyakorlati alkalmazás alapjául szolgált, mint például orvosi rezidensek kórházakhoz rendelése, egyetemi felvételi eljárások, vesecsere-programok. A stabil allokációk elméletéért és a piactervezés területén végzett munkájukért Lloyd Shapley és Alvin E. Roth kapta a 2012-es közgazdasági Nobel-díjat (Nobel Prize, 2012a).

későbbi tesztek során ennek még sajnálatos módon kiderülhet az ellenkezője.

Morrill (2010) az egyoldali párosítási feladat megoldásra a stabilitással szemben egy alternatív megközelítést, a Pareto-hatékonyságot javasolja. Érvelése szerint a szobatárs probléma esetében a stabilitás figyelmen kívül hagyja azt a meghatározó fizikai korlátot, hogy a szobatársaknak szobákra van szüksége, és ennélfogva az túlságosan szigorú. Ugyanis, hogyha egy szobabeosztás már kialakult, akkor hiába szeretne két különböző szobában lévő egyén összeköltözni, egyoldalúan egyikük sem költöztetheti ki a saját szobatársát, és az új párosítás nem jöhet létre. Ez különösen fontos lehet a vesecserék esetében, ahol a párosítás után is kiderülhet még inkompatibilitás.

Az eddig említett megközelítések párok kialakítsára vonatkoznak, ugyanakkor a gyakorlatban vannak olyan feladatok, amelyek esetében előfordulhatnak nagyobb méretű csoportok is. Például a vesecserék esetében van olyan program, amelyben megengedettek a hármas párok (Biró, 2006), a szobatársak tekintetében pedig gyakran 3- vagy 4-fős szobákkal találkozunk. A szobatárs problémának azt a változatát, amelyben háromfős szobák kialakítása a feladat, 3-dimenziós szobatárs problémának (3D-SR) nevezzük. Ismereteink szerint egy kivétellel (Arkin és szerzőtársai (2009) 2-stabil párosítás megközelítése) valamennyi 3-dimenziós felírás NP-teljes eredményre vezet. Ez azt jelenti, hogy ezen feladatok esetén bonyolultságelméleti szempontból nincs hatékony módszer a megoldás meghatározására, így a gyakorlatban nagyméretű problémák esetén nem tudjuk megadni azt. Továbbá megjegyezzük, hogy mindössze egy olyan megközelítéssel találkoztunk (Lam és Plaxton (2019) teljes ciklikus listákra vonatkozó eredménye), amely a háromnál magasabb dimenzió esetét is tárgyalja.

Segev és szerzőtársai (2005) a vesecserékre egy olyan gyakorlatban is alkalmazott megközelítést tekintettek, amely egy súlyozott párosítási problémára vezethető vissza (Biró, 2006). A szerzők szimulációs kísérletek alapján úgy találták, hogy módszerükkel országos szinten jobb eredményt lehetne elérni, mint az "első találatot elfogadó" párosítási eljárást alkalmazó vesecsere központok esetében.

A disszertáció célja röviden a következőképpen fogalmazható meg:

- A súlyozott párosítási megközelítés kiterjesztése egyoldali párosítási kontextusban a csoportok tetszőleges méretére. Kifejezett célunk olyan keret meghatározása, amelyben a minimalizálási és maximalizálási célokat egyaránt tudjuk vizsgálni.
- Az így megfogalmazott feladatok elméleti megoldhatóságának, vagy másképpen azok

bonyolultságának vizsgálata.

- A feladatok gyakorlati megoldhatóságának és tulajdonságainak vizsgálata széleskörű szimulációs kísérletek segítségével az irodalomban található heurisztikák és saját algoritmusok eredményeinek összehasonlításával.

A súlyozott párosítási feladat tetszőleges m csoportméretre történő kiterjesztésére az m-dimenziós párosítási problémát vesszük alapul. Ez egy gráfparticionálási problémaként kezeli a csoportosítási feladatot, amelyben Pareto-hatékony megoldást határozunk meg. A disszertácó középpontjában ennek egy speciális esete áll, amelyben a szereplők egy euklideszi tér pontjainak felelnek meg, a kapcsolataikat a közöttük adódó euklideszi távolságok adják meg, a cél pedig m főből álló csoportok létrehozása úgy, hogy a csoportokon belüli távolságok négyzetösszege a konkrét feladattól függően minimális vagy maximális legyen.

Vegyük észre, hogy a feladat során a célfüggvény értékéhez a csoporton belüli valamennyi élt figyelembe kell venni. Ebből következően a vesecserékre háromfősnél nagyobb csoportok kialakítására nem tudjuk interpretálni, hiszen ennél az alkalmazásnál tulajdonképpen körök kialakítása a feladat. Megjegyezzük továbbá, hogy a vesecserékre a gyakorlatban helyesebb lenne olyan megoldási koncepció, amely megengedi egyszerre a két- és háromfős csoportokat is, valamint figyelembe tudja venni a háromfős csoportok esetén a két különböző csere-irányt is.

Az előző megfontolások miatt az euklideszi térben felírt problémát szobatársak párosításaként, vagy másképpen kollégiumi szobákhoz rendeléseként kezeljük, és *m-szobatárs problémának*[2,3] nevezzük. Az euklideszi tér dimenziói a hallgatók tulajdonságait reprezentálják, és az euklideszi távolságok határozzák meg, hogy mennyire lennének jó szobatársak. A cél egy kollégium hallgatóinak beosztása egyenlő méretű szobákba úgy, hogy az számukra a legelőnyösebb legyen.

A csoportosítás módjára két különböző megközelítést tárgyalunk, amelyek mellett különböző érvek szólnak. Az egyikben azt tesszük fel, hogy olyan csoportok kialakítása a kedvező, amelyben hasonló egyének szerepelnek. Ekkor homogén csoportokról vagy klaszterekről beszélünk. Emögött a motivációt a hasonló érdeklődési területek és tulajdonságok révén feltételezhetően kialakuló jobb kapcsolatok adják. Ekkor egy minimalzálási

[2]Vegyük észre, hogy az m-szobatárs probléma csupán elnevezésében hasonlít a Gale és Shapley (1962) által megfogalmazott szobatárs problémára. Az általunk tárgyalt feladat nem preferencia-sorrendekre épül, és nem cél stabil megoldás meghatározása.

[3]Ugyanerre a problémára Kondor (2018) konferenciaelőadásomban és 2015-ös k-szobatárs probléma című TDK dolgozatomban a k-szobatárs probléma elnevezést használtuk. Ugyanakkor, hogy konzisztensek legyünk a szélesebb szakirodalomban megjelenő, kapcsolódó problémákkal, ezt módosítottuk.

problémát tekintünk. A másik megközelítésben ezzel ellenkezőleg, a napjainkban egyre nagyobb szerepet kapó diverzitás fontosságával összhangban, olyan szobabeosztásokat kívánunk megadni, amelyben a szobatársak a lehető legváltozatosabb tulajdonságokkal rendelkeznek. Ekkor heterogén csoportok kialakítása céljából egy maximalizálási problémát vizsgálunk.

Hagyományosan a *klasztereket* olyan csoportoknak tekintjük, amelyek elemei egymáshoz hasonlóak. Ebből eredően a mi esetünkben szigorúan véve az m-dimenziós párosítás és az m-szobatárs problémáknak csupán a minimalizálási verziója tekinthető egyenletes klaszterezési feladatnak, vagyis olyan problémának, amelyben a cél egymáshoz hasonló elemekből álló, egyenlő méretű csoportok kialakítása. A mi esetünkben a maximalizálási feladat célja diverz csoportok kialakítása. Ugyanakkor az irodalomban található olyan megközelítés (lásd például Feo és Khellaf (1990), valamint Feo és szerzőtársai (1992) által tekintett k-particionálás probléma), amelyben a szereplők közötti távolságokat kompatibilitási értékekként interpretálják, így a klaszterek kialakítását egy maximalizálási feladat írja le. Emiatt, és részben az egyszerűbb tárgyalás érdekében is, a *klaszterezés* elnevezést megengedően használjuk, és olyan csoportokként tekintünk a klaszterekre, amelyeket egy meghatározott minimalizálási vagy maximalizálási probléma megoldásaként alakítottunk ki. Ezzel összhangban pedig a dolgozat során az egyenlő méretű csoportok kialakításának problémáit összefoglalóan *egyenletes klaszterezési problémáknak* nevezzük.

A 2. Fejezetben részletes áttekintést adunk az alkalmazási lehetőségekről.

A 3. Fejezetben először illusztráljuk az m-szobatárs probléma nehézségét az esetek számának megadásával. Ezt követően bevezetjük a csoportosítási problémák elméleti nehézségének vizsgálatához szükséges eszköztárat, és áttekintjük a csoportosítási feladatok nehézségi eredményeit. Végül kiterjesztjük a kapcsolódó nehézségi eredményeket általános dimenzióban.

A 4. Fejezetben bemutatjuk a stabil szobatársak problémát, valamint tárgyaljuk ennek változatait és a kapcsolódó bonyolultságelméleti eredményeket. Ezután megadjuk az m-szobatárs probléma formális definícióját, valamint tárgyaljuk a modell előnyeit és hátrányait a stabil szobatársak probléma ellenében.

Az 5. Fejezetben olyan algoritmusokat mutatunk be a megengedett megoldások konstrukciójára, amelyek polinomiális futásidejűek és valamilyen konkrétumot adnak a megoldás szuboptimalitására. Kétféle módszert tekintünk, az approximációs eljárásokat és a

kúp optimalizálást.

A 6. Fejezetben bemutatjuk az irodalomban található legfontosabb heurisztikus eljárásokat, és új módszereket is megkonstruálunk. Ezeknek két csoportját tárgyaljuk. Az elsőbe a klaszterelemzéshez kapcsolódó eljárások tartoznak, amelyek alapvetően alacsony futásidővel rendelkeznek, ugyanakkor nem garantált, hogy a megoldásban a csoportok száma egyenlő lesz. Hogy az m-szobatárs problémára ezek alkalmazhatóak legyenek, megfogalmazunk 6 különböző heurisztikus eljárást a klaszterek kiegyenlítésére, amelyeket a 7. Fejezetben tesztelünk. A második csoportban olyan eljárásokat tárgyalunk, amelyek konstrukciójukból adódóan alkalmasak egyenlő elemszámú klaszterek előállítására. Ezek közül többnek is közös vonása, hogy párok cseréjével próbál meg javítani a megoldáson. A mi hozzájárulásunk az eljárások ezen csoportjához, hogy megvizsgáljuk a nagyobb méretű cserék lehetőségét, és megfogalmazunk három heurisztikus eljárást, amelyek kettes és hármas cserék segítségével keresik az optimumot. Ezeket szintén bevonjuk a 7. Fejezetben elvégzett elemzésbe.

A 7. Fejezet során végrehajtjuk a probléma egy több lépésből álló vizsgálatát egy széles elemzési keretben. Az irodalomban rendszerint az optimalizálási problémáknak csak az egyik, vagy a minimalizálási vagy a maximalizálási verzióját tekintik, és erre értékelik ki a heurisztikák egy szűkebb halmazát. A mi esetünkben egyszerre vizsgáljuk a feladat mindkét megfogalmazását, és az eljárások széles körét bevonjuk az elemzésbe.

A 8. Fejezetben összegezzük a dolgozat eredményeit, és további kutatási irányokat jelölünk meg.

2. fejezet

A felhasznált módszerek

2.1. Az m-dimenziós párosítási problémák definiálása és bizonyítás redukció által

A disszertáció 3. Fejezetében kiterjesztjük az egyenlő méretű csoportok létrehozására vonatkozó általános dimenziós nehézségi eredményeket. A nehézségi eredmények kiterjesztéséhez az egyenletes MSSC problémából indulunk ki. Huygen tétele[1] alapján a klasztereken belüli pontoknak a klaszter centroidjától vett négyzetes távolságainak összege megegyezik a klaszter pontjainak egymástól vett négyzetes távolságainak az adott klaszter elemszámának kétszeresével leosztott összegével. Ugyanez egyszerűbben, formálisan megadva a $C_1, \ldots, C_k$ klaszterek egységes m elemszáma esetén

$$\sum_{s=1}^{k} \sum_{x_i \in C_s} \left\| x_i - \left(\frac{1}{m} \sum\nolimits_{x_j \in C_s} x_j \right) \right\|^2 = \frac{1}{2m} \sum_{s=1}^{k} \sum_{x_i, x_j \in C_s} \|x_i - x_j\|^2.$$

Ezt felhasználva átírhatjuk az egyenletes MSSC problémát úgy, hogy annak célfüggvényében a pontok közötti euklideszi távolságok szerepeljenek. A probléma általánosításához ezt követően lecseréljük az euklideszi távolságot egy, az ℓ_p normán alapuló távolságmértékre, valamint a feladat minimalizálási verziója mellett a maximalizálási célt is tekintjük. Az így kapott feladatokat a 2.1.1. Optimalizálási problémával definiáljuk.

2.1.1. Optimalizálási probléma. pMIN-mDM (pMAX-mDM).

Input: *$C = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ pontok halmaza, valamint m és p egész számok, ahol $k = n/m$ egész.*

[1] Huygen tételét először Edwards és Cavalli-Sforza (1965) bizonyította (Novick, 2009).

Output: *A pontok azon $C_1, \ldots, C_k$ egyenlő méretű csoportokból álló partíciója, amely minimalizálja (maximalizálja) a*

$$\sum_{s=1}^{k} \sum_{x_i \in C_s} \sum_{x_j \in C_s} \|x_i - x_j\|_p^p$$

célfüggvényt.

Az optimalizálási probléma NP-nehézségének bizonyításához megmutatjuk, hogy az optimalizálási problémához tartozó eldöntési probléma NP-teljes (Ausiello és szerzőtársai, 2003). Ennek a bizonyítását egy másik NP-teljes problémából történő redukcióval tesszük meg. A bizonyítás lényege, hogy ha adott egy $\mathcal{A}$ NP-teljes eldöntési probléma, és ez alapján szeretnék bizonyítani egy $\mathcal{B}$ eldöntési probléma NP-teljességét, akkor először is be kell látnunk, hogy $\mathcal{B}$ NP-beli, másodsorban pedig meg kell mutatnunk, hogy $\mathcal{A}$ tetszőleges esete visszavezethető polinomiális időben a $\mathcal{B}$ feladatra, így ha meg tudnánk oldani $\mathcal{B}$-t, akkor meg tudnánk oldani $\mathcal{A}$-t is. Ugyanakkor mivel tudjuk, hogy $\mathcal{A}$ NP-teljes, ezért $\mathcal{B}$-nek legalább olyan nehéznek kell lennie, mint $\mathcal{A}$.

2.2. Új heurisztikus eljárások megkonstruálása

A disszertációban tárgyaljuk az m-szobatárs probléma gyakorlati megoldhatóságát. Ehhez a 7. fejezetben számos heurisztikus eljárást hasonlítunk össze különböző feladatok megoldására. Az elemzésben szereplő eljárások köréhez mi is hozzájárultunk néhány algoritmus megkonstruálásával, amelyeket az alábbiakban mutatunk be.

2.2.1. Kiegyenlítő eljárások `eq1-6`

Az összevonó hierarchikus klaszterezés alkalmas vágással (`cluster`, lásd Dr. Kovács és szerzőtársai (2011)), a k-közép++ algoritmus (`kmeans`, lásd Arthur és Vassilvitskii (2007)) és a fuzzy c-közép egyenlő klaszterméretekkel módszer (`eqFCM`, lásd Höppner és Klawonn (2008)) közül egyik sem ad garantáltan egyenlő elemszámú csoportokat, ezért megadunk hat heurisztikus eljárást a csoportok kiegyenlítésére. Ezek mindegyike a 0. lépésben ellenőrzi, hogy a klaszterek száma k-val egyenlő-e. Amennyiben ez nem teljesül, úgy a legnagyobb elemszámmal rendelkező klaszterből kiválasztjuk azt az elemet, amely a legtávolabb helyezkedik el a klaszter középpontjától, és ebből egy új, egyelemű csoportot hozunk létre. Ezt addig ismételjük, amíg már nem maradnak üres klaszterek. Ezt

követően az egyes eljárások működési elvei az alábbiak szerinti:

1. kiegyenlítő módszer (`eq1`)

   A legnagyobb elemszámú klasztertől kezdve, egymás után egyenlíti ki a csoportokat. Egy adott csoport esetén a többi klaszterközépponttól való távolság alapján sorol át annyi elemet a klaszterből, amennyi az ideális csoport elemszám eléréséhez szükséges. Minden lépésben azt az elemet helyezzük át, amelyik a legközelebb van valamely másik klaszter középpontjához. Az átsorolásokat követően 'letiltjuk' (másképpen megjelöljük) az adott csoportot, hogy ne kerülhessenek vissza az elemek, és átlépünk a következő legnagyobb elemszámú klaszterre. Bármely kiegyenlítés során legfeljebb $k(k-1)(m-1)/2$ lépésre van szükség. Az eljárás pszeudokódját a 2.1. Algoritmus írja le.[2]

Az 1. kiegyenlítő eljárás (`eq1`)

Input $C = \{C_1, \ldots, C_k\}$ nemüres klaszterek $c = \{c_1, \ldots, c_k\}$ klaszterközéppontokkal, ahol $c_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j$ a C_i klaszter középpontja, valamint a klaszterek egyenlő m mérete

(1) $i \leftarrow \arg\max_i \{|C_i| : C_i \in C\}$ [a legnagyobb klaszter indexe]

(2) **while** $(|C_i| > m)$

$j, l \leftarrow \arg\min_{j,l} \{d(x_j, c_l) : x_j \in C_i, c_l \in c, l \neq i\}$

$C_i \leftarrow C_i \backslash x_j,\ C_l \leftarrow C_l \cup \{x_j\},\ c_l$ frissítése

end

(3) $C \backslash C_i,\ c \backslash c_i,\ i \leftarrow \arg\max_i \{|C_i| : C_i \in C\}$ [halmazok és index frissítése]

(4) **if** $(|C_i| > m)$

goto Step (2)

end

2.1. Algoritmus. Az 1. kiegyenlítő eljárás (`eq1`)

[2] A további kiegyenlítő eljárások ennek kisebb módosításaival következnek, így azok pszeudokódjait nem közöljük.

2. kiegyenlítő módszer (`eq2`)

   A legnagyobb elemszámú klasztertől kezdve 'lecsorgatja' a többletet, amíg a csoportok ki nem egyenlítődnek. Egy lépésben átsorol egy adott klaszterből egy elemet valamely másik csoportba a klaszterközéppontoktól való távolság alapján. Mindig azt az elemet helyezzük át, amelyik a legközelebb van valamelyik másik csoport középpontjához. Az átsorolás után letiltja azt a klasztert, amelyből áthelyezésre került az elem, majd átlép a következő legnagyobb elemszámú csoportra. Ezt iteráljuk, amíg van olyan még nem letiltott csoport, amelynek elemszáma nagyobb az ideálisnál, egyébként pedig feloldjuk a tiltásokat, és újraindítjuk az algoritmust. A maximálisan szükséges lépések száma ebben az esetben is $k(k-1)(m-1)/2$.

3. kiegyenlítő módszer (`eq3`)

   A legkisebb elemszámú klasztertől kezdve, egymás után egyenlíti ki a csoportokat az 1. kiegyenlítő módszerhez hasonló módon. Egy adott csoportba annyi elemet sorolunk át, amennyi az ideális csoport elemszám eléréséhez szükséges. Az átsorolás a hiánnyal rendelkező csoport klaszterközéppontja és a többi, legalább két elemmel rendelkező csoportban lévő elemek távolsága alapján történik. Mindig azt az elemet helyezzük át, amelyik a legközelebb van az éppen kiegyenlíteni kívánt klaszterhez. Az algoritmus legfeljebb $(k-1)(m-1)$ áthelyezés után biztosan terminál.

4. kiegyenlítő módszer (`eq4`)

   A legkisebb elemszámú klaszter felől indulva, és afelé 'csorgatja' a többletet, amíg a csoportok ki nem egyenlítődnek. Egy lépésben az éppen kiegyenlíteni kívánt klaszter középpontjának és a többi, legalább két elemmel rendelkező csoport elemeinek a távolságát tekinti. Mindig azt az elemet sorolja át, amelyik a legközelebb van az aktuálisan tekintett klaszterhez. Az algoritmus legfeljebb $(k-1)\left((m-2)k/2+1\right)$ áthelyezés után biztosan terminál.

5. kiegyenlítő módszer (`eq5`)

   Az eljárás az 1. és a 3. kiegyenlítő módszereket ötvözi. Miután végrehajtotta a legnagyobb elemszámú csoport kiegyenlítését, letiltja azt, majd a legkisebb klaszter kiegyenlítésével folytatja, amelyet szintén letilt, ha azzal végzett. Ezt a két fázist iterálja egymás után, amíg vannak az ideálistól eltérő méretű klaszterek.

6. kiegyenlítő módszer (`eq6`)

   Egyszerre csökkenti a többletet a legnagyobb elemszámú klaszter felől és tölt fel egy

hiányt a legkisebb csoportoknál. Egy lépésben átsorol egy elemet az elérhető (nem letiltott) legnagyobb elemszámú klaszterből, letiltja azt, majd ha még lehetséges, átrak egy elemet a legkisebb csoportba, és ezt a klasztert is letiltja. A két fázist addig iteráljuk, amíg az összes csoport ki nem egyenlítődik.

2.2.2. Fuzzy c-közép egyenlő elemszámú klaszterekkel (`eqFCMv2`)

A fuzzy c-közép egyenlő klaszterméretekkel eljárás (c itt a klaszterek számát jelöli, lásd Höppner és Klawonn (2008)) eredményeképpen egy olyan $U = [u_{ij}] \subset \mathbb{R}^{c \times n}$ hozzátartozási mátrixot kapunk eredményül, amely minden $j, 1 \leq j \leq n$ pontra egyértelműen meghatározza, hogy mekkora mértékben tartozik az $i, 1 \leq i \leq c$ klaszterekhez. Ez alapján megkonstruálhatunk egy olyan csoportosítást is, amelyben minden klaszterbe azonos számú elemet sorolunk.

Az eljárás során az u_{ij} hozzátartozási mértékek szerint haladunk csökkenő sorrendben. Egy adott u_{ij} változó esetén, ha a j elemet még nem osztottuk be sehová sem, és az i csoportban van még szabad hely, akkor a j elemet az i csoporthoz rendeljük. Ellenkező esetben ugrunk a következő u_{ij} értékre, míg az összes elemet be nem osztottuk.

2.2.3. Az LCW algoritmus hármas cserékkel (`LCWv2`, `LCWv3` és `LCWv4`)

Az LCW algoritmus (lásd Weitz és Lakshminarayanan (1996)) úgy próbál megengedett megoldást találni az optimalizálási problémára, hogy egy véletlenszerű beosztásból kiindulva párok cseréjével lépésenként javít a célfüggvényértéken. Ha már nem lehet párok cseréjével javítani a célfüggvényértéken, akkor az algoritmus terminál. Ugyanakkor az optimalizálási feladat szempontjából az LCW módszer megoldása nem feltétlenül optimális. Ugyanis előfordulhat, hogy habár egy pár cseréjével már nem lehet javítani a célfüggvényértéken, egy hármas vagy nagyobb méretű cserével - ahol az elemek különböző klaszterekből származnak, és egyik sem marad a helyén - még lenne rá lehetőség. A disszertációban megvizsgáljuk az LCW módszer kiterjesztésének lehetőségét hármas, vagy akár nagyobb elemszámú cserékkel is.

Az LCW eljárás egy lépésben azt vizsgálja, hogy mennyivel javulna a célfüggvény értéke, hogyha egy csoport egy adott elemét felcserélnénk egy másik csoport valamely elemével. Ez egy lépésben $(k-1)m$ darab eset kiértékelését jelenti. Ha az LCW algoritmus metódusát alkalmaznánk, és s cserére jelölt elemet tekintenénk, akkor minden egyes javító

lépésnél összesen

$$\binom{k-1}{s-1} m^{s-1} D_s$$

esetet kellene figyelembe vennünk, ahol $D_s = sD_{s-1} + (-1)^s, D_0 = 1$ a fixpont nélküli s-edfokú permutációk (vagyis olyan cserék, amelyeknél egyik elem sem marad helyben; $D_5 = 44, D_6 = 265, D_7 = 1854$; lásd Király és Tóth (2011)) számát megadó rekurzív sorozat. A lehetséges esetek száma ekkor már kis k és m értékeknél is vállalhatatlanná válik.

Így a fentiek miatt az algoritmus továbbgondolásánál a kettes cserék mellett csupán a hármas cseréket vesszük figyelembe. Ezt háromféleképpen tesszük meg, így a hármas cserékre három különböző heurisztikát konstruálunk. Az első változat (`LCWv2`) során, amelynek pszeudokódját a 2.2. Algoritmus írja le, miután megpróbáltunk kettes cserékkel javítani a szobák össztávolságának értékén, ugyenezt megtesszük hármas cserékre szorítkozva is.

LCW Algoritmus második verzió - hármas cserék (`LCWv2`)

---

(1) Egy tetszőleges $X = [x_{ip}]$ kezdőmegoldás megkonstruálása, ahol $x_{ip} = 1$, ha $i = (p-1) * S + 1, (p-1) * S + 2, \ldots, (p-1) * S + S$ és $p = 1, 2, \ldots, G$, valamint 0 minden más esetben.

(2) $R \leftarrow DX$, ahol $D = [d_{ij}]$ az elemek közötti távolságok mátrixa.
Flag $\leftarrow$ false [ezzel jelöljük, ha csere történik a későbbiek során]
$i \leftarrow 0$

(3) *Megnézzük, hogy lehet-e javítani úgy, hogy az i elemet kicseréljük egy másik csoportban lévő elemmel.*

$i \leftarrow i + 1$
if $(i \leq N)$
$t \leftarrow$ az a csoport, amelyben az i elem van, vagyis amelyre $x_{it} = 1$
$k \leftarrow \arg\max_{j \in J} w_j$, ahol $w_j = (r_{iq} - r_{it}) + (r_{jt} - r_{jq}) - 2d_{ij}$ és
$J = \{j | x_{jq} = 1, 1 \leq q \leq G, q \neq t\}$
if $(w_k > 0)$
$x_{kq} \leftarrow 0, x_{kt} \leftarrow 1, x_{iq} \leftarrow 1, x_{it} \leftarrow 0$ [kicseréljük a két elemet]

$R \leftarrow DX$ [frissítjük az R mátrixot]

Flag $\leftarrow$ true [jelöljük, hogy csere történt]

end if

goto Step (3)

end if

(4) *Ha történt csere a legutóbbi iteráció során, akkor újra végigmegyünk az összes elemen.*

if (Flag == true)

Flag $\leftarrow$ false, $i \leftarrow 0$

goto Step (3)

end if

(5) *Megnézzük, hogy lehet-e javítani hármas cserével úgy, hogy az i elemet és két másik, különböző csoportokban lévő elemet felcserélünk.*

$i \leftarrow i + 1$

if $(i \leq N)$

$t \leftarrow$ az a csoport, amelyben az i elem van, vagyis amelyre $x_{it} = 1$

$(k_1, k_2) \leftarrow \underset{(j_1,j_2)\in J\cdot J}{\arg\max}\, W(j_1, j_2)$, ahol

$W(j_1, j_2) = r_{iq_2} + r_{j_1 r} + r_{j_2 q_1} - (r_{it} + r_{j_1 q_1} + r_{j_2 q_2} + d_{ij_1} + d_{ij_2} + d_{j_1 j_2})$

és

$$J \cdot J = \{(j_1, j_2) \mid x_{j_1 q_1} = 1, 1 \leq q_1 \leq G, q_1 \neq t, \\ x_{j_2 q_2} = 1, 1 \leq q_2 \leq G, q_2 \neq t, q_1 \neq q_2\}$$

if $(W(k_1, k_2) > 0)$

[kicseréljük a három elemet]

$x_{k_1 q_1} \leftarrow 0, x_{k_1 t} \leftarrow 1, x_{k_2 q_2} \leftarrow 0, x_{k_2 q_1} \leftarrow 1, x_{iq_2} \leftarrow 1, x_{it} \leftarrow 0$

$R \leftarrow DX$ [frissítjük az R mátrixot]

Flag $\leftarrow$ true [jelöljük, hogy csere történt]

end if

goto Step (5)

end if

(6) *Megnézzük, hogy történt-e csere a legutóbbi iteráció során, és ha nem, akkor az algoritmus terminál.*

if (Flag == false)

    stop [nem lehet tovább javítani cserékkel]

else

    Flag ← false, $i \leftarrow 0$

    goto Step (5)

end if

2.2. Algoritmus. LCW Algoritmus második verzió - hármas cserék (`LCWv2`)

Az LCW algoritmus hármas cserékkel kiegészített második változatában (`LCWv3`) első lépésben kettes cserékkel próbálunk meg javítani, majd ha ezzel már nem tudunk, akkor a hármas cserék következnek. Ezt a két fázist egészen addig végezzük el egymás után ismételten, amíg már sem kettes, sem hármas cserékkel nem tudtunk javítani.

Végül a harmadik változatban (`LCWv4`) minden egyes javítási lépésnél egyszerre tekintjük a lehetséges kettes, illetve hármas cseréket. Ezek közül azt a lehetőséget választjuk, amellyel a legtöbbet javítunk. Ha kettes és hármas cserével is ugyanakkora mértékű javulást érnénk el, akkor a kettes cserét preferáljuk. Az algoritmus akkor terminál, ha semelyik elem esetén sem lehet már kettes vagy hármas cserével javítani.

2.3. Szimulációk a heurisztikus módszerek összehasonlítására

A disszertáció 7. fejezetében a gyakorlati megoldhatóság vizsgálata részben szimulációkkal történik, amelyek során hallgatói mintákat generálunk, a vizsgálatban résztvevő algoritmusokat futtatjuk a mintákon, és végül kiértékeljük, hogy melyek végeztek a legjobban. Az elemzésben részt vevő algoritmusokat három csoportba soroljuk, és a módszereket a 2.3. Táblázatban összegezzük.

A kísérletek során a hallgatói minták generálásánál általánosan a tulajdonságok száma 3, és minden tulajdonság a $[0, 10]$ intervallumból vehet fel egész értéket. Az elemzés elvégzéséhez szükséges programokat MATLAB-ban implementáltuk, az algoritmusok futtatását pedig egy Intel Core i5-8600K 3.60GHz processzorral és 16,0 GB RAM-mal rendelkező számítógépen hajtottuk végre.

Konstruktív módszerek

`cluster`	Összevonó hierarchikus klaszterezés alkalmas vágással, amely távolságmértékként négyzetes euklideszi távolságokat, összevonó eljárásként pedig Ward módszert alkalmaz (részletekért lásd Dr. Kovács és szerzőtársai (2011))
`eq1-6`	Heurisztikus eljárások, amelyek az elemek és a klaszterközéppontok távolságai alapján, lépésenként egy elem áthelyezésével egyenlítik ki a csoportokat

Középpontok számítását és hozzárendelést alternáló eljárások

`kmeans`	A k-közép++ módszer, amely a klaszterközéppontok számítását és a pontoknak a hozzájuk legközelebb lévő klaszterhez történő hozzárendelését alternálja (Arthur és Vassilvitskii, 2007)
`eqFCM`	A fuzzy c-közép egyenlő klaszterméretekkel módszer, amely prototípusok és hozzátartozási mértékek számítását alternálja (Höppner és Klawonn, 2008)
`eqFCMv2`	Az `eqFCM` módszer futása után elvégezzük a csoportok kiegyenlítését úgy, hogy a hozzátartozási mértékek szerint csökkenő sorrendben haladva rendeljük hozzá az elemeket klaszterekhez
`MalinenFranti`	A `kmeans` eljáráson alapuló alternáló módszer, ahol az elemek hozzárendelése a klaszterekhez a Magyar módszer segítségével történik (Malinen és Fränti, 2014)
`JittaKlami`	Valószínűségszámításon alapuló módszer, amely az elemek allokációinak és a klaszterek paramétereinek becsléseit alternálja (Jitta és Klami, 2018)

Lokális keresést alkalmazó heurisztikák

`LCW`	Az eljárás párok cseréjével próbál meg javítani a célfüggvényértéken (Weitz és Lakshminarayanan, 1996)

`LCWv2-4`	A módszerek párok és hármas cserék segítségével próbálnak javítani a célfüggvényértéken különböző konstrukciók szerint
`TLCW`	Az `LCW` módszer mohó konstrukcióval és tabu memóriával, ahol egy lokális megoldás megtalálása után végrehajtjuk a legjobb nem javító cserét, és újraindítjuk a keresést (Gallego és szerzőtársai, 2013)
`SO`	A `TLCW` módszer stratégiai ingázással ötvözve, amely a keresés során nem megengedett megoldásokat is érint (Gallego és szerzőtársai, 2013)
`Costaetal`	A LIMA-VNS módszer alkalmazása, amely növekvő szomszédságokból egy pontot véletlen cserék útján kiválasztva indítja újra az `LCW` eljárást (Costa és szerzőtársai, 2017)

2.3. Táblázat. Az elemzésben szereplő heurisztikus módszerek összefoglaló táblázata.

Amikor ötvözzük a kiegyenlítő eljárások valamelyikét egy nem egyenlő csoportokat eredményező heurisztikus módszerrel, akkor azt a kiegyenlítő eljárás nevének utótagként való alkalmazásával tesszük meg, például `cluster_eq3`. Mivel a kiegyenlítő eljárások önmagukban is alkalmazhatóak egyenlő elemszámú klaszterek létrehozására, ezért ezeket így is bevesszük az összehasonlításba.

Az eljárások összevetése során egyes módszerek minimalizálási és maximalizálási verzióját is szerepeltetjük, ha az átírás magától értetődő. Így például az `LCW` algoritmus esetében tekintünk `LCWmin` és `LCWmax` algoritmusokat is rendre a minimalizálási, valamint a maximalizálási problémákra. Hasonlóan a hármas cseréket megvalósító eljárásokra, továbbá a `TLCW`, az `SO` és `Costaetal` algoritmusokra is. A hármas cseréket is alkalmazó módszerekre azt is megnézzük, hogy más heurisztikák eredményeit, mint kezdőmegoldást alkalmazva, tudunk-e javítani az eredeti célfüggvényértéken.

2.3.1. A kiegyenlítő eljárások vizsgálata

Az `eq1-eq6` kiegyenlítő módszerek összehasonlítása Monte-Carlo (MC) szimulációval történik. Az összehasonlítást minden esetben valamelyik rögzített alap klaszterező eljárás mellett tesszük meg, amely az összevonó hierarchikus klaszterezés alkalmas vágással

(`cluster`), a k-közép++ (`kmeans`), vagy a fuzzy c-közép (`eqFCM`) heurisztikák valamelyike.

Egy szcenárió esetén véletlenül generált hallgatókat tekintünk, alkalmazzuk az alap klaszterező eljárást, majd az így kapott csoportosításra futtatjuk valamennyi kiegyenlítő módszert (`eq1-eq6`), amelyekről meghatározzuk, hogy egymáshoz képest mennyire teljesítettek jól. Ezután a módszerek összehasonlítása különböző mutatók mentén történik. Egy MC szimuláció során 10000 mintát generálunk, így a mutatók az egyes szcenáriók aggregált eredményeit tükrözik. A mutatók stabilitásának vizsgálatához ezt a folyamatot 100-szor megismételjük, és meghatározzuk a 100 futás alapján a mutatók különböző statisztikáit is: a mediánt, a minimum és maximum értékeket, valamint a 25%-os, illetve 75%-os percentiliseket.

Ahhoz, hogy egy szcenárió esetén meghatározzuk, hogy az egyes kiegyenlítő eljárások relatíve mennyivel teljesítenek jobban vetélytársaiknál, a következő értékelést alkalmazzuk. Legyen $c_1, c_2, \ldots, c_6$ rendre az `eq1`, `eq2`, ..., `eq6` kiegyenlítő eljárás által kapott költség, és jelölje

$$c_{terj} = \max\{c_1, \ldots, c_6\} - \min\{c_1, \ldots, c_6\}$$

a költségek terjedelmét. Ekkor a

$$p_i = \begin{cases} \sum_{\substack{1 \le j \le 6 \\ j \neq i}} \frac{c_j - c_i}{c_{terj}}, & \text{ha } c_{terj} > 0, \\ 0, & \text{egyébként} \end{cases}$$

relatív pontszám mutatja, hogy az i-edik kiegyenlítő eljárás relatíve mennyivel teljesít jobban a többinél. A terjedelemmel való skálázásra azért van szükség, mert a generált hallgatóktól függően a minimum és maximum költségek közötti távolság elég változékony lehet. A skálázás révén a legjobban és legrosszabbul teljesítő módszerek skálázott költségei közötti távolság mindig 1, a többi eljárás skálázott költségkülönbségei pedig ezzel arányosan adódnak.

Az algoritmusok értékelésre a következő mutatókat vezetjük be:

p0 : Azt mutatja, hogy az algoritmus hányszor adta vissza a legjobb eredményt (döntetlenek is számítanak), osztva az ismétlések számával.

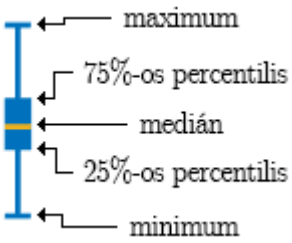
p1 : 6-tól 1-ig pontozzuk az eljárásokat, és ezek átlagát vesszük algoritmusonként. A legjobb eljárás 6 pontot kap, a legrosszabb pedig egyet. Döntetlenek esetén az algoritmusok egységesen a helyezésekért járó pontok átlagát kapják. Így ha a két legjobban teljesítő

algoritmus ugyanolyan jól szerepelt, akkor mindkettő 5,5 pontot kap. Ha az élen hármas döntetlen alakult ki, akkor mindhárom eljárás 5 pontot kap. A pontszámok egy lehetséges alakulása rendre az `eq1`, ..., `eq6` módszerekre: (legjobb) [5,5; 5,5; 4; 2; 2; 2] (legrosszabb).

p2 : Átlagosan mennyire teljesít jól az algoritmus a többi módszerhez képest, vagyis a p_i relatív pontszámok átlaga.

T : Átlagos futásidők.

Az algoritmusok összehasonlításához használt ábrákon a 100 futtatás eredményéből számított statisztikák megjelenítésére gyertyaszerű ábrákat alkalmazunk. Ezek a 2.1. ábrán jelölt módon mutatják a maximum (felső vízszintes kék vonal), a 75%-os percentilis (a középső széles terület felső éle), a medián (a középső vízszintes sárga vonal), a 25%-os percentilis (a középső széles terület alsó éle) és a minimum (az alsó vízszintes kék vonal) értékét.



2.1. ábra. A kiegyenlítő eljárások statisztikai mutatóinak megjelenítésére használt gyertyaszerű ábrák magyarázata.

2.3.2. Optimalitási pontszám a nagyobb hallgatói elemszámokra

A lokális kereséseket megvalósító eljárásokat $n = 600$ fős hallgatói minták esetében is vizsgáljuk különböző m szobaméretek mellett. Ekkor az $m = 2$, vagyis kétfős szobák esetét leszámítva a tényleges optimumok keresése az összes lehetséges szobabeosztás költségének végignézésével már biztosan nem megvalósítható. Ezért az eljárásokat egymáshoz képest értékeljük ki az *optimalitási pontszám* felhasználásával.

2.3.1. Definíció. *Jelölje M a hallgatói minták számát. Jelölje továbbá x_i^{max} és x_i^{min} a heurisztikák által megtalált legnagyobb, illetve legkisebb célfüggvényértékeket az i-edik minta esetén. Tegyük fel, hogy $x_i^{max} > x_i^{min}$ $\forall i$. Legyen végül x_i^{alg} egy tetszőleges 'alg' algoritmus által megtalált célfüggvényérték az i-edik minta esetén. Ekkor az 'alg' algoritmus p^{alg}*

optimalitási pontszáma

$$p^{alg} = \frac{1}{M} \sum_{i=1}^{M} \frac{x_i^{alg} - x_i^{\min}}{x_i^{\max} - x_i^{\min}}.$$

Az optimalitási pontszám lényege, hogy különböző csoportszámok (k) és szobaméretek (m) mellett is azonos skálán tudjuk összehasonlítani az eredményeket, mivel $p^{alg} \in [0, 1]$ mindig teljesül.

2.4. A lehetséges szobabeosztások megkonstruálása kis k és m esetén

Az m-szobatárs probléma kisméretű eseteire meg tudjuk határozni a 2Min-mDM és 2Max-mDM problémák optimumát, így ezt használhatjuk az algoritmusok eredményeinek értékeléséhez. Az optimumokat az összes lehetséges szobabeosztás célfüggvény szerinti értékének meghatározásával, és ezen értékeken a minimum és maximum keresésével adjuk meg.

Az összes lehetséges szobabeosztás felírására számokat rendelünk a hallgatókhoz 1-től n-ig. Ezután egy rekurzív eljárás segítségével lépésenként alakítjuk ki a szobákat. A rekurzív algoritmus pszeudokódjának megadásához az alábbi definíciókat és jelöléseket vezetjük be.

2.4.1. Definíció. *Jelölje $k > 1$ a szobák számát, $m > 1$ a szobák (azonos) méretét és $n = k \cdot m$ a hallgatók számát. Legyen $S = \{1, \ldots, n\}$ az összes hallgató halmaza. Legyen továbbá $R \subset S, |R| = m$ egy beosztott szoba és $\mathcal{R} = \{R | R \subset S, |R| = m\}$ egy szoba lehetséges beosztásainak halmaza. Legyen megadott k és m értékek mellett az r-szobabeosztások ($0 \leq r \leq k$) halmaza*

$$\begin{aligned}\mathcal{P}_r = \Big\{ & (R_1, \ldots, R_r, s_1, \ldots, s_l) \mid \\ & 0 \leq r \leq k;\ \emptyset \neq R_i \in \mathcal{R}\ \forall i;\ R_i \cap R_j = \emptyset\ \forall i, j, i \neq j; \\ & 0 \leq l \leq n;\ l = n - r \cdot m;\ s_i \in S\ \forall i;\ s_i \notin R_j\ \forall i, j; \\ & s_i \neq s_j\ \forall i, j, i \neq j;\ (\cup_{i=1}^{r} R_i) \cup \left(\cup_{i=1}^{l} \{s_i\}\right) = S \Big\}.\end{aligned}$$

Ez olyan részleges szobabeosztások halmaza, ahol azon szobák száma, amelyekhez már hozzárendeltünk m hallgatót, r. Ekkor $\mathcal{P}_0$ egy egyelemű halmaz, amelynek eleme annak felel

meg, amikor még egy hallgatót sem osztottunk be. $\mathcal{P}_k$ pedig a lehetséges (teljes) szobabeosztások halmaza, amelynek elemei azok a beosztások, ahol minden hallgatót hozzárendeltük valamelyik szobához.

2.4.2. Definíció. *Jelölje $k > 1$ a szobák számát, m a szobák (azonos) méretét és így $n = k \cdot m$ a hallgatók számát. Legyen $D \in \mathbb{R}^{n \times n}$ a hallgatók közötti euklideszi távolságok négyzeteinek mátrixa. Legyen $0 \leq r \leq k$ a már beosztott szobák száma, és ennek megfelelően legyen $P = (R_1, \ldots, R_r, s_1, \ldots, s_l) \in \mathcal{P}_r$ egy r-beosztás. Legyen $d : \mathcal{P}_r \to \mathbb{R}_0^+$ egy beosztás-költségfüggvény, amely megadja egy r-beosztás esetén a már beosztott szobákban lévő hallgatók szobán belüli távolságainak összegét:*

$$d(P) = \sum_{m=1}^{r} \sum_{\substack{i,j \in R_m, \\ i<j}} D_{ij}.$$

Legyen továbbá $\mathcal{D} = \{(P, d(P)) \,|P \in \mathcal{P}_r, 0 \leq r \leq k\}$ az összes lehetséges beosztás-költség pár halmaza, és jelölje $2^{\mathcal{D}}$ a $\mathcal{D}$ részhalmazainak halmazát. Végül pedig legyen $h : \mathcal{P}_r \times \mathbb{R}_0^+ \to 2^{\mathcal{D}}$ egy olyan szoba(halmaz)-hozzárendelés, ahol a $P \in \mathcal{P}_r$ r-beosztásra

$$h\left(P, d(P)\right) = \Big\{ \left(P', d(P')\right) \,\Big|\, \left(P', d(P')\right) \in \mathcal{D}, \; P' = \left(R_1, \ldots, R_r, R_{r+1}, s'_1, \ldots, s'_{l-m}\right) \in \mathcal{P}_{r+1}, \; \min\{s_1, \ldots, s_l\} \in R_{r+1} \Big\}.$$

Az összes lehetséges szobabeosztás, azok költségeinek, valamint a minimum- és maximumértékek meghatározásához használt rekurzív eljárás pszeudokódját a 2.4. Algoritmus írja le.

Rekurzív eljárás a lehetséges szobabeosztások meghatározására

---

(1) *Változók inicializálása.*

$C = \emptyset$ [globális változó: az összes költség halmaza]

$Cmax = NaN, Cmin = NaN$ [globális változók: C szélsőértékei]

$r = 0$ [beosztott szobák száma]

(2) *A* **kroomcases**$(r, P, d(P))$ *rekurzív függvény futtatása.*

if $(r < k)$

$H \leftarrow h\big(P, d(P)\big)$ [$r+1$-szobabeosztások generálása]

for $\big(P', d(P')\big)$ in H

kroomcases$(r+1, P', d(P'))$

end for

else [költséghalmaz és szélsőértékek frissítése]

$C \leftarrow C \cup d(P)$

if (isnan$(Cmax)$) **or** $(Cmax < d(P))$

$Cmax \leftarrow d(P)$

end if

if (isnan$(Cmin)$) **or** $(Cmin > d(P))$

$Cmin \leftarrow d(P)$

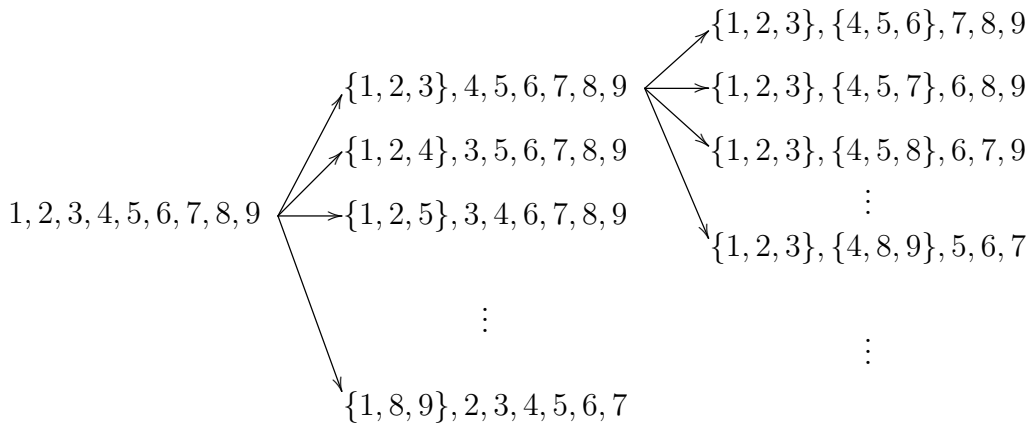
end if

end if

2.4. Algoritmus. Rekurzív eljárás a lehetséges szobabeosztások költségeinek és a költségek minimum és maximum értékeinek meghatározásához

Az algoritmus működését $n = 9$ és $m = 3$ esetén a 2.2. ábra szemlélteti.

2.4.3. Állítás. *A 2.4. Algoritmus az összes lehetséges szobabeosztást megkonstruálja, és mindegyiket pontosan egyszer.*



2.2. ábra. A `kroomcases` eljárás rekurzív lépéseinek ábrázolása az összes lehetséges szobabeosztás megkonstruálására $n = 9$ és $m = 3$ esetén.

3. fejezet

Az értekezés eredményei

3.1. Az m-dimenziós párosítási problémák NP-nehézségi eredményei

A következőekben kiterjesztjük az egyenlő méretű csoportok létrehozására vonatkozó általános dimenziós nehézségi eredményeket. Megmutatjuk expliciten, hogy ha a csoportok m mérete legalább 3, akkor az egyenletes MSSC feladat különböző távolságfelírások mellett, a célfüggvény minimalizálására és maximalizálására is NP-nehéz. Ezzel egyúttal alternatív bizonyítást adunk a Feo és Khellaf (1990) által tárgyalt mDM problémához tartozó optimalizálási probléma NP-nehézségére is. Az optimalizálási feladatra az alábbiakban *maximális súlyú m-dimenziós párosítás* (MAX-mDM) problémaként hivatkozunk, és $m \geq 2$ egész érték mellett tekintjük. Továbbá belátjuk, hogy az előző minimalizálási verziója, a *minimális súlyú m-dimenziós párosítás* (MIN-mDM) szintén NP-nehéz, amennyiben $m \geq 3$. Utóbbi jelentősége abban rejlik, hogy az az egyenletes MSSC probléma általánosításaként is tekinthető, amelyben a pontok közötti távolságok tetszőlegesek lehetnek. A MAX-mDM és MIN-mDM problémákra összefoglalóan *m-dimenziós párosítási problémákként* hivatkozunk. Elméleti eredményeinket Kondor (2022a) műhelytanulmányban közöltük, az m-dimenziós párosítási problémákra vonatkozó összefoglaló táblázatokat pedig Kondor (2022b) tanulmányban szerepeltetjük.

Vegyük észre, hogy az egyenletes MSSC probléma ekvivalens a 2MIN-mDM problémával. A következőekben belátjuk a pMIN-mDM és pMAX-mDM problémák NP-nehézségét $m \geq 3$ esetén különböző p értékek mellett.

3.1.1. Tétel. (Kondor (2022a)) *A* pMIN-mDM *optimalizálási probléma NP-nehéz bármely rögzített* $m \geq 3$ *és* $p \geq 1$ *egészekre.*

Bizonyítás. A tétel bizonyításához belátjuk, hogy a pMIN-mDM problémához tartozó 3.1.2. Eldöntési probléma NP-teljes.

3.1.2. Eldöntési probléma. A pMIN-mDM problémához tartozó eldöntési probléma.
Input: *$C = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ pontok halmaza, valamint m és p egész számok, ahol $k = n/m$ egész, és $W \in \mathbb{Q}^+$.*
Kérdés: *Van-e a pontok C halmazának olyan $C_1, \ldots, C_k$ diszjunkt, m csúcsot tartalmazó halmazokból álló partíciója, amelyre*

$$\sum_{s=1}^{k} \sum_{x_i \in C_s} \sum_{x_j \in C_s} \|x_i - x_j\|_p^p \leq W$$

teljesül?

A probléma NP-beli. A bizonyítás hátralévő részében Pyatkin és szerzőtársai (2017) lépéseit követjük. A redukciót az m-dimenziós párosítás élsúlyozatlan gráfon (mDM-$\{0,1\}$) eldöntési problémából végezzük el, amelyet a 3.1.3. Eldöntési probléma ír le. Erről Feo és Khellaf (1990) belátja, hogy NP-teljes az mDM probléma NP-nehézségi bizonyításának egy lépéseként.

3.1.3. Eldöntési probléma. m-dimenziós párosítás élsúlyozatlan gráfon (mDM-$\{0,1\}$).
Input: *Egy $G = (V, E)$ gráf, ahol $|V| = km, m \geq 3$, k, l pozitív egészek.*
Kérdés: *Van-e a csúcsok V halmazának olyan $V_1, V_2, \ldots, V_k$ diszjunkt, m csúcsot tartalmazó halmazokból álló partíciója, amelyre azon élek száma, amelyeknek mindkét végpontja ugyanabban a V_i halmazban található, nagyobb mint l?*

A bizonyításhoz megmutatjuk, hogy az mDM-$\{0,1\}$ probléma tetszőleges esetét meg tudnánk oldani (polinomiális időben), hogyha a pMIN-mDM problémához tartozó eldöntési problémát a redukció során kapott paraméterekkel meg tudnánk oldalni (polinomiális időben).

Tekintsük az mDM-$\{0,1\}$ probléma egy tetszőleges esetét $|V| = km$ darab csúccsal és $|E| = q$ darab éllel. Rögzítsük a p pozitív egész értékét, és legyen $d = q$, valamint $W = 4q(m-1) - 4(l+1)$. Jelölje $C \in \{0,1\}^{km \times d}$ a G gráf incidenciamátrixát, vagyis $x_{it} = 1$, ha a $v_i \in V$ csúcs az e_t él egyik végpontja, és $x_{it} = 0$ egyébként. Ekkor a

C soraira tekinthetünk $\mathbb{R}^d$-beli pontokként. Ezen pontok $n/k = m$ méretű klaszterekbe történő rendezései pontosan megfeleltethetőek a gráf csúcsainak m méretű részhalmazokra való particionálásainak.

Legyen $A_{st} = \sum_{i:x_i \in C_s} \sum_{j:x_j \in C_s} |x_{it} - x_{jt}|^p$ a C_s részhalmaz t koordináta szerinti hozzájárulása a célfüggvényhez. A pMIN-mDM célfüggvényét ekkor felírhatjuk mint

$$\sum_{t=1}^{d} \sum_{s=1}^{k} A_{st}.$$

Ha két olyan x_i és x_j pontot tekintünk, amelyek t-edik koordinátája 1, akkor a következő két eset egyike teljesül rájuk: a pontok 1) ugyanabba a C_{s_1} részhalmazba tartoznak, vagy 2) különböző, C_{s_1} és C_{s_2} csoportokba lettek particionálva. Jelöle $A_t = \sum_{s=1}^{k} A_{st}$ a t-edik koordináta hozzájárulását a célfüggvényhez, és jelölje $A_t^{(1)}$, valamint $A_t^{(2)}$ ennek értékeit a két különböző esetben. Ekkor a hozzájárulás az első esetben

$$A_t^{(1)} = A_{s_1 t} = 2|1-1|^p + 4(m-2)|1-0|^p = 4(m-2),$$

míg a második esetben

$$A_t^{(2)} = A_{s_1 t} + A_{s_2 t} = 4(m-1)|1|^p = 4(m-1).$$

Végül jelölje b azon élek számát, amelyeknek mindkét végpontja ugyanabban a részhalmazban van. Ekkor egy egyenletes partíció célfüggvényértéke kifejezhető a b függvényeként, vagyis

$$A(b) = (q-b)A_t^{(2)} + bA_t^{(1)} = 4q(m-1) - 4b. \tag{3.1}$$

$A(b)$ csökkenő b-ben, amelyből következik, hogy $A(b) \leq W$ akkor és csak akkor, hogyha $b \geq l+1$. □

Az előző párjaként a 3.1.4. Tétel a pMAX-mDM NP-nehézségét mondja ki.

3.1.4. Tétel. (Kondor (2022a)) *A pMAX-mDM optimalizálási probléma NP-nehéz bármely rögzített $m \geq 3$ és $p \geq 2$ egészekre.*

Bizonyítás. A bizonyítás során a 3.1.1. Tétel bizonyításának lépéseit követjük a következő változtatásokkal. A pMAX-mDM optimalizálási problémához tartozó eldöntési problémában a célfüggvény értékének nagyobb, vagy egyenlőnek kell lennie, mint W.

A p értékét úgy kell megválasztani, hogy a $p \geq 2$ egyenlőtlenség is teljesüljön rá, és a célfüggvény célértéke legyen $W = 4q(m-1) + (l+1)(2^{p+1} - 4)$. Amikor a G gráf

incidenciamátrixát tekintjük, akkor ebben az esetben egy módosított mátrixot adunk meg. A C minden egyes oszlopában az egyik egyes értéket cseréljük le -1-re, és hagyjuk a másik értéket változatlanul. Az előzőekből következően

$$A_t^{(1)} = 2^{p+1} + 4(m-2),$$
$$A_t^{(2)} = 4(m-1), \text{ és}$$
$$A(b) = 4q(m-1) + b(2^{p+1} - 4).$$

Mivel $p \geq 2$, $A(b)$ növekvő b-ben, így $A(b) \geq W$ akkor és csak akkor, hogyha $b \geq l+1$. □

Mivel az egyenletes MSSC probléma ekvivalens a 2MIN-mDM problémával, ezért a 3.1.1. Tételből egyből következik az egyenletes MSSC probléma NP-nehézsége is általános m-re.

3.1.5. Következmény. (Kondor (2022a)) Az egyenletes MSSC probléma bármely rögzített $m \geq 3$ egész esetén NP-nehéz.

A 3.1.1. Tételből szintén következik, hogy az m-dimenziós párosítási probléma minimalizálási verziója is NP-nehéz, hiszen az tartalmazza a 2MIN-mDM problémát.

3.1.6. Következmény. (Kondor (2022a)) A minimális súlyú m-dimenziós párosítás optimalizálási probléma (MIN-mDM) NP-nehéz bármely rögzített $m \geq 3$ esetén.

Az m-dimenziós párosításokra vonatkozó nehézségi eredményeket a 3.1. és 3.2. táblázatok összegzik. A kapcsolódó eredményeket zárójelben hivatkozzuk, a 3.2. táblázatban a következő számozott rövidítéseket alkalmazzuk: (1) Bertoni és szerzőtársai (2012), (2) Lin és szerzőtársai (2016), (3) Kel'manov és Pyatkin (2016), (4) Pyatkin és szerzőtársai (2017), (5) Kondor (2022a). A kérdőjelekkel jelölt eredmények ismereteink szerint nyitott problémák.

	Maximális súlyú m-dimenziós párosítás (MAX-mDM)		
	általános	pMAX-mDM	
	eset	$p=1$	$p \geq 2$
$m=2$	polinomiális (Edmonds, 1965)		
$m \geq 3$	NP-nehéz (Feo és Khellaf, 1990)	?	NP-nehéz (Kondor, 2022a)

3.1. Algoritmus. Nehézségi eredmények a maximális súlyú m-dimenziós párosítás problémára és annak speciális eseteire. Az m a csoportok méretét, a p pedig az ℓ_p norma paraméterét jelöli. Forrás: Kondor (2022b).

		Minimális súlyú m-dimenziós párosítás (MIN-mDM)			
		általános	pMIN-mDM		
		eset	$p=1$	$p=2$	$p\geq 3$
$m=2$		polinomiális (Edmonds, 1965)			
$m=3$		NP-nehéz (5)	NP-nehéz (5)	NP-nehéz (4)	NP-nehéz (5)
$3<m<n/2$		NP-nehéz (5)			
$m=n/2$	$d=1$	?	?	polinomiális (1)	?
	$d=2$	?	?	polinomiális (2)	?
	d általános	NP-nehéz (5)	NP-nehéz (5)	NP-nehéz (3)	NP-nehéz (5)

3.2. Algoritmus. Nehézségi eredmények a minimális súlyú m-dimenziós párosítás problémára és annak speciális eseteire. Az m a csoportok méretét, a d az euklideszi tér dimenzióját, a p pedig az ℓ_p norma paraméterét jelöli. Forrás: Kondor (2022b).

3.2. Az m-szobatárs probléma, valamint előnyei és hátrányai a stabil szobatársak problémához képest

Az m-szobatárs probléma formális definíciója a következő. Legyen n a felvételt nyert hallgatók száma, és legyen m egységesen a szobánkénti férőhelyek száma. Az egyszerűség kedvéért legyen n az m egész többszöröse, a szobák száma pedig ennél fogva legyen $k = n/m$. A szobákat jelölje $C_1, \ldots, C_k$. Tegyük fel továbbá, hogy azt, hogy két ember mennyire lenne megfelelő szobatárs, bizonyos tulajdonságok alapján döntjük el a következő módon. Legyen d a *tulajdonságok* száma, és tegyük fel, hogy a tulajdonságok tetszőleges valós értékek lehetnek.[1] Ennek következtében egy hallgató megfeleltethető egy $\mathbb{R}^d$-beli koordinátának, és az összes hallgató elhelyezhető egy d-dimenziós euklideszi térben. Jelölje $x_1, \ldots, x_n \in \mathbb{R}^d$ a hallgatókat és legyen $D \in \mathbb{R}^{d\times d}$ a hallgatók kölcsönös távolságainak mátrixa. Jelölje továbbá $x_i \in C_s$, ha az i-edik hallgatót az s-edik szobába osztottuk be.

Feltesszük, hogy azt, hogy az x_i és x_j hallgatók mennyire lennének jó szobatársak egymás számára a közöttük lévő euklideszi távolság, vagyis a $d_{ij} = \|x_i - x_j\|$ érték reprezentálja. A *jó* szobabeosztásra két különböző megközelítést tekintünk. Homogén csoportok

[1] Az elemzés során az egyszerűség kedvéért azt tesszük fel, hogy a tulajdonságoknak megfelelő értékek a $[0, 10]$ intervallumból vehetnek fel egész értékeket.

kialakítására egy minimalizálási problémát adunk meg:

$$\min \sum_{s=1}^{k} \sum_{x_i, x_j \in C_s} d_{ij}^2, \qquad \text{(2Min-}m\text{DM)}$$
$$\text{s.t. } |C_s| = m \ \forall s.$$

A heterogén csoportosítás megadásához a maximalizálási problémát a

$$\max \sum_{s=1}^{k} \sum_{x_i, x_j \in C_s} d_{ij}^2, \qquad \text{(2Max-}m\text{DM)}$$
$$\text{s.t. } |C_s| = m \ \forall s$$

alakban írjuk fel. A megegyező célfüggvények garantálják, hogy egy egységes modell keretein belül vizsgálhassuk az m-szobatárs probléma minimalizálási és maximalizálási verzióit.

Elméleti szempontból az egyik legfontosabb különbség a stabil szobatárs és az m-szobatárs problémák között, hogy amíg előbbi stabil megoldást ad eredményül, addig utóbbi egy Pareto-hatékony megoldási keretet határoz meg. Az utóbbi által adott megoldás tehát nem feltétlenül stabil, viszont bizonyos esetekben jobb választás lehet a gyakorlati probléma leírására (lásd Morrill (2010)).

A stabilitási koncepció esetében megmutattuk, hogy a kétfős csoportokra előfordulhat, hogy a megoldások halmaza az üres halmaz, továbbá gyenge preferenciák esetében annak eldöntése, hogy létezik-e gyengén stabil párosítás, NP-teljes feladat. A megközelítés magasabb dimenziós eredményeinél pedig láthattuk, hogy egy kivétellel az összes felírás csupán a háromfős csoportokra vonatkozik, és szintén egy kivétellel az összes megfogalmazás NP-teljes feladatra vezet.

Ezzel szemben az m-szobatárs problémában, optimalizálási problémáról lévén szó, mindig van megoldás. Emellett párok, vagyis $m = 2$ esetén Edmonds (1965) algoritmusa révén mindig meg tudjuk adni a megoldást polinomiális futásidőben. Legalább háromfős csoportokra ugyanakkor már ennél a megközelítésnél is nehézségekkel találjuk szembe magunkat. A 3.1.1. és 3.1.4. Tételek alapján tudjuk, hogy a 2Min-mDM, valamint a 2Max-mDM problémák NP-nehezek. Ennek értelmében pedig, habár tudjuk, hogy bármely klaszterméretre létezik optimum, azt a nagyobb méretű problémák, vagyis a hallgatók nagyobb száma mellett képtelenek vagyunk meghatározni (Kondor, 2022b).

Gyakorlati szempontból az m-szobatárs szerinti felírásnak - egyúttal az egyéb metrikus

térbeli megközelítéseknek is - az az előnye a stabil szobatársak problémával szemben, hogy nincs szükség arra, hogy a hallgatók egyéni preferencia-sorrendeket adjanak meg. Ez olyan esetekben jelent reálisabb megközelítést a szobabeosztásokra, amikor vannak olyan hallgatók, akiknek nagy valószínűséggel nincs semmilyen preferencia-sorrendje a többiekre nézve. Ez a helyzet fordul elő a kollégiumi felvételek során, amikor sokaknak jellemzően nincs információja a többi hallgatóról, és így rangsort sem tudnak felállítani közöttük.

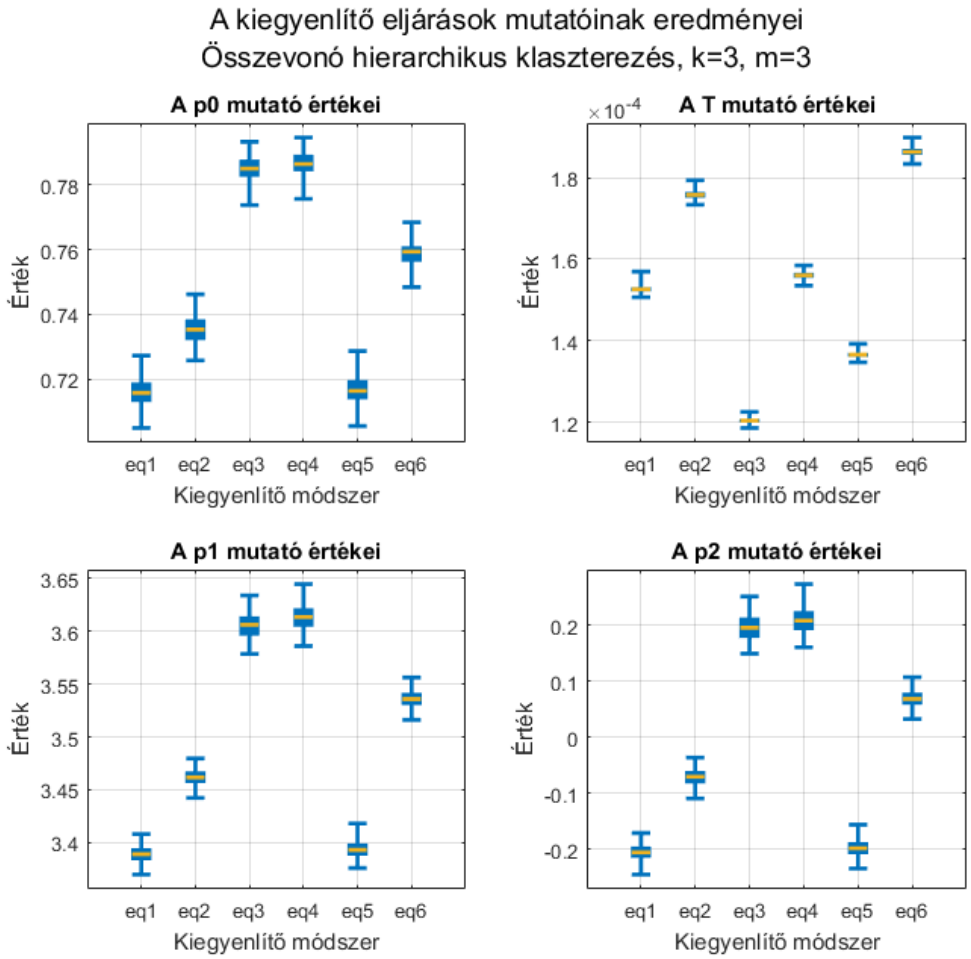
A modell további pozitívuma a preferencia-listák elhagyásában rejlik. Ugyanis, a preferencia-sorrendek esetén nincs mértékbeli különbség a rangsor elemei között. Vagyis ha egy x hallgató rangsorában y és z egyénekre vonatkozóan $y \succ_x z$ szerepel (vagyis x szempontjából y jobb, mint z), akkor nem tudjuk, hogy y mennyivel jobban preferált z-vel szemben. Ezzel ellentétben a távolságok többletinformációt adnak, így a gyakorlatban hozzájárulhatnak egy jobb szobabeosztás kialakításához.

Az m-szobatárs modell egyik negatívuma a stabil szobatársak problémával szemben, hogy míg utóbbiban az y és z hallgatóknak lehetett eltérő preferenciája a másikról, addig előbbiben a távolság révén a szerepük szimmetrikus.

A felírás másik hátránya, hogy hiányzik az egyéni preferenciák megadásának lehetősége. Elképzelhető ugyanis, hogy két vagy több hallgató, akik már ismerik egymást, mindenképpen ugyanabba a szobába szeretnének kerülni. Ahhoz, hogy ezt figyelembe tudjuk venni, egy általánosabb modell, például az m-dimenziós párosítás alkalmazása lenne a megoldás, amelyben a hallgatók között tetszőleges élsúlyokkal egyéni távolságokat is megadhatunk. Ebben az esetben a minimalizálási problémára akár olyan élsúlyokat is definiálhatunk, amely révén két hallgató biztosan különböző szobákba kerül, amelyre egyik előzőleg említett modell sem alkalmas. Az általános modell további tárgyalása ugyanakkor nem célja jelen dolgozatnak, de érdekes további kutatási irány lehet.

3.3. A kiegyenlítő eljárások összehasonlításának eredményei

A kiegyenlítő eljárások eredményeit $k = 3$ és $m = 3$ esetére a 3.1., 3.2. és 3.3. ábrák mutatják be.



3.1. ábra. A kiegyenlítő eljárások (`eq1-eq6`) mutatóinak eredményei. Összevonó hierarchikus klaszterezés, $k = 3, m = 3$.

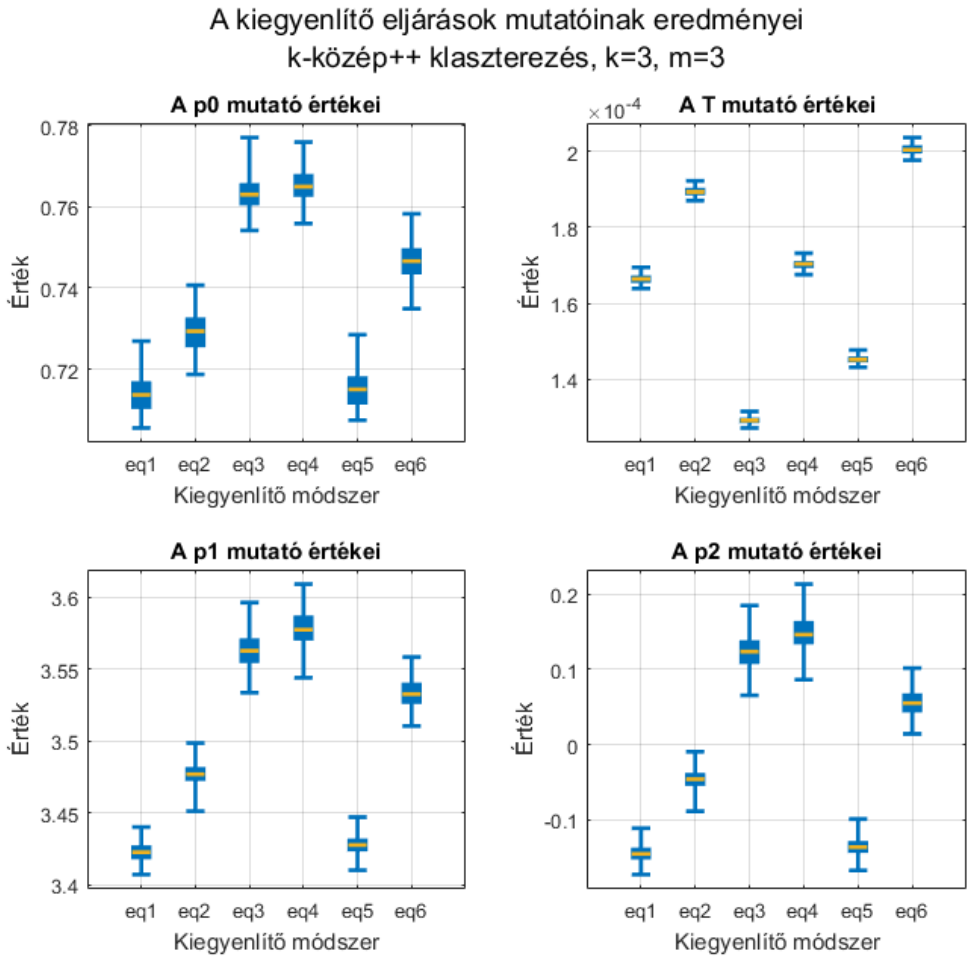
A futásidő (T mutató) szempontjából minden esetben az `eq3` heurisztika teljesített a legjobban. Ezt az `eq5` követi nem sokkal lemaradva, majd az `eq1` és az `eq4` következik. A legrosszabb futásidőket minden esetben az `eq2` és az `eq6` módszerek produkálták. A gyertyák magassága alapján a sorrend stabil.

Aszerint, hogy melyik algoritmus az esetek hány százalékában adta a legjobb eredményt (p0 mutató), valamennyi esetben az `eq3` és `eq4` módszerek teljesítettek a legjobban, és ezeket követi nagyobb lemaradással az `eq6` eljárás. Megjegyezzük, hogy az `eqFCM` eljárásra a gyertyák közötti átfedések miatt a sorrend esetenként eltérő lehet.

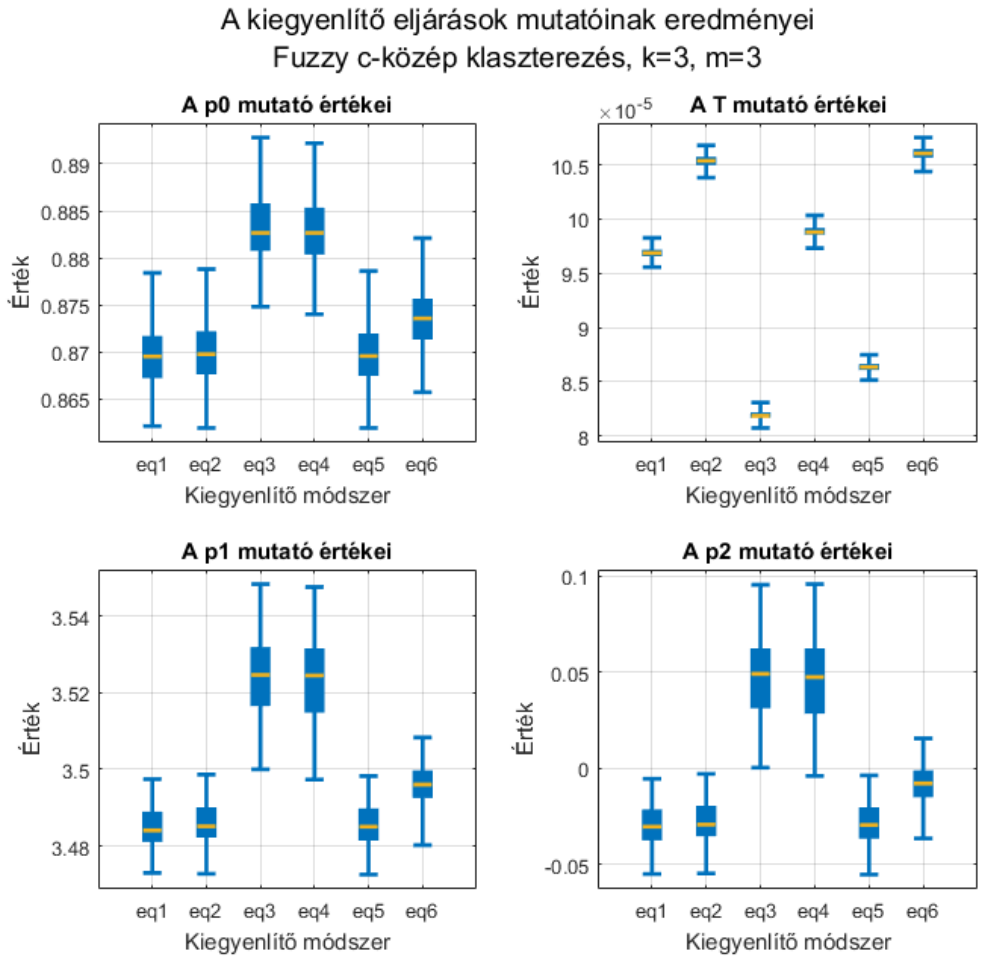
A p1, illetve p2 mutatók rendkívül nagy hasonlóságot mutatnak a p0 mutatóval, így ezek szerint is egyértelmű az értékelés. Minden esetben az `eq3` és `eq4` algoritmusok teljesítenek a legjobban.

Az elemzést $k = 5$ csoport, illetve $d = 10$ dimenzió esetében is elvégeztük, és több-

nyire hasonló eredményeket kaptunk. Az elemzés eredményeképpen összességében az `eq4` módszert értékeljük a legjobbnak, amely futásidejét tekintve közepesen, a többi mutató szerint viszont általánosságban nagyon jól teljesített. Emellett a további kísérletek során rendkívül kedvező futásideje miatt az `eq3` módszert is szerepeltetjük.



3.2. ábra. A kiegyenlítő eljárások (`eq1-eq6`) mutatóinak eredményei. k-közép++ klaszterezés, $k = 3, m = 3$.

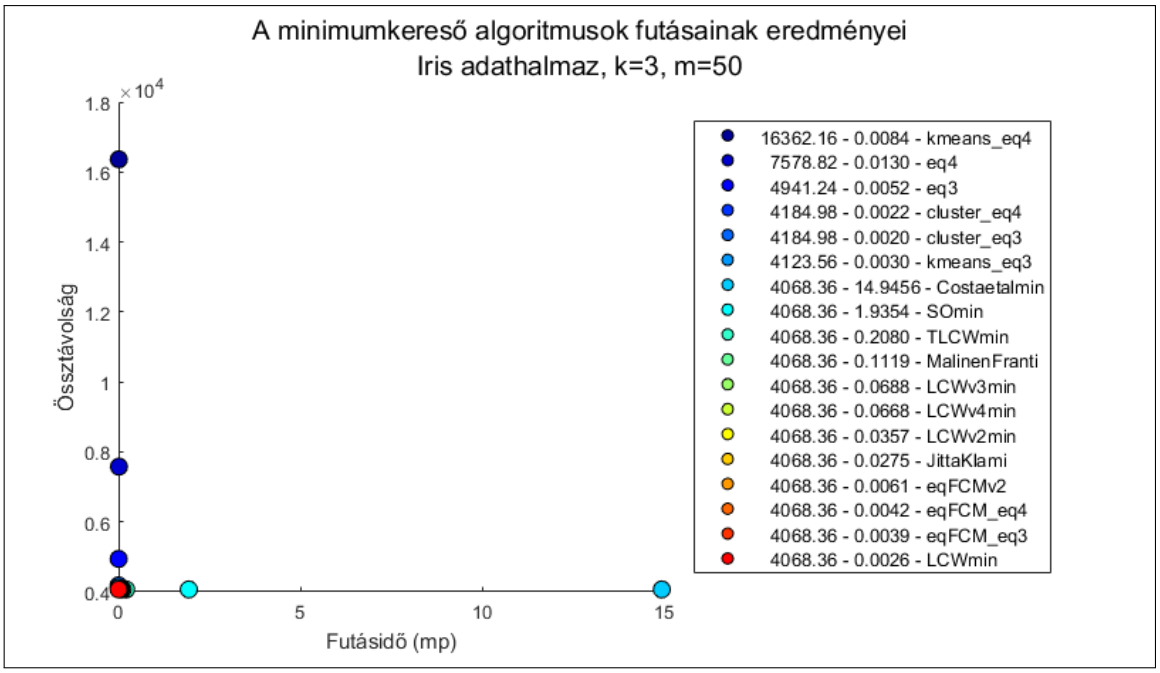


3.3. ábra. A kiegyenlítő eljárások (`eq1-eq6`) mutatóinak eredményei. Fuzzy c-közép klaszterezés, $k = 3, m = 3$.

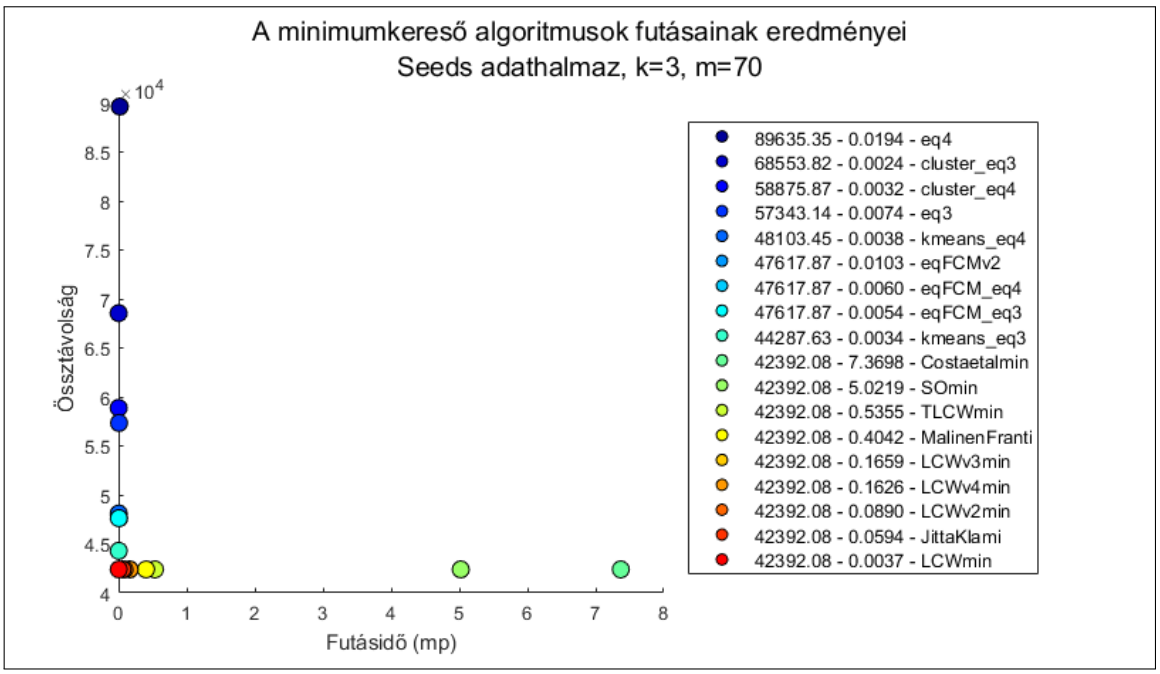
3.4. Kísérletek valós adatokon

A heurisztikus eljárásokat összehasonlítottuk az irodalomban is használt ‘Iris’ és ‘Seeds’ adathalmazokon (lásd Malinen és Fränti (2014); Costa és szerzőtársai (2017); Rujeerapaiboon és szerzőtársai (2019)). Ezek olyan valós adathalmazok, amelyek egyenlő elemszámú klasztereket tartalmaznak. A megjelölt forrásokban ezeket a minimalizálási problémák esetén alkalmazták, és ismert rájuk a minimális célfüggvényérték. Ebből kifolyólag a fejezet során mi is a minimumkereső algoritmusok összehasonlításával foglalkozunk.

Az elemzéshez az algoritmusokat futtatjuk az adatokon, mérjük a futásidejüket, majd kiszámítjuk a megtalált megoldások célfüggvényértékét. Az eredményeket az össztávolságok és célfüggvényértékek terében a 3.4. és 3.5. ábrák mutatják be, valamint ezeket az algoritmusok feliratain is feltüntetjük.



3.4. ábra. A minimumkereső algoritmusok futásainak eredményei az Iris adathalmaz esetében. $k = 3, m = 50$. A feliratok magyarázata: célfüggvényérték - futásidő - algoritmus neve.



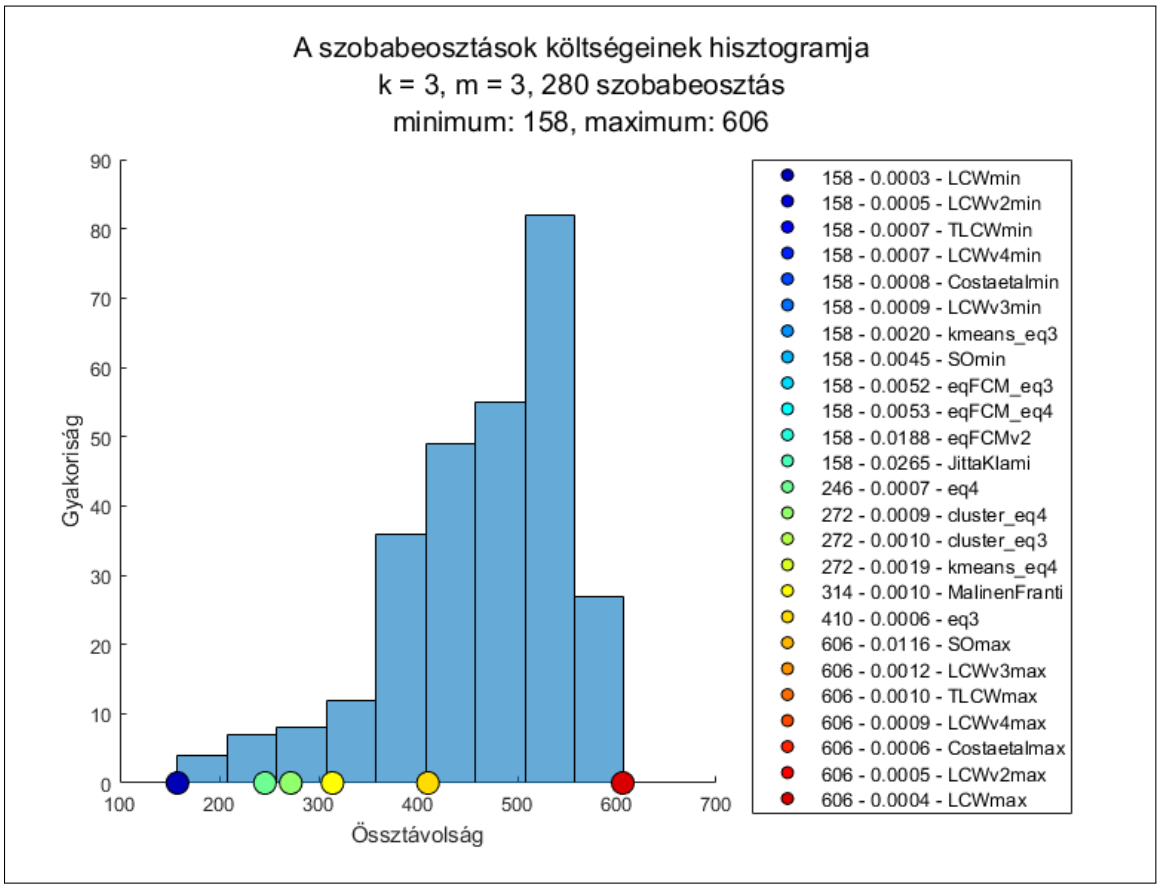
3.5. ábra. A minimumkereső algoritmusok futásainak eredményei a Seeds adathalmaz esetében. $k = 3, m = 70$. A feliratok magyarázata: célfüggvényérték - futásidő - algoritmus neve.

Az eredmények feliratai elsősorban a megtalált célfüggvényérték és másodsorban a futásidő szerint vannak sorbarendezve. Látható, hogy a heurisztikák többsége ugyanazt a

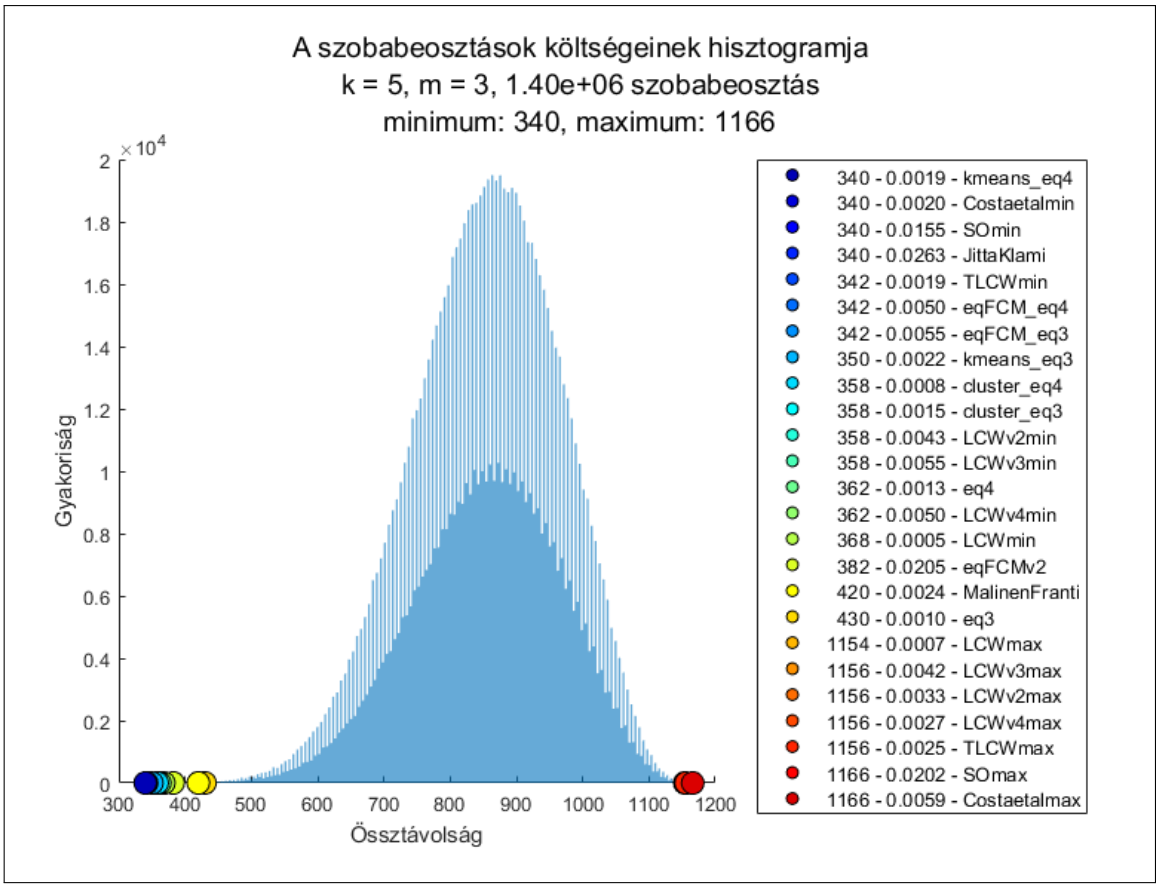
minimumértéket találja meg, amely egyben az irodalom alapján a tényleges minimumnak gondolt érték. A minimumértéket megtaláló eljárások között a futásidőkben elég nagy eltérések mutatkoznak, amely egyértelműen az algoritmusok konstrukciójából adódik. Amelyik módszer komplexebb módon próbál meg javítani a célfüggvényértéken, annak a futásideje óhatatlanul nagyobb lesz. Ezzel együtt viszont általános esetben várhatóan jobban teljesítenek a megadott megoldás tekintetében, mint az egyszerűbb társaik.

A hagyományos klaszterelemzéshez kapcsolódó módszerek, habár futásidő szempontjából jól teljesítenek, általánosan rosszabb - egyik esetben több, mint 4-szer rosszabb - célfüggvényértékeket találnak meg. Az alacsony futásidő kedvező lehet az adatpontok számának növekedésével, ugyanakkor a gyenge teljesítmény nem túl bíztató.

3.5. Eredmények kis k és kis m értékek mellett



3.6. ábra. A szobabeosztások költségeinek hisztogramja, valamint a minimum- és maximumkereső algoritmusok futásainak eredményei. Egy szimulált eset, $k = 3, m = 3$, 280 lehetséges szobabeosztás, a szobabeosztások generálásának és az optimumok keresésének futásideje 0,1541 mp. Ferdeség: -0,5339. A feliratok magyarázata: célfüggvényérték - futásidő - algoritmus neve.



3.7. ábra. A szobabeosztások költségeinek hisztogramja, valamint a minimum- és maximumkereső algoritmusok futásainak eredményei. Egy szimulált eset, $k = 5, m = 3$, 1,40e+06 lehetséges szobabeosztás, a szobabeosztások generálásának és az optimumok keresésének futásideje 62,06 mp. Ferdeség: -0,2814. A feliratok magyarázata: célfüggvényérték - futásidő - algoritmus neve.

Az elemzésnek ebben a lépésében kis k és m értékek esetén, szimulált adathalmazokon hasonlítjuk össze a különböző módszereket. Ekkor az összes lehetséges beosztást végignézve meghatározzuk az elérhető legnagyobb, illetve legkisebb költségeket, és ezekhez képest értékeljük ki a heurisztikákat. Az elemzésben a véletlenszerűen előállított adathalmazok dimenziója $d = 3$, vagyis a szimulált hallgatók 3 tulajdonsággal rendelkeznek.

A 3.6. ábra $k = 3$ és $m = 3$ választás mellett egy szimulált adathalmaz esetében ábrázolja a lehetségek szobabeosztások költségeinek hisztogramját. Emellett megmutatja a beosztások költségének minimumát és maximumát, valamint azt is, hogy mik az egyes heurisztikák által megtalált célfüggvényértékek és futásidők. Hasonlóan, a hallgatók nagyobb számát tekintve, $k = 5$ és $m = 3$ esetét mutatja (1,40e+06 lehetséges szobabeosztásra) a 3.7. ábra.

A 3.3. Táblázat egy 200 elemű mintán elvégzett szimuláció eredményeit mutatja be

(a minimalizálási esetben csupán a legjobban teljesítő eljárásokra). A táblázatban a 'Távolságok' oszlop az optimumtól való átlagos távolságot, a 'Futásidők' oszlop az átlagos futásidőt, míg a '# legjobb eredmény' oszlop azt mutatja meg, hogy az adott algoritmus hány esetben adta a legjobb eredményt. Az értékek alatt zárójelben a szórásokat tüntetjük fel. A táblázat sorai az utolsó oszlop szerint vannak rendezve.

A futásidők tekintetében észrevehetjük, hogy az egyes módszerek között nagyságrendbeli eltérések is megjelennek. Az `LCW` (minimumkereső és maximumkereső) algoritmusok kitűnnek azzal, hogy nagyon alacsony futásidő mellett is az esetek jelentős százalékában megtalálták az optimumot.

A maximumkereső eljárások között egyértelműen azok teljesítenek jobban, amelyek szofisztikáltabb javító módszert alkalmaznak a megoldás keresése során. Ezek az `SOmax` és `Costaetalmax` algoritmusok. Érdemes ugyanakkor azt is megfigyelni, hogy az `SOmax` eljárás futásideje egy nagyságrenddel nagyobb, mint az azt követő `Costaetalmax` módszeré. A hármas cseréket megvalósító algoritmusok tekintetében az `LCWv4max` teljesít a legjobban.

A minimumkereső eljárásoknál szintén a szofisztikált módszerek érték el a legjobb eredményeket. A konstruktív módszerek, a klaszterelemzéshez kapcsolódó heurisztikák, a `JittaKlami`, valamint a `MalinenFranti` algoritmusok a legjobban teljesítő módszerektől jóval lemaradtak (az esetek legfeljebb 30%-ában találták meg a legjobb megoldást), így nem kerültek felsorolásra. Továbbá a hármas cseréket megvalósító eljárások között az átlagos távolságok szempontjából az `LCWv3min` teljesít a legjobban, míg ha azt nézzük, hogy melyik módszer hányszor találta meg a legjobb megoldást, akkor bár csak egy hajszálnyival, de az `LCWv4min` módszer nyer. Futásidő szempontjából viszont a kettő közül egyértelműen az utóbbi a győztes.

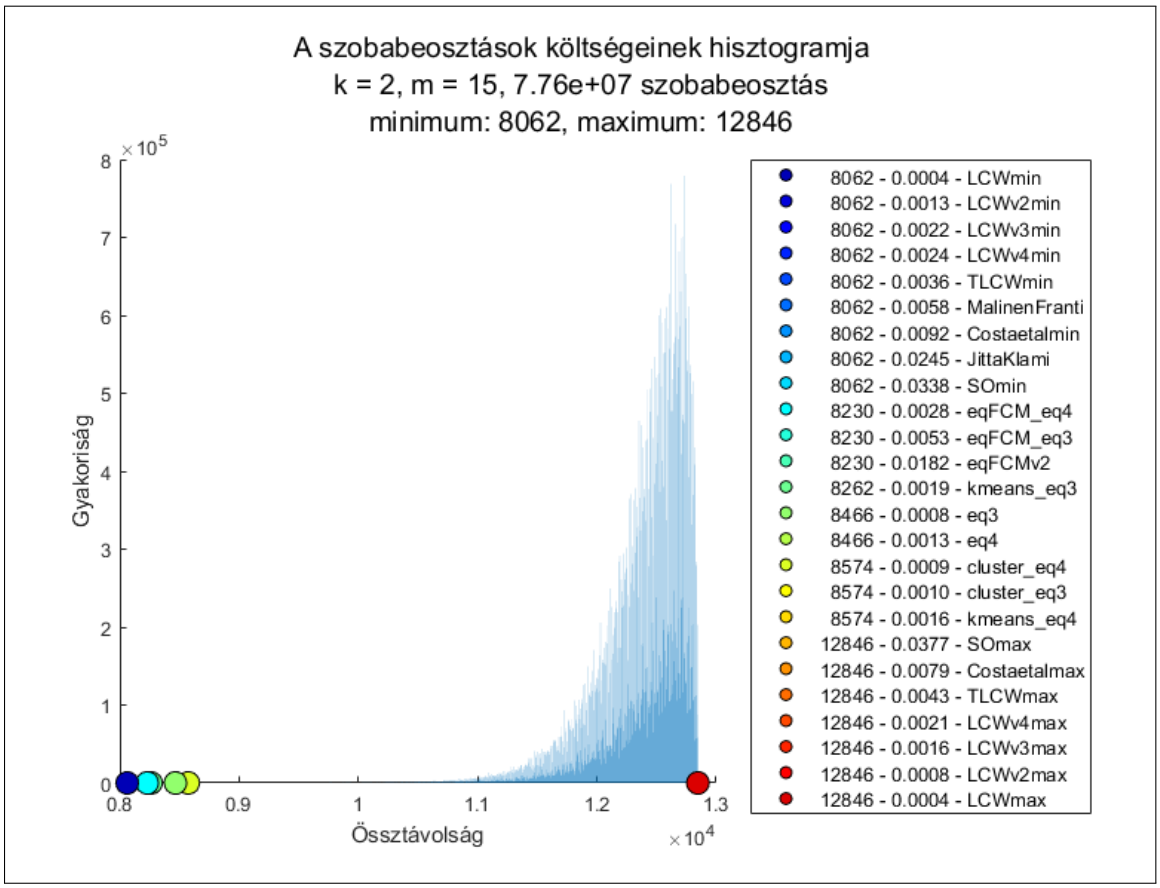
Hogy sokkal jobb rálátásunk legyen a jól teljesítő algoritmusok működésére, az elemzést az elemszámok növelése mellett a lokális kereső módszerek további vizsgálatával folytatjuk. Ugyanakkor a továbbiakban a hármas cserék közül csak egyet vizsgálunk, amely az `LCWv4` módszer.

	Távolságok	Futásidők	# legjobb eredmény
LCWmax	6,96	**0,0004**	78
	(9,03)	***(0,0001)***	
LCWv2max	5,64	0,0010	85
	(7,71)	*(0,0002)*	
LCWv3max	5,31	0,0017	90
	(7,65)	*(0,0004)*	
LCWv4max	4,91	0,0015	97
	(7,52)	*(0,0003)*	
TLCWmax	3,76	0,0014	109
	(5,73)	*(0,0002)*	
Costaetalmax	0,53	0,0023	180
	(1,85)	*(0,0006)*	
SOmax	**0,20**	0,0143	**191**
	(0,92)	*(0,0022)*	

	Távolságok	Futásidők	# legjobb eredmény
LCWmin	21,89	**0,0004**	106
	(32,92)	***(0,0001)***	
TLCWmin	15,54	0,0014	112
	(25,38)	*(0,0003)*	
LCWv2min	7,95	0,0020	141
	(16,88)	*(0,0006)*	
LCWv3min	4,62	0,0034	161
	(11,85)	*(0,0009)*	
LCWv4min	5,06	0,0029	162
	(13,98)	*(0,0005)*	
SOmin	4,47	0,0126	168
	(13,60)	*(0,0035)*	
Costaetalmin	**0,63**	0,0029	**190**
	(3,02)	*(0,0009)*	

3.3. Táblázat. A maximum- (felül) és minimumkereső (alul) algoritmusok futásainak eredményei: átlagos távolság az optimális megoldástól (és szórása), átlagos futásidő (és szórása), valamint azon esetek száma, amikor az adott algoritmus adta a legjobb eredményt. $k = 5, m = 3$, 200 szimulált eset.

3.5.1. A minimalizálási és maximalizálási feladatok aszimmetriája



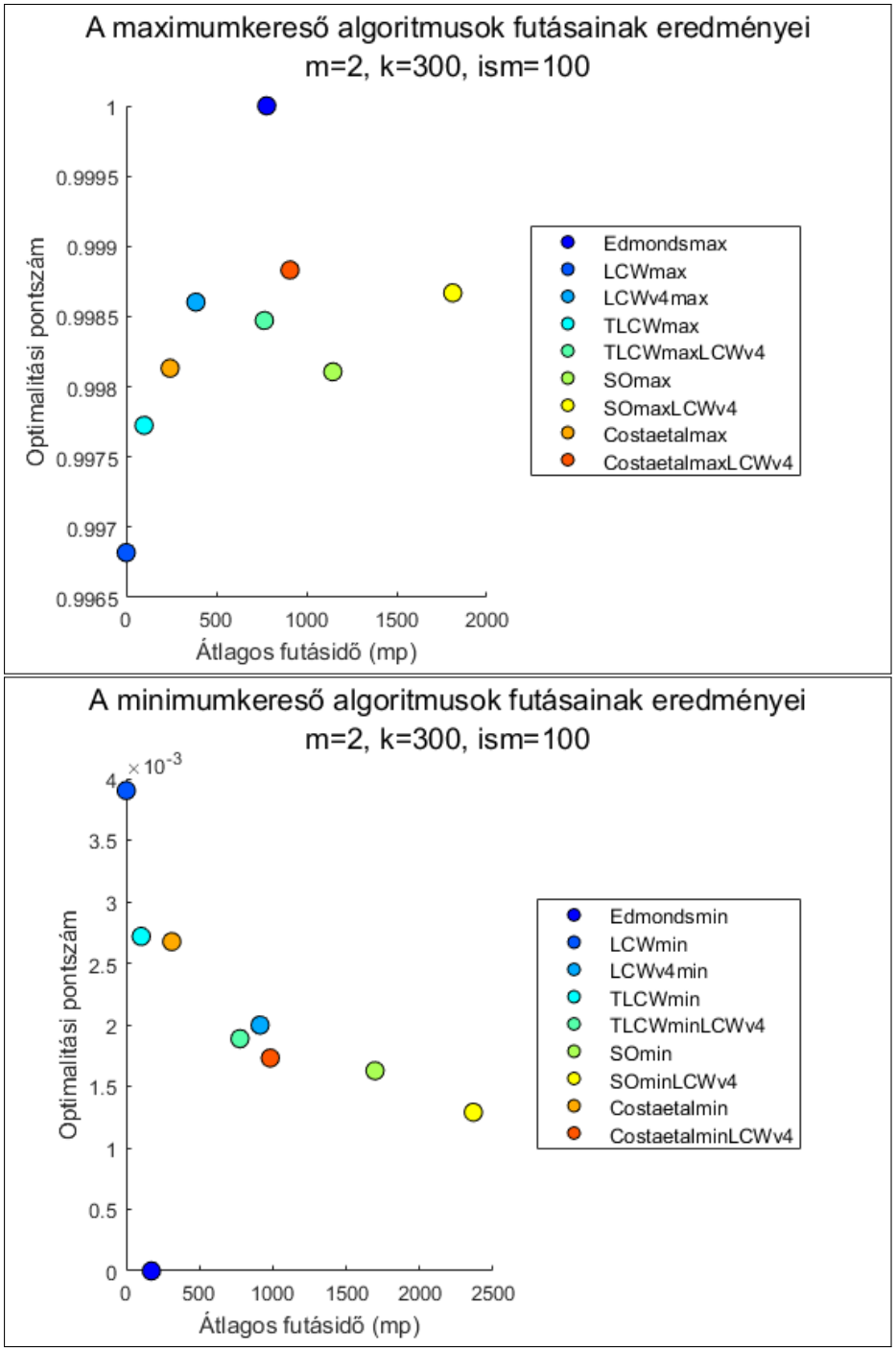
3.8. ábra. A szobabeosztások költségeinek hisztogramja, valamint a minimum- és maximumkereső algoritmusok futásainak eredményei. Egy szimulált eset, $k = 2, m = 15$, 7,76e+07 lehetséges szobabeosztás, a szobabeosztások generálásának és az optimumok keresésének futásideje 54 p 6 mp. Ferdeség: $-0{,}9044$. A feliratok magyarázata: célfüggvényérték - futásidő - algoritmus neve.

Felhívjuk a figyelmet egy érdekes jelenségre, amelynek egyértelmű szemléltetéséhez a 3.8. ábrát használjuk fel. Az ábra $k = 2, m = 15$ paraméterek esetén mutatja a lehetséges szobabeosztások költségeinek hisztogramját. Vegyük észre, hogy az eloszlás bal széle rendkívül vékony a jobb széléhez képest. Ebben az esetben azt mondjuk, hogy az eloszlás balra ferde, amelyet a kiszámított ferdeség érték negatív mivolta is megerősít: -0,9044. Ez a kereső eljárások szempontjából a minimalizálási és maximalizálási problémák lehetséges aszimmetriájára hívja fel a figyelmet.

Az eddig bemutatott, hisztogrammal ábrázolt valamennyi minta ferdesége negatív. Emellett több paraméterpárra ($k = 3, m = 3$; $k = 3, m = 4$; $k = 4, m = 3$; $k = 5, m = 3$) is egyenként 200 hallgatói mintát generálva, és az egyes minták esetén a szobabeosztások költségeit meghatározva a ferdeségi értékek kivétel nélkül mind negatívak voltak.

3.6. Eredmények nagyobb hallgatói elemszámok mellett

3.6.1. Edmonds algoritmusa mint viszonyítási alap



3.9. ábra. A maximum- és minimumkereső algoritmusok futásainak eredményei, Edmonds algoritmusát is beleértve. $m = 2, k = 300$, 100 szimulált eset.

Párok esetében ismert, hogy polinomiális időben meg tudjuk határozni az optimális megoldást. Emiatt ebben az esetben a heurisztikus eljárások eredményét össze tudjuk hason-

lítani a tényleges optimummal. A 3.9. ábra 100 generált hallgatói minta esetén mutatja az eredményeket $m = 2$ hallgatóból álló csoportok mellett, ahol a csoportok száma $k = 300$. Az eljárások között szerepeltetünk egy-egy Edmonds algoritmusára építő minimum-, illetve maximumkereső eljárást, amelyekre rendre `Edmondsmin` és `Edmondsmax` néven hivatkozunk[2]

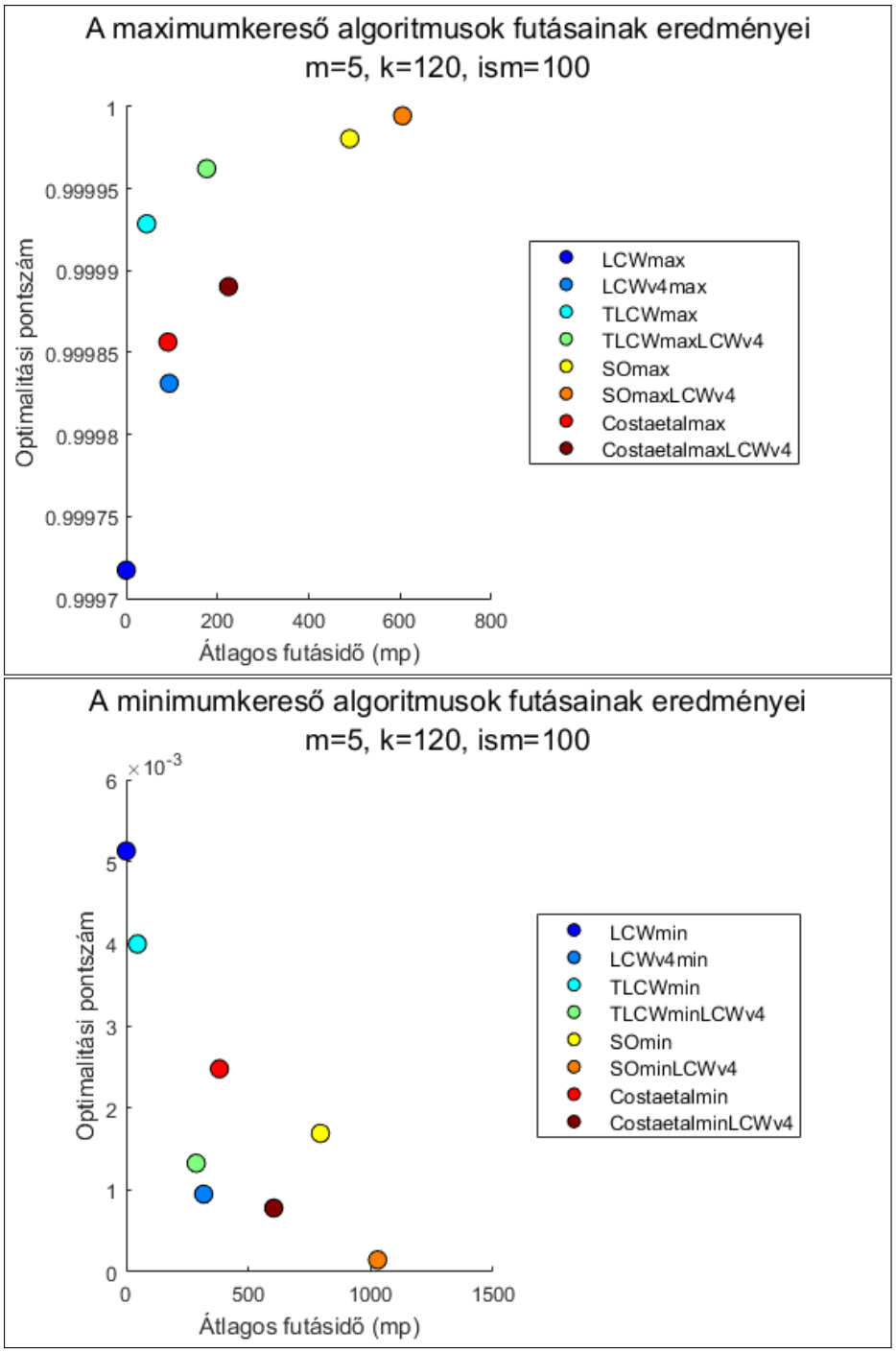
A 3.9. ábra alapján azt mondhatjuk, hogy relatíve közel tudunk kerülni az optimumhoz, akár még önmagában az `LCW` eljárással is. Emellett a szofisztikáltabb módszerek jelentősen tudnak javítani a célfüggvényértéken, ugyanakkor úgy tűnik, hogy még ezekkel sem tudjuk teljesen lefaragni az `LCW` megoldása és az optimum közötti különbséget. Megjegyezzük, hogy a maximumkereső eljárásoknál az `LCWv4` módszer önmagában is relevánsnak tűnik, a minimumkereső eljárásoknál ugyanakkor valamivel rosszabbul teljesített, mint a `TLCWminLCWv4` és `CostaetalminLCWv4` módszerek. Valamely másik heurisztika után alkalmazva az `LCWv4` a legtöbb esetben jelentősen tud javítani a célfüggvényértéken.

3.6.2. Eredmények legalább háromfős csoportokra

A 3.10. ábra 100 generált hallgatói minta esetén mutatja az eredményeket $m = 5$ hallgatóból álló csoportok mellett, ahol a csoportok száma $k = 120$. Optimalitási pontszám szempontjából az alábrákon egymáshoz közeli eredményeket látunk. Észrevehetjük, hogy az `LCWv4max` módszer önmagában redundáns, hiszen a `TLCWmax` módszer mind futásidő, mind optimalitási pontszám tekintetében jobban teljesít. A minimalizálási esetben az `LCWv4min` eljárásnak ugyanakkor már önmagában is van relevanciája, hiszen futásidő és optimalitási pontszám szerint is jobban teljesít, mint a `Costaetalmin` vagy az `SOmin` módszer. A minimalizálási és maximalizálási esetre is igaz, hogy más heurisztikák eredményeit kezdőértékként felhasználva az `LCWv4` módszer jelentősen tudott javítani azok célfüggvényértékén.

Végül vegyük észre, hogy a minimalizálási probléma esetén az algoritmusok futtatása jelentősen több időt igényel, mint a maximalizálási probléma során. A `Costaetal` módszerre egymintás t-próbával tesztelve a H0 hipotézist, miszerint a minimum- és maximumkeresési idők különbségeinek átlaga 0, bármilyen szignifikancia-szint mellett elvethetjük. A jelenség fennáll a többi eljárás esetében is, így ez is tükrözi a minimalizálási és maximalizálási feladatok aszimmetriáját.

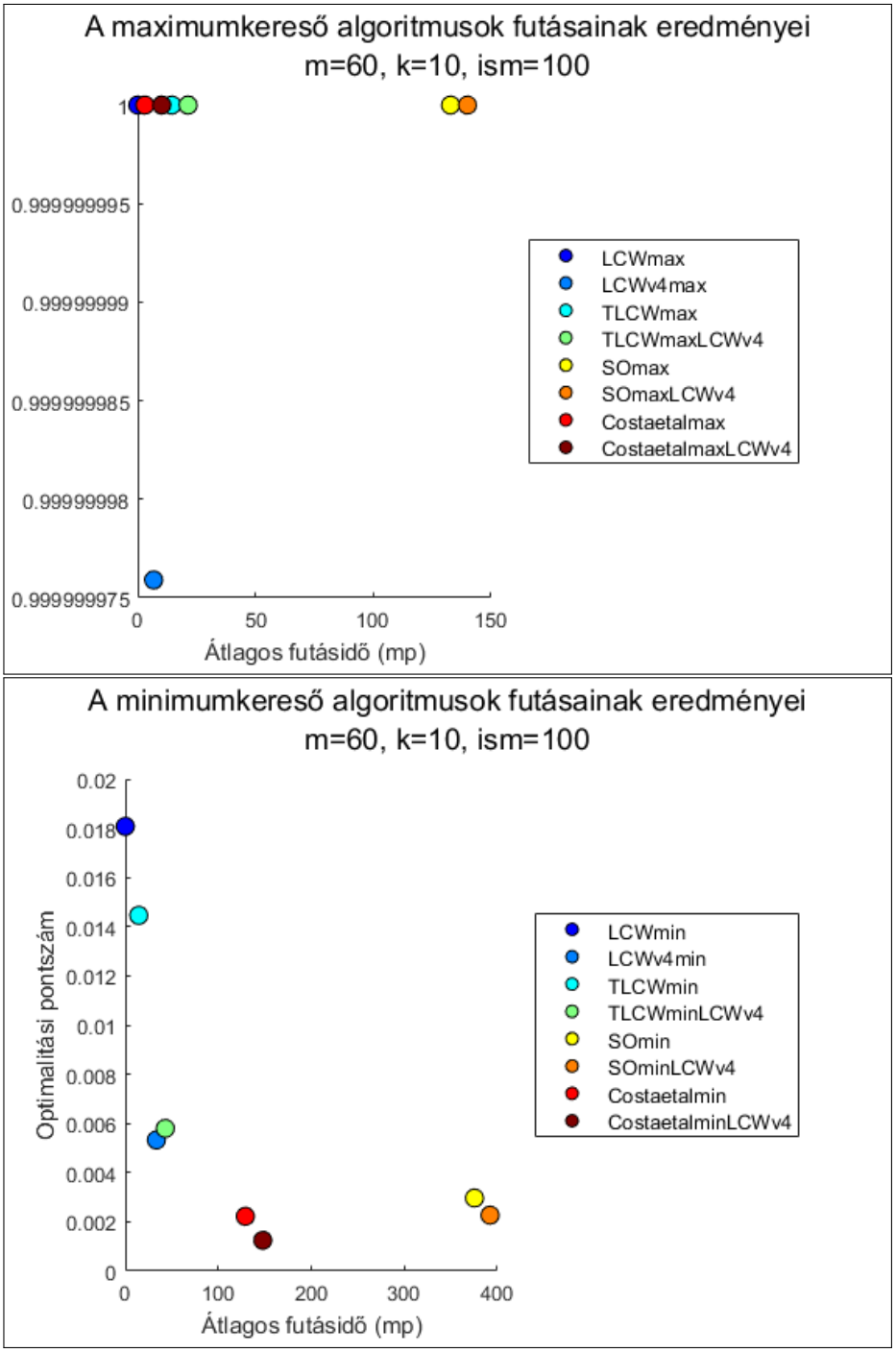
[2]Az algoritmusok MATLAB kódja nyíltan hozzáférhető, lásd Saunders (2022).



3.10. ábra. A maximum- és minimumkereső algoritmusok futásainak eredményei. $m = 5, k = 120$, 100 szimulált eset.

A 3.11. ábra $m = 60$ főből álló csoportok és $k = 10$ csoport mellett mutatja az eredményeket 100 generált hallgatói mintából számítva. A maximumkereső eljárások esetében gyakorlatilag minden eljárás minden esetben megtalálta ugyanazt az optimumot - az `LCWv4` eredmény 1e-8 nagyságrendű eltérésének hátterében valószínűleg numerikus pontatlanság áll. Ennélfogva hasonló esetekben még az egyszerű `LCWmax` eljárás is elegendő

lehet az optimum keresésére.



3.11. ábra. A maximum- és minimumkereső algoritmusok futásainak eredményei. $m = 60, k = 10$, 100 szimulált eset.

A minimumkereső heurisztikák ábrája esetében azt láthatjuk, hogy az optimalitási pontszám mentén az értékek sokkal jobban szétterülnek, mint a csoportok nagy száma esetén. Megfigyelhetjük, hogy az `LCWv4min` módszer segítségével itt is jelentősen tudunk javítani más eljárások célfüggvényértékén. Megjegyezzük továbbá, hogy habár az

`LCWv4min` jobbnak tűnik a `TLCWminLCWv4` módszernél, azok eredményei elég közel esnek egymáshoz, így a viszonyuk nem feltétlenül stabil.

4. fejezet

Hivatkozások

Arkin, E. M., Bae, S. W., Efrat, A., Okamoto, K., Mitchell, J. S. B. és Polishchuk, V. (2009). Geometric stable roommates. *Information Processing Letters*, 109(4):219–224. DOI: 10.1016/j.ipl.2008.10.003.

Arthur, D. és Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035. ISBN 978-0-898716-24-5.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A. és Protasi, M. (2003). *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer-Verlag Berlin Heidelberg 1999. ISBN: 978-3-642-63581-6. DOI: 10.1007/978-3-642-58412-1.

Bertoni, A., Goldwurm, M., Lin, J. és Saccà, F. (2012). Size constrained distance clustering: separation properties and some complexity results. *Fundamenta Informaticae*, 115(1):125–139. DOI: 10.3233/FI-2012-644.

Biró, P. (2006). Stabil párosítási modellek és ezeken alapuló központi párosító programok. *Szigma*, 37(3-4):153–175. `https://journals.lib.pte.hu/index.php/szigma/article/view/1096`.

Costa, L. R., Aloise, D. és Mladenović, N. (2017). Less is more: basic variable neighborhood search heuristic for balanced minimum sum-of-squares clustering. *Information Sciences*, 415-416:247–253. DOI: 10.1016/j.ins.2017.06.019.

Edmonds, J. (1965). Maximum matching and a polyhedron with $0, 1$-vertices. *Journal of Research of the National Bureau of Standards*, 69B(1-2):125–130. DOI: 10.6028/jres.069b.013.

Edwards, A. W. F. és Cavalli-Sforza, L. L. (1965). A method for cluster analysis. *Biometrics*, 21(2):362–375. DOI: 10.2307/2528096.

Feo, T. A. és Khellaf, M. (1990). A class of bounded approximation algorithms for graph partitioning. *Networks*, 20(2):181–195. DOI: 10.1002/net.3230200205.

Feo, T. A., Goldschmidt, O. és Khellaf, M. (1992). One-half approximation algorithms for the k-partition problem. *Operations Research*, 40:S170–S173. `https://www.jstor.org/stable/3840846`.

Gale, D. és Shapley, L. S. (1962). College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15. DOI: 10.2307/2312726.

Gallego, M., Laguna, M., Marti, R. és Duarte, A. (2013). Tabu search with strategic oscillation for the maximally diverse grouping problem. *Journal of the Operational Research Society*, 64(5):724–734. DOI: 10.1057/jors.2012.66.

Höppner, F. és Klawonn, F. (2008). Clustering with size constraints. In *Computational Intelligence Paradigms*, volume 137 of *Studies in Computational Intelligence*, pages 167–180. ISBN 978-3-540-79473-8. DOI: 10.1007/978-3-540-79474-5_8.

Jitta, A. és Klami, A. (2018). On controlling the size of clusters in probabilistic clustering. In *Thirty-Second AAAI Conference on Artificial Intelligence*, volume 32, pages 3350–3357. Palo Alto, CA: AAAI Press. `https://ojs.aaai.org/index.php/AAAI/article/view/11793`.

Kel'manov, A. V. és Pyatkin, A. V. E. (2016). On the complexity of some quadratic Euclidean 2-clustering problems. *Computational Mathematics and Mathematical Physics*, 56:491–497. DOI: 10.1134/S096554251603009X.

Király, B. és Tóth, L. (2011). Kombinatorika jegyzet és feladatgyűjtemény. Pécsi Tudományegyetem.

Kondor, G. (2018). k-szobatárs probléma metrikus térben – klaszterezés egyenlő elemszámú és kisméretű csoportokkal. In *Tavaszi Szél 2018 Konferencia = Spring Wind 2018: Konferenciakötet II.*, pages 549–563. ISBN: 9786155586316.

Kondor, G. (2022a). NP-hardness of m-dimensional matching problems. *Műhelytanulmány.*

Kondor, G. (2022b). Egyoldali párosítási piacok nehézségi eredményei magasabb dimenzióban. *Közgazdasági Szemle. Megjelenés alatt.*

Dr. Kovács, E., Szüle, B., Fliszár, V. és Vékás, P. (2011). *Pénzügyi adatok statisztikai elemzése: Egyetemi tankönyv.* Tanszék Kft., Budapest.

Lam, C.-K. és Plaxton, C. G. (2019). On the existence of three-dimensional stable matchings with cyclic preferences. In Fotakis, D. és Markakis, E., editors, *Algorithmic Game Theory*, SAGT 2019. Lecture Notes in Computer Science, vol 11801. Springer, Cham. DOI: 10.1007/978-3-030-30473-7_22.

Lin, J., Bertoni, A. és Goldwurm, M. (2016). Exact algorithms for size constrained 2-clustering in the plane. *Theoretical Computer Science*, 629:80–95. DOI: 10.1016/j.tcs.2015.10.005.

Malinen, M. I. és Fränti, P. (2014). Balanced k-means for clustering. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, S+SSPR 2014, pages 32–41. Springer, Berlin, Heidelberg. DOI: 10.1007/978-3-662-44415-3_4.

Morrill, T. (2010). The roommates problem revisited. *Journal of Economic Theory*, 145 (5):1739–1756. DOI: 10.1016/j.jet.2010.02.003.

Nobel Prize. (2012a). Press release. NobelPrize.org. `https://www.nobelprize.org/prizes/economic-sciences/2012/press-release/`.

Nobel Prize. (2012b). Scientific background. NobelPrize.org. `https://www.nobelprize.org/uploads/2018/06/advanced-economicsciences2012.pdf`.

Novick, B. (2009). Norm statistics and the complexity of clustering problems. *Discrete Applied Mathematics*, 157(8):1831–1839. DOI: 10.1016/j.dam.2009.01.003.

Pyatkin, A., Aloise, D. és Mladenović, N. (2017). NP-hardness of balanced minimum sum-of-squares clustering. *Pattern Recognition Letters*, 97:44–45. DOI: 10.1016/j.patrec.2017.05.033.

Rujeerapaiboon, N., Schindler, K., Kuhn, D. és Wiesemann, W. (2019). Size matters: Cardinality-constrained clustering and outlier detection via conic optimization. *SIAM Journal on Optimization*, 29(2):1211–1239. DOI: 10.1137/17M1150670.

Saunders, D. (2022). Weighted maximum matching in general graphs. MATLAB Central File Exchange. Letöltve: 2022. február 22. `https://www.mathworks.com/matlabcentral/fileexchange/42827-weighted-maximum-matching-in-general-graphs`.

Segev, D. L., Gentry, S. E., Warren, D. S., Reeb, B. és Montgomery, R. A. (2005). Kidney paired donation and optimizing the use of live donor organs. *Journal of the American Medical Association*, 293(15):1883–1890. DOI: 10.1001/jama.293.15.1883.

Weitz, R. R. és Lakshminarayanan, S. (1996). On a heuristic for the final exam scheduling problem. *Journal of the Operational Research Society*, 47(4):599–600. DOI: 10.1057/jors.1996.72.

5. fejezet

Saját publikációk jegyzéke

Folyóirat cikkek

- Bihary, Zsolt ; Csóka, Péter ; Kondor, Gábor (2018). A részvénytartás spektrális kockázata hosszú távon. Közgazdasági Szemle, 65 : 7-8 pp. 687-700. DOI: 10.18414/KSZ.2018.7-8.687
- Csóka, Péter ; Kondor, Gábor (2019). Delegációk igazságos kiválasztása társadalmi választások elméletével. Közgazdasági Szemle, 66 : 7-8 pp. 771-787. DOI: 10.18414/KSZ.2019.7-8.771
- Csóka, Péter ; Kondor, Gábor (2020). Csődszabályok pénzügyi hálózatokban. Alkalmazott Matematikai Lapok, 37 : 2 pp. 233-245. 10.37070/AML.2020.37.2.08
- Kovács-Szamosi, Rita ; Kondor, Gábor ; Varga, József (2021). Derivatív-ügyletek az iszlám bankrendszerben. Köz-Gazdaság, 16 : 4 pp. 203-221. DOI: 10.14267/RETP2021.04.12
- Kondor, Gábor (2022). Egyoldali párosítási piacok nehézségi eredményei magasabb dimenzióban. Közgazdasági Szemle. Megjelenés alatt.

Műhelytanulmányok

- Kondor, Gábor (2022). NP-hardness of m-dimensional matching problems.