Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Számítástudományi és Információelméleti Tanszék

Zarankiewicz-számok alsó becslése szemi random algoritmusokkal

Szakdolgozat

Készítette
Borsik Balázs

Konzulens
Héger Tamás

2025. december 21.

Tartalomjegyzék

HALLGATÓI NYILATKOZAT

Alulírott *Borsik Balázs*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem. A Függelékben egyértelműen megjelöltem, hogy a mesterséges intelligencia eszközeit alkalmaztam-e a dolgozat elkészítéséhez; amennyiben igen, annak módját és mértékét a táblázatban közöltem. Tudomásul veszem, hogy a mesterséges intelligenciával generált tartalomért – annak mértékétől függetlenül – teljes felelősséggel tartozom.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövege pedig a BME Címtárban regisztrált személyek számára elérhető legyen. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2025. december 21.

Borsik Balázs
hallgató

Kivonat

A Zarankiewicz-számok vizsgálatát Kazimierz Zarankiewicz lengyel matematikus vetette fel először 1951-ben megjelent cikkében. Az s, t, m, n paraméterekkel definiált Zarankiewicz-szám ($Z(s, t, m, n)$) az a legnagyobb k egész szám, melyre egy $m \times n$-es 0-1 mátrixban elhelyezhető k darab 1-es úgy, hogy ne keletkezzen olyan $s \times t$–es részmátrix, melynek minden eleme 1-es.

A Zarankiewicz-számok általános paraméterekre való meghatározása máig nyitott kérdés, melynek megoldása bizonyítottan nehéz feladat, és több, a véges matematikában ismert problémát is lezárna.

Jelen dolgozat célja egy olyan szemi-random algoritmus kidolgozása és megvalósítása volt, amely kis paraméterek esetén megközelíti, vagy el is éri a szakirodalomban rögzített pontos értékeket, ugyanakkor a korábbiaknál nagyobb paramétertartományokban is képes észszerű időn belül releváns eredményeket szolgáltatni.

A dolgozat keretében először ismertetjük az eddigi eredmények egy részét és azok matematikai hátterét. Ezt követően bemutatásra kerülnek a kipróbált algoritmusok, ezek matematikai háttere, valamint az implementáció részletei. Végezetül az algoritmus által generált alsó becsléseket összevetjük a szakirodalomban fellelhető táblázatok adataival, értékelve a módszer hatékonyságát és skálázhatóságát. A kapott eredmények a legtöbb esetben elérik a szakirodalomban eddig közölt értékeket. Számos esetben új vagy jobb becslést adunk, továbbá azonosítunk és kijavítunk néhány hibás szakirodalmi értéket is.

Abstract

The problem of Zarankiewicz numbers was first proposed by the Polish mathematician Kazimierz Zarankiewicz in a paper published in 1951. The Zarankiewicz number with parameters s, t, m, n, denoted as $Z(s, t, m, n)$, is defined as the maximum integer k such that an $m \times n$ 0-1 matrix can contain k ones without containing an $s \times t$ submatrix consisting entirely of ones.

Determining the exact value of Zarankiewicz numbers for general parameters remains an open question. It is a provenly difficult problem, the solution of which would settle several known issues in finite mathematics.

The objective of this thesis was to develop and implement a semi-random algorithm that approaches, or even matches, the exact values found in the literature for small parameters, while also being capable of providing relevant results for larger parameter ranges within a reasonable time frame.

First, the thesis reviews existing results and their mathematical background. Subsequently, the tested algorithms, their theoretical foundations, and the details of their implementation are presented. Finally, the lower bounds generated by the algorithm are compared with data available in the literature, evaluating the efficiency and scalability of the method. The obtained results match the values reported in the literature in most cases. In several instances, we provide new or improved bounds, and also identify and correct some errors found in the literature.

1. Bevezetés

A Zarankiewicz-probléma egy extremális gráfelméleti probléma, amit Kazimierz Zarankiewicz vetett fel először az 1951-ben megjelent cikkében [22]. Eredeti megfogalmazásában így hangzik:

1.1. Definíció. (Zarankiewicz-probléma (eredeti megfogalmazás)) *Vegyünk egy $m \times n$-es mátrixot, amely csak 0 és 1 értékeket tartalmaz. Legkevesebb hány darab 1-es szükséges, hogy mindenképp tartalmazzon $s \times t$-es csupa 1-es részmátrixot?*

A problémát ma már legtöbbször az alábbi széles körben használt gráfelméleti megfogalmazásában tárgyaljuk.

1.2. Definíció. *Egy $G = (A, B, E)$ páros gráf $K_{s,t}$-mentes, ha nem tartalmaz olyan s elemű csúcshalmazt A-ban és t elemű csúcshalmazt B-ben, amelyek egy $K_{s,t}$-vel izomorf részgráfot alkotnak.*

Megjegyzés. Definíció szerint egy $K_{s,t}$-mentes gráf nem feltétlen $K_{t,s}$-mentes.

1.3. Definíció. (Zarankiewicz-szám) *Az (m, n) méretű, $K_{s,t}$-mentes páros gráfok éleinek maximális számát $Z_{s,t}(m, n)$-nel jelöljük, és Zarankiewicz-számnak nevezzük.*

Az eredeti megfogalmazás ekvivalens $Z_{s,t}(m, n) + 1$-el, ez az alábbi mátrixos áttérésből azonnal következik. Egy (A, B, E) páros gráfhoz gyakran hozzárendeljük a hozzá tartozó szomszédsági mátrixot, melynek mérete $|A| \times |B|$. Ebben a megfeleltetésben a mátrix u-adik sora egy A-beli csúcsot, a v-edik oszlopa pedig egy B-beli csúcsot reprezentál. A két csúcs között pontosan akkor létezik él, ha a szomszédsági mátrix (u, v) pozíciójában az érték 1. Ellenkező esetben az érték 0 lesz. Tehát minden $K_{s,t}$-mentes páros gráf pontosan megfeleltethető egy olyan mátrixnak, amely nem tartalmaz $s \times t$ méretű csupa 1-es részmátrixot.

Megjegyzés. A dokumentumban a *gráf* kifejezés páros gráfokra utal. Amikor egy (A, B, E) páros gráfról beszélünk akkor az A csúcsosztály mérete m, a B csúcsosztály mérete pedig n, abban az esetben, ha ennek ellenkezője nincs hangsúlyozva.

A Zarankiewicz-probléma tekinthető a Turán-tétel páros gráfokra értelmezett változatának. A Turán-tétel 1941-ben jelent meg Turán Pál (*On an extremal problem in graph theory* című) cikkében [21], és az alábbi kérdésre adott választ: legfeljebb hány éle lehet egy n csúcsú egyszerű gráfnak anélkül, hogy tartalmazna K_t részgráfot. Erre a kérdésre Turán minden (n, t) paraméter esetén megadta a választ, és a hozzá tartozó optimális konstrukciót is. A Zarankiewicz-szám ennek a kérdésnek a páros gráfokkal feltett verziója, és a Turán által kimondott problémával ellentétben a Zarankiewicz-számokra nem ismert minden paraméterre érvényes megoldás vagy konstrukció.

A problémát a múlt század közepe óta aktívan vizsgálják, és az eredmények jellemzően két nagy csoportra bonthatóak: felső, valamint konstrukciókon alapuló alsó becslésekre.

A felső becslések közül az egyik legismertebb eredmény az 1954-es Kővári–T. Sós–Turán tétel [14], ami általános felső becslést adott (s, t, m, n) paraméterekre, ezt a felső becslésekről szóló fejezetben fogjuk bizonyítani. A becslés hátránya, hogy szigorú egyenlőtlenségeket

használ, így nem adhat pontos eredményt egyik esetre sem. Viszont $s = 2, t = 2, m = n$ paraméterek esetén aszimptotikusan pontos, és a következőt állítja: $Z_{2,2}(n, n) \leq n^{3/2} + n = O(n^{3/2})$, és végtelen sok n-re $Z_{2,2}(n, n) \geq \Omega(n^{3/2})$.

Reiman [16] már olyan felső becslést adott, ami adhat egyenlőséget. Továbbá megmutatta, hogy az egyenlőség kapcsolatban áll bizonyos matematikai struktúrák, úgynevezett projektív síkok létezésével. A Reiman által bemutatott felső becslést Hyltén-Cavallius [8] is bebizonyította.

Roman [17] ezt az irányt tovább javította, ő már több paraméterre éles felső becslést adott, továbbá az egyenlőség feltételeit általánosabb matematikai struktúrákkal hozta összefüggésbe, úgynevezett $t - (v, k, \lambda)$ dizájnokkal, melyekről majd az alsó becslésekről szóló fejezetben lesz szó.

Guy [7] a paraméterek szempontjából az aszimmetrikus esetben, azaz nagyjából $n > c(s, t)m^s$ feltétel mellett (itt $c(s, t)$ egy, az s-től és t-től függő konstans) határozta meg $Z_{s,t}(m, n)$ pontos értékét.

Az eddig bemutatott becslésekben a Jensen-egyenlőtlenségnek [10] (lásd 2.3. fejezetben.) alapvető szerepe van. Füredi [6] és Nikiforov [15] további ötletekkel javították az ismert felső becslések egy részét.

A dolgozatban alapvetően alsó becslésekkel fogunk foglalkozni; ennek ellenére a 2. fejezetben részletesen bemutatunk néhány felső becslést is, mert azok megmutatják, hogy az optimális vagy ahhoz közeli élszámú gráfok milyen tulajdonságokkal rendelkeznek, és ez számunkra is körülhatárolja, hogy milyen gráfokat keressünk.

Alsó becslések terén különböző kimerítő, heurisztikus keresések adhatnak pontos értékeket, de ezen módszerek hátránya, hogy a paraméterek növelésével a futásidő fenntarthatatlanná válik. A másik megközelítés az, hogy elméleti úton adunk konstrukciókat, melyek végtelen sok, de közel sem az összes paraméterre adnak pontos értékeket. A Zarankiewicz-számokat elérő gráfok konstruálása viszont nehéz feladat, hiszen az egyenlőséggel teljesíthető felső becslésekből tudjuk, hogy bizonyos paraméterekre a kérdés egyenértékű jó néhány matematikai struktúra létezésének eldöntésével. Ha általános megoldást találnánk a Zarankiewicz-problémára, azzal a projektív síkok, és általánosabban, a $t - (v, k, \lambda)$ blokkrendszerek létezésének vagy nemlétezésének kérdését is meg tudnánk válaszolni; ezek a kérdések viszont évtizedek óta nyitott, sokat kutatott, nehéz problémák [13, 23.o.], [1], továbbá lásd még [12]. Emiatt általános paraméterekre nem várható, hogy a Zarankiewicz-számok pontos értékét meg lehet határozni.

A Zarankiewicz-probléma más matematikai kérdésekkel is kapcsolatban áll, ilyen például a páros Ramsey-számok kérdése, amit az alábbi módon definiálhatunk. Legyen $r(m, n)$ az a legkisebb egész szám, amelyre teljesül, hogy a $K_{r,r}$ éleinek bármely két színnel való színezése tartalmaz egyszínű $K_{m,n}$ részgráfot. Irving [9] megmutatta, hogy a Zarankiewicz-számokra vonatkozó felső becslések közvetlenül felhasználhatóak a páros Ramsey-számok felső becslésére.

Szintén jellemző kutatási irány konkrét kisebb paraméterekre alsó, illetve felső becsléseket keresni. A korai szakirodalomban, például Guy (*A Many-facetted problem of Zarankiewicz* című) cikkében [7], ezeket az eredményeket jellemzően hosszadalmas, manuális esetszétválasztáson alapuló módszerekkel érték el. Ennek eredményeként a Guy által közölt táb-

lázatban néhány esetben hibás eredmény jelent meg. A frissebb cikkekben számítógépes keresésekkel, vagy egyéb konstrukciókkal próbálják meg az ismert pontos alsó és felső becslések paramétertartományát tovább bővíteni, ezekre jó példa a 2020-ban megjelent [2] cikk (A. Collins, A. Riasanovsky, J. Wallace, S. Radziszowski), vagy Tan 2022-es kézirata [20], amelyek egyaránt közölnek felső becsléseket és pontos értékeket. Ezzel szemben Kay 2016-os kézirata [11] kizárólag pontos értékeket közöl, míg a Davies, Gill és Horsley 2024-ben kéziratként megjelent tanulmánya [4] kifejezetten a felső becslések javítására összpontosít. A dolgozat eredményeit a 6. fejezetben ezekkel a munkákkal vetjük össze. Ezen összehasonlítás során a Collins és tsai. [2] által közölt táblázatban korábban nem ismert hibát is azonosítottunk.

Jelen dolgozat célkitűzése heurisztikus megközelítéssel minél nagyobb paramétertartományra kellően jó alsó becsléseket adni. Ezt részben randomizált algoritmusok segítségével érjük el. A cél a szakirodalomból ismert alsó becslések reprodukálása, helyenként megjavítása, illetve kiterjesztése volt nagyobb paramétertartományra észszerű futásidő mellett.

A téma kutatását az Önálló laboratórium tárgy keretében kezdtük meg, ahol a vizsgálatok fókusza még kizárólag a $K_{2,2}$-mentes esetekre korlátozódott. Bár a jelenlegi projekt nem a korábbi eredmények közvetlen folytatása, az ott szerzett tapasztalatok alapvető fontosságúnak bizonyultak az algoritmus újragondolásához. A programkódot a korábbi tanulságok és új ötletek ötvözésével teljes egészében újraimplementáltuk, az Önálló laboratórium során kapott eredmények (gráfok) nem lettek bemenetként felhasználva ezen projekt során.

A dolgozat felépítése a következő: A 2. fejezetben bemutatunk néhány felső becslést, a 3. fejezetben érintünk néhány fontosabb alsó becslést, melyek később motivációt adnak az algoritmus tervezéséhez. A 4. fejezetben bemutatásra kerülnek a program lépései, használt algoritmusok, majd az 5. fejezetben kitérünk a program technikai hátterére. Ezután a kapott eredményeket a 6. fejezetben ismertetjük és összevetjük a szakirodalomban található eredményekkel.

2. Felső becslések

Zarankiewicz problémájára leginkább felső becslések terén születtek ígéretes próbálkozások. Ennek elsődleges oka az, hogy több matematikai módszer áll rendelkezésünkre mikor felső becslést próbálunk adni, az alsó becslések konstrukciós és kimerítő kereséses megközelítéseihez képest. A felső becslésekhez alkalmazható például a Jensen-egyenlőtlenség, amelynek most két változatát fogjuk kimondani, majd bizonyítani.

2.1. A Jensen-egyenlőtlenség

A Jensen-egyenlőtlenség, kimondásánál, majd bizonyításánál támaszkodni fogunk a konvexitás fogalmára, ezért a későbbiekben használt definíciók az alábbiak lesznek:

2.1. Definíció. (Konvex tartomány) *Egy $H \subseteq \mathbb{R}^2$ tartományt* konvexnek *nevezünk, ha bármely két $A, B \in H$ pontjára teljesül, hogy az őket összekötő szakasz minden pontja is H-ban van. Formálisan:*

$$\forall\, A, B \in H \quad AB \subseteq H,$$

ahol AB az A és B pontokat összekötő szakasz.

Megjegyzés. A definícióból rögtön adódik, hogy ha A, B konvex, akkor $A \cap B$ is az.

2.2. Definíció. (Függvény konvexitása) *Egy $f : I \to \mathbb{R}$ függvényt az $I \subseteq \mathbb{R}$ intervallumon (ahol lehet $I = \mathbb{R}$) akkor nevezünk konvexnek, ha*

$$\{(x, y) : x \in I,\ y \geq f(x)\} \subseteq \mathbb{R}^2 \text{ egy konvex tartomány.}$$

Megjegyzés. A definícióban látott ponthalmaz a függvény grafikonja feletti tartomány.

2.3. Tétel. (Jensen-egyenlőtlenség) *Legyen $f : I \to \mathbb{R}$ egy konvex függvény az $I \subseteq \mathbb{R}$ intervallumon és $x_1, \ldots, x_n \in I$. Ekkor*

$$f\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right) \leq \frac{1}{n}\sum_{i=1}^{n} f(x_i).$$

2.4. Tétel. (Jensen-egyenlőtlenség egészekre élesített változata) *Legyen $f : I \to \mathbb{R}$ egy konvex függvény az $I \subseteq \mathbb{R}$ intervallumon és $x_1, \ldots, x_n \in I$ egészek, ahol $x_1 \leq \cdots \leq x_n$. Ekkor az $x'_1, \ldots, x'_n \in \mathbb{Z}$ egészekre, melyekre $\sum_{i=1}^{n} x'_i = \sum_{i=1}^{n} x_i$ és minden i-re $x'_i \in \{p, p+1\}$, ahol $p = \left\lfloor \frac{1}{n}\sum_{i=1}^{n} x_i \right\rfloor \in \mathbb{Z}$*

$$\sum_{i=1}^{n} f(x'_i) \leq \sum_{i=1}^{n} f(x_i).$$

Mindkét bizonyítás esetében az alábbi segédlemmát fogjuk alkalmazni.

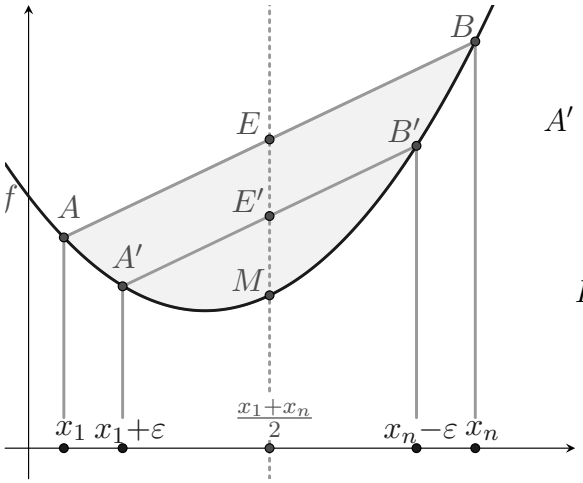
2.5. Lemma. *Legyen $f : I \to \mathbb{R}$ egy konvex függvény az $I \subseteq \mathbb{R}$ intervallumon és $x_1, \ldots, x_n \in I$, ahol $x_1 \leq \cdots \leq x_n$. Ekkor bármely $\varepsilon \leq \frac{x_n - x_1}{2}, \varepsilon \in \mathbb{R}$-re igaz, hogy:*

$$\sum_{i=1}^{n} f(x_i) = f(x_1) + f(x_2) + \cdots + f(x_n) \geq f(x_1 + \varepsilon) + f(x_2) + \cdots + f(x_{n-1}) + f(x_n - \varepsilon).$$

Bizonyítás. A tétellel ekvivalens, hogy

$$f(x_1) + f(x_n) \geq f(x_1 + \varepsilon) + f(x_n - \varepsilon), \text{ azaz}$$

$$\frac{f(x_1) + f(x_n)}{2} \geq \frac{f(x_1 + \varepsilon) + f(x_n - \varepsilon)}{2}.$$



Tekintsük az alábbi pontokat a síkon:

$$A = (x_1,\ f(x_1)), \qquad B = (x_n,\ f(x_n))$$

$$A' = (x_1{+}\varepsilon,\ f(x_1{+}\varepsilon)), \qquad B' = (x_n{-}\varepsilon,\ f(x_n{-}\varepsilon))$$

$$E = \left(\frac{x_1 + x_n}{2},\ \frac{f(x_1) + f(x_n)}{2}\right)$$

$$E' = \left(\frac{x_1 + x_n}{2},\ \frac{f(x_1 + \varepsilon) + f(x_n - \varepsilon)}{2}\right)$$

$$M = \left(\frac{x_1 + x_n}{2},\ f\left(\frac{x_1 + x_n}{2}\right)\right).$$

Mivel f konvex, az f grafikonja feletti, továbbá az AB egyenes alatti tartomány is konvex, ezért a tartományok metszete: T is konvex. (Az ábrán szürkével jelölt régió)

$A', B' \in T \Rightarrow A'B' \subseteq T$ (Def. 2.1.), tehát $A'B'$ minden pontja is T-n belül van. Illetve ez igaz A, B-re is.

Ezek alapján AB felezőpontja: $E = (\frac{x_1+x_n}{2},\ \frac{f(x_1)+f(x_n)}{2})$ T határán van, továbbá

$A'B'$ felezőpontja: $E' = (\frac{x_1+x_n}{2},\ \frac{f(x_1+\varepsilon)+f(x_n-\varepsilon)}{2}) \in T$.

Mivel $E, E' \in T$ és az x koordinátájuk megegyezik, ezért csak az y koordinátában lehet eltérés, de mivel E a T tartomány határán van, ezért következik, hogy:

$$\frac{f(x_1) + f(x_n)}{2} \geq \frac{f(x_1 + \varepsilon) + f(x_n - \varepsilon)}{2}.$$

Ez pedig, ahogy a bizonyítás elején beláttuk, ekvivalens az eredeti állítással. □

Megjegyzés. Látható, hogy mivel M a T tartomány alsó határán van, ezért E' sosem lehet M alatt, ezért az $\varepsilon = \frac{x_n - x_1}{2}$ választással lesz $f(x_1 + \varepsilon) + f(x_2) + \cdots + f(x_{n-1}) + f(x_n - \varepsilon)$ minimális.

Most pedig nézzük meg az eredeti Jensen-egyenlőtlenség bizonyítását.

Bizonyítás. Az állítás ekvivalens azzal, hogy

$$f\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right) n \leq \sum_{i=1}^{n} f(x_i).$$

Alkalmazzuk a segédlemmát (Lemma 2.5.)

$$f(x_1) + f(x_n) \geq f(x_1 + \varepsilon) + f(x_n - \varepsilon).$$

Használjuk a $d(j) = |\frac{1}{n}\sum_{i=1}^{n} x_i - x_j|$ jelölést (x_j átlagtól való távolsága), ekkor legyen $\varepsilon = d(1)$, ha $d(1) < d(n)$, egyébként pedig $\varepsilon = d(n)$. Cseréljük le az $x_1, \ldots, x_n$ számokat az $x_1 + \varepsilon, x_2, \ldots, x_{n-1}, x_n - \varepsilon$ számokra. A lemma szerint $\sum_{i=1}^{n} f(x_i)$ ezáltal csökken míg $\sum_{i=1}^{n} x_i$ nem változik. Ismételjük ezt a lépést, míg $x_1 = x_n$ teljesül. Mivel minden lépésben nő azon x_i-k száma, melyek egyenlők $\frac{1}{n}\sum_{i=1}^{n} x_i$-vel, maximum n lépés után elérjük ezt. □

Következhet az egészekre élesített változat bizonyítása.

Bizonyítás (Tétel 2.4.). Alkalmazzuk a segédlemmát a legkisebb lehetséges $\varepsilon = 1$ választással, hiszen $x_i \in \mathbb{Z}$, $1 \leq i \leq n$. Ezt addig alkalmazzuk, amíg igaz, hogy $(x_n - x_1) \geq 2$. Véges számú lépés után kapunk egy $x'_1, \ldots, x'_n$ halmazt, ahol $\forall i$-re, $x'_i \in \{p, p+1\}$, ezzel pedig beláttuk az állítást. Mivel $\sum_{i=1}^{n} x_i$ nem változik, $p = \left\lfloor \frac{1}{n}\sum_{i=1}^{n} x_i \right\rfloor$ kell legyen.

Most, hogy megismertük a Jensen-egyenlőtlenséget, nézzünk meg két felső becslést a Zarankiewicz-problémára. Ahhoz, hogy ezt megtehessük, be kell vezetnünk egy lemmát.

2.6. Lemma. (Kulcsegyenlőtlenség) *Legyen $G = (A, B, E)$ egy $K_{s,t}$-mentes $m \times n$-es páros gráf, ahol $|A| = m$ és $|B| = n$. Jelölje d_i a B csúcsosztály i-edik csúcsának fokszámát. Ekkor G-re teljesül:*

$$\sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s}.$$

Bizonyítás. Tekintsük azt az M halmazt, amely az A csúcshalmaz összes s elemű részhalmazát tartalmazza

$$M = \{\, S \subseteq A : |S| = s \,\} = \{S_1, S_2, \ldots, S_{|M|}\}.$$

Mondjuk azt, hogy egy $v \in B$ csúcs *illeszkedik* egy $S \in M$ halmazhoz, ha v szomszédja S minden csúcsának. Jelöljük az illeszkedéseket továbbiakban az alábbi módon: $S \subseteq N(v)$.

Számoljuk meg az illeszkedéseket először az egyik oldalról. A B-beli i-edik csúcsnak d_i szomszédja van az A oldalon, így ehhez pontosan $\binom{d_i}{s}$ darab $S \in M$ illeszkedik. Összesen tehát

$$\left|\{\, (v,S) \mid v \in B,\ S \in M,\ S \subseteq N(v) \,\}\right| = \sum_{i=1}^{n} \binom{d_i}{s}$$

illeszkedés van a gráfban.[1]

Most nézzük meg az illeszkedéseket a másik oldalról. Mivel G nem tartalmaz $K_{s,t}$ részgráfot, bármely $S \in M$-hez legfeljebb $(t-1)$ különböző $v \in B$ illeszkedhet, hiszen t ilyen csúcs az S beli csúcsokkal már $K_{s,t}$-t alkotna. Az M halmaz mérete $|M| = \binom{m}{s}$, ezért az összes illeszkedés a gráfban legfeljebb

$$\left|\{\, (v,S) \mid v \in B,\ S \in M,\ S \subseteq N(v) \,\}\right| \leq (t-1)\binom{m}{s}.$$

[1]Jogosan merülhet fel az a kérdés, hogy mi történik, ha valamelyik i-re $d_i < s$. Ekkor a $\binom{d_i}{s} = 0$ kell legyen, hiszen ekkor nincs illeszkedés. Ennek tisztázására az $\binom{x_1}{x_2}$ ezen definícióját használjuk a továbbiakban: Legyen $\binom{x_1}{x_2}$ egy x_1 elemű halmaz x_2 elemű részhalmazainak száma. Ez a definíció értelmezhető az $x_1 < x_2$-re is, és persze az eredmény 0.

Mivel ugyanazokat az illeszkedéseket vizsgáltuk meg kétféleképpen, így ezeket összehasonlítva megkapjuk a kívánt egyenlőtlenséget:

$$\sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s}.$$

□

2.2. Becslések

Most, hogy beláttuk a kulcsegyenlőtlenséget, nézzük meg az egyik legismertebb általános felső becslést a Zarankiewicz-problémára.

2.7. Tétel. (Kővári, T. Sós, Turán [14], [13]) *Legyen $G = (A, B, E)$ egy $K_{s,t}$-mentes páros gráf, ahol $|A| = m$, $|B| = n$. Ekkor G éleinek számára teljesül az alábbi becslés:*

$$e(G) < (t-1)^{\frac{1}{s}} m n^{1-\frac{1}{s}} + (s-1)n.$$

Bizonyítás. Induljunk ki a kulcsegyenlőtlenségből (Lemma 2.6.):

$$\sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s}.$$

A Jensen-egyenlőtlenséget (Tétel 2.3.) fogjuk használni az

$$f(x) = \binom{x}{s} = \frac{x(x-1)\cdots(x-s+1)}{s!}$$

függvényre. Ez a függvény $\forall x \in \mathbb{R}$ számra értelmezhető, és $x \in \mathbb{N}$ esetén értéke éppen $\binom{x}{s}$. A probléma az, hogy ez a függvény nem konvex a teljes $[0, \infty)$ intervallumon, csak ha $x \geq s-1$. Ezért vegyük azt a módosított függvényt, ami az ilyen esetekben 0, azaz

$$\binom{x}{s} := \begin{cases} 0, & \text{ha } x < s-1, \\ \frac{x(x-1)\cdots(x-s+1)}{s!}, & \text{ha } x \geq s-1. \end{cases}$$

Ez már egy konvex $[0, \infty) \to \mathbb{R}$ függvény. A Jensen-egyenlőtlenség miatt:

$$\sum_{i=1}^{n} \binom{d_i}{s} \geq n\binom{\frac{\sum_{i=1}^{n} d_i}{n}}{s} = n\binom{\frac{e(G)}{n}}{s}.$$

Ezután már csak ki kell hozni $e(G)$-t, hogy megkapjuk a becslést, amit egy egyszerű, de $s \geq 2$ esetén szigorú becsléssel megtehetünk:

$$n\frac{(\frac{e(G)}{n} - (s-1))^s}{s!} < n\binom{\frac{e(G)}{n}}{s} \leq \sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s} < (t-1)\frac{m^s}{s!}.$$

Átrendezve :

$$e(G) < (t-1)^{\frac{1}{s}} m n^{1-\frac{1}{s}} + (s-1)n$$

□

2.8. Tétel. (Roman-féle felső korlát [17]) *Legyen $G = (A, B, E)$ egy $K_{s,t}$-mentes páros gráf, ahol $|A| = m$, $|B| = n$, és $p \geq s - 1$ tetszőleges egész. Ekkor G éleinek számára teljesül az alábbi becslés:*

$$e(G) \leq \frac{t-1}{\binom{p}{s-1}}\binom{m}{s} + n \cdot \frac{(p+1)(s-1)}{s}.$$

Bizonyítás. Vegyük a kulcsegyenlőtlenség (Lemma 2.6.) során kapott alakot.

$$\sum_{i=1}^{n}\binom{d_i}{s} \leq (t-1)\binom{m}{s}.$$

Roman eredeti cikkében ezt az alakot használta tovább. Azonban annak érdekében, hogy látható legyen, hogy Roman módszere nem csak $f(x) = \binom{x}{s}$ esetén működik, folytassuk egy tetszőleges konvex $f(x)$ függvénnyel (lásd [3]).

Definiáljunk egy új függvényt tetszőleges $a > 0,\ a \in \mathbb{R},\ c \in \mathbb{R}$ paraméterekre:

$$F(x) = af(x) - x + c - t,$$

amely szintén konvex (ez könnyen látható, ha f kétszer deriválható, hiszen $F''(x) = a \cdot f''(x)$, ahol a és $f''(x)$ egyszerre pozitívak).

Válasszuk meg az a és c paramétereket úgy, hogy teljesüljön:

$$F(p) = F(p+1) = 0.$$

Ekkor:

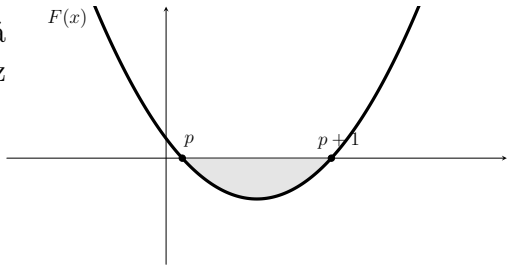
$$F(p) = af(p) - p + c = 0,$$
$$F(p+1) = af(p+1) - (p+1) + c = 0.$$

Ezekből:

$$a = \frac{1}{f(p+1) - f(p)}, \qquad c = \frac{pf(p+1) - (p+1)f(p)}{f(p+1) - f(p)}.$$

Mivel $F(x)$ konvex, $F(p) = 0$ és $F(p+1) = 0$, továbbá p és $p+1$ szomszédos egész számok, ezért bármely egész $x_1, \ldots, x_n$ számokra belátható, hogy:

$$\sum_{i=1}^{n} F(x_i) \geq 0 \quad \text{ha } x_i \in \mathbb{Z}.$$



Ezért:

$$\sum_{i=1}^{n} F(x_i) = \sum_{i=1}^{n}\big(af(x_i) - x_i + c\big) = a\sum_{i=1}^{n} f(x_i) - \sum_{i=1}^{n} x_i + nc \geq 0.$$

Átrendezés után:

$$\sum_{i=1}^{n} x_i \leq \frac{\sum_{i=1}^{n} f(x_i)}{f(p+1) - f(p)} + n \cdot \frac{pf(p+1) - (p+1)f(p)}{f(p+1) - f(p)}.$$

Ha $f(x) = \binom{x}{s}$ helyettesítést alkalmazzuk, akkor megkapjuk a Roman-féle egyenlőtlenséget. Ekkor

$$\sum_{i=1}^{n} x_i \leq \frac{\sum_{i=1}^{n} \binom{x_i}{s}}{\binom{p}{s-1}} + n \cdot \frac{(p+1)(s-1)}{s}.$$

Egyenlőség pontosan akkor teljesül, ha $x_i \in \{p,\, p+1\}$ minden $1 \leq i \leq n$ esetén.

Ha az előbb megkapott alakban kicseréljük x_i-ket d_i-kre azaz a fokszámokra, akkor:

$$\sum_{i=1}^{n} d_i \leq \frac{\sum_{i=1}^{n} \binom{d_i}{s}}{\binom{p}{s-1}} + n \cdot \frac{(p+1)(s-1)}{s}.$$

Majd a

$$\sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s}$$

kulcsegyenlőtlenséget (Lemma 2.6.) felhasználva megkapjuk a végső alakot. Az élszám azaz $\sum_{i=1}^{n} d_i$ felülről becsülhető:

$$e(G) = \sum_{i=1}^{n} d_i \leq \frac{t-1}{\binom{p}{s-1}}\binom{m}{s} + n \cdot \frac{(p+1)(s-1)}{s}.$$

□

Megjegyzés. Gondoljuk, meg, hogy mikor lehet egyenlőség, hol használtunk a képletben becsléseket?

Amikor $\sum_{i=1}^{n} F(x_i)$ értékét becsültük: $\sum_{i=1}^{n} F(x_i) \geq 0$, akkor úgy definiáltuk $F(x)$-et, hogy $x \in \{p, p+1\}$-re $F(x) = 0$ igaz legyen, tehát ennél a becslésnél pontosan akkor marad meg az egyenlőség, ha minden x_i-re igaz, hogy $x_i \in \{p, p+1\}$. De még használtunk egy becslést, mégpedig a kulcsegyenlőtlenségben (Lemma 2.6.) az illeszkedések számolásánál:

$$\sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s}.$$

Itt pedig pontosan akkor fogunk egyenlőséget kapni, ha igaz, hogy az A csúcsosztályban bárhogy választunk ki s darab csúcsot, azoknak mindig $(t-1)$ közös szomszédjuk lesz B-ben. A kérdés már csak az, hogy mikor létezik ilyen struktúra, hiszen akkor meg is találtuk $Z_{s,t}(m,n)$-t. A válasz pedig az, hogy nem minden s, t, m, n esetén léteznek ilyen struktúrák. A kérdéssel részletesebben az alsó korlátok fejezetben foglalkozunk.

Most, hogy megkaptuk a Roman-féle egyenlőtlenséget, nézzük meg, hogy miért is jó nekünk ez az elegáns megoldás, számoljunk végig egy konkrét példát kizárólag a Jensen-egyenlőtlenség egészekre élesített változatát (Tétel 2.4.) kihasználva.

Becslések összehasonlítása

Az alábbiakban illusztráció céljából, a fenti becslési módszereket fogjuk alkalmazni $Z_{2,2}(9, 18)$-ra.

Szeretnénk belátni, hogy $Z_{2,2}(9,18)$ kisebb mint 46. Tegyük fel indirekten, hogy létezik $K_{9,18}$-ban 46 élű $K_{2,2}$-mentes részgráf. A nagyobbik oldalon a fokszámok $d_1, d_2, \ldots, d_{18}$. Első lépésként írjuk fel a kulcsegyenlőtlenséget:

$$\sum_{i=1}^{18} \binom{d_i}{2} \leq (2-1)\binom{9}{2} = 36.$$

Mivel $\binom{x}{2}$ konvex függvény, a Jensen-egyenlőtlenség egészekre élesített változata (Tétel 2.4.) miatt a minimum akkor adódik, ha a fokszámok az átlaghoz a lehető legközelebb vannak. 46 él esetén az átlagos fokszám $\frac{46}{18} = 2,\dot{5}$, tehát a legkiegyenlítettebb eloszlás 10 darab 3 fokszámú csúcs és 8 darab 2 fokszámú csúcs. Ezért

$$\sum_{i=1}^{18} \binom{d_i}{2} \;\geq\; 10\binom{3}{2} + 8\binom{2}{2} = 10 \cdot 3 + 8 \cdot 1 = 38.$$

Összevetve a kettőt $38 \leq 36$, ellentmondás. Tehát $Z_{2,2}(9,18) \neq 46$.

Vizsgáljuk meg a 45 élű esetet. Ekkor az átlagos fokszám $\frac{45}{18} = 2,5$, tehát a legkiegyenlítettebb eloszlás 9 darab 3 fokszámú, és 9 darab 2 fokszámú csúcs. Ezért

$$\sum_{i=1}^{18} \binom{d_i}{2} \;\geq\; 9\binom{3}{2} + 9\binom{2}{2} = 9 \cdot 3 + 9 \cdot 1 = 36,$$

ami pontosan megegyezik a másik oldalról adódó felső korláttal. Így nincs ellentmondás. Következésképpen $Z_{2,2}(9,18) \leq 45$-öt nem tudjuk kizárni ezzel a becsléssel.

Megjegyzés. A levezetésből jól látszik, hogy amennyiben létezik ilyen struktúra, akkor a gráfnak pontosan 9 darab 3 és 9 darab 2 fokú csúcsának kell lennie a nagyobbik 18 elemű osztályában, és a kisebbik 9 elemű osztály bármely két csúcsának pontosan egy közös szomszédjának kell lennie. (Érdekesség: létezik ilyen gráf.)

Megjegyzés. Ha nem az élesített Jensen-egyenlőtlenséget használtuk volna, hanem a klasszikus, valós számokra használt alakját, akkor nem kaptunk volna ellentmondást a 46 élű esetre. Ekkor ugyanis az átlagos fokszám $\frac{46}{18} = 2.\dot{5}$, és a Jensen-egyenlőtlenség szerint

$$\sum_{i=1}^{18} \binom{d_i}{2} \;\geq\; 18 \cdot \binom{\frac{46}{18}}{2} = 18 \cdot \frac{\frac{46}{18}\left(\frac{46}{18}-1\right)}{2} = 18 \cdot \frac{\frac{46}{18} \cdot \frac{24}{18}}{2} = \frac{46 \cdot 28}{36} = 35.\dot{7}.$$

Ez az érték nem haladja meg a másik oldalról adódó felső korlátot, hiszen $\binom{9}{2} = 36$, így pusztán a valós számokra értelmezett Jensen-egyenlőtlenség nem ad ellentmondást.

A Roman-becslésből valójában ugyanazt ez eredményt kapjuk meg, mint a Jensen-egyenlőtlenség egészekre élesített változatából, csak gyorsabban:

$$e(G) \leq \frac{t-1}{\binom{p}{s-1}}\binom{m}{s} + n \cdot \frac{(p+1)(s-1)}{s},$$

ami $Z_{2,2}(9,18)$-ra (a kisebb értéket: 9-et érdemes m helyére behelyettesíteni, mivel ebből kapunk élesebb becslést):

$$e(G) \leq \frac{1}{p}\binom{9}{2} + 18 \cdot \frac{(p+1)}{2}.$$

$p = 1$-re: $e(G) \leq 54$.

$p = 2$-re: $e(G) \leq 45$.

$p = 3$-ra: $e(G) \leq 48$.

Mint látható, $p = 2$-re megkaptuk a korábban igazolt megoldást. Mivel $p = 3$-ra rosszabb eredményt kaptunk, mint $p = 2$-re, ezért nem is kaphatunk jobb eredményt, hiszen az egyenlőtlenség jobb oldala is egy konvex függvény p-re, így tetszőleges $p \geq 3$-ra a felső becslés értéke nagyobb mint $p = 2$-re.

3. Alsó becslések

Mint látható, több jól használható felső becslés is található a Zarankiewicz-problémára, bár ezek a becslések sok esetben jelentősen eltérhetnek a pontos értékektől. Például $Z_{2,2}(15, 15)$-re a Roman-féle felső becslés 63, miközben a pontos érték 61. Alsó becslésekre kevesebb képlet található, ezekben az esetekben a különféle konstrukciók, vagy adott tartományon végzett kimerítő keresések nyújthatnak jobb korlátokat. Mivel a szakdolgozat célja az ismert alsó becslések javítása volt, így az alsó korlátok vizsgálata különösen fontos szempont volt a program tervezése során. Az alábbiakban bemutatunk néhány, pontos becslést adó matematikai struktúrát és hozzájuk kapcsolódó motiváló jelenséget.

A Roman-féle felső becslés 2.8. pontosan meghatározza az egyenlőség feltételét. Ez akkor teljesülhet, ha az egyik csúcsosztály minden csúcsának fokszáma p vagy $p + 1$, tehát a csúcsok fokszámai közt legfeljebb 1 eltérés lehet, továbbá a másik csúcsosztály bármely s pontjának pontosan $(t - 1)$ közös szomszédja van. Ez az alábbi struktúrák, úgynevezett blokkrendszerek létezésével hozható kapcsolatba.

Forrás: *Damásdi, G., Héger, T. and Szőnyi, T., 2013. The Zarankiewicz problem, cages and geometries.* [3]

3.1. Definíció. (Illeszkedési struktúra) *Egy illeszkedési struktúra egy $(\mathcal{P}, \mathcal{B}, \mathcal{I})$ hármas, ahol:*

- *$\mathcal{P}$ egy halmaz, elemei a „pontok”,*
- *$\mathcal{B}$ egy $\mathcal{P}$-től különböző halmaz, elemei a „blokkok”,*
- *$\mathcal{I} \subseteq \mathcal{P} \times \mathcal{B}$ az „illeszkedési reláció”, vagyis azon pont-egyenes párok halmaza, melyek illeszkednek egymásra.*

Megjegyzés. Az előbb bemutatott definícióban több blokk is illeszkedhet ugyanazon ponthalmazokra, azaz megengedünk úgynevezett többszörös blokkokat. Léteznek úgynevezett illeszkedési rendszerek, melyek csak különböző blokkokat tartalmazhatnak (pontosabban az egyes blokkokra illeszkedő pontok halmazának kell különbözőnek lennie). Ilyenek konstruálása lényegesen nehezebb feladat, mint a megengedőbb struktúráké [19].

A $\lambda = 1$ esetben viszont nincs lényegi különbség a kettő között, hiszen ha két blokkra ugyanazok a $P_1, \ldots, P_k$ pontok illeszkednek, és $k \geq 2$ (az egyelemű blokkok nem különösebben érdekesek), akkor például a P_1 és a P_2 pontokra nem csak $\lambda = 1$ közös blokk illeszkedne.

3.2. Definíció. (Illeszkedési gráf) *Minden illeszkedési struktúra megfeleltethető egy $G = (A, B, E)$ páros gráfnak, ahol $A = \mathcal{P}$, $B = \mathcal{B}$, és $u \in A$, $v \in B$ között akkor van él, ha $(u, v) \in \mathcal{I}$. Ezt a gráfot hívjuk az illeszkedési struktúra* illeszkedési gráfjának.

3.3. Definíció. ($t - (v, K, \lambda)$ blokkrendszer) *Legyen $\emptyset \neq K \subset \mathbb{Z}^+$. Egy $(\mathcal{P}, \mathcal{B})$ illeszkedési struktúrát $t - (v, K, \lambda)$* blokkrendszernek *nevezünk, ha:*

- *$|\mathcal{P}| = v$,*
- *minden $B \in \mathcal{B}$ esetén $|B| \in K$,*
- *minden t különböző pont pontosan λ közös blokkban szerepel.*

Megjegyzés. Egy $t-(v,K,\lambda)$ blokkrendszer illeszkedési gráfja, egy $(v\times b)$ méretű, $K_{t,\lambda+1}$ mentes páros gráf, mivel teljesül, hogy bármely t különböző pont pontosan λ blokkban szerepel.

Ha $K=\{k\}$, akkor egyszerűen $t-(v,k,\lambda)$ blokkrendszerről beszélünk. Ilyen blokkrendszer esetén (lásd [13]):

$$b=|B|=\lambda\binom{v}{t}\Big/\binom{k}{t},\quad r=\frac{bk}{v}=\lambda\binom{v-1}{t-1}\Big/\binom{k-1}{t-1}$$

ahol b a blokkok száma, r pedig az egy pontra eső blokkok száma. Feltételezzük, hogy $k<v$ és $\lambda\geq 1$.

A (t,v,k,λ) paramétereket akkor nevezzük *megengedettnek*, ha pozitív egészek, teljesül:

$$2\leq t\leq k<v,\quad b=\lambda\binom{v}{t}\Big/\binom{k}{t}\in\mathbb{Z},\quad r=\lambda\binom{v-1}{t-1}\Big/\binom{k-1}{t-1}\in\mathbb{Z}$$

Egy (t,v,k,λ) blokkrendszer létezésének alapfeltétele a megengedettség, de nem minden megengedett paraméterhalmazra létezik ilyen blokkrendszer.

3.4. Tétel. *Legyenek adottak az $s,t,m,n\in\mathbb{Z}$ paraméterek, továbbá egy $p\in\mathbb{Z}$ paraméter. Ekkor $Z_{s,t}(m,n)$ pontosan akkor egyezik meg a paraméterek szerinti Roman-becsléssel, ha létezik $t-(n,\{p,p+1\},s-1)$ blokkrendszer $b=m$ blokkal. (Ez természetesen csak olyan a p-re teljesülhet, mellyel kiegészítve a paramétereket a Roman-becslés minimális.)*

Bizonyítás. A 2.8. Tételben bemutatott egyenlőségi feltétel könnyen látható, hogy megfelelő. Hiszen teljesül, hogy minden t különböző pont pontosan $(s-1)$ blokkban szerepel, továbbá, hogy minden blokk p, vagy $p+1$ fokú, tehát a blokkrendszer illeszkedési gráfja a Roman-féle felső becslésnek megfelelő élszámú gráf. Mivel a becslés levezetése során alkalmazott egyenlőtlenségekben csak ilyen eloszlás mellett állhat fenn egyenlőség, a feltétel szükséges is. □

Egy fontos példa illeszkedési struktúrára a projektív sík.

3.5. Definíció. *Egy* projektív sík *egy $2-(q^2+q+1,q+1,1)$ blokkrendszer, ahol $q\geq 2$ tetszőleges egész. Ez a q szám a projektív sík* rendje.

Projektív síkok létezése bizonyított, ha q prímhatvány, és sejtés, hogy csak prímhatvány q-ra létezik q rendű projektív sík [13].

A projektív síkok segítségével illusztrálunk egy általános jelenséget, miszerint optimális $K_{s,t}$-mentes gráfok gyakran tartalmaznak részgráfként kisebb csúcsszámú, optimális $K_{s,t}$-mentes gráfokat, ez a tulajdonság általános $t-(v,k,\lambda)$ blokkrendszerekre is igaz (bővebben [3]).

3.6. Állítás. *Legyen adott egy q egész, $v=q^2+q+1$ ponttal. Legyen c olyan egész, amelyre $c(c-1)<2q$. Ekkor*

$$Z_{2,2}(v-c,v)\leq(v-c)(q+1).$$

Egyenlőség elérhető, ha létezik q-rendű projektív sík.

BIZONYÍTÁS. Tekintsük a projektív sík incidenciagráfját, hagyjunk el c pontot az egyik osztályból. A kapott gráf $(v-c)\times v$ csúcsú és $(v-c)(q+1)$ élt tartalmaz.

Alkalmazzuk a Roman-féle felső becslést (Tétel 2.8.) az $s=2, t=2$ és $p=q$ paraméterekre.

$$Z_{2,2}(v-c,v) \le \frac{1}{q}\binom{v-c}{2} + v\frac{q+1}{2} = \frac{(v-c)(v-c-1)}{2q} + \frac{v(q+1)}{2}$$

$$= \frac{(v-c)(q^2+q-c)}{2q} + \frac{v(q^2+q)}{2q}$$

$$= \frac{(v-c)(q^2+q)-c(v-c)}{2q} + \frac{(v-c)(q^2+q)+c(q^2+q)}{2q}$$

$$= (v-c)(q+1) + \frac{c(q^2+q)-c(v-c)}{2q} = (v-c)(q+1) + \frac{c(v-1)-c(v-c)}{2q}$$

$$= (v-c)(q+1) + \frac{c(c-1)}{2q}$$

Mivel $Z_{2,2}(v-c,v)$ értéke szükségképpen egész, a fenti kifejezésnek vehetjük az alsó egészrészét. A feltételünk ($c(c-1)<2q$) éppen azt biztosítja, hogy a tört tagra:

$$0 \le \frac{c(c-1)}{2q} < 1.$$

Így a teljes kifejezés alsó egészrésze pontosan $(v-c)(q+1)$, ami igazolja az állítást. □

A következőkben egy roppant kézenfekvő, ám meglepően hatékony ötletet mutatunk be, melyet már Guy is használt [7]-ben, és azóta is gyakran alkalmazzák (lásd például Collins és tsai. [2]).

3.7. Tétel. (Optimális csúcselvétel) *Ha $G=(A,B,E)$ egy $K_{s,t}$-mentes, (m,n) méretű páros gráf, és $\delta(V)$ a $V\in\{A,B\}$ osztályban vett minimális fokszám, akkor*

$$Z_{s,t}(m-1,n) \ge e(G)-\delta(A),$$
$$Z_{s,t}(m,n-1) \ge e(G)-\delta(B).$$

A tétel magától értetődő, hiszen egy minimális fokszámú csúcsot törölve A-ból, illetve B-ből egy $K_{s,t}$-mentes $(m-1,n)$, illetve $(m,n-1)$ méretű részgráfot kapunk. Meglepő, hogy mennyi esetben kaphatunk pontos becslést $Z_{s,t}(m,n)$-re ennek az egyszerű becslésnek a használatával. A fenti tételben a gráf élszáma $e(G)=Z_{s,t}(m,n)$, és a minimális fokszámok felülről becsülhetőek $\delta(A)\le\left\lfloor\frac{Z_{s,t}(m,n)}{m}\right\rfloor$, illetve $\delta(B)\le\left\lfloor\frac{Z_{s,t}(m,n)}{m}\right\rfloor$ képletekkel. Ezeket behelyettesítve az alábbi következményt kapjuk.

3.8. Következmény. *Optimális (m,n) méretű gráfból indulva*

$$Z_{s,t}(m-1,n) \ge Z_{s,t}(m,n) - \left\lfloor\frac{Z_{s,t}(m,n)}{m}\right\rfloor,$$
$$Z_{s,t}(m,n-1) \ge Z_{s,t}(m,n) - \left\lfloor\frac{Z_{s,t}(m,n)}{n}\right\rfloor.$$

Nézzünk meg erre egy ábrát $Z_{2,2}(m,n)$-re.

$m\backslash n$	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
7	**21**														
8	22	24													
9	**24**	26	**29**												
10	25	28	31	**34**											
11	27	30	33	36	39										
12	28	32	36	39	42	**45**									
13	**30**	33	37	40	44	48	**52**								
14	31	35	**39**	42	45	49	**53**	**56**							
15	**33**	36	40	44	47	51	**55**	**58**	**61**						
16	34	38	42	46	**50**	53	**57**	60	64	**67**					
17	**36**	39	43	47	51	55	59	63	**67**	70	**74**				
18	37	**41**	45	49	53	57	61	65	69	73	77	**81**			
19	**39**	42	46	**51**	55	60	64	68	72	76	80	84	88		
20	40	**44**	**48**	52	57	61	**66**	70	75	80	84	88	92	96	
21	**42**	**45**	49	54	**59**	**63**	67	72	**77**	**81**	85	90	95	100	**105**

1. táblázat. $Z_{2,2}(m,n)$ értékei, optimális csúcselvételek jelölésével; $m,n \in [7,21]$ és $m \geq n$.

Az ábrán láthatóak $Z_{2,2}(m,n)$ pontos értékei. Ha $Z_{2,2}(m,n)$-ből megkapjuk $Z_{2,2}(m,n-1)$-et a 3.8. következmény használatával, akkor a táblázat (m,n) és $(m,n-1)$ koordinátáit közös zöld kerettel ábrázoljuk. Hasonlóan a piros keretek $Z_{2,2}(m,n)$-ből $Z_{2,2}(m-1,n)$ megkaphatóságát jelzik.

Megjegyzés. Lehetséges, hogy létezik olyan $Z_{2,2}(m,n)$ élű gráf, amire $\delta(A)$ vagy $\delta(B)$ kisebb, mint a fenti becslések, így a táblázat piros és zöld kereteinek elkészítéséhez használtnál szigorúbb becsléshez juthatnánk. Tehát lehetséges olyan eset, hogy egy kereten kívüli (m,n)-ből is eljuthatunk $(m-1,n)$-be (vagy $(m,n-1)$-be) a 3.7. Tételt felhasználva, mivel semmi nem garantálja, hogy egy $Z_{2,2}(m,n)$ élű $K_{2,2}$ mentes gráfban a minimális fok a $\delta(A)$-ra, illetve $\delta(B)$-re fent használt becslésekkel megegyezik, ilyenre példa: Tan [20] kézirata alapján van olyan $(15,15)$ méretű optimális, azaz 61 élű $K_{2,2}$-mentes gráf, melyben $\delta(A) = \delta(B) = 3$, melyből $Z_{2,2}(14,15) \geq 58$ következik.

Ahogy látható, sok esetben ez az egyszerűnek tűnő becslés is elég lehet arra, hogy az eredmények jó részét megtaláljuk. Félkövérrel azokat az eseteket jelöltük, melyeket nem lehet biztosan megkapni egy tartományon belüli másik gráfból a 3.8. következmény használatával. Mint látható, kevés ilyen eset van, és ez egyben azt is jelenti, hogy ha ezeket a gráfokat megtalálnánk, akkor mindössze a fenti következmény használatával meg lehet kapni az összes pontos értéket a bemutatott tartományban. Érdekesség, hogy elég $Z_{2,2}(21,21)$ értékét tudnunk ahhoz, hogy egészen $Z_{2,2}(17,8)$-ba eljussunk, közben 32 pontos érték, avagy a táblázat $26.\dot{6}\%$-ának megtalálásával.

Megjegyzés. Ez a megfigyelés a fordított irányú keresést is motiválja: egy meglévő, optimális $K_{s,t}$-mentes gráfhoz egy új csúcsot adva, és azt a másik osztály lehető legtöbb csúcsával összekötve (a $K_{s,t}$-mentességet megőrizve) szintén jó alsó becslésekhez juthatunk. Ezt a módszert Collins és társai [2] is alkalmazták kisebb paraméterű pontos értékek meghatározására.

4. Használt algoritmusok

A szakdolgozat célja alapvetően a kiválasztott s, t, m, n paraméterekre $Z_{s,t}(m, n)$-et élszámban minél jobban megközelítő, $K_{s,t}$-mentes gráf előállítása volt heurisztikus algoritmusok implementálásával. A következőkben részletesen ismertetjük az alkalmazott algoritmusokat, valamint bemutatjuk azok működését és hatékonyságát. Majd miután az algoritmus egyes változatait, lépéseit áttekintettük, az 5. fejezetben részletesen is bemutatjuk magát a program felépítését, programozási megfontolásokat.

Alapvető követelmény, hogy a talált gráfokat ne lehessen mohó módon kiegészíteni, erre az alábbi definíciót vezetjük be:

4.1. Definíció. *Egy $K_{s,t}$-mentes gráf* triviálisan kiegészíthető*, ha létezik olyan a gráfban még nem szereplő él, amelynek hozzáadása után a gráf továbbra is $K_{s,t}$-mentes marad.*

A későbbiekben minden programváltozat triviálisan nem kiegészíthető gráfokat találhat csak, hiszen ez alapvető feltétele a Zarankiewicz-számok elérésének.

A program alapötlete az volt, hogy kiindulunk egy (m, n) méretű, $K_{s,t}$-mentes gráfból (kezdetben az üres, azaz élmentes gráfból), a hiányzó élek közül néhányat behúzunk, majd a keletkező $K_{s,t}$-részgráfokat megszüntetjük egy-egy élük törlésével. A kapott gráf $K_{s,t}$-mentes lesz. Az eljárást a kapott gráfra több iteráción keresztül alkalmazzuk, hogy minél nagyobb élűszámú gráfokat kapjunk. $K_{s,t}$-mentes gráfok előállításához mindenképp el kell tudnunk dönteni, hogy keletkezett-e $K_{s,t}$ az élbehúzási fázisban. Ezért ennek a problémának a megoldása minden programváltozatban kelleni fog.

4.1. $K_{s,t}$ detektálás

A $K_{s,t}$ detektálás, kijelenthető, hogy a program egyik legköltségesebb része, éppen ezért ennek a feladatnak a kellően hatékony megoldása kulcsfontosságú. Igazából két oldalról lehet megközelíteni a problémát. Az egyik megoldás az, hogy az élek behúzásakor nem foglalkozunk azzal, hogy létrejönnek-e új $K_{s,t}$-k, majd amikor az iteráció végére értünk, akkor kell az egész gráfra megkeresnünk az összes $K_{s,t}$-t. A másik opció az, hogy minden e él behúzásánál megnézzük, hogy keletkeznek-e új $K_{s,t}$-k, melyek tartalmazzák az adott e élt. Ennek nagy előnye, hogy tökéletesen beleillik a program szerkezetébe, hiszen mindig lefuttatható az éppen behúzott élre, majd eltárolva a kapott $K_{s,t}$-ket az iteráció végén nagyon könnyű ezeket „kiritkítani". Ezen érvek miatt ezzel a megközelítéssel haladtunk tovább. A program minden esetben szomszédsági mátrixban tárolta a gráfokat, mivel a gyakorlatban ez bizonyult a gyorsabb megoldásnak (erről bővebben az 5. fejezetben).

Megjegyzés. Jogosan merülhet fel a kérdés, hogy nem lenne-e egyszerűbb a $K_{s,t}$-t képző éleket be sem húzni, és ezzel megspórolni a „kiritkítás" fázisát. A későbbi, 4.2.1. fejezetben bemutatott mohó algoritmus pont ezt a módszert veszi alapul, ennél pedig az előbb bemutatott irány jobbnak bizonyult, ezért a mohó megközelítést nem fejlesztettük tovább.

Az Önálló labor tárgy keretében végzett munkám során már előkerült a $K_{2,2}$ detektálás feladata, az akkori algoritmus az alábbi módon működött:

1. Algoritmus $K_{s,t}$ detektálás algoritmusa $K_{2,2}$-es esetben.

Bemenet: $G(A, B, E)$ gráf; $\{u, v\}$ él $(u \in A, v \in B)$
$\mathcal{S} \leftarrow \emptyset$ ▷ Eredményhalmaz inicializálása
for $u' \in A$ **do**
  if $u' \neq u \wedge \{u', v\} \in E$ **then**
    for $v' \in B$ **do**
      if $v' \neq v \wedge \{u, v'\} \in E \wedge \{u', v'\} \in E$ **then**
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{\{u, u'\} \cup \{v, v'\}\}$ ▷ $K_{2,2}$ hozzáadva az eredményhalmazhoz
      end if
    end for
  end if
end for
return $\mathcal{S}$ ▷ Visszatérés az összes megtalált részgráffal

Nagyobb $K_{s,t}$ paraméterekre két módszert próbáltunk ki. Az első a $K_{2,2}$ detektálás alábbi általánosítása. Nevezzük ezt a módszert működési elve alapján Pingpong metódusnak.

2. Algoritmus $K_{s,t}$ detektálás algoritmusa (Pingpong metódus).

Bemenet: $G(A, B, E)$ gráf; $\{u, v\}$ él $(u \in A, v \in B)$
$\mathcal{S} \leftarrow \emptyset$ ▷ Eredményhalmaz inicializálása
for $u_2 \in A$ **do**
  if $u_2 \neq u \wedge \{u_2, v\} \in E$ **then**
    for $v_2 \in B$ **do**
      if $v_2 \neq v \wedge \{u, v_2\} \in E \wedge \{u_2, v_2\} \in E$ **then**

        for $u_3 \in A$, ahol $index(u_3) > index(u_2)$ **do** ▷ Rendezés: ismétlések elkerülése
          if $u_3 \neq u \wedge \{u_3, v\} \in E \wedge \{u_3, v_2\} \in E$ **then**

            ... ▷ A ciklusok folytatódnak felváltva.
            ... ▷ Ha $s \neq t$: a kisebb oldal végeztével csak a hiányzót töltjük.

            for $v_t \in B$, ahol $index(v_t) > index(v_{t-1})$ **do**
              if $v_t \notin \{v, \ldots, v_{t-1}\} \wedge (\forall x \in \{u, \ldots, u_s\} : \{x, v_t\} \in E)$ **then**
                $\mathcal{S} \leftarrow \mathcal{S} \cup \{\{u, \ldots, u_s\} \cup \{v, \ldots, v_t\}\}$ ▷ $K_{s,t}$ elmentve
              end if
            end for
            ...
          end if
        end for
      end if
    end for
  end if
end for
return $\mathcal{S}$ ▷ Visszatérés az összes megtalált részgráffal

Az algoritmus felváltva veszi hozzá a csúcsokat a két csúcsosztályból, és így építi fel az $(s+t)$ darab csúcsot a $K_{s,t}$-t keresve. Minden ciklusban $index(u_i) > index(u_{i-1})$ feltétellel élünk (ugyanígy v-re), hogy egy kombinációt biztosan csak egyszer vizsgáljunk. Látható, hogy ezen módszer nagy hátránya az elméleti lépésszáma. Mivel minden ciklusban $O(m)$ vagy másik oldalon $O(n)$ lépést teszünk, és $(s-1)$ ciklus van A-ban, $(t-1)$ B-ben, ezért az algoritmus lépésszáma $O(m \cdot n \cdot m \cdots) = O(m^{s-1} \cdot n^{t-1})$.

Megjegyzés. Ha a $K_{s,t}$ tárolás nem cél, csak azok detektálása, akkor ez a módszer a gyakorlatban meglepően jó lesz futásidő szempontból, erről bővebben az 5.1. fejezetben fogunk beszélni.

A fent bemutatott $K_{2,2}$ detektálás másik általánosítása, az alább bemutatott Horgony metódus.

3. Algoritmus $K_{s,t}$ detektálás algoritmusa (Horgony metódus).

Bemenet: $G(A, B, E)$ gráf; $\{u, v\}$ él ($u \in A, v \in B$); s, t paraméterek
$\mathcal{S} \leftarrow \emptyset$ ▷ Eredményhalmaz inicializálása
$N(v) \leftarrow \{x \in A \mid \{x, v\} \in E\}$ ▷ v szomszédai az A oldalon
$U_{\text{jelöltek}} \leftarrow N(v) \setminus \{u\}$ ▷ u-t kizárjuk a jelöltek közül
if $|U_{\text{jelöltek}}| < s - 1$ **then**
    return $\mathcal{S}$ ▷ Nincs elég szomszéd egy $K_{s,t}$-hez
end if

for minden $U' \subseteq U_{\text{jelöltek}}$ részhalmazra, ahol $|U'| = s - 1$ **do** ▷ Most megvan az s db csúcs az A oldalon: $U' \cup \{u\}$
    $V_{\text{jelöltek}} \leftarrow \{y \in B \setminus \{v\} \mid \forall x \in (U' \cup \{u\}) : \{x, y\} \in E\}$ ▷ Azok a $y \in B$ csúcsok, melyek szomszédosak $(U' \cup \{u\})$-val
    if $|V_{\text{jelöltek}}| \geq t - 1$ **then**
        for minden $V' \subseteq V_{\text{jelöltek}}$ részhalmazra, ahol $|V'| = t - 1$ **do** ▷ Most megvan a t db csúcs a B oldalon: $V' \cup \{v\}$
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{U' \cup \{u\}\} \cup \{V' \cup \{v\}\}$ ▷ $K_{s,t}$ megtalálva: $U' \cup \{u\}$ és $V' \cup \{v\}$
        end for
    end if
end for
return $\mathcal{S}$ ▷ Visszatérés az összes megtalált részgráffal

Mint látható, ez a módszer a Pingpong eljárással ellentétben nem felváltva vizsgálja az oldalakat, hanem először rögzíti az A oldali v-vel szomszédos lehetséges csúcsokat, kiválaszt egy u-t nem tartalmazó, $(s-1)$ elemű U' csúcshalmazt, melyre $U' \subseteq N(v)$. Ezt követően megkeresi a B oldalon azokat a csúcsokat, amelyek U' minden elemével össze vannak kötve. Jelölje ezt a halmazt $V_{\text{jelöltek}} = \{y \in B \setminus \{v\} \mid y \in N(x),\ \forall x \in U' \cup \{u\}\}$. Ezután kiválasztja $V_{\text{jelöltek}}$ minden $(t-1)$ elemű V' részhalmazát, majd visszatér a talált $K_{s,t}$ részgráfokkal.

Most pedig vizsgáljuk meg lépésszám alapján ezt az algoritmust. Az A-ban $(s-1)$ elemű, v-vel szomszédos kombinációk végigpróbálása megint $O(m^{s-1})$ lépés lesz, viszont a belső ciklusban itt elég egyszer végigmenni B csúcsain, majd a kapott $V_{\text{jelöltek}}$ halmazból kell a $(t-1)$ elemű kombinációkat eltárolnunk. Látható, hogy a kombinációk gyártása miatt az

elméleti lépésszám a belső ciklusban is $O(n^{t-1})$, így ugyanazt a futásidőt kapjuk. Összességében az elméleti lépésszám nem lehet jobb, mint $O(m^{s-1} \cdot n^{t-1})$, hiszen ha egy adott élt behúzva a gráf egy teljes páros gráffá válik, akkor az algoritmusnak szükségszerűen $\binom{m}{s-1} \cdot \binom{n}{t-1}$ darab $K_{s,t}$ részgráfot kell megtalálnia és eltárolnia. A gyakorlatban viszont sokkal jobb futásidőre is számíthatunk, mivel alapvetően a kiinduló gráfjaink $K_{s,t}$-mentes gráfok, ezért nagyon valószínűtlen, hogy közel teljes páros gráfot fogunk kapni bemeneti gráfként. Így $V_{\text{jelöltek}}$ mérete várhatóan nem sokkal lesz nagyobb t-nél (számolhatunk $t + c$-vel, ahol c konstans), ezzel pedig a gyakorlatban a belső ciklus $O(n)$ lépés lesz, mivel a kombinációk gyártása, már nem lesz befolyásoló tényező. Ezért a gyakorlati lépésszám során $O(m^{t-1} \cdot n)$ lépésre számíthatunk. Ezen állítás, és az általunk készített mérések alapján (erről bővebben 5.1. fejezet) a program a Horgony metódust használja a $K_{s,t}$-k eltárolására a továbbiakban.

4.2. Programváltozatok, algoritmusok

A program több lépésben fejlődött, majd a legjobbnak bizonyuló módszerek összessége adta a program végső formáját. Kezdetben a $K_{2,2}$-avagy C_4-mentes esettel foglalkoztunk. Emiatt a különböző programváltozatok összehasonlítása is a $K_{2,2}$-es eseten lesz bemutatva. Az algoritmusokat összehasonlító elemzést egységes, 1800 másodperces futási időkorlát mellett végeztük. A kiértékeléshez az alábbi metrikákat fogjuk használni.

Megjegyzés. A dolgozatban a bemutatott mérőszámokat négy tartományra fogjuk vizsgálni, melyek a következőek $(10 \leq m, n \leq 20)$, $(20 \leq m, n \leq 30)$, $(30 \leq m, n \leq 40)$, $(10 \leq m, n \leq 40)$. Viszonyítási alapként, pedig az általunk kapott legjobb eredményeket fogjuk használni, „pontos" értékeknek venni.

A **Pontos illeszkedés** azt mutatja meg, hogy az adott módszer hány százalékban adott „pontos" megoldást a vizsgált intervallumban.

Az **Átlagos eltérés** azt mutatja meg, hogy a vizsgált módszernek mennyi a pontos értékektől való átlagos eltérése az adott paramétertartományban. Ehhez összeadjuk minden esetre az optimumtól való távolságot, majd ezt osztjuk az esetszámmal.

Az **Él-lefedettség** azt mutatja meg, hogy az adott módszer a vizsgált intervallumon belül összességében mennyire közelíti meg a legjobb általunk ismert eredményeket. Ehhez a módszer által talált legjobb gráfok élszámait összeadjuk, majd elosztjuk az általunk kapott legjobb eredmények élszámainak összegével.

Minden módszer leírása végén egy eredménytáblázat található, mely az előbb bemutatott metrikákat tartalmazza.

Mielőtt rátérnénk a komplexebb heurisztikák tárgyalására, elsőként nézzük meg a leginkább magától értetődő programváltozatot, a mohó algoritmust.

4.2.1. Mohó algoritmus (randomizált)

Algoritmus: Véletlen sorrendben végig megyünk az üres gráf összes lehetséges élén. Ha egy él behúzása $K_{s,t}$-t hozna létre, akkor nem húzzuk be, különben behúzzuk.

Az így a kapott gráf biztos, hogy nem triviálisan kiegészíthető, hiszen ha az lenne, akkor az utólag behúzható élt már behúztuk volna, amikor az algoritmus során vizsgáltuk. Továbbá

biztos, hogy $K_{s,t}$-mentes, hiszen csak olyan éleket húztunk be, melyek nem hoznak létre $K_{s,t}$-t.

Megjegyzés. Az éleket csak sorban vizsgálva mindig ugyanazt a megoldást kapnánk, ráadásul ez a megoldás nagyon aszimmetrikus lenne, például az elsőnek vizsgált csúcs össze lenne kötve az egész másik csúcsosztállyal. Éppen ezért vizsgáljuk véletlen sorrendben az éleket, így pozitív valószínűséggel találhatjuk meg a $Z_{s,t}(m,n)$ élű gráfot, bár ez gyakorlatban nagyobb paraméterekre igen valószínűtlen.

Eredmények			
Tartomány (min,max)	**Pontos illeszkedés**	**Átlagos eltérés**	**Él-lefedettség**
$(10, 20)$	27,3%	2,05	96,6%
$(20, 30)$	0%	12,26	90,52%
$(30, 40)$	0%	24,5	88,3%
$(10, 40)$	13,91%	8,75	92,8%

2. táblázat. Mohó algoritmus eredményei.

4.2.2. Moser–Tardos módszer

A program alapötlete a Moser–Tardos-féle randomizált módszer implementálása volt, mivel ez az algoritmus garanciákat ad az eredmények várható értékére. Később azonban kiderült, hogy pusztán ezen módszer használatával nehezen érhetőek el a kívánt eredmények. Mivel a program kezdeti változatának motivációját ez a módszer adta, fontosnak tartjuk legalább érintőlegesen bemutatni a matematikai hátterét, amiről átfogóbb leírás a [18] cikkben található.

4.2. Definíció. (Függetlenségi gráf) *Legyenek $E_1, E_2, \ldots, E_n$ események egy valószínűségi térből. A G gráfot függetlenségi gráfnak nevezzük, ha csúcshalmaza $\{1, 2, \ldots, n\}$, és minden i-re az E_i esemény teljesen független az*

$$\{E_j : j \neq i,\ (j,i) \notin E(G)\}$$

események halmazától.

Forrás: *Rónyai Lajos: Véletlen és algoritmusok 2011* [18]

4.3. Tétel. (LLL (Lovász lokális lemma) egy változata)) *Legyen $\mathcal{P}$ független valószínűségi változok egy véges halmaza, az Ω valószínűségi térből. Legyenek $E_1, \ldots, E_n$ események ezekből a valószínűségi változókból származtatva úgy, hogy minden i-re van egy olyan $P_i \subseteq \mathcal{P}$, melyre E_i bekövetkezése csak a P_i-beli változók értékeitől függ. G legyen egy függetlenségi gráf, $d \geq 1$, és teljesüljön:*

- $\mathbb{P}(E_i) \leq p$ *minden i-re,*
- *G-ben a csúcsok foka legfeljebb d,*

- $4dp \leq 1$.

Ekkor $\mathbb{P}\left(\bigcap_{i=1}^{n} \overline{E_i}\right) > 0$.

Az E_i eseményekre úgy gondolunk mint elkerülendő problémákra; a cél tehát egy $\omega \in \bigcap_{i=1}^{n} \overline{E_i}$ elemi esemény bekövetkezése. Az *LLL* igazolja adott feltétel mellett, hogy a $\bigcap_{i=1}^{n} \overline{E_i}$ nem lehetetlen esemény.

Az *LLL*-nek létezik egy algoritmikus változata, a Moser–Tardos-algoritmus, amely az alábbi lépésekből áll.

Moser–Tardos-algoritmus:

1. Kiértékeljük a $\xi \in \mathcal{P}$ változókat egy véletlen $\omega \in \Omega$ helyen. Ha egyik E_i esemény sem következik be, akkor ω egy elemi esemény a $\bigcap_{i=1}^{n} \overline{E_i}$ metszetből, és az eljárás véget ér.

2. Ha valamelyik E_j esemény bekövetkezett, akkor újra kiértékeljük a P_j-beli ξ változókat a saját eloszlásuk szerint, függetlenül. A $\mathcal{P} \setminus P_j$ változókat változatlanul hagyjuk. Ezután ismét ellenőrizzük, hogy bekövetkezik-e valamelyik E_i esemény, és szükség esetén ismételjük a 2. lépést.

Megjegyzés. Moser és Tardos bizonyították, hogy a 2. lépést várhatóan legfeljebb $\frac{n}{d-1}$ alkalommal kell elvégeznünk.

Az eredeti ötlet az imént bemutatott Moser–Tardos-algoritmus megvalósítása volt. Mivel kezdetben csak a $K_{2,2}$-mentes (C_4-mentes) definiálva esetre koncentráltunk, ezért az algoritmusok is erre az esetre lesznek bemutatva. Szerencsére a legjobbnak bizonyuló változat egyszerűen átültethető az általános $K_{s,t}$-mentes esetre is, így felhasználható volt későbbi programváltozatokban is. Az E_i nemkívánatos eseményt kezdetben kétféleképpen állítottuk be:

1. E_i esemény: Egy csúcsosztályban két csúcsnak van két közös szomszédja.

2. E_i esemény: C_4 képződik.

A második (C_4-re épülő) változat jobb eredményeket adott, ami érthető, hiszen az elsőként definiált eseményhez több C_4 is hozzárendelhető. Így ez a megközelítés kevésbé illik rá a nem kívánt esetekre. Érdekesség, hogy ekkor mindkét vizsgált algoritmus még mindig gyengébben teljesített a mohó algoritmusnál, ezért későbbiekben utolsó lépésként minden kimeneti gráfot triviálisan kiegészítettünk a mohó algoritmus futtatásával. Ezután az újrasorsolásban tértünk el az eredeti Moser–Tardos-algoritmustól, mivel jelenleg egy C_4 újrasorsolásánál csak éleket veszünk el a gráfból, sosem adunk hozzá új éleket, hiszen a C_4 definíciójából adódóan mind a négy élnek szerepelnie kell a gráfban. Ezért valójában az ismertetett módon futtatva a Moser–Tardos-algoritmust egy „mintavétel és igazítás” (sample and modify) elven működő eljárást kapunk: generálunk egy véletlen gráfot, majd a „hibákat” (a keletkező C_4-eket) lokálisan igazítjuk (egy C_4-ből egy véletlenszerűen választott élt törlünk, amíg nem marad C_4); lásd [18]. Az újrasorsolás helyett megpróbáltuk minél kevesebb törléssel megszüntetni az összes C_4-et a gráfban, ehhez kezdetben egyesével töröltük a C_4-ek éleit addig, amíg az összes gráfban található C_4 meg nem szűnt. Ez a módszer a mohó algoritmussal kombinálva már jobb eredményeket hozott, mint a mohó algoritmus

önmagában. De a C_4 „kiritkításon” még lehetett fejleszteni annak érdekében, hogy összesen minél kevesebb élt kelljen törölni. A későbbiekben az alábbi módszerrel dolgoztunk: számon tartjuk, hogy egy él mennyi C_4-ben szerepel, majd a legtöbb C_4-ben szereplő élek közül választjuk ki az adott lépésben törlendő élt: ezzel a módosítással szemi-random algoritmust kapunk, ahol egy randomizált generálási fázist egy determinisztikus javítási fázis követ. Ezt ismételve kapjuk meg a végső C_4 mentes gráfot. A program alapja erőteljesen támaszkodik a Moser–Tardos, pontosabban a mintavétel és igazítás algoritmusra, az így kapott program alapszerkezete az alábbi lépésekből áll (a $K_{s,t}$ detektálást a 4.1. fejezetben leírtak alapján végezzük):

4. Algoritmus Algoritmus alapja.

Input: m, n, élbehúzási valószínűség p
$G \leftarrow$ üres $m \times n$ páros gráf
for minden lehetséges él $e = (u, v)$, ahol $u \in A$, $v \in B$ **do**
    if véletlen() $< p$ **then**
        húzzuk be az e élt G-be
        jegyezzük fel azokat a $K_{s,t}$-eket, amelyek az e él behúzásával keletkeztek
    end if
    while a gráf G tartalmaz legalább egy $K_{s,t}$-et **do**
        keressük meg azt az l élt, amely a legtöbb $K_{s,t}$-ben szerepel
        távolítsuk el l-t G-ből
    end while
end for
Output: a kapott $K_{s,t}$-mentes páros gráf G

Végig iterálunk adott s, t paraméterek mellett a kívánt (m, n) paramétertartományon, majd egy kezdetben üres (m, n) nagyságú gráfra a fent bemutatott lépéseket végezzük el. A kapott eredménygráfon lefuttatjuk a mohó algoritmust (ezt az eredménygráfokon minden esetben megtesszük elmentés előtt), hogy biztosan triviálisan nem kiegészíthető gráfot kapjunk.

Mivel a program kiindulási alapját a Moser–Tardos-algoritmus képezi, a továbbiakban ezen a néven hivatkozunk rá, annak ellenére is, hogy a fejlesztett eljárás alapvető mechanizmusa már a mintavétel és igazítás elvét követi.

Az algoritmus 1800 másodpercen keresztül többszöri futtatásának legjobb kapott eredményei az alábbi táblázatban láthatóak.

Eredmények			
Tartomány (min,max)	**Pontos illeszkedés**	**Átlagos eltérés**	**Él-lefedettség**
$(10, 20)$	31,82%	1,95	96,75%
$(20, 30)$	0%	12,23	90,54%
$(30, 40)$	0%	24,33	88,38%
$(10, 40)$	12,9%	8,71	92,83%

3. táblázat. Moser–Tardos-algoritmus okosított változatának eredményei.

A következő fejlesztés az élbehúzási valószínűségek futás közbeni módosítása volt. A módszer alapgondolata az volt, hogy a gráf struktúrájának változásával együtt változzanak az élbehúzási valószínűségek is, hogy több esetben kapjunk a felső becslésekből, konstrukciókból megismert gráfokhoz hasonló fokszámeloszlású struktúrákat. Ennek elérése érdekében az alábbi módszer került bevezetésre az algoritmusban.

4.2.3. Dinamikusan karbantartott valószínűségek

Az élbehúzási valószínűségekre futás közben a gráf struktúrájával együtt változó valószínűségeket alkalmaztunk. Ennek a megközelítésnek a motivációja az volt, hogy a Roman-féle felső korlát (Tétel 2.8.) esetén akkor és csak akkor lehet egyenlőség, ha a megfelelő osztály minden csúcsának foka p vagy $p + 1$. Tehát intuitívan a kisebb fokszámú csúcsokra illeszkedő éleket szeretnénk behúzni nagyobb valószínűséggel. Jelenleg a program öt módszert tartalmaz valószínűségek detektálására, melyek nagyon hasonló eredményeket adnak. Az összehasonlításhoz az alábbi módszert fogjuk használni.

Megjegyzés. Sokáig egy heurisztikusabb változatot használtunk, mely matematikailag sokkal megalapozatlanabb, mint az alábbi, jelenleg is használt kód. A 6. fejezetben bemutatott eredmények nagy része ezzel a régebbi módszerrel lett elérve.

5. Algoritmus Élbehúzási valószínűségek kódrészlet.

```
int m_, n_;
int edge_count_; // élek száma G-ben
int upper_bound_; // Roman-féle felső becslés
int degree_m_[]; // csúcsok fokszámai A-ban
int degree_n_[]; // csúcsok fokszámai B-ben
double expected_m_; // m-beli csúcs várható fokszáma
double expected_n_; // n-beli csúcs várható fokszáma

double get_p(int u, int v){
  double expected_p = (upper_bound_ - edge_count_) / (m_ * n_ - edge_count_);

  double prob_A = (expected_m_ - degree_m_[u]) / (n_ - degree_m_[u]);
  double prob_B = (expected_n_ - degree_n_[v]) / (m_ - degree_n_[v]);

  double term_A = (expected_p + 2.0 * prob_A) / 3.0;
  double term_B = (expected_p + 2.0 * prob_B) / 3.0;

  return (term_A + term_B) / 2.0;
}
```

A behúzási valószínűségek meghatározásánál az elsődleges cél az volt, hogy a folyamat végére a gráf élszámának várható értéke megegyezzen a Roman-féle felső becsléssel, miközben a fokszámeloszlás a lehető legegyenletesebb marad.

Jelölje E a gráf éleinek halmazát, R pedig a Roman-féle felső becslés értékét. Ekkor a fenti kód garantálja, hogy a lehetséges, de még be nem húzott $e \notin E$ élek behúzási valószínűségeinek összege pontosan a hiányzó élek száma:

$$\sum_{e \notin E} \texttt{get\_p}(e) = R - |E|,$$

így a kapott gráf élszáma minden iteráció végén várhatóan R.

A dinamikusan karbantartott algoritmust használó programváltozat eredményei az alábbi táblázatban láthatóak:

Eredmények			
Tartomány (min,max)	**Pontos illeszkedés**	**Átlagos eltérés**	**Él-lefedettség**
$(10, 20)$	36,36%	1,64	97,28%
$(20, 30)$	0%	11,58	91,05%
$(30, 40)$	0%	23,38	88,84%
$(10, 40)$	17,54%	8,18	93,27%

4. táblázat. Dinamikusan karbantartott valószínűségek eredményei.

4.2.4. Belső iterációk futtatása

A Moser–Tardos-algoritmus, és az alapötlet része is, hogy ne csak üres gráfokon kezdjük meg az iterációkat, hanem a behúzás-éltörlés ciklusok több iterációban is lefussanak, mielőtt újrakezdenénk a futtatást.

Így a program következő eleme ez a lépés volt, ezt a fejlesztést a dinamikus valószínűségeket használó programváltozaton teszteltük le, de későbbi fejlesztések is profitálnak ebből a módszerből. Az összehasonlításokhoz $1, 3, 6, 12$ belső iterációt futtattunk váltogatva minden iterációban.

Eredmények			
Tartomány (min,max)	**Pontos illeszkedés**	**Átlagos eltérés**	**Él-lefedettség**
$(10, 20)$	75,76%	0,53	99,12%
$(20, 30)$	0%	9,11	92,96%
$(30, 40)$	0%	21,35	89,81%
$(10, 40)$	26,81%	6,75	94.45%

5. táblázat. Belső iterációk eredményei.

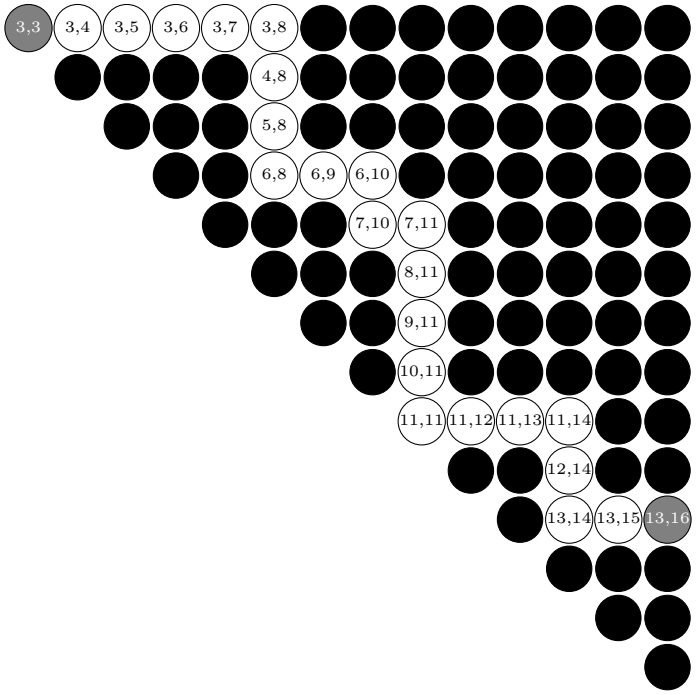
4.2.5. Optimális kiegészítés

A program egyik legtöbbet hozzáadó fejlesztése a optimális kiegészítés volt. Magától értetődő, hogy ha már van egy jó gráf, amit valamilyen módon újra tudunk használni a futás során, akkor nem éri meg nulláról indulni. Hiszen az eddig meglévő gráfokhoz hasonló gráfok felépítése is időt vesz igénybe, így ezzel a megoldással ez a költség többé-kevésbé megspórolható. A másik matematikailag megalapozottabb érv, pedig a 3.7. Tételben kimondott alsó becslés volt, és annak meglepően jó eredményei a vizsgált tartományokon. A tétel közvetlen implementációja ugyan csúcselvételen alapulna, azonban ezt a programban fordított irányban valósítottuk meg. Ennek oka az, hogy általában nehéz kellően nagy méretű, megfelelő gráfot találni, amelyből csúcselvétellel kisebb (m, n) paraméterekre is jó megoldások nyerhetőek. Ezzel szemben a választott megoldás lehetővé teszi, hogy a program közvetlenül felhasználja a korábbi futások során kapott eredményeket. Az algoritmus kiindulópontként az addig elért legjobb $(m, n-1)$, illetve $(m-1, n)$ méretű gráfokat használja, és ezek kiegészítésével keres megoldást az (m, n) paraméterpárra. A végső, program által is használt algoritmus így néz ki:

Mielőtt elkezdenénk futtatni az algoritmusunkat $Z_{s,t}(m, n)$-re, keressük meg az eddigi legjobb $Z_{s,t}(m-1, n)$ vagy $Z_{s,t}(m, n-1)$ értéket, és olvassuk be a hozzá tartozó gráfot. A program alapértelmezetten a két legjobb talált gráfot menti, így a két legjobb $Z_{s,t}(m-1, n)$ és a két legjobb $Z_{s,t}(m, n-1)$ adja a kiinduló gráfok halmazát. A program ezekből választja az éppen soron következőt kiinduló gráfnak. Például, ha az egyik $(m-1, n)$ dimenziós gráfot választja ($(m, n-1)$-re analógan), akkor az újonnan hozzávett $u_m \in A$ csúcsból kimenő éleket véletlen sorrendben megpróbáljuk hozzávenni a gráfhoz. Ehhez először végigmegyünk véletlen sorrendben az érintetlen B csúcsosztály csúcsain. Ha az éppen vizsgált $v' \in B$ csúcs hozzá vétele u_m szomszédaihoz nem hoz létre $K_{s,t}$-t (ehhez a 4.1. fejezetben bemu-

tatott algoritmust használjuk), akkor kiválasztjuk, ellenkező esetben nem. Így kapunk egy $N(u_m)$ csúcshalmazt az érintetlen B csúcsosztályból, ezt a folyamatot $\frac{mn}{4} + 5$-ször megismételjük, majd ezután a legnagyobb elemszámú csúcshalmaz lesz az újonnan hozzávett csúcs szomszédainak halmaza. Látható, hogy az így kapott gráf továbbra is $K_{s,t}$-mentes marad, hiszen a kiinduló gráf is $K_{s,t}$-mentes volt, és csak olyan új élt vettünk hozzá a gráfhoz, ami nem hozz létre új $K_{s,t}$-t.

Hogy biztosak lehessünk benne, hogy a csúcs hozzáadás is hasonlóan jól működik, mint az elvétel (Tétel 3.7.), elvégeztünk egy tesztet, amiben a $Z_{2,2}(3,3)$-ról indítottuk el az előbb bemutatott algoritmust. Az eredmények meggyőzőek lettek, ezt jól szemlélteti, hogy a $Z_{2,2}(3,3)$-tól egészen $Z_{2,2}(13,16)$-ig el lehet jutni ezzel az egyszerű módszerrel úgy, hogy közben csak $Z_{2,2}(m,n)$ pontos értékein haladunk át.



1. ábra. $Z_{2,2}(3,3)$-ból $Z_{2,2}(13,16)$-ba vezető út kizárólag optimális kiegészítésekkel.

Az ábrán látható, a $Z_{2,2}(13,13)$ élű gráf megtalálásához tartozó útvonal. A jelölt értékek mindegyikére sikerült pontos értéket találni a fent bemutatott módszerrel. Most pedig nézzük meg az eredménytáblázatot.

Eredmények			
Tartomány (min,max)	**Pontos illeszkedés**	**Átlagos eltérés**	**Él-lefedettség**
$(10, 20)$	100%	0	100%
$(20, 30)$	96,97%	0,05	99,96%
$(30, 40)$	43,94%	1,68	99,2%
$(10, 40)$	77,62%	0,44	99,64%

6. táblázat. Optimális kiegészítés eredményei.

Mint látható, ez a fejlesztés lendített a legnagyobbat az eredményeken, míg az eddigi módszerekkel a $(20, 30)$ tartományban egyetlen pontos értéket sem sikerült találnunk, addig ez a módszer a tartományon belül található 66 esetből 64-ben beállította az általunk ismert legjobb megoldást.

4.2.6. Optimális csúcselvétel

Sokáig az algoritmus kizárólag az előbb bemutatott kiegészítéses módszert használta fel, kezdetben nem tűnt érdemesnek foglalkozni a másik iránnyal, hiszen a becslés ugyanolyan jól működik mindkét irányban.

A program működéséből kifolyólag, ha találunk egy kiugró eredményt, akkor az a kiegészítések miatt a nagyobb paraméterekre is ráhatással lesz, mivel az (m, n)-re talált gráfnál az $(m, n + 1)$ és $(m + 1, n)$ gráfok élszáma legalább az optimális kiegészítés által hozzáadható élszámmal nagyobb lesz. A másik irányban viszont $(m, n - 1)$ és $(m - 1, n)$-re még semmi nem garantálja, hogy a különbség legfeljebb a nagyobbik gráf megfelelő osztályának minimum fokszáma lesz. Ahhoz, hogy ez a garancia is igaz legyen szükség lesz optimális csúcselvételek futtatására is az eredményhalmazon.

A probléma megoldása külön programként lett implementálva, mivel nem tervezési időben merült fel. A segédprogram felépítése így néz ki:

Beolvassuk az összes eddigi gráfot az eredményhalmazból (a mi esetünkben ez adott (m, n) méret esetén két gráf lesz), majd elmentjük a gráf összes csúcsának fokszámait, és megjegyezzük, mindkét csúcsosztály legkisebb fokú csúcsát. Minden (m, n) paraméterpárra legfeljebb a 10 legjobb **különböző** gráfot mentjük el, az alábbi összehasonlítás szerint: élszám szerint csökkenő, majd a legkisebb összegű sor szerinti növekvő, majd a legkisebb összegű oszlop szerinti növekvő sorrendben. Ha két gráf ezekben a tulajdonságokban megegyezik, akkor csak az elsőnek feldolgozottat fogjuk elmenteni.

6. Algoritmus Gráfok összehasonlítása kódrészlet.

```
bool operator<(const GraphRowColumn& lhs, const GraphRowColumn& rhs)
{
  if (lhs.graph.edges == rhs.graph.edges)
  {
    if (lhs.getMinRowSum() == rhs.getMinRowSum())
      return lhs.getMinColSum() < rhs.getMinColSum();
    else
      return lhs.getMinRowSum() < rhs.getMinRowSum();
  }
  return lhs.graph.edges < rhs.graph.edges;
}
```

Ezután végigmegyünk az elmentett gráfokon a legnagyobbtól a legkisebbig.

Minden feldolgozandó (m, n) méretű gráfból képzünk két kisebb gráfot egyszer az egyik, egyszer a másik csúcsosztályból való legkisebb fokszámú csúcs elvételével. Az így kapott két gráfnak frissítjük a fokszámait, majd hozzáadjuk őket az elmentett gráfokhoz.

Miután az összes gráffal végeztünk, mindegyikhez elmentjük az élszám szerinti két legjobb kapott gráfot, és ezzel az algoritmus végére értünk.

Eredmények			
Tartomány (min,max)	**Pontos illeszkedés**	**Átlagos eltérés**	**Él-lefedettség**
$(10, 20)$	100%	0	100%
$(20, 30)$	98,48%	0,02	99,98%
$(30, 40)$	45,45%	1,67	99,2%
$(10, 40)$	79,84%	0,4	99,67%

7. táblázat. Optimális csúcselvétel eredményei.

Az eredményeket a 4.2.5. fejezetben előbb bemutatott Optimális kiegészítés módszer eredményhalmazán való futtatás adta, a $(10, 40)$ tartományban 17 esetben sikerült javulást elérnie az alábbi módszernek, a javulások minden esetben egy extra élt jelentettek.

Mivel git verziókezelőt használtunk, ezért a főprogramhoz hasonlóan a csúcselvételes algoritmust is külön mappában futtattuk, majd csak azokat a gráfokat mentettük, ahol sikerült javítani az eddig talált élszámon (erről bővebben az 5. fejezetben). Látható, hogy a fent leírt algoritmusban simán lehet, hogy egy olyan csúcsot veszünk el, ami nem csökkenti a másik csúcsosztály minimum fokszámát, egy olyan helyett, ami csökkenthetné azt. Ennek a jelenségnek le is teszteltük, hogy mekkora hatása van az eredményekre. Ehhez az eredményhalmaz, csúcsainak sorrendjét megváltoztattuk a futások előtt (ez megváltoztatja az algoritmus kimenetét, hiszen minden lépésben a legnagyobb indexű minimális fokszámú csúcsot vesszük el), továbbá több iterációt alkalmaztunk, azt találtuk, hogy az esetek $5 - 10\%$-ában kaptunk jobb eredményeket, ezért a jelenlegi algoritmus háromszor megkeveri a szomszédsági mátrixok sorait, és oszlopait, továbbá összesen 30 iterációt végez az

eredményhalmazon minden futásnál. Ennél több iteráció nem hozott további javulást az eredmények szempontjából. A $K_{3,5}$ esetében hozta ez a módszer a legnagyobb javulást eredmények terén. A többi s, t paraméterek esetében is mindig tudott javítani az eredményeken, de nem olyan jelentősen. Az alábbi táblázatok, a javulások mértékét mutatják be, $K_{3,5}$ és $K_{3,3}$ esetében. Látható, hogy $K_{3,5}$-re kiugróan jó eredményeket kaptunk, a $K_{3,3}$-ra kapott adatok pedig inkább az átlagos javulást mutatják be.

$m\backslash n$	17	18	19	20	21	22	23	24	25	26	27	28	29	30
17							1	2	3	4				1
18						2	2	3	2	3				1
19					1	2	2	3	3	4	2	2	2	1
20				1	2	4	4	3	5	8	4	3	4	2
21				2	2	3	3	4	6	9	6	4	6	2
22				3	2	5	4	5	6	8	8	7	8	
23				3	2	5	5	6	6	8	8	7	8	
24				2	3	5	6	7	7	8	8	8	8	
25			1	2	5	6	7	7	6	8	8	9	4	
26				3	7	7	7	7	6	8	10	9	7	
27				2	6	6	6	7	7	9	11	10	10	
28				1	5	6	7	7	8	9	11	10	10	
29				1	6	7	8	8	9	9	11	10	11	
30				1	5	6	7	6	8	9	11	11	12	

8. táblázat. Csúcselvétel javításai $K_{3,5}$-mentes esetben.

$m\backslash n$	17	18	19	20	21	22	23	24	25	26	27	28	29	30
17					1	1	2	2	2					
18						1								1
19					1	1			1					
20				2	2	2	1	1	1					
21					3	2	2	1						
22						2	3	1			1	1	1	
23														
24								2						
25														
26														
27														
28														
29														
30														

9. táblázat. Csúcselvétel javításai $K_{3,3}$-mentes esetben.

Koncentráljunk az érdekesebb esetre, a $K_{3,5}$-re. Látható, hogy az $n = 30$-as oszlopban $m > 21$ esetén az optimális csúcselvétel egyetlen esetben sem vezetett javuláshoz. Ennek oka feltehetően az, hogy a program egy jó konstrukciót talált a $Z_{3,5}(22, 30)$ esetre, és az optimális kiegészítés miatt, ezt a konstrukciót felhasználva eljutott egészen $Z_{3,5}(30, 30)$-ig. Ugyanakkor ezek a struktúrák az $n < 30$ esetekben nem kerültek bemenetként felhasználásra. Ezért történhetett meg, hogy a legjobb $(30, 30)$-as 465-élű konstrukcióból bármelyik A-beli csúcs eltávolításával jobb eredményt kaptunk volna, mint az akkori legjobb $(29, 30)$-as 438-élű konstrukció.

Érdekesség, hogy a $K_{3,6}$-mentes gráfok első iterációjának futtatása során, amikor az eredményeket a $K_{3,5}$-re kapott értékekkel vetettük össze, megfigyelhető volt, hogy míg az $(m, n) = (29, 30)$ paraméterpárra már jobb eredmények születtek, addig az $(m, n) = (30, 30)$ esetén az értékek még elmaradtak.

Ezen megfigyelés vezetett a $K_{3,5}$-re kapott eredmények részletesebb vizsgálatához, amelynek során fény derült a program hiányosságára. Ez a felismerés végül az előzőekben bemutatott algoritmus kidolgozásához vezetett.

5. Program felépítése, programozási megoldások

Az alábbiakban ki fogunk térni a program szerkezetére, tervezésére és a döntések okaira.

A projekt során Git verziókezelő rendszert használtunk, és itt is tároltuk az eredményhalmazt, ennek hátránya az Önálló laboratórium során vált világossá számunkra. Mivel a teljes eredményhalmaz minden futással megváltozott, ezért m, n paraméterenként öt mentett gráffal számolva ez $40 \cdot 40 \cdot 5 = 8000$ fájl folyamatos változását jelentette futásonként. Ez a mennyiség jelentősen lelassította a verziókezelés működését. Az új projektben, ezért változtattunk ezen a módszeren. Minden paraméterhez már csak a két legjobb gráfot mentettük el, a programot pedig külön könyvtárban futtattuk, a kiinduló adatok átmásolását követően. Csak azok a gráfok kerültek elmentésre, melyek javulást hoztak az eddig talált legjobb eredményekhez képest. Ez a megoldás nemcsak a verziókezelés hatékonyságát javította, hanem lehetővé tette a program párhuzamos futtatását is. Legtöbb esetben négy példány futott egyszerre, saját lokális eredményhalmazzal dolgozva.

A programot minden esetben Windows operációs rendszeren, egy Ryzen 5 3600 processzorral futtattuk. A kód `C++` nyelven készült, a `C++20` szabványnak megfelelően. A fordítás során minden esetben engedélyezve volt az `-O3` optimalizáció, így a mért futásidők is ennek figyelembevételével értendőek. A nyelv kiválasztásánál fontos szempont volt az elemi műveletek sebessége, és ebben a `C++` nagyon jó, mivel nagyon hardverközeli nyelv, továbbá nagyon fontos volt az objektumorientáltság támogatása is. Mivel az Önálló laboratórium során alábecsültük a program komplexitását, így az ott alkalmazott funkcionálisabb megközelítés nem bizonyult megfelelőnek, az ötletek, algoritmusok halmazának növekedésével a projekt egyre fenntarthatatlanabbá vált. Ezért a program egy sokkal objektumorientáltabb szemlélettel lett újraírva, szem előtt tartva a bővíthetőséget, és az osztályok közti felelősségek elkülönítését.

A program egyik legfontosabb tervezési kérdése a gráfok tárolásának módja, itt két lehetőség merült fel: a szomszédsági mátrix, vagy az éllistás tárolás. Az éllista előnye, hogy kevesebb a memória igénye, mindössze $O(e)$, viszont ez esetünkben elhanyagolható, mivel legfeljebb 40×40-es gráfokkal dolgozunk, melyek memória igénye igen kicsi, egy 40×40-es gráf maximális memória igénye 32 bites `int` típust használva mindössze $40 \cdot 40 \cdot 32 = 51200$ bit, ami mindössze 6.4 kilobyte, így egy modern rendszer számára teljes mértékben figyelmen kívül hagyható.

Felmerülhet, hogy minek használunk 32 bites adattípust 1 biten is tárolható bináris információk tárolására. Ez a kérdés bennünk is felmerült éppen ezért a kezdeti mohó algoritmus két verzióban is elkészült, egy egyszerű változatban, amely `int`-et használ minden él tárolására és egy bitenként optimalizált változatban, amely `std::bitset` segítségével csupán 1 bitet foglal el élenként. Ezek közül a gyakorlatban a klasszikus `int`-es változat bizonyult a gyorsabb megoldásnak, egyes esetekben akár 20%-os futásidőbeli különbséggel. Ennek az lehetett az oka, hogy a modern processzorok 32 és 64 bites regisztertméretre vannak hangolva, ezért a bitenkénti címzés és a bitmanipulációk több kiegészítő utasítást igényelnek, melyek lassíthatják a számításokat. Állításunkat az [5]-ös cikk is alátámasztja.

Az éllista és a szomszédsági mátrix összevetése során a szomszédsági mátrix bizonyult hatékonyabbnak. Ennek oka feltehetően az, hogy az algoritmus nagymértékben támaszkodik a csúcspárok közötti élek beszúrására és törlésére, amelyek a mátrixos reprezentációban gyorsabban végrehajthatóak. Az összehasonlítás során a $K_{2,2}$-mentes esetet vizsgáltuk

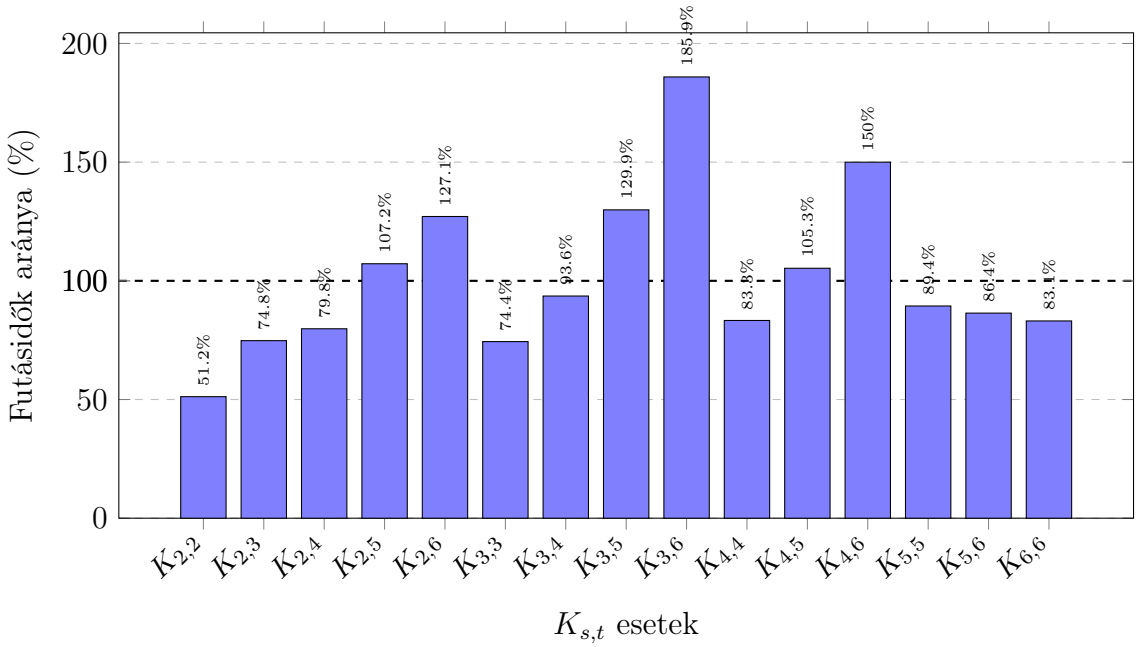
$2 \leq m, n \leq 40$ paraméterek esetén, és ezen futások során a szomszédsági mátrix minden esetben gyorsabbnak bizonyult, mint az éllistát használó verzió.

5.1. $K_{s,t}$ detektálás

A másik kulcsfontosságú kérdés a $K_{s,t}$ detektálás volt, aminek algoritmusát már bemutattuk a 4.1. fejezetben. Viszont a programozási megvalósításáról még nem esett szó. Mivel egy nagyon költséges feladatról van szó, ezért fontos volt egy kellően optimális megoldás találása. A 4.1. ($K_{s,t}$ detektálás) fejezetben két módszer került bemutatásra, az egyik a Pingpong, a másik a Horgony metódus volt. Ezek alapos összehasonlítása kulcsfontosságú feladat volt a különböző módszerek megértéséhez. A Horgony metódus során a kombinációk képzését kezdetben rekurzióval valósítottuk meg, a Pingpong algoritmus, pedig a 2. algoritmusban látható egymásba ágyazott ciklusokkal lett megvalósítva. Az egyszerű tesztelés, cserélhetőség érdekében a KstStore absztrakt ősosztály került bevezetésre. Ez az osztály definiálja azt az interfészt, amelyet minden specifikus $K_{s,t}$ detektáló implementáció megvalósít, így biztosított a konfigurációs fájlból történő egyszerű állíthatóság.

Először a Horgony metódus általános megoldását valósítottuk meg, de ez a módszer a $K_{2,2}$-es esetben lényegesen lassabbnak bizonyult, mint a specifikus fordítási időben ismert ciklusokat használó verzió, pedig ebben az esetben a módszer megegyezik a Pingpong metódussal. Ez adta a motivációt a Pingpong metódus $2 \leq s, t \leq 6$ való paraméterekre való specifikus kiterjesztésére. Ebben a megvalósításban minden paraméterpárhoz külön fordítási időben ismert osztály tartozik, ezek közül futás időben választjuk ki az éppen használandót. Ennek a megközelítésnek az egyik fő előnye, hogy az s, t értékek a leszármazott osztályokban (például K23Store, K44Store) fordítási időben ismert konstansok. Ez lehetővé teszi, hogy a fordító hatékonyabb optimalizációkat alkalmazzon, ami teljesítménynövekedést eredményezhet a futás közben meghatározott paraméterekkel dolgozó megoldásokhoz képest. A két módszer össze is lett hasonlítva kétféleképpen is. Teljes program futásidejének összevetése minden s, t paraméterre az alábbi ábrán látható, az ábrán az Egyik oldal metódus, a Ping-pong metódushoz vett futásidejének viszonyítása látható százalékos formátumban.

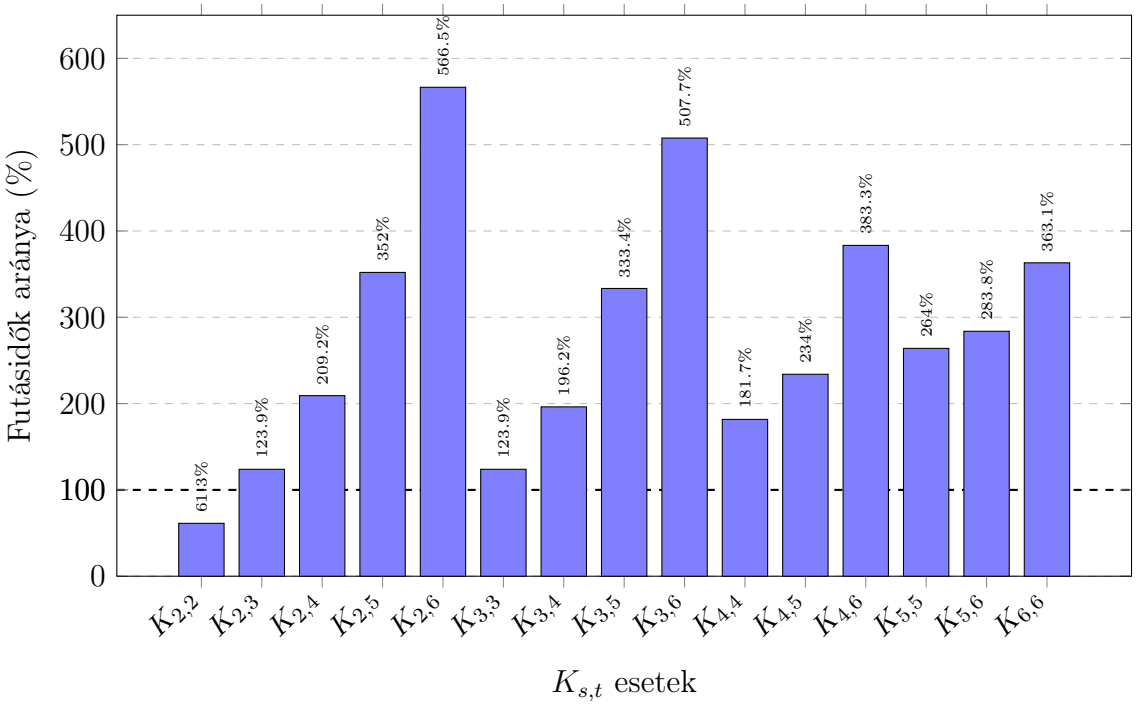
Megjegyzés. Mivel a végleges algoritmust hasonlítottuk össze, így tehát ez a mérés a teljes program gyakorlati futásidejét tartalmazza, nem csak a $K_{s,t}$ detektálásra vonatkozó részidőket.



2. ábra. Pingpong specifikus változata Horgony metódus általános változatához képest (100% = Horgony metódus futásideje).

Ahogy látható, a specifikus Pingpong metódus gyorsabbnak bizonyult a legtöbb esetben, de az aszimmetrikusabb esetekben a Horgony metódus bizonyult a jobb megoldásnak.

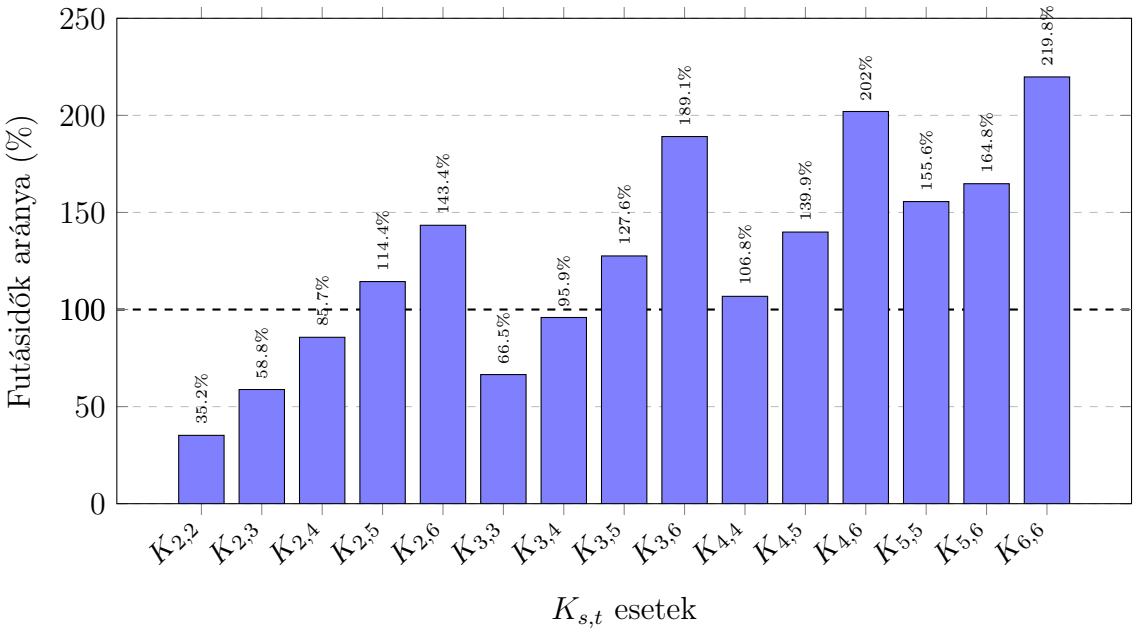
Külön a $K_{s,t}$ detektálásra is futtattunk teszteket, és legnagyobb meglepetésre ebben az esetben már más eredményeket kaptunk. A bemenetet mindig az éppen aktuális legjobb eredményhalmazaink adták, a teszteket minden esetben ugyanazon a gráfhalmazon futtattuk, a fájlból olvasás, és egyéb műveletek nem tartoztak bele a kapott futásidőkbe. Először a $K_{s,t}$-k eltárolását teszteltük le, a bemenet alábbi módosításával: vesszük a legjobb eredményhalmazt, majd a nem behúzott élei közül, véletlenszerűen kiválasztunk $\lfloor m \cdot n \cdot 0.05 \rfloor$ darab élt. A teszt közben ezekre az élekre fogjuk meghívni a $K_{s,t}$ eltároló algoritmust, ezzel szimulálva a programon belüli egy iteráció során végzendő műveleteket. Az első tesztben az eltárolás logika helyett mindkét osztály ugyanazt a mock függvényt hívta, ezért maga az eltárolás logika sebessége nem befolyásolta a teszteredményeket.



3. ábra. Pingpong specifikus változat futásideje eltárolás logika nélkül (100% = Horgony metódus futásideje).

Látható, hogy ebben az esetben nagy meglepetésre a Pingpong metódus a $K_{2,2}$-es esetet leszámítva minden esetben lassabbnak bizonyult. Még a szimmetrikus esetekben is, ahol a program teljes futásidejét vizsgáló mérések során gyorsabb volt. A különbség ráadásul nem elhanyagolható, egyes esetekben a futásidő akár a Horgony metódus futásidejének 5,5-szöröse volt.

Ennek megértése érdekében a ugyanez a teszt ugyanezen paraméterekkel is lefuttatásra került, annyi különbséggel, hogy a mock függvény helyett mindkét módszer eltárolta a futás során képződött $K_{s,t}$-ket.

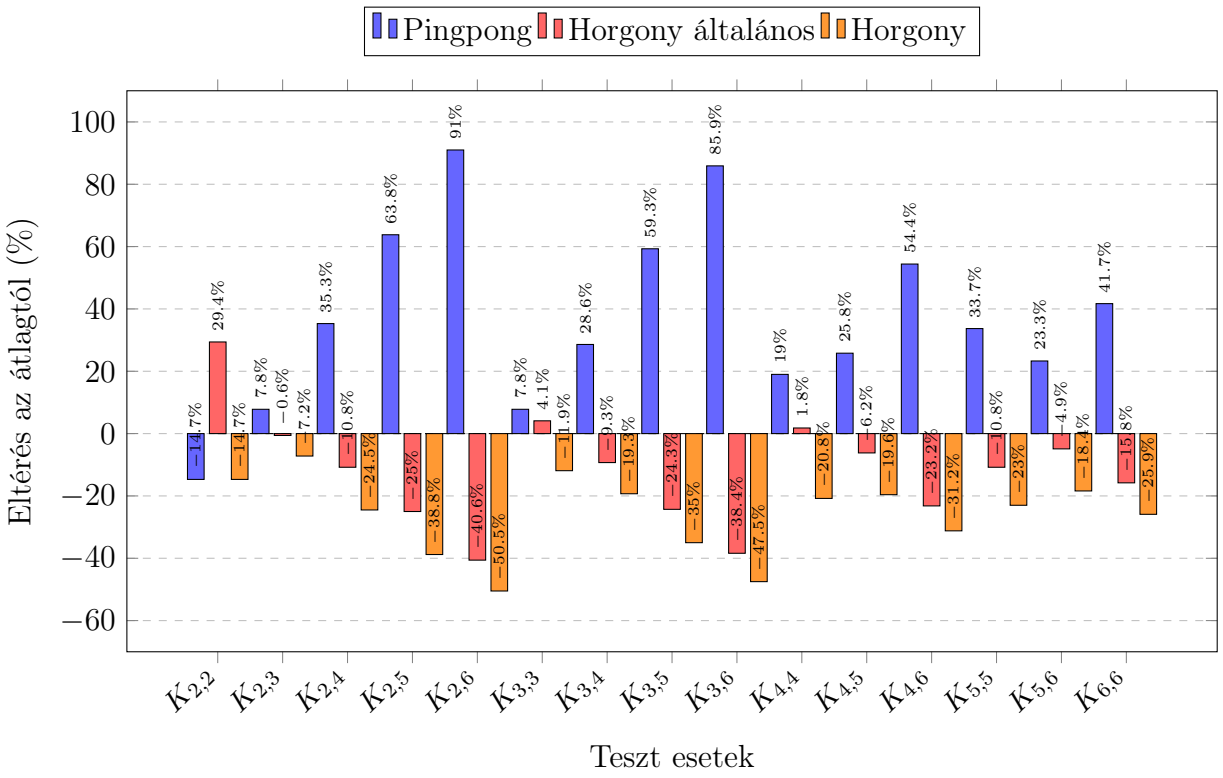


4. ábra. Pingpong specifikus változat futásideje eltárolás logikával (100% = Horgony metódus futásideje).

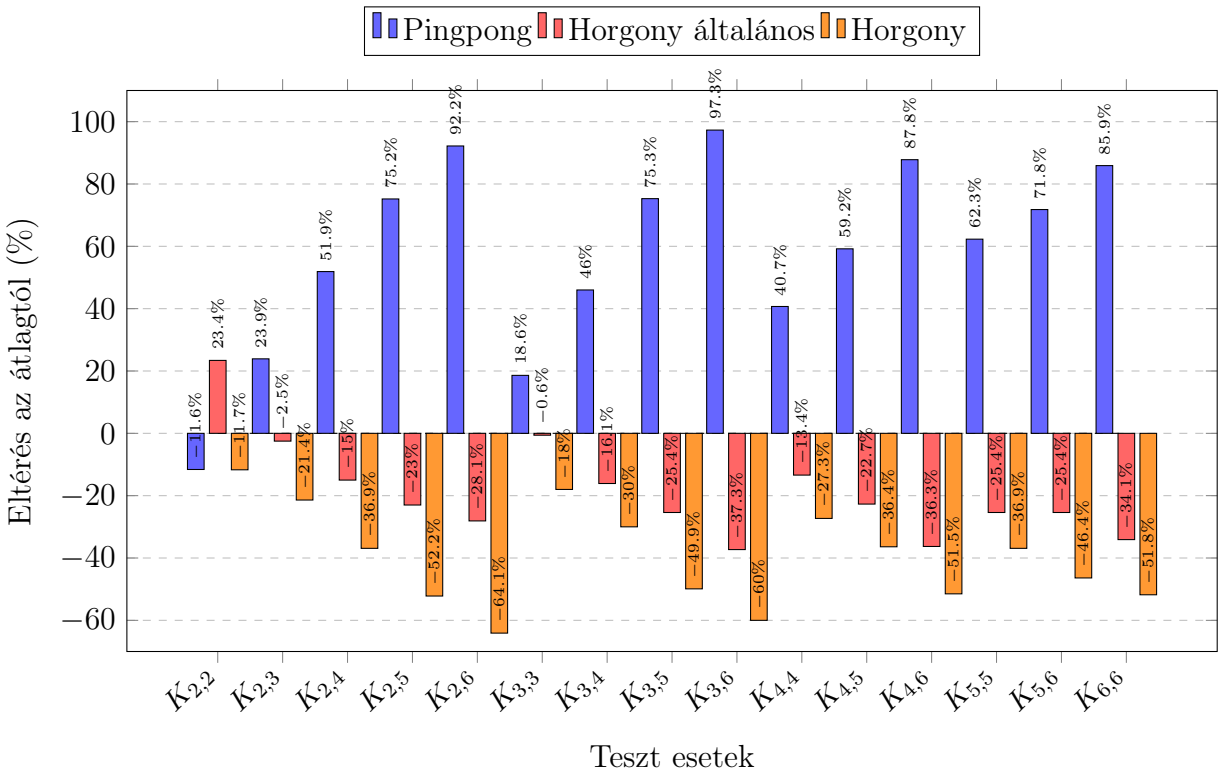
Látható, hogy a futásidőbeli különbségek többé kevésbé lefeleződtek, a Pingpong metódus alig érzékelhető lassulást mutatott a korábbi változathoz képest, ellenben a Horgony metódus, esetében a futásidők majdhogynem kétszeresére nőttek. Ezekből az eredményekből kiindulva, arra a következtetésre jutottunk, hogy a $K_{s,t}$-k létrehozáshoz szükséges dinamikus tárolók (esetünkben `std::vector`) létrehozása, esetén az overhead kifejezetten nagy. Ennek eredményeként a teljesen futásidőben változtatható, változatot elvetettük, és ehelyett két új módszert valósítottunk meg.

Az első módszer az egymásba ágyazott ciklusokat használó Pingpong metódusból ismert, minden $K_{s,t}$-re specifikus osztály megvalósítása volt, a Horgony algoritmust használva. Fontos megjegyezni, hogy a $K_{2,2}$ esetben a Pingpong és Horgony metódusok megegyeznek, így ebben az esetben mindkét specifikus változat ugyanazt az osztályt használja, ezért a később bemutatott eredmények is az egyéb külső behatásoktól eltekintve többé kevésbé azonosak lesznek.

A másik megvalósítás pedig ennek egy fél-generikus változata, a `Constants` osztály által meghatározott $s \leq$ `MAX_S` és $t \leq$ `MAX_T` paraméterekre. Ha nagyobb paraméterekre szeretnénk futtatni a programot, akkor nem kell legenerálnunk a specifikus osztályokat, hanem elég a `Constants` namespace-ben átírnunk ezt a két paramétert, és újrafordítás után futtathatjuk is a nagyobb paraméterekkel dolgozó programot, a dinamikus tárolók okozta futásidő növekedés nélkül. Amennyiben ezt nem tesszük meg, és a konfigurációs fájlban olyan s, t értékek találhatóak, melyek nem felelnek meg a `Constants` osztályban meghatározott értékeknek, a program „unsupported $K_{s,t}$" hibaüzenettel leáll. A fél-generikus változatban a kombinációk a teljesen generikus változathoz hasonlóan rekurzióval vannak implementálva, a specifikus megvalósítás pedig egymásba ágyazott ciklusokat használ ezek előállítására. Az alábbiakban ezen módszerek tesztjei során kapott eredményeket mutatjuk be, hogy ezzel jobban megérthessük a futásidőbeli különbségeket.



5. ábra. Mérési eredmények összehasonlítása az átlaghoz képest ($K_{s,t}$ detektálás olyan éleken, melyek nem hoznak létre $K_{s,t}$-t).



6. ábra. Mérési eredmények összehasonlítása az átlaghoz képest ($K_{s,t}$ eltárolás $\lfloor m \cdot n \cdot 0.05 \rfloor$ hozzáadott élre, melyek mindegyike létrehoz $K_{s,t}$-t).

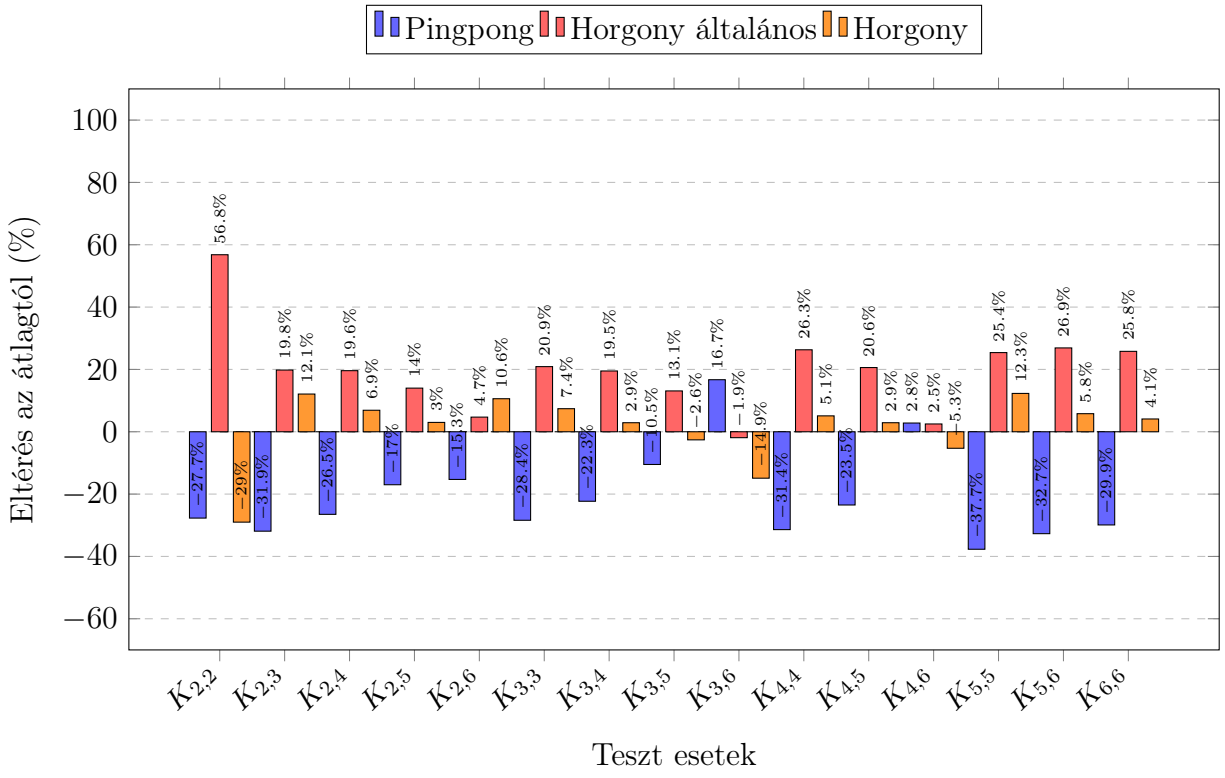
Mint látható a 6. ábrán, a Horgony metódus új megvalósításaiban a $K_{s,t}$-k eltárolása, már nem jelent problémát, éppen ezért az 5. és 6. ábra eredményei nagyon hasonlók. Megfigyelhető a Horgony metódus két megvalósításának viszonyából, hogy a $K_{s,t}$-k eltárolása esetén a rekurzióval képzett kombinációk miatt a futásidőbeli különbség tovább növekszik a módszer generikus és specifikus megvalósításai közt.

A Horgony algoritmus elméleti lépésszáma $K_{s,t}$-t tartalmazó gráfokon $O(m^{s-1}n^{t-1})$, amely lényegesen rosszabb, mint a $K_{s,t}$-mentes eset $O(m^{s-1}n)$ lépésszáma (hiszen $K_{s,t}$-mentes gráfokra a belső n elemű ciklus megtétele után nem kell $(t-1)$ elemű kombinációkat gyártanunk). Ezzel szemben az 5. és a 6. ábrákon szemléltetett futásidőkben nem tapasztaltunk lényegi változást. Ennek oka az lehet, hogy kevés $K_{s,t}$ esetében a belső ciklus során kevés $(t-1)$ elemű kombinációt kell gyártanunk, így a gyakorlatban ez tekinthető $O(n)$ lépésnek.

Megjegyzés. A program futása során egy iteráción belül szinte mindig kevesebb élt veszünk hozzá, mint a 6. ábrán bemutatott esetben.

Itt még azt lehet érdekes megfigyelni, hogy az aszimmetrikus esetekben láthatóan sokkal jobban teljesít a Horgony algoritmus, ennek az lehet az oka, hogy ebben az esetben különösen megéri a kisebb osztály kombinációival kezdeni.

Most, hogy láttuk, hogy a Horgony algoritmus szinte minden esetben jobban teljesít a Pingpong metódusnál nézzünk egy olyan esetet, amikor ez nem igaz.

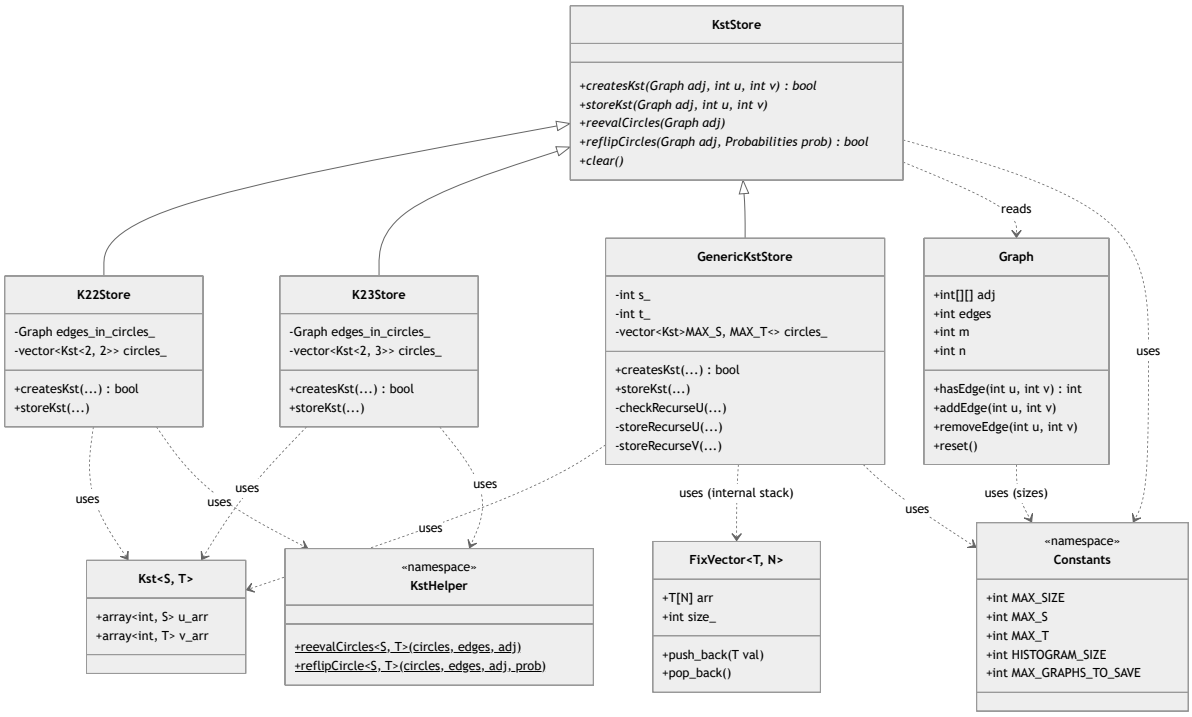


7. ábra. Mérési eredmények összehasonlítása az átlaghoz képest ($K_{s,t}$ detektálás eltárolás nélkül (mohó algoritmus) $\lfloor m \cdot n \cdot 0.05 \rfloor$ hozzáadott élre, melyek mindegyike létrehoz $K_{s,t}$-t).

Ahogy látható, két nagyon aszimmetrikus esetet kivéve minden esetben a Pingpong algoritmus adta a jobb futásidőket. Nézzük meg, hogy mi is történhetett: a program alapvetően

minden élbehúzáskor a $K_{s,t}$ eltároló algoritmust futtatja, hiszen az iteráció végeztével ki kell, hogy tudjuk törölni a gráfból a $K_{s,t}$-ket. Egyetlen esetben nincs szükségünk a $K_{s,t}$-k eltárolására, mégpedig a 4.2.1. fejezetben bemutatott mohó algoritmus során. Hiszen itt azokat az éleket, melyek $K_{s,t}$-t képeznének sosem húzzuk be. A mohó algoritmust minden programváltozat használja a belső iteráció(k) végeztével az eredménygráfok összes behúzatlan élére. A Pingpong metódus azért adhatott ilyen jó eredményeket erre a feladatra, mivel semmilyen előfeldolgozást nem igényel, a mohó fázisban pedig a találatok tárolásának szükségtelensége miatt a keresés az első tiltott részgráf megtalálásakor azonnal megszakítható. Ennek következtében szerencsés esetben a beágyazott ciklusoknak csupán az első néhány iterációja fut le, így a legjobb eset lépésszáma mindössze $\Omega(s+t)$. Ellenben a Horgony módszer esetén a best case lépésszám $\Omega(m+s+n)$ lépés, ennyi lépés alatt pedig a Pingpong metódus már a kisebb indexekre több kombinációt is megvizsgálhat.

Ezen futások eredményeképpen a program a $K_{s,t}$-k eltárolásának algoritmusához minden esetben a Horgony módszer adott s,t paraméterekre specifikus osztályát használja, amennyiben megvan valósítva, ellenkező esetben a fél-generikus változatot. $K_{s,t}$ detektáláshoz pedig minden esetben a Pingpong algoritmust, kivéve a $K_{3,6}$-os esetben, mivel ekkor a Horgony megközelítés jobb eredményeket adott.



8. ábra. Osztály diagram a program $K_{s,t}$ detektáló részéről.

Az ábra a program $K_{s,t}$-detektálásért, megszüntetésért felelős osztályokat mutatja be.

Láthatóak a `KstStore` ősosztály előbb bemutatott megvalósításai: a `K22Store` és `K23Store` osztályok, melyek a Horgony metódus specifikus változatai, illetve a `GenericKstStore`, amely a módszer fél-generikus megvalósítása.

A `Kst` osztály egy konkrét $K_{s,t}$ részgráfot reprezentál, míg a `FixVector` segédosztály a

rekurzív keresések során használt, rögzített méretű ideiglenes tároló. A `KstHelper` névtér olyan segédfüggvényeket tartalmaz, melyek a $K_{s,t}$-k kiritkítására szolgálnak, így ezeket a függvényeket a leszármazott osztályoknak nem kell implementálniuk, használhatják ezt is. A `KstHelper`-ben található függvények a 4.2.2. fejezetben bemutatott módszert valósítják meg a $K_{s,t}$-k megszüntetésére.

A `Constants` névtérben vannak definiálva a különböző megengedett fordítási időben ismert paraméter maximumok.

5.2. Valószínűségi módszerek

A program eredményessége szempontjából kritikus tényező a 4.2.3.-as fejezetben tárgyalt, dinamikusan változó élbehúzási valószínűségek. Mivel a generált gráfok topológiáját döntően ez a valószínűségi függvény határozza meg, ennek finomhangolásával jelentős javulás érhető el. Az Önálló labor során szerzett tapasztalatok rávilágítottak arra, hogy a valószínűségi modellek egyszerű cserélhetősége alapvető architekturális követelmény. Ennek megfelelően a kód újratervezésekor kiemelt figyelmet kapott a modularitás és a bővíthetőség.

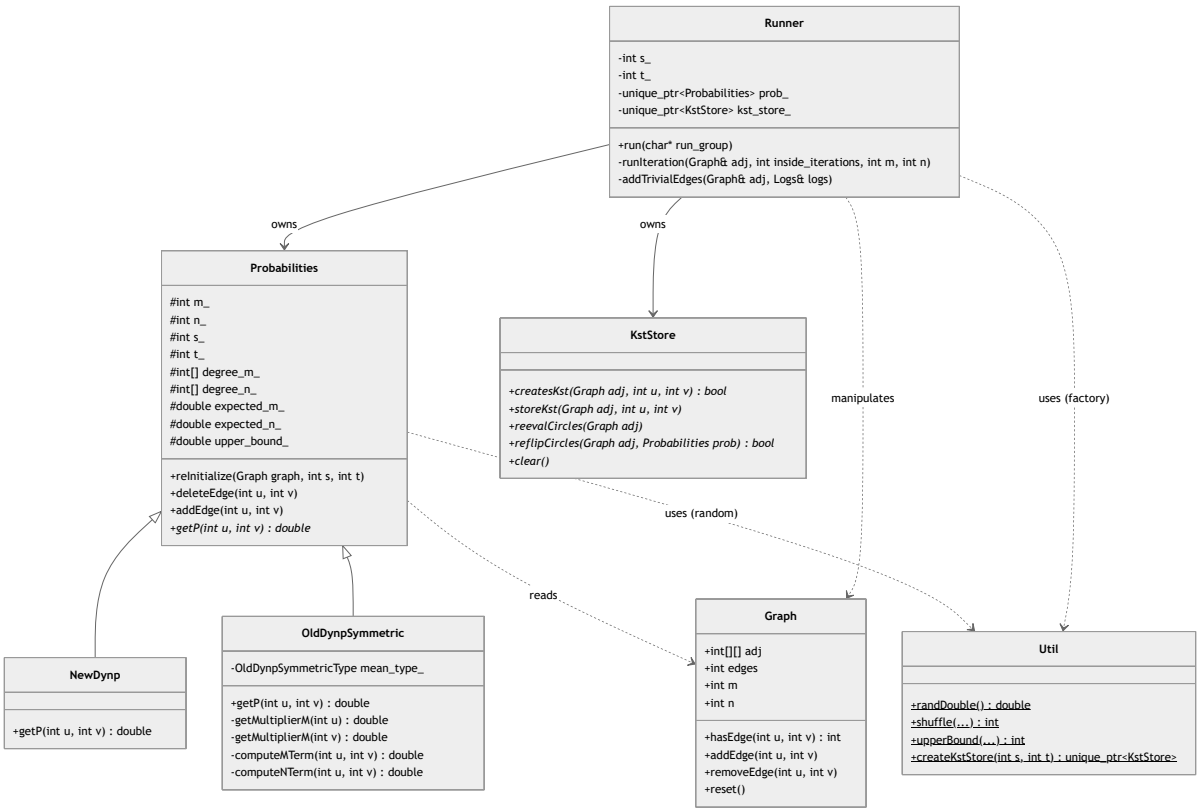
E cél elérése érdekében bevezetésre került a `Probabilities` absztrakt ősosztály, ami egységes interfészt biztosít a különböző stratégiák számára, lehetővé téve a polimorf viselkedést. Ez lehetővé teszi, hogy a `Runner` anélkül használja a különböző stratégiákat, hogy ismernie kellene azok belső működését. A tervezési minta egyik fő előnye a matematikai logika és a gráfépítés algoritmikus folyamata közötti teljes szeparáció. A `Probabilities` ősosztály egy tisztán virtuális `getP(u, v)` függvényt definiál, amely a két csúcs között behúzandó él valószínűségét határozza meg. Emellett az osztály egységesen kezeli az olyan közös állapotinformációkat, mint a csúcsok foka vagy a várható élszámok. Ezeket a `deleteEdge` és `addEdge` virtuális metódusok automatikusan karbantartják, így a leszármazott osztályoknak kizárólag a specifikus matematikai logika implementálására kell fókuszálniuk.

Az alábbi kódrészlet a rendszer bővíthetőségét mutatja be.

7. Algoritmus Új valószínűségi algoritmus létrehozása kódrészlet.

```
#pragma once

#include "probability/Probabilities.h"

class NewDynp : public Probabilities
{
 public:
  using Probabilities::Probabilities;

  double get_p(int u, int v) override
  {
  double expected_p = (upper_bound_ - edge_count_) / (m_ * n_ - edge_count_);

  double prob_A = (expected_m_ - degree_m_[u]) / (n_ - degree_m_[u]);
  double prob_B = (expected_n_ - degree_n_[v]) / (m_ - degree_n_[v]);

  double term_A = (expected_p + 2.0 * prob_A) / 3.0;
  double term_B = (expected_p + 2.0 * prob_B) / 3.0;

  return (term_A + term_B) / 2.0;
  }
};
```

A felépítés jelentős rugalmasságot biztosít, új heurisztikák kipróbálásához elegendő egy új osztályt származtatni a `Probabilities`-ből, és implementálni benne a `getP` metódust. A konfigurációs fájlban történő egyszerű átállítást követően a program már az új stratégiát használja. Ennek köszönhetően a különböző megközelítések összehasonlítása, kipróbálása sokkal egyszerűbben megtehető.



9. ábra. Osztály diagram a program központi részéről.

Az ábrán látható a `Runner` osztály, amely az algoritmus futtatását végzi, továbbá meghívja a `Probabilities` ősosztály `addEdge`, `deleteEdge` függvényeit a gráf minden egyes élének módosításakor, ez biztosítja azt, hogy egy él valószínűségének a lekérdezésekor az éppen használt módszer ismerje a csúcsok fokszámait, ne kelljen végigiterálni a szomszédsági mátrixon.

5.3. Egyéb támogató osztályok

A program matematikai magjának hatékony működéséhez elengedhetetlen egy támogató környezet, amely a konfigurációt, az adatok perzisztenciáját és a futás monitorozását végzi.

A rendszer központi eleme a `Config` osztály, amely Singleton tervezési mintát valósít meg. Ez biztosítja, hogy a futtatási paraméterek a program bármely pontján elérhetőek legyenek anélkül, hogy azokat paraméterként át kellene adni az objektumok között. A konfigurációs `.conf` kiterjesztésű fájlok betöltését a `ConfigLoader` osztály végzi. Ezek a fájlok gyakorlatban egyszerű szövegfájlok, melyek bizonyos paramétereket állítanak be. Például az `iterations = 20000` sorral adhatjuk meg, hogy adott m, n paraméterekre mennyi iteráció legyen, a `min = 2` és `max = 40` sorokkal pedig az (m, n) futási tartomány határozható meg.

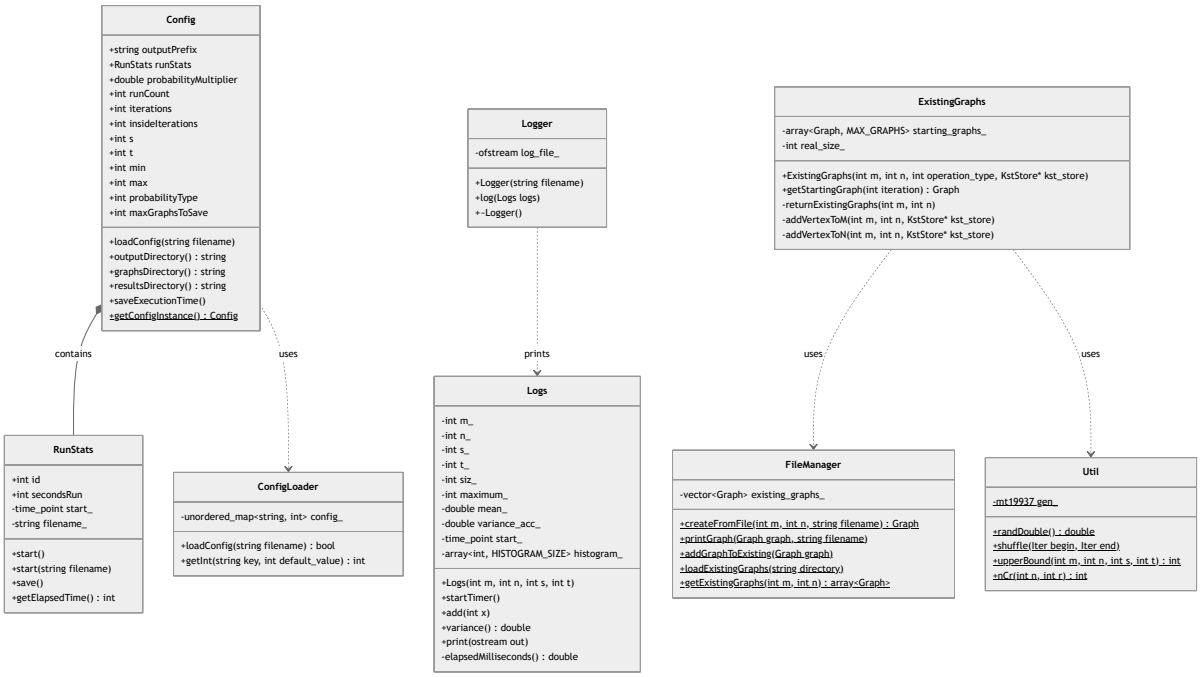
1. Konfiguráció Példa konfigurációs fájl bemeneti paraméterekkel.

```
// config.conf
runCount = 10
iterations = 100
insideIterations = 2 // comment
s = 3
t = 6
min = 3
max = 30
maxGraphsToSave = 2
// probabilityType = default
```

Az adatkezelésért a `FileManager` osztály felel, amely a gráfok fizikai tárolását végzi, elkülönítve ezen feladatokat az algoritmustól. Ezzel szorosan együttműködik az `ExistingGraphs` osztály, amely a 4.2.5. fejezetben (Optimális kiegészítés) bemutatott logikát valósítja meg, így a központi logikából egy függvényhívással megkapható a kiinduló gráf.

A futások kiértékelhetőségét a `Logger` és a `RunStats` osztályok biztosítják. A `Logger` a futás közbeni állapotokat és a gráfok statisztikai jellemzőit rögzíti. A `RunStats` két dolgot rögzít a futás hosszát másodpercben, és az adott futás azonosítóját. A program minden indulásnál beolvassa, majd kilépésnél frissíti ezeket az adatokat. Ezáltal minden futáshoz azonosító alapján vissza kereshetőek a log-ok és eredmények (ezeket általában nem verzióztam).

Emellett a projekt során még sok kisebb segéd program készült, erre jó példa a 3.7. fejezetben említett optimális csúcselvétel. Ez a módszer nem illeszkedik közvetlenül az algoritmus alapstruktúrájába, ezért külön segédprogramként került implementálásra. Az eredmények helyességének ellenőrzését is külön program valósítja meg.



10. ábra. Osztály diagram a program infrastrukturális részéről.

Az ábrán program infrastrukturális komponensei láthatóak, amelyek a konfiguráció, az adatok és a futások nyomon követését biztosítják. Ez a komponens biztosítja az algoritmus matematikai magjának átláthatóbb működését, miközben a kiinduló gráfok és a futási eredmények kezelése elkülönül a fő logikától.

6. Eredmények

Ebben a fejezetben be fogjuk mutatni a kapott eredményeket, mivel a $K_{2,2}$-mentes eset a leginkább kutatott probléma, és mi is ezzel foglalkoztunk a legtöbbet, ezért az eredmények összehasonlításában először kifejezetten a $Z_{2,2}(m,n)$-re kapott eredményeket fogjuk összevetni a szakirodalommal, majd külön fejezetben kitérünk az $s = t$ esetre. Az $s \leq t$ paraméterekre kapott táblázatok, pedig a függelékben lesznek felsorolva, mivel ezekre kevesebb szakirodalmi eredmény található.

Az alább bemutatott táblázatok, mind Zarankiewicz-számok alsó becsléseit tartalmazzák. A félkövérrel jelölt elemek pontos értéket reprezentálnak. A programunk által újonnan felfedezett, eddig nem ismert pontos értékeket (beleértve a szakirodalom által korábban nem vizsgált eseteket is) bekeretezéssel jelöltük.

Azokat az értékeket, amelyeknél a program nem érte el az ismert alsó becslést, aláhúzással jelöltük (ilyen eset kevés van, mindössze 4 darab az összes táblázatban). Az értékeket, amelyeknél a program nem érte el az ismert felső becslést, formázatlanul hagytuk.

Az összehasonlításhoz használt felső becsléseket a szakirodalomból vettük, ha az adott paraméterekre nem találtunk felső becslést, mindig a Roman-féle felső becslést használtuk.

A szakirodalomban az alábbi hibákat találtuk, ezeknek javítását a táblázatunkban az előbb bemutatott jelölésrendszer mellett csillaggal jelöltük. Collins és tsai. [2] által közölt táblázatokban összesen négy hibát találtunk, a $Z_{6,6}(10,10)$ esetre 95 szerepelt pontos értékként, ami nagyobb, mint a Roman-féle felső becslés (91). A programunk talált 91 élű konstrukciót, így a táblázatba ez pontos értékként került be megcsillagozva. A további három hiba, pedig a $K_{5,5}$-mentes táblázatban volt, $(7,16)$, $(7,17)$, $(7,18)$ esetekben, a cikkben a Roman-féle felső becslésből kapott érték eggyel javított változata került be pontos értékként. A programunk mindhárom esetben talált ezeknél eggyel nagyobb élszámú, így már a Roman-becsléssel megegyező konstrukciót, így ezek is pontos értékekként megcsillagozva kerültek be a végső táblázatunkba.

Ezek mellett még egy elgépelést találtunk. Ez a Davies–Gill–Horsley [4] cikkben volt, a $K_{4,5}$-mentes esetben publikált eredmények $m = 15$ sorában az értékek nagyságrendekkel nagyobbak voltak, mint az ismert felső becslések, ezeket az értékeket nem jelöltük külön a táblázatunkban, mivel biztosan elgépelésről volt szó.

6.1. $Z_{2,2}(m,n)$-re kapott eredmények

Jelenleg ismert pontos értékeket *Kay* [11] cikkéből vettük. A cikk a $2 \leq m \leq 23$ és $2 \leq n \leq 38$ eseteket vizsgálja, $Z_{2,2}(23,23)$-ig minden esetben és a teljes tartományban is jó pár esetben pontos értéket adva. Frissebb felső becsléseket *Tan* [20] cikke tartalmaz, ahol $Z_{2,2}(m,n)$ értékei $2 \leq m \leq 24$ és $2 \leq n \leq 35$ tartományban szerepelnek. A táblázat $m \leq 20$ és $n \leq 26$ esetén minden értéket pontosan közöl, és több aszimmetrikus párra is kiterjed ezen határokon túl. A főátlót, pedig a *Collins és tsai.* [2] által közölt táblázatból vettük, ahol egészen $(31,31)$-ig pontos értékeket kapunk $m = n$-re.

Az általunk alkalmazott módszerekkel előállított táblázat minden eddig ismert alsó becslést megtalált, továbbá összesen 81 eddig nem ismert esetben sikerült beállítanunk a legjobb

ismert felső becsléseket (bár ezek egy része a mások által nem vizsgált tartományból származik).

A $K_{2,2}$-es esetre fordított teljes futásidő 67,22 óra volt. Megjegyzendő azonban, hogy a program már körülbelül fél óra futtatás után elérte a szakirodalomban ismert alsó becslések nagy részét, és csupán 9 esetben adott azoknál gyengébb eredményt.

$m\backslash n$	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**
3		**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**
4			**9**	**10**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**
5				**12**	**14**	**15**	**17**	**18**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**
6					**16**	**18**	**19**	**21**	**22**	**24**	**25**	**27**	**28**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**
7						**21**	**22**	**24**	**25**	**27**	**28**	**30**	**31**	**33**	**34**	**36**	**37**	**39**	**40**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**
8							**24**	**26**	**28**	**30**	**32**	**33**	**35**	**36**	**38**	**39**	**41**	**42**	**44**	**45**	**47**	**48**	**50**	**51**	**53**	**54**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**	**64**	**65**	**66**	**67**	**68**
9								**29**	**31**	**33**	**36**	**37**	**39**	**40**	**42**	**43**	**45**	**46**	**48**	**49**	**51**	**52**	**54**	**55**	**57**	**58**	**60**	**61**	**63**	**64**	**66**	**67**	**69**	**70**	**72**	**73**	**74**	**75**	**76**
10									**34**	**36**	**39**	**40**	**42**	**44**	**46**	**47**	**49**	**51**	**52**	**54**	**55**	**57**	**58**	**60**	**61**	**63**	**64**	**66**	**67**	**69**	**70**	**72**	**73**	**75**	**76**	**78**	**79**	**81**	**82**
11										**39**	**42**	**44**	**45**	**47**	**50**	**51**	**53**	**55**	**57**	**59**	**60**	**62**	**63**	**65**	**66**	**68**	**69**	**71**	**72**	**74**	**75**	**77**	**78**	**80**	**81**	**83**	**84**	**86**	**87**
12											**45**	**48**	**49**	**51**	**53**	**55**	**57**	**60**	**61**	**63**	**65**	**66**	**68**	**70**	**72**	**73**	**75**	**76**	**78**	**79**	**81**	**82**	**84**	**85**	**87**	**88**	**90**	**91**	**93**
13												**52**	**53**	**55**	**57**	**59**	**61**	**64**	**66**	**67**	**69**	**71**	**73**	**75**	**78**	**79**	**81**	**82**	**84**	**85**	**87**	**88**	**90**	**91**	**93**	**94**	**96**	**97**	**99**
14													**56**	**58**	**60**	**63**	**65**	**68**	**70**	**72**	**73**	**75**	**78**	**80**	**82**	**84**	**86**	**87**	**89**	**91**	**92**	**94**	**96**	**98**	**99**	**101**	**102**	**104**	**105**
15														**61**	**64**	**67**	**69**	**72**	**75**	**77**	**78**	**80**	**82**	**85**	**86**	**88**	**91**	**93**	**95**	**96**	**98**	**100**	**102**	**105**	**106**	**108**	**109**	**111**	**112**
16															**67**	**70**	**73**	**76**	**80**	**81**	**83**	**85**	**87**	**90**	**91**	**93**	**96**	**98**	**100**	**102**	**103**	**106**	**108**	**110**	**112**	113	115	117	118
17																**74**	**77**	**80**	**84**	**85**	**87**	**89**	**91**	**94**	**96**	**98**	**101**	**102**	**105**	**107**	**109**	**111**	**113**	**115**	**117**	**119**	**121**	**123**	**125**
18																	**81**	**84**	**88**	**90**	**91**	**93**	**96**	**99**	**101**	**103**	**106**	108	110	112	114	116	118	**121**	**123**	**125**	**127**	**129**	**131**
19																		**88**	**92**	**95**	**96**	**98**	**100**	**103**	**106**	**108**	**111**	**114**	115	117	119	121	124	**127**	**129**	**131**	132	134	**137**
20																			**96**	**100**	**101**	**103**	**105**	**108**	**111**	**113**	**116**	**120**	121	123	125	127	129	132	**135**	136	138	**141**	**143**
21																				**105**	**106**	**108**	**110**	112	**116**	**118**	**121**	**125**	126	128	130	132	135	138	141	142	144	147	149
22																					**108**	**110**	**114**	117	120	**123**	**126**	**130**	**132**	133	136	138	140	143	**147**	148	150	153	155
23																						**115**	**118**	121	125	**128**	**131**	**135**	**138**	139	141	143	146	148	151	154	156	159	161
24																							**122**	126	129	133	136	140	**144**	145	147	149	151	154	156	159	162	165	168
25																								**130**	134	138	141	145	**150**	151	153	155	157	160	162	164	167	171	**175**
26																									**138**	142	146	150	**155**	156	158	161	163	165	168	170	172	176	180
27																										**147**	151	155	**160**	162	163	166	168	170	173	175	178	181	185
28																											**156**	160	**165**	**168**	169	171	174	176	178	181	183	187	191
29																												**165**	170	**174**	175	177	180	182	184	187	189	192	196
30																													**175**	**180**	181	183	186	188	190	193	195	198	202
31																														**186**	187	189	192	194	196	199	201	204	207
32																															189	191	194	196	200	203	206	208	212
33																																195	198	202	204	207	210	213	217
34																																	204	208	210	213	216	219	222
35																																		213	216	219	222	225	228
36																																			221	224	227	231	234
37																																				228	232	235	238
38																																					237	241	244
39																																						245	248
40																																							253

10. táblázat. $Z_{2,2}(m,n)$ alsó becslések

6.2. Nagyobb szimmetrikus paraméterekre kapott eredmények

A $K_{3,3}$-mentes esetben a $2 \leq m \leq 16$ és $2 \leq n \leq 23$ tartományban találtunk pontos értékeket, az összehasonlításhoz a $K_{2,2}$-es esetben is használt [2, 11, 20] cikkek eredményeinek unióját vettük alapul, továbbá a 2024-ben megjelent *Davies–Gill–Horsley* [4] cikket.

A program a $Z_{3,3}(16,16)$-os eseten kívül minden esetben beállította az ismert alsó becsléseket, továbbá 9 alkalommal a szakirodalomban vizsgált tartományon belül az elméleti felső becslést is elérte, ezzel javítva a korábbi eredményeken és igazolva a pontos értéket. A $K_{3,3}$-mentes esetre fordított futásidő összesen 19,03 óra volt.

$m\backslash n$	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
3	**8**	**10**	**12**	**14**	**16**	**18**	**20**	**22**	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**	**42**	**44**	**46**	**48**	**50**	**52**	**54**	**56**	**58**	**60**	**62**
4		**13**	**16**	**18**	**21**	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**	**42**	**44**	**46**	**48**	**50**	**52**	**54**	**56**	**58**	**60**	**62**	**64**	**66**	**68**
5			**20**	**22**	**25**	**28**	**30**	**33**	**36**	**38**	**41**	**44**	**46**	**49**	**52**	**54**	**57**	**60**	**62**	**64**	**66**	**68**	**70**	**72**	**74**	**76**	**78**	**80**
6				**26**	**29**	**32**	**36**	**39**	**42**	**45**	**48**	**50**	**53**	**56**	**58**	**61**	**64**	**66**	**69**	**72**	**74**	**77**	**80**	**82**	**85**	**88**	**90**	**93**
7					**33**	**37**	**40**	**44**	**47**	**50**	**53**	**56**	**60**	**63**	**66**	**69**	**72**	**75**	**78**	**81**	**84**	**87**	**90**	**92**	**95**	**98**	**100**	**103**
8						**42**	**45**	**50**	**53**	**57**	**60**	**64**	**67**	**70**	**74**	**77**	**81**	**84**	**87**	**90**	**94**	97	100	104	**108**	**112**	**114**	**117**
9							**49**	**54**	**59**	**64**	**67**	**70**	**73**	**77**	**81**	**85**	**89**	**93**	**96**	**100**	103	106	109	112	116	120	123	127
10								**60**	**64**	**68**	**73**	**77**	**81**	**85**	**90**	**94**	**98**	**102**	106	110	112	116	119	123	126	130	133	137
11									**69**	**74**	**80**	**84**	**88**	**92**	**96**	**101**	106	111	**116**	**121**	123	127	130	133	137	141	144	148
12										**80**	**86**	**91**	**96**	**99**	**103**	108	114	120	126	**132**	134	138	141	144	148	152	155	159
13											**92**	**98**	**104**	**107**	**110**	**116**	119	125	131	137	140	145	148	153	156	161	165	169
14												**105**	**112**	**115**	**118**	**124**	127	131	136	142	146	152	156	161	165	169	174	178
15													**120**	**123**	**126**	**132**	135	139	144	150	156	160	164	169	173	178	183	188
16														127	132	137	141	147	152	158	163	168	172	177	181	187	192	196
17															138	143	149	155	161	167	171	176	181	186	191	196	201	206
18																150	156	162	168	175	180	185	191	196	201	206	211	216
19																	164	170	177	183	189	194	201	206	211	216	221	226
20																		178	185	192	198	203	210	215	221	226	231	236
21																			193	200	207	212	218	224	230	235	240	245
22																				208	216	220	227	233	239	244	250	255
23																					225	230	236	242	249	254	260	265
24																						237	244	251	258	263	269	275
25																							252	260	268	273	279	284
26																								268	276	282	288	294
27																									285	291	297	303
28																										298	305	311
29																											313	320
30																												327

11. táblázat. $Z_{3,3}(m,n)$ alsó becslések

A $K_{4,4}$-mentes esetben is az előbb használt ([2, 11, 20, 4] cikkek adták a referencia adatokat. A program erre az esetre összesen 24,31 órát futott, és az összes ismert alsó becslést beállította.

$m\backslash n$	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
4	**15**	**18**	**21**	**24**	**27**	**30**	**33**	**36**	**39**	**42**	**45**	**48**	**51**	**54**	**57**	**60**	**63**	**66**	**69**	**72**	**75**	**78**	**81**	**84**	**87**	**90**	**93**
5		**22**	**26**	**30**	**33**	**37**	**41**	**45**	**48**	**52**	**56**	**60**	**63**	**66**	**69**	**72**	**75**	**78**	**81**	**84**	**87**	**90**	**93**	**96**	**99**	**102**	**105**
6			**31**	**36**	**39**	**43**	**47**	**51**	**55**	**59**	**63**	**67**	**71**	**75**	**78**	**82**	**86**	**90**	**93**	**97**	**101**	**105**	**108**	**112**	**116**	**120**	**123**
7				**42**	**45**	**49**	**54**	**58**	**63**	**68**	**72**	**77**	**82**	**87**	**90**	**95**	**100**	**105**	**108**	**112**	**116**	**120**	**123**	**127**	**131**	**135**	**138**
8					**51**	**55**	**60**	**65**	**70**	**75**	**80**	**85**	**90**	**95**	**99**	**104**	109	114	118	123	127	132	136	141	145	150	154
9						**61**	**67**	**72**	**78**	**84**	**88**	**94**	**99**	**104**	**109**	114	120	126	130	135	139	144	149	153	158	163	168
10							**74**	**79**	**86**	**93**	**97**	**103**	**109**	**115**	**120**	126	132	138	142	147	152	158	162	167	172	177	183
11								**86**	**93**	**100**	**105**	**111**	**117**	123	129	136	143	150	155	160	166	172	177	182	188	192	198
12									**100**	**108**	**114**	120	126	133	139	145	152	159	164	171	176	182	188	194	200	205	211
13										**117**	122	129	135	142	149	155	162	168	174	181	187	193	199	205	212	218	224
14											129	136	143	150	158	165	172	179	186	193	199	205	212	218	225	231	237
15												144	151	159	167	175	183	191	198	205	212	218	225	231	238	244	251
16													159	168	176	184	193	201	209	217	225	231	238	245	251	258	265
17														177	186	194	203	211	220	229	238	244	251	258	265	271	278
18															196	204	213	222	232	241	250	257	265	272	279	286	293
19																213	222	232	242	251	260	268	275	283	291	298	306
20																	232	242	252	261	270	278	286	294	302	310	319
21																		252	262	271	280	289	298	306	314	324	334
22																			272	281	290	300	308	316	326	337	347
23																				291	301	311	319	328	340	350	362
24																					311	323	331	341	353	364	375
25																						336	343	354	366	378	390
26																							352	363	376	388	400
27																								373	386	398	410
28																									396	408	421
29																										418	431
30																											441

12. táblázat. $Z_{4,4}(m,n)$ alsó becslések

A $K_{5,5}$-mentes esetben [2, 4] cikkek szolgáltak referenciaként. Egy híján az összes eset-

ben elértük az ismert felső becsléseket, továbbá a fejezet elején említett három hibát is javítottuk. A program futásideje 12,22 óra volt.

$m\backslash n$	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
5	**24**	**28**	**32**	**36**	**40**	**44**	**48**	**52**	**56**	**60**	**64**	**68**	**72**	**76**	**80**	**84**	**88**	**92**	**96**	**100**	**104**	**108**	**112**	**116**	**120**	**124**
6		**33**	**38**	**43**	**48**	**52**	**57**	**62**	**67**	**72**	**76**	**81**	**86**	**91**	**96**	**100**	**105**	**110**	**115**	**120**	**124**	**128**	**132**	**136**	**140**	**144**
7			**44**	**50**	**56**	**60**	**66**	**72**	**78**	**84**	**88**	**93***	**98***	**103***	**108**	**112**	**117**	**122**	**127**	**132**	**136**	**141**	**146**	**151**	**156**	**160**
8				**57**	**64**	**68**	**74**	**80**	**86**	**92**	**97**	**103**	**109**	**115**	120	**126**	**132**	137	**143**	148	**154**	**160**	165	**171**	176	181
9					**72**	**76**	**82**	**88**	**95**	**101**	**108**	**114**	**121**	**128**	134	140	146	152	158	164	170	176	182	188	194	200
10						**84**	**90**	**97**	**104**	**110**	**117**	**124**	**131**	**138**	144	151	158	164	171	177	184	191	197	203	210	216
11							**98**	**106**	**113**	**120**	**127**	**135**	**142**	149	157	164	171	179	186	192	199	206	213	220	226	233
12								**114**	**122**	**130**	**138**	**146**	**154**	162	170	177	185	192	200	207	214	221	228	236	243	250
13									**132**	**140**	**149**	**156**	164	172	181	190	197	205	213	221	229	238	246	253	261	269
14										**150**	**160**	167	176	184	193	202	210	218	226	235	244	253	262	271	280	288
15											**171**	177	187	196	205	214	222	232	242	250	260	270	279	288	297	306
16												186	197	206	216	225	236	247	258	266	276	285	294	304	314	324
17													209	217	229	238	249	260	271	279	290	299	310	320	331	342
18														228	240	252	261	272	284	293	304	314	326	336	348	360
19															253	266	274	284	295	305	316	326	338	348	360	372
20																280	288	297	308	318	329	340	351	362	373	384
21																	298	308	319	330	341	352	363	375	387	398
22																		319	332	344	355	367	378	389	401	413
23																			343	355	368	380	392	404	415	427
24																				368	381	394	407	420	432	444
25																					394	407	420	433	446	459
26																						420	433	447	460	473
27																							446	460	473	487
28																								473	488	502
29																									501	515
30																										529

13. táblázat. $Z_{5,5}(m,n)$ alsó becslések

A $K_{6,6}$-mentes esetre csak egy cikkben [2] találtunk eredményeket. Két esetben nem sikerült beállítanunk a pontos értékeket, javítottuk a fejezet elején említett hibát. A program erre az esetre szánt futásideje összesen 12,78 óra volt.

$m\backslash n$	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
6	**35**	**40**	**45**	**50**	**55**	**60**	**65**	**70**	**75**	**80**	**85**	**90**	**95**	**100**	**105**	**110**	**115**	**120**	**125**	**130**	**135**	**140**	**145**	**150**	**155**
7		**46**	**52**	**58**	**64**	**70**	**75**	**81**	**87**	**93**	**99**	**105**	**110**	**116**	**122**	**128**	**134**	**140**	**145**	**151**	**157**	**163**	**169**	**175**	**180**
8			**59**	**66**	**73**	**80**	**85**	**92**	**99**	**106**	**113**	**120**	**125**	131	137	143	149	155	161	167	173	179	185	191	197
9				**74**	**82**	**90**	**95**	**102**	**109**	**116**	**123**	**130**	**137**	144	151	158	165	172	178	185	192	198	205	212	218
10					**91***	**100**	**105**	**112**	**120**	**127**	**135**	**142**	**150**	158	165	173	181	189	196	204	211	219	226	234	241
11						**110**	**115**	**122**	**130**	**138**	**147**	**155**	**163**	171	180	188	196	204	212	220	227	235	243	251	259
12							**125**	**132**	**141**	**150**	**158**	**167**	**176**	184	193	202	210	219	227	236	244	253	261	269	277
13								**142**	**152**	**161**	**170**	**180**	**189**	198	207	216	225	234	243	253	262	271	280	289	297
14									**162**	**172**	**182**	191	201	211	221	231	241	250	260	270	280	290	299	308	317
15										183	193	204	214	224	234	244	254	264	275	285	296	306	315	324	334
16											205	217	226	237	248	259	270	280	291	301	311	322	332	342	352
17												230	239	251	263	273	285	296	307	318	328	339	349	359	370
18													250	263	275	286	298	310	322	333	344	355	366	377	388
19														275	288	300	312	324	336	349	360	372	383	394	406
20															302	314	327	339	351	364	376	389	401	413	425
21																327	341	354	366	379	392	405	418	430	442
22																	354	368	381	394	408	421	434	447	460
23																		381	395	409	423	436	450	463	477
24																			410	424	439	453	467	480	494
25																				439	454	469	483	497	512
26																					469	484	500	515	529
27																						500	516	531	546
28																							532	548	564
29																								565	581
30																									598

14. táblázat. $Z_{6,6}(m,n)$ alsó becslések

6.3. Forráskódok, talált gráfok

A projekttel kapcsolatos fájlok, eredmények megtalálhatóak az alábbi linken:
`https://github.com/balazsborsik/Zarankiewicz`

Ábrák jegyzéke

Táblázatok jegyzéke

Algoritmusok jegyzéke

Irodalomjegyzék

[1] Charles J Colbourn. *CRC handbook of combinatorial designs*. CRC press, 2010.

[2] Alex F. Collins és tsai. „Zarankiewicz Numbers and Bipartite Ramsey Numbers". *Journal of Algorithms and Computation* 47.1 (2016. jún.), 63–78. old.

[3] Gábor Damásdi, Tamás Héger és Tamás Szőnyi. „The Zarankiewicz problem, cages and geometries". *Annales Universitatis Scientiarum Budapestinensis de Rolando Eötvös Nominatae - Sectio Mathematica* 56.1 (2013), 3–37. old.

[4] Sara Davies, Peter Gill és Daniel Horsley. *Improved upper bounds on Zarankiewicz numbers*. 2024. arXiv: `2411.18842 [math.CO]`. URL: `https://arxiv.org/abs/2411.18842`.

[5] Agner Fog. „Optimizing software in C++". *URL: http://www. agner. org/optimize/optimizing_ cpp. pdf* (2006).

[6] Zoltán Füredi. „An upper bound on Zarankiewicz'problem". *Combinatorics, Probability and Computing* 5.1 (1996), 29–33. old.

[7] Richard K Guy. „A many-facetted problem of Zarankiewicz". *The Many Facets of Graph Theory: Proceedings of the Conference held at Western Michigan University, Kalamazoo/MI., October 31-November 2, 1968*. Springer. 2006, 129–148. old.

[8] C Hyltén-Cavallius. „On a combinatorical problem". *Colloquium Mathematicae*. 6. köt. 1. 1958, 61–65. old.

[9] Robert W Irving. „A bipartite Ramsey problem and the Zarankiewicz numbers". *Glasgow Mathematical Journal* 19.1 (1978), 13–26. old.

[10] JLWV Jensen. „On the convex functions and inequalities between mean values". *Acta Math* 30 (1906), 175–193. old.

[11] Andrew Kay. *Efficient algorithms for the Zarankiewicz problem*. Technical Report. School of Computing és Digital Technology, Birmingham City University, 2016. URL: `https://andrewkay.name/maths/Efficient_algorithms_for_the_Zarankiewicz_problem.pdf`.

[12] Peter Keevash. *The existence of designs*. 2024. arXiv: `1401.3665 [math.CO]`. URL: `https://arxiv.org/abs/1401.3665`.

[13] György Kiss és Tamás Szőnyi. *Finite Geometries*. Boca Raton, FL, USA: CRC Press, Taylor & Francis Group, 2020. ISBN: 978-1-4987-2165-3.

[14] Tamás Kővári, Vera T. Sós és Pál Turán. „On a problem of K. Zarankiewicz". *Colloquium Mathematicae* 3 (1954), 50–57. old.

[15] Vladimir Nikiforov. „A contribution to the Zarankiewicz problem". *Linear algebra and its applications* 432.6 (2010), 1405–1411. old.

[16] Istvan Reiman. „Über ein problem von K. Zarankiewicz". *Acta mathematica hungarica* 9.3-4 (1958), 269–273. old.

[17] Steven Roman. „A Problem of Zarankiewicz". *Journal of Combinatorial Theory, Series B* 18.3 (1975), 289–292. old.

[18] Lajos Rónyai. *Véletlen és algoritmusok*. Budapest: Typotex, 2011.

[19] Tamás Szőnyi. *Szimmetrikus struktúrák*. Budapest: Typotex, 2013.

[20] Jeremy Tan. „An attack on Zarankiewicz's problem through SAT solving". *arXiv preprint arXiv:2201.12345* (2022).

[21] Paul Turán. „On an external problem in graph theory". *Mat. Fiz. Lapok* 48 (1941), 436–452. old.

[22] Kazimierz Zarankiewicz. „Problem p 101". *Colloq. Math.* 2. köt. 301. 1951, 5. old.

Függelék

F.1. További táblázatok $s \leq t$ esetekre

$K_{2,3}$	$K_{2,4}$	$K_{2,5}$	$K_{2,6}$	$K_{3,4}$	$K_{3,5}$	$K_{3,6}$	$K_{4,5}$	$K_{4,6}$	$K_{5,6}$
41,53	6,25	7,64	4,86	9,86	5,14	7,78	14,58	7,22	10,00

15. táblázat. Az aszimmetrikus $K_{s,t}$ gráfok generálásának futtási ideje (óra)

$m\backslash n$	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
2	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
3	7	9	10	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
4	9	12	13	15	16	18	19	21	22	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
5	11	14	16	18	20	22	23	25	26	28	29	31	32	34	35	37	38	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
6	13	16	19	21	24	25	27	30	31	33	34	36	37	39	40	42	43	45	46	48	49	51	52	54	55	57	58	60	61	62	63	64	65	66	67	68	69	70
7	15	18	22	24	28	29	31	34	36	37	39	42	43	45	46	48	49	51	52	54	55	57	58	60	61	63	64	66	67	69	70	72	73	75	76	78	79	81
8	17	20	24	27	31	33	35	38	40	42	44	46	48	50	52	54	56	58	59	61	62	64	65	67	68	70	71	73	74	76	77	79	80	82	83	85	86	88
9	19	22	27	30	34	36	39	42	45	46	49	51	54	56	57	60	62	64	66	68	69	72	73	75	76	78	79	81	82	84	85	87	88	90	91	93	94	96
10	21	24	30	33	37	40	42	46	50	51	54	56	60	61	63	66	68	70	72	74	76	78	80	82	84	86	87	90	91	93	94	96	97	99	100	102	103	105
11	23	26	32	36	40	44	46	50	55	56	58	61	65	66	69	71	74	76	78	80	82	84	86	88	90	92	94	96	98	100	102	104	106	108	110	112	113	115
12	25	28	34	38	43	48	50	53	58	60	63	66	70	72	74	77	79	82	84	87	89	91	93	96	98	100	102	104	106	108	110	112	114	116	118	120	122	123
13	27	30	36	41	46	52	54	56	61	64	67	71	75	78	80	82	85	88	90	93	95	98	100	104	105	108	109	111	114	116	118	120	122	124	126	128	130	132
14	29	32	38	44	49	56	58	60	65	68	72	76	80	84	86	88	91	94	96	100	102	104	107	110	112	114	117	119	121	123	126	128	130	132	134	136	138	140
15	31	34	40	46	51	58	61	64	68	72	76	80	85	90	91	94	96	100	102	106	108	111	114	116	119	122	124	126	129	131	134	136	138	141	143	145	147	149
16	33	36	42	49	54	61	64	68	72	76	81	84	90	96	97	99	102	106	108	112	115	117	120	123	126	130	131	134	137	139	142	144	146	149	151	153	156	158
17	35	38	44	52	57	64	68	72	76	80	85	89	94	100	102	104	108	111	114	117	120	123	127	130	132	136	138	141	144	147	149	152	154	157	160	162	164	167
18	37	40	46	54	59	66	72	76	79	84	90	94	98	104	107	110	113	116	120	123	127	130	133	136	139	143	145	148	151	154	157	160	162	165	168	170	173	175
19	39	42	48	57	62	69	75	79	83	88	94	98	102	108	112	115	118	121	125	129	133	136	139	143	146	149	152	155	158	161	164	167	170	173	176	178	181	184
20	41	44	50	60	65	72	78	83	87	92	98	103	106	112	117	120	123	127	131	134	139	142	146	149	152	156	159	162	165	168	171	175	178	181	184	187	189	192
21	43	46	52	62	67	74	81	86	91	96	102	107	111	116	122	125	128	132	136	140	145	148	152	156	159	162	166	169	172	176	178	182	185	188	192	195	197	200
22	45	48	54	64	70	77	84	90	95	100	106	112	116	120	126	130	133	137	141	146	150	155	158	162	165	169	173	176	179	183	186	189	193	196	199	203	206	208
23	47	50	56	66	73	80	87	93	99	104	110	116	121	125	131	135	139	142	146	151	156	161	165	168	171	175	180	183	186	189	193	197	200	204	207	210	214	217
24	49	52	58	68	75	82	90	97	103	108	114	120	126	130	136	140	145	148	152	156	162	166	170	174	178	181	186	190	193	197	200	204	208	211	214	218	222	225
25	51	54	60	70	78	85	93	100	106	112	118	125	131	135	141	145	150	153	157	161	167	172	176	180	184	188	192	196	200	204	208	211	215	218	222	226	229	233
26	53	56	62	72	81	88	96	104	110	116	122	130	135	140	146	150	156	158	162	166	172	177	181	186	190	194	198	202	207	211	215	218	222	226	229	233	237	241
27	55	58	64	74	83	90	99	108	113	120	126	135	140	145	151	155	162	164	167	171	177	182	186	191	196	200	205	209	214	217	222	225	229	233	237	241	245	248
28	57	60	66	76	86	93	102	112	117	124	130	140	144	150	156	160	168	170	173	176	182	187	192	197	202	206	211	215	220	224	228	232	236	241	245	248	252	256
29	59	62	68	78	89	96	105	116	121	128	134	144	149	155	161	165	173	176	179	182	187	192	197	202	207	212	217	222	227	231	235	239	243	248	252	256	260	263
30	61	64	70	80	91	98	108	120	125	132	138	148	154	160	166	170	178	181	184	188	192	197	202	207	212	218	223	228	233	237	242	246	250	255	260	263	267	271
31	63	66	72	82	94	101	110	122	128	135	142	152	158	165	171	175	183	187	190	194	197	202	207	213	218	223	229	234	239	244	248	253	257	262	267	271	275	279
32	65	68	74	84	97	104	113	125	131	139	146	156	162	169	176	180	188	193	196	199	203	207	212	218	223	229	235	240	245	250	254	260	264	269	274	278	283	287
33	67	70	76	86	99	106	116	128	134	142	150	160	166	174	181	185	193	199	202	205	209	213	217	224	229	234	240	246	251	256	261	266	271	275	281	285	290	294
34	69	72	78	88	102	109	118	130	137	146	154	164	170	178	186	190	198	205	208	211	214	219	223	229	235	240	246	251	257	262	267	273	277	282	288	292	297	302
35	71	74	80	90	105	112	121	133	140	150	158	168	175	183	191	195	203	211	214	217	220	225	229	234	240	246	251	257	263	268	273	279	284	289	294	299	304	309
36	73	76	82	92	107	114	124	136	143	153	162	172	180	188	196	200	208	216	220	223	226	231	235	239	246	252	257	262	268	274	280	285	291	295	300	306	311	316
37	75	78	84	94	109	117	126	138	146	156	166	176	185	192	200	205	213	222	225	228	232	236	241	245	252	257	263	268	274	280	286	291	297	302	307	313	318	323
38	77	80	86	96	111	120	129	141	149	160	170	180	190	196	204	210	218	228	231	234	238	242	246	251	257	263	269	274	280	286	292	298	303	308	313	320	325	330
39	79	82	88	98	113	122	132	144	152	164	173	184	195	201	208	215	223	234	237	240	243	247	252	257	262	269	274	280	286	291	298	304	309	314	320	326	332	337
40	81	84	90	100	115	125	134	146	155	167	176	188	200	206	213	220	228	240	243	245	249	253	258	262	267	274	280	286	292	297	304	310	316	321	326	332	339	344

16. táblázat. $Z_{2,3}(m,n)$ alsó becslések

$m\backslash n$	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
3	10	12	13	15	16	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
4	13	16	17	19	21	22	24	25	27	28	30	31	33	34	36	37	38	39	40	41	42	43	44	45	46	47	48
5	16	20	21	23	26	27	30	31	33	34	36	37	39	40	42	43	45	46	48	49	51	52	54	55	57	58	60
6	19	23	25	27	30	32	35	37	39	40	42	44	46	48	49	51	52	54	55	57	58	60	61	63	64	66	67
7	22	26	29	31	35	37	40	42	44	46	49	51	53	55	57	58	60	63	64	66	67	69	70	72	73	75	76
8	25	29	33	35	40	42	45	48	50	52	56	57	59	62	64	66	68	70	72	74	76	78	80	81	83	85	86
9	28	32	37	39	44	47	50	54	56	58	62	64	66	69	72	73	75	78	80	82	84	86	88	90	92	94	96
10	31	35	41	43	48	52	55	60	61	64	68	71	73	75	78	81	83	85	88	90	92	95	97	99	101	103	105
11	34	38	45	47	52	57	60	66	67	70	74	77	79	82	85	88	90	93	96	98	100	103	105	108	110	112	114
12	37	41	48	51	56	61	65	71	73	76	80	84	86	89	92	96	98	101	104	106	109	111	114	116	119	121	123
13	40	44	52	55	60	66	70	76	78	81	86	91	93	96	99	103	106	108	111	114	117	120	122	125	128	130	133
14	43	47	56	59	64	71	75	81	84	87	92	98	99	102	106	110	113	116	119	122	126	128	131	134	137	140	142
15	46	50	60	63	68	75	80	86	90	93	98	105	106	109	113	117	121	124	126	130	135	136	139	142	145	148	151
16	49	53	63	67	72	80	85	91	96	98	103	110	112	116	120	124	128	131	134	138	144	145	148	151	154	157	160
17	52	56	66	71	76	85	90	96	102	104	108	115	118	122	127	131	136	138	141	146	152	154	156	159	162	166	170
18	55	59	69	75	80	90	94	101	108	110	113	120	124	128	134	138	143	146	149	154	160	163	165	167	171	175	179
19	58	62	72	79	84	94	98	106	114	116	119	125	130	135	140	145	150	153	157	162	168	172	173	176	179	183	188
20	61	65	75	83	88	98	103	111	120	122	125	130	136	141	146	152	157	161	165	170	176	181	182	185	188	192	196
21	64	68	78	87	92	102	108	116	126	128	131	135	142	147	152	158	163	167	172	178	184	190	191	193	196	200	204
22	67	71	81	91	96	106	113	121	132	134	136	141	148	153	158	164	170	174	179	186	192	198	200	202	205	208	212
23	70	74	84	95	100	110	118	125	136	139	142	147	154	159	164	170	176	181	186	193	200	207	208	210	214	217	221
24	73	77	87	98	104	114	123	129	140	144	148	153	160	165	170	176	182	188	192	200	208	216	217	219	223	226	230
25	76	80	90	102	108	118	127	133	144	149	154	159	166	171	176	182	188	194	199	207	216	225	226	228	232	235	238
26	79	83	93	106	112	122	131	138	148	154	160	165	172	177	182	188	194	201	206	214	223	232	234	236	240	243	246
27	82	86	96	110	116	126	136	143	152	159	166	170	178	183	188	193	200	207	213	221	230	239	241	244	247	251	254
28	85	89	99	113	120	130	140	147	157	164	172	176	184	189	194	200	206	214	220	228	237	246	249	252	255	259	262
29	88	92	102	117	124	134	145	152	162	169	178	182	190	195	200	206	212	220	226	235	244	253	257	261	263	267	270
30	91	95	105	121	128	138	150	157	167	174	184	188	195	201	206	212	218	226	233	241	250	259	264	269	271	275	278

17. táblázat. $Z_{2,4}(m,n)$ alsó becslések

$m\backslash n$	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**
3	**13**	**15**	**16**	**18**	**19**	**21**	**22**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
4	**17**	**20**	21	**24**	**25**	**27**	**28**	**30**	**31**	**33**	**34**	**36**	**37**	**39**	**40**	**42**	**43**	**45**	**46**	**48**	**49**	**50**	**51**	**52**	**53**	**54**
5	**21**	**25**	26	**29**	**31**	**33**	**35**	**37**	38	40	**42**	**44**	**45**	**47**	**48**	**50**	**51**	**53**	**54**	**56**	**57**	**59**	**60**	**62**	**63**	**65**
6	**25**	**30**	31	34	**37**	39	**42**	**44**	45	**48**	**50**	**52**	**54**	**56**	57	**60**	**61**	**63**	**64**	**66**	**67**	**69**	**70**	**72**	**73**	**75**
7	**29**	**34**	36	39	**43**	45	**48**	50	52	**56**	57	**60**	**62**	**64**	**66**	**68**	**70**	**72**	**74**	**76**	**78**	**80**	81	**84**	**85**	**87**
8	**33**	**38**	41	44	**49**	51	54	57	59	**63**	**65**	67	**70**	72	74	**77**	**79**	**81**	**83**	**85**	**87**	**89**	**91**	**93**	**95**	**97**
9	**37**	**42**	46	49	54	57	60	**64**	66	**70**	**73**	75	**78**	**81**	**83**	85	**88**	90	**93**	95	97	**100**	**102**	**104**	**106**	**108**
10	**41**	**46**	51	54	60	63	66	70	73	77	**81**	83	86	**90**	**92**	94	**97**	**100**	**102**	104	**107**	**110**	**112**	114	**117**	**120**
11	**45**	**50**	56	59	66	69	72	77	80	84	88	90	94	**98**	100	103	106	**110**	**112**	114	**117**	**120**	**122**	124	**127**	**130**
12	**49**	**54**	**61**	64	72	75	78	84	87	91	**96**	98	101	**106**	**109**	111	**115**	**118**	**121**	**124**	126	129	132	135	**138**	140
13	**53**	**58**	**66**	69	77	80	84	90	94	98	**104**	106	109	114	**118**	120	123	127	**131**	133	136	139	142	145	148	150
14	**57**	**62**	**71**	74	82	86	90	96	101	105	**112**	113	117	122	126	129	132	136	140	143	145	149	152	**156**	158	161
15	**61**	**66**	**76**	79	87	92	96	102	108	112	**120**	121	125	130	135	137	140	145	**150**	152	155	158	162	166	169	172
16	**65**	**70**	**81**	84	92	98	102	108	114	119	**127**	129	132	138	**144**	146	149	154	**160**	161	165	168	172	175	179	183
17	**69**	**74**	**85**	89	97	104	108	114	120	126	134	136	139	146	**153**	154	158	163	**169**	171	174	178	182	185	189	193
18	**73**	**78**	**90**	94	102	110	114	120	127	133	141	144	147	154	**162**	163	167	172	178	181	183	187	192	195	199	203
19	**77**	**82**	**95**	99	107	116	120	126	134	140	148	152	155	162	**171**	172	175	181	187	190	193	196	201	205	209	213
20	**81**	**86**	**100**	104	112	122	126	132	141	147	155	160	163	169	178	180	184	190	196	200	203	206	211	215	219	223
21	**85**	**90**	**105**	109	117	128	132	138	147	154	162	168	171	176	185	188	192	198	204	209	212	215	220	225	229	233
22	**89**	**94**	**109**	114	122	133	138	144	154	161	169	176	179	183	192	196	200	206	213	218	222	225	230	235	239	244
23	**93**	**98**	**113**	119	127	138	143	150	161	168	176	184	186	190	199	204	209	214	221	227	231	235	239	244	249	254
24	**97**	**102**	**117**	124	132	144	149	156	168	174	183	192	194	198	206	212	217	222	229	236	241	244	248	254	259	264
25	**101**	**106**	**121**	129	137	150	155	162	174	180	190	200	202	206	213	220	226	230	237	245	250	253	258	263	269	274
26	**105**	**110**	**125**	134	142	156	161	168	180	186	197	208	210	213	220	228	234	238	245	253	258	262	267	272	278	284
27	**109**	**114**	**129**	139	147	162	167	174	186	192	204	216	218	221	227	236	242	246	253	261	266	271	276	281	288	293
28	**113**	**118**	**133**	144	152	168	173	180	192	198	211	224	226	229	235	244	250	254	261	269	275	280	285	290	298	303
29	**117**	**122**	**137**	149	157	174	179	186	198	204	218	232	234	236	242	252	258	263	269	277	284	289	294	299	307	313
30	**121**	**126**	**141**	154	162	180	185	192	204	210	225	240	242	244	249	259	266	271	277	285	292	297	303	308	316	323

18. táblázat. $Z_{2,5}(m,n)$ alsó becslések

$m\backslash n$	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**
3	**16**	**18**	**19**	**21**	**22**	**24**	**25**	**27**	**28**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**
4	**21**	**24**	25	**28**	29	**31**	**33**	**34**	**36**	**37**	**39**	**40**	**42**	**43**	**45**	**46**	**48**	**49**	**51**	**52**	**54**	**55**	**57**	**58**	**60**
5	**26**	**30**	31	**34**	**36**	**38**	**40**	**42**	**44**	**46**	**48**	**50**	**52**	**53**	**55**	**56**	**58**	**59**	**61**	**62**	**64**	**65**	**67**	**68**	**70**
6	**31**	**36**	37	40	**43**	45	**48**	50	52	**55**	**57**	**59**	**61**	**63**	**65**	**67**	**69**	70	72	74	**76**	**78**	**79**	**81**	**82**
7	**36**	**42**	43	46	50	52	**56**	**58**	60	**63**	**66**	**68**	70	72	**75**	**77**	**79**	**81**	**83**	**85**	**87**	**89**	**91**	**93**	**95**
8	**41**	**47**	49	52	57	59	**64**	66	68	**72**	74	**77**	**80**	**82**	**85**	**87**	89	**92**	**94**	**96**	**98**	**100**	**102**	**104**	**106**
9	**46**	**52**	55	58	64	66	**72**	74	76	**81**	83	86	**90**	**92**	94	**97**	**100**	**102**	**105**	**107**	**110**	**112**	114	**117**	119
10	**51**	**57**	61	64	70	73	**79**	82	84	**90**	92	95	**99**	101	104	107	110	112	115	**118**	**121**	**123**	**126**	**128**	**131**
11	**56**	**62**	67	70	77	80	86	89	92	**98**	100	104	**108**	111	114	117	**121**	123	126	129	132	135	137	140	**143**
12	**61**	**67**	73	76	84	87	93	97	100	106	109	112	117	**121**	123	127	**132**	134	137	140	143	146	149	152	155
13	**66**	**72**	79	82	91	94	100	105	108	114	118	121	126	130	133	137	**142**	144	147	151	154	157	160	164	167
14	**71**	**77**	85	88	97	101	107	112	116	122	126	130	135	140	143	147	**152**	154	158	161	165	169	171	175	179
15	**76**	**82**	91	94	103	108	114	120	124	130	135	139	144	150	152	156	162	165	168	171	176	180	183	186	191
16	**81**	**87**	97	100	109	115	120	128	132	138	144	148	153	**160**	161	166	172	176	179	182	187	191	195	198	202
17	**86**	**92**	**103**	106	115	122	127	136	140	146	152	157	162	**170**	171	176	182	187	190	193	198	202	206	210	214
18	**91**	**97**	**109**	112	121	129	134	144	148	154	160	166	171	**180**	181	186	192	198	200	203	208	213	217	221	225
19	**96**	**102**	**115**	118	127	136	141	151	155	162	168	175	180	**190**	191	195	202	209	210	214	218	224	228	232	236
20	**101**	**107**	**121**	124	133	143	148	158	163	170	176	183	189	**199**	201	204	212	**220**	221	225	229	235	239	244	248
21	**106**	**112**	**127**	130	139	150	155	165	171	178	184	191	198	208	210	213	222	**231**	232	235	239	246	250	255	259
22	**111**	**117**	**133**	136	145	156	162	172	178	186	192	199	207	217	220	223	232	**242**	243	246	250	256	261	266	271
23	**116**	**122**	**138**	142	151	163	169	179	185	193	199	207	216	226	230	233	242	**253**	254	257	261	266	272	277	282
24	**121**	**127**	**144**	148	157	170	176	186	193	200	207	216	225	235	240	243	251	262	264	267	272	276	283	288	294
25	**126**	**132**	**150**	154	163	176	183	193	201	208	215	225	234	244	250	253	260	271	274	277	282	287	293	299	305
26	**131**	**137**	**156**	160	169	183	190	200	208	216	223	234	243	253	260	263	269	280	284	287	292	298	303	310	316
27	**136**	**142**	**162**	166	175	190	196	207	216	224	231	243	252	262	270	272	278	289	293	297	302	308	314	321	327
28	**141**	**147**	**168**	172	181	197	203	214	224	231	239	252	261	271	280	282	287	298	302	307	312	318	324	332	338
29	**146**	**152**	**173**	178	187	204	210	221	232	239	247	261	270	280	290	292	296	306	311	317	322	328	335	343	348
30	**151**	**157**	**178**	184	193	211	217	228	240	246	255	269	278	289	300	302	305	315	320	327	332	338	345	353	359

19. táblázat. $Z_{2,6}(m,n)$ alsó becslések

$m\backslash n$	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
3	**11**	**13**	**15**	**17**	**19**	**21**	**23**	**25**	**27**	**29**	**31**	**33**	**35**	**37**	**39**	**41**	**43**	**45**	**47**	**49**	**51**	**53**	**55**	**57**	**59**	**61**	**63**
4	**14**	**17**	**20**	**22**	**25**	**28**	**30**	**33**	**36**	**38**	**40**	**42**	**44**	**46**	**48**	**50**	**52**	**54**	**56**	**58**	**60**	**62**	**64**	**66**	**68**	**70**	**72**
5	**17**	**21**	**25**	**27**	**30**	**33**	**36**	**39**	**42**	**44**	**47**	**50**	**52**	**55**	**58**	**60**	**63**	**66**	**68**	**71**	**74**	**76**	**79**	**82**	**84**	**87**	**90**
6	**20**	**25**	**30**	**32**	**35**	**39**	**42**	**46**	**50**	**52**	**56**	**60**	**62**	**65**	**68**	**70**	**73**	**76**	**78**	**81**	**84**	**86**	**89**	**92**	**94**	**97**	**100**
7	**23**	**28**	**33**	**37**	**40**	**44**	**48**	**52**	**56**	**59**	**63**	**66**	**70**	**73**	**77**	**80**	83	**87**	**90**	93	96	98	101	104	107	110	113
8	**26**	**32**	**37**	**42**	**45**	**50**	**54**	**59**	**64**	**67**	**70**	**74**	78	82	85	89	92	96	100	103	107	110	113	116	120	123	127
9	**29**	**36**	**41**	**46**	**50**	**56**	**60**	**66**	**72**	**75**	**78**	82	86	90	94	99	103	107	112	115	120	123	127	130	134	137	140
10	**32**	**40**	**45**	**51**	**55**	**61**	**66**	72	78	82	86	90	94	99	103	107	111	116	120	125	130	135	**140**	142	146	149	153
11	**35**	**43**	**48**	**55**	**60**	**66**	**72**	78	84	89	94	99	103	107	112	116	121	126	132	137	140	145	150	154	158	162	166
12	**38**	**46**	**52**	**60**	**65**	**72**	**78**	84	90	96	102	108	112	116	122	126	131	136	141	146	152	157	162	168	172	176	181
13	**41**	**49**	**56**	**64**	**70**	**77**	**83**	89	96	102	108	114	119	124	130	135	141	147	151	156	162	168	174	180	185	190	196
14	**44**	**52**	**60**	**68**	**75**	**82**	**89**	95	102	108	114	120	126	132	138	144	150	155	161	168	174	180	186	192	198	204	210
15	**47**	**55**	**63**	**72**	**79**	**86**	**94**	100	108	114	121	128	135	140	147	153	160	165	171	177	184	190	197	204	211	218	**225**
16	**50**	**58**	**67**	**76**	**83**	**91**	**100**	106	114	120	128	136	144	148	155	162	169	175	182	188	194	200	208	216	224	232	**240**
17	**53**	**61**	**71**	**80**	**88**	**96**	**105**	112	120	126	134	142	150	156	163	171	178	185	193	199	204	210	218	226	234	242	250
18	**56**	**64**	**75**	**84**	**92**	**101**	110	118	126	132	140	148	156	163	172	180	188	195	204	210	216	222	228	236	244	252	260
19	**59**	**67**	**78**	**88**	**97**	**106**	116	124	133	139	146	154	162	171	180	188	197	205	214	221	228	232	238	246	254	262	270
20	**62**	**70**	**82**	**92**	101	111	122	130	140	146	153	160	169	178	188	197	206	215	224	232	240	244	248	256	264	272	280
21	**65**	**73**	**86**	**96**	105	115	127	134	144	152	159	167	175	185	195	205	215	225	234	243	252	256	260	266	274	282	290
22	**68**	**76**	**90**	**100**	110	120	132	140	150	159	166	174	182	193	204	214	224	234	244	254	264	268	272	277	284	292	300
23	**71**	**79**	**93**	**103**	114	124	137	145	155	165	172	180	189	200	211	221	232	243	254	265	276	280	284	289	294	302	310
24	**74**	**82**	**97**	**107**	118	129	142	150	160	171	178	186	195	206	218	228	240	252	264	276	288	292	296	301	306	312	320
25	**77**	**85**	**101**	**111**	122	133	146	156	166	177	185	193	202	212	224	234	246	258	270	282	294	299	304	309	315	321	330
26	**80**	**88**	**105**	**115**	126	138	151	161	172	183	192	200	208	218	230	240	252	264	276	288	300	306	312	318	324	330	340
27	**83**	**91**	**108**	**118**	130	142	155	166	178	189	199	207	215	224	236	246	258	270	282	294	306	314	321	327	333	340	350
28	**86**	**94**	**112**	**122**	134	147	160	171	184	196	206	214	222	231	242	252	264	276	288	300	312	322	329	336	343	350	360
29	**89**	**97**	**116**	**126**	138	152	165	177	190	203	213	221	229	238	248	259	272	284	296	307	319	329	338	345	353	361	369
30	**92**	**100**	**120**	**130**	142	157	169	182	195	210	220	228	236	245	254	266	279	291	304	315	326	336	346	354	362	371	378

20. táblázat. $Z_{3,4}(m,n)$ alsó becslések

$m\backslash n$	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
3	**14**	**16**	**18**	**20**	**22**	**24**	**26**	**28**	**30**	**32**	**34**	**36**	**38**	**40**	**42**	**44**	**46**	**48**	**50**	**52**	**54**	**56**	**58**	**60**	**62**	**64**
4	**18**	**21**	**24**	**26**	**29**	**32**	**34**	**37**	**40**	**42**	**45**	**48**	**50**	**52**	**54**	**56**	**58**	**60**	**62**	**64**	**66**	**68**	**70**	**72**	**74**	**76**
5	**22**	**26**	**30**	32	**36**	**40**	**42**	**45**	**48**	**50**	**53**	**56**	**58**	**61**	**64**	**66**	**69**	**72**	**74**	**77**	**80**	**82**	**85**	**88**	**90**	**93**
6	**26**	**31**	**36**	38	42	**46**	49	**53**	**56**	59	**63**	**66**	69	72	75	78	81	84	87	**90**	**93**	**96**	**98**	**101**	**104**	**106**
7	**30**	**36**	**42**	44	48	52	57	**61**	**65**	68	**72**	**76**	**80**	82	**86**	**90**	92	95	99	102	**106**	**110**	112	**116**	**120**	122
8	**34**	**40**	**46**	50	54	59	64	68	72	76	80	84	89	93	96	100	104	108	112	115	119	122	126	129	133	136
9	**38**	**45**	**51**	56	60	66	71	76	81	85	90	94	99	102	107	111	115	120	123	127	131	135	139	143	146	150
10	**42**	**50**	**56**	62	66	72	78	84	90	94	99	104	108	112	118	123	127	133	136	141	145	149	153	156	160	164
11	**46**	**55**	**61**	68	72	79	85	92	99	103	108	112	117	122	128	134	138	144	148	154	157	163	166	171	175	179
12	**50**	**60**	**66**	73	78	86	92	100	108	112	117	122	127	132	138	144	150	156	161	168	171	176	180	185	189	193
13	**54**	**64**	**70**	79	84	93	99	108	117	121	126	132	137	142	148	154	160	166	172	179	184	190	194	199	203	208
14	**58**	**68**	**75**	84	90	98	105	114	123	127	135	140	146	152	158	164	170	178	184	192	198	204	208	213	218	224
15	**62**	**72**	**80**	90	96	105	112	121	129	135	144	150	156	162	169	175	181	188	195	203	211	216	222	228	233	240
16	**66**	**76**	**85**	96	102	112	118	128	136	143	152	160	166	172	180	186	193	200	207	216	225	230	236	243	247	255
17	**70**	**80**	**90**	101	108	118	125	134	143	151	160	167	174	180	189	196	203	210	218	226	235	244	250	258	262	270
18	**74**	**84**	**94**	106	114	124	132	141	150	159	168	174	182	189	198	205	213	222	230	238	246	255	264	273	277	285
19	**78**	**88**	**99**	111	120	130	138	148	158	167	176	183	190	198	206	214	223	232	241	250	259	268	278	288	292	300
20	**82**	**92**	**104**	116	126	136	145	155	166	175	184	192	200	207	214	224	234	244	253	262	272	282	290	300	307	315
21	**86**	**96**	**109**	121	131	141	151	162	174	183	192	201	210	216	223	234	244	254	264	274	285	296	304	312	322	330
22	**90**	**100**	**114**	126	136	147	158	168	180	190	200	210	220	225	232	244	254	266	276	286	297	308	317	326	337	345
23	**94**	**104**	**118**	131	142	153	165	175	187	197	208	219	230	234	242	254	264	276	287	298	309	320	330	340	352	360
24	**98**	**108**	**123**	136	147	159	172	181	194	204	216	228	240	244	252	263	274	286	298	310	321	332	343	354	366	375
25	**102**	**112**	**128**	141	153	165	178	188	200	211	224	235	248	253	262	272	285	297	309	321	333	345	356	368	380	390
26	**106**	**116**	**133**	146	158	170	184	195	207	219	232	243	256	262	271	282	296	308	320	332	345	357	370	382	394	405
27	**110**	**120**	**138**	151	164	176	191	202	213	226	240	251	262	271	280	291	305	317	330	343	356	370	384	396	408	420
28	**114**	**124**	**142**	**156**	169	182	198	209	220	233	247	258	270	280	289	300	314	326	340	354	368	382	396	409	422	435
29	**118**	**128**	**147**	**161**	175	188	205	216	227	239	254	265	278	288	298	309	323	337	352	366	380	394	408	422	436	450
30	**122**	**132**	**152**	**166**	180	194	211	222	234	246	261	273	286	296	306	317	332	345	360	375	390	405	420	435	450	465

21. táblázat. $Z_{3,5}(m,n)$ alsó becslések

$m\backslash n$	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
3	**17**	**19**	**21**	**23**	**25**	**27**	**29**	**31**	**33**	**35**	**37**	**39**	**41**	**43**	**45**	**47**	**49**	**51**	**53**	**55**	**57**	**59**	**61**	**63**	**65**
4	**22**	**25**	**28**	**30**	**33**	**36**	**38**	**41**	**44**	**46**	**49**	**52**	**54**	**57**	**60**	**62**	**64**	**66**	**68**	**70**	**72**	**74**	**76**	**78**	**80**
5	**27**	**31**	**35**	37	**41**	**45**	47	50	53	**56**	**59**	**62**	**64**	**67**	**70**	**72**	**75**	**78**	**80**	**83**	**86**	**88**	**91**	**94**	**96**
6	**32**	**37**	**42**	44	48	52	**56**	59	62	**66**	69	72	**76**	79	82	**86**	89	92	**96**	99	**102**	**105**	**108**	**110**	**113**
7	**37**	**43**	**49**	51	55	60	64	68	72	76	80	84	88	91	**95**	98	101	**105**	108	**112**	**115**	118	**122**	**125**	128
8	**42**	**49**	**56**	58	62	67	72	77	82	86	90	95	98	102	106	110	114	118	122	125	129	133	137	141	145
9	**47**	**54**	**61**	65	69	75	81	86	92	95	101	105	109	114	118	122	127	131	136	139	143	147	151	156	160
10	**52**	**60**	**67**	72	76	83	89	95	102	105	112	116	121	126	130	135	140	144	149	153	158	162	167	171	175
11	**57**	**66**	**73**	79	83	90	97	103	110	115	121	126	132	137	142	146	152	157	162	167	172	176	181	185	190
12	**62**	**72**	**79**	86	90	98	105	112	120	125	131	136	142	148	153	158	164	169	175	180	185	190	195	200	205
13	**67**	**78**	**85**	93	97	106	113	121	130	135	141	147	153	159	165	170	176	182	188	193	199	204	210	215	219
14	**72**	**84**	**91**	100	104	114	121	130	140	144	150	157	164	170	176	182	188	195	201	207	213	218	225	230	235
15	**77**	**89**	**96**	106	111	121	129	138	148	154	160	166	174	180	187	193	200	207	214	221	228	233	240	245	251
16	**82**	**94**	**102**	113	118	129	137	146	156	162	170	176	184	190	198	205	212	219	227	235	243	248	255	260	267
17	**87**	**99**	**108**	120	125	136	144	154	164	171	179	187	194	201	208	216	223	231	240	249	258	263	270	275	282
18	**92**	**104**	**114**	126	132	144	152	161	172	180	189	198	204	211	219	227	235	243	252	261	270	276	284	290	296
19	**97**	**109**	**120**	133	139	152	160	169	180	189	199	209	215	222	230	238	246	255	264	273	283	291	299	306	311
20	**102**	**114**	**126**	140	146	159	167	177	188	198	209	220	225	233	240	250	258	268	276	286	295	304	313	320	326
21	**107**	**119**	**131**	146	153	166	175	185	196	207	219	231	236	244	251	261	270	280	289	299	308	317	326	334	340
22	**112**	**124**	**137**	152	160	173	183	193	203	216	229	242	246	254	262	272	282	292	302	312	321	330	340	349	355
23	**117**	**129**	**143**	158	167	180	190	201	211	225	239	253	257	265	273	284	294	304	314	324	333	342	353	362	369
24	**122**	**134**	**149**	164	174	187	198	209	220	234	249	264	268	276	284	295	306	316	326	335	345	355	366	375	383
25	**127**	**139**	**155**	170	181	194	205	217	228	242	257	272	278	287	295	306	317	327	337	346	357	367	379	388	396
26	**132**	**144**	**161**	176	188	201	213	225	236	250	265	280	288	298	306	316	328	338	348	358	369	380	391	401	409
27	**137**	**149**	**166**	182	195	208	221	233	244	258	273	288	298	308	317	326	339	349	359	370	380	392	404	413	422
28	**142**	**154**	**172**	188	202	215	228	241	252	266	281	296	308	318	327	336	349	359	370	381	391	404	416	426	436
29	**147**	**159**	**178**	194	208	221	236	249	260	274	289	304	317	327	337	347	359	369	381	392	403	416	428	439	449
30	**152**	**164**	**184**	200	215	228	244	257	268	282	297	312	326	337	347	358	369	380	392	403	414	428	440	451	462

22. táblázat. $Z_{3,6}(m,n)$ alsó becslések

$m\backslash n$	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
4	**19**	**22**	**25**	**28**	**31**	**34**	**37**	**40**	**43**	**46**	**49**	**52**	**55**	**58**	**61**	**64**	**67**	**70**	**73**	**76**	**79**	**82**	**85**	**88**	**91**	**94**
5	**23**	**27**	**31**	**35**	**38**	**42**	**46**	**50**	**53**	**57**	**61**	**65**	**68**	**72**	**76**	**80**	**83**	**86**	**89**	**92**	**95**	**98**	**101**	**104**	**107**	**110**
6	**27**	**32**	**37**	**42**	45	50	**55**	**60**	**63**	**67**	**71**	**75**	**78**	**82**	**86**	**90**	**93**	**97**	**101**	**105**	**108**	**112**	**116**	**120**	**123**	**127**
7	**31**	**37**	**43**	**49**	52	57	62	**67**	71	76	**81**	85	**90**	94	**99**	103	**108**	112	116	120	124	128	132	136	140	144
8	**35**	**42**	**49**	**56**	59	64	69	75	80	86	91	96	101	106	111	116	**121**	**126**	**131**	**136**	**140**	**145**	148	152	157	162
9	**39**	**46**	53	60	66	71	77	83	88	94	100	105	111	117	121	127	132	138	142	148	153	159	164	169	173	178
10	**43**	**51**	58	65	72	78	85	91	97	103	110	116	122	129	133	139	145	151	155	161	167	172	177	183	189	195
11	**47**	**56**	63	71	78	85	92	99	106	113	120	127	134	141	146	151	157	163	168	174	180	187	192	199	206	212
12	**51**	**61**	68	76	84	92	100	108	115	123	128	136	143	150	156	163	171	177	183	189	195	201	207	213	219	226
13	**55**	**66**	73	82	90	99	108	116	124	133	138	146	153	161	167	175	183	190	197	204	210	217	222	229	235	242
14	**59**	**70**	78	88	96	106	116	124	133	143	148	157	164	172	179	187	195	203	211	219	225	232	238	245	251	258
15	**63**	**75**	83	93	102	111	121	132	139	149	156	166	173	181	189	198	207	216	225	234	240	246	253	260	266	272
16	**67**	**80**	88	99	108	118	128	138	148	158	166	177	184	191	199	209	219	229	239	249	256	261	268	275	282	288
17	**71**	**85**	93	105	114	125	135	145	156	166	174	185	192	200	209	220	231	242	253	264	272	277	284	291	298	304
18	**75**	**90**	98	111	120	131	142	152	163	174	183	193	201	210	219	229	240	251	262	273	282	288	296	303	311	318
19	**79**	**94**	**103**	116	126	138	149	160	171	182	192	201	211	220	229	239	250	261	272	283	292	300	308	316	324	332
20	**83**	**98**	**108**	122	132	144	156	167	178	190	200	210	220	229	239	249	260	271	282	293	303	312	321	329	337	346
21	**87**	**102**	**113**	**128**	138	150	163	174	186	198	208	218	229	239	249	259	270	281	292	303	314	324	332	341	350	359
22	**91**	**106**	**118**	**134**	144	157	170	181	194	205	216	227	238	249	260	270	281	292	303	314	325	335	344	354	363	372
23	**95**	**110**	**123**	**140**	150	163	176	189	201	212	225	236	247	259	270	280	292	303	314	325	336	347	357	367	377	386
24	**99**	**114**	**127**	144	156	169	183	196	208	220	233	244	256	268	280	291	302	313	324	335	347	358	370	380	390	400
25	**103**	**118**	**132**	150	161	175	190	204	216	228	241	253	265	277	289	301	312	323	335	346	358	370	382	392	403	413
26	**107**	**122**	**137**	**156**	167	182	196	212	224	236	249	261	274	286	299	311	322	335	346	358	370	382	394	404	416	427
27	**111**	**126**	**142**	**162**	173	189	203	220	232	244	257	270	283	296	308	321	333	346	357	369	381	393	406	417	429	440
28	**115**	**130**	**147**	**168**	179	195	210	228	240	252	264	278	292	305	318	331	343	356	368	380	393	405	418	429	442	453
29	**119**	**134**	**151**	**172**	185	201	217	236	248	260	272	287	301	315	328	340	353	367	379	392	405	417	430	442	454	466
30	**123**	**138**	**156**	**177**	191	207	224	244	256	268	280	295	310	324	337	350	363	377	390	403	416	429	442	455	467	479

23. táblázat. $Z_{4,5}(m,n)$ alsó becslések

$m\backslash n$	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
4	**23**	**26**	**29**	**32**	**35**	**38**	**41**	**44**	**47**	**50**	**53**	**56**	**59**	**62**	**65**	**68**	**71**	**74**	**77**	**80**	**83**	**86**	**89**	**92**	**95**
5	**28**	**32**	**36**	**40**	**43**	**47**	**51**	**55**	**58**	**62**	**66**	**70**	**73**	**77**	**81**	**85**	**88**	**92**	**96**	**100**	**103**	**106**	**109**	**112**	**115**
6	**33**	**38**	**43**	**48**	51	56	**61**	**66**	69	73	77	81	85	89	**93**	**97**	**101**	**105**	**108**	**112**	**116**	**120**	**123**	**127**	**131**
7	**38**	**44**	**50**	**56**	59	64	69	74	79	84	88	93	97	102	106	111	115	120	124	129	133	138	142	147	151
8	**43**	**50**	**57**	**64**	67	72	78	83	89	94	100	105	110	115	121	126	131	136	141	146	150	155	159	164	168
9	**48**	**56**	**64**	**72**	75	80	86	92	98	104	110	116	122	128	133	139	144	150	155	160	165	170	175	180	185
10	**53**	**61**	69	77	83	88	95	102	108	115	121	128	134	140	146	152	158	164	169	175	180	186	191	197	202
11	**58**	**67**	75	83	91	96	104	111	118	126	132	140	147	154	159	165	171	177	184	190	196	202	208	214	220
12	**63**	**73**	81	90	98	104	112	120	128	135	144	152	160	168	173	179	185	191	198	205	211	218	224	230	237
13	**68**	**79**	87	96	105	112	121	129	137	146	153	162	170	179	185	192	198	205	212	219	226	233	240	246	253
14	**73**	**85**	93	103	112	120	130	138	147	157	164	173	182	191	198	205	212	219	226	233	240	248	255	262	269
15	**78**	**91**	99	110	120	128	138	147	157	168	174	184	193	202	210	218	225	232	240	247	255	263	271	278	286
16	**83**	**96**	105	116	127	136	147	156	166	177	185	195	205	215	222	231	239	247	255	262	269	277	286	294	302
17	**88**	**102**	111	123	134	144	155	165	175	186	196	205	215	225	233	242	251	260	268	276	284	292	300	309	318
18	**93**	**108**	117	130	141	152	163	173	184	195	207	216	226	236	245	255	263	273	281	290	299	308	316	324	333
19	**98**	**114**	123	136	148	159	171	182	193	204	216	226	237	247	257	267	276	285	294	304	314	323	331	340	349
20	**103**	**120**	129	143	155	166	180	190	202	214	225	236	247	258	269	279	289	299	308	318	328	338	347	356	366
21	**108**	**126**	135	150	162	174	189	199	211	223	235	246	257	269	280	291	302	312	322	332	343	352	362	372	382
22	**113**	**131**	141	157	169	182	197	207	220	232	244	256	267	279	291	303	315	325	336	346	356	366	377	387	397
23	**118**	**136**	147	164	176	189	205	216	229	242	253	266	277	289	302	314	327	338	350	360	370	380	391	401	412
24	**123**	**141**	153	171	183	197	213	224	238	251	263	276	287	299	313	325	339	351	363	373	384	394	405	416	427
25	**128**	**146**	**159**	177	190	205	220	233	247	260	273	285	298	310	324	336	351	364	376	387	398	409	420	431	443
26	**133**	**151**	**165**	184	197	212	228	241	256	270	282	295	308	321	334	347	362	376	389	401	412	423	435	446	458
27	**138**	**156**	**171**	191	204	219	235	249	265	279	292	305	319	332	346	358	374	388	401	414	426	437	449	461	473
28	**143**	**161**	**177**	**198**	211	226	243	258	274	288	301	314	329	343	357	370	385	400	413	427	440	451	463	475	488
29	**148**	**166**	**183**	**205**	218	233	251	266	282	297	311	324	339	353	368	382	396	411	425	440	453	465	477	490	503
30	**153**	**171**	**189**	**212**	225	240	259	274	291	305	320	334	349	364	379	393	407	423	437	453	466	479	492	504	518

24. táblázat. $Z_{4,6}(m,n)$ alsó becslések

$m\backslash n$	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
5	**29**	**33**	**37**	**41**	**45**	**49**	**53**	**57**	**61**	**65**	**69**	**73**	**77**	**81**	**85**	**89**	**93**	**97**	**101**	**105**	**109**	**113**	**117**	**121**	**125**
6	**34**	**39**	**44**	**49**	**54**	**58**	**63**	**68**	**73**	**78**	**82**	**87**	**92**	**97**	**102**	**106**	**111**	**116**	**121**	**126**	**130**	**135**	**140**	**145**	**150**
7	**39**	**45**	**51**	**57**	**63**	67	73	**79**	**85**	**91**	95	100	105	110	115	120	125	130	135	140	**145**	**150**	**155**	**160**	**165**
8	**44**	**51**	**58**	**65**	**72**	76	82	88	94	100	106	112	118	124	129	135	140	146	152	157	163	168	174	180	185
9	**49**	**57**	**65**	**73**	**81**	85	91	98	104	111	117	124	131	137	144	150	157	163	169	176	182	189	195	201	207
10	**54**	**63**	**72**	**81**	**90**	94	100	107	114	122	129	136	143	150	157	164	170	177	184	191	197	204	211	218	225
11	**59**	**68**	77	86	95	103	109	117	125	132	140	148	155	163	170	178	185	192	199	207	213	221	228	235	243
12	**64**	**74**	84	93	102	111	118	127	135	143	152	160	168	176	184	192	200	208	216	225	231	238	245	252	260
13	**69**	**80**	91	100	109	118	127	136	145	153	162	172	181	190	199	207	215	224	232	241	248	255	262	270	278
14	**74**	**86**	**98**	107	117	126	136	146	155	164	174	183	192	201	211	220	228	238	246	255	263	271	279	287	295
15	**79**	**92**	**105**	114	124	134	145	156	165	175	186	195	204	214	224	233	242	252	260	269	278	286	295	304	312
16	**84**	**98**	**112**	121	132	142	153	165	175	186	198	207	217	228	237	246	256	266	275	284	293	303	312	321	330
17	**89**	**103**	**117**	127	139	150	162	174	184	195	207	218	228	239	250	260	270	281	291	300	309	319	328	338	347
18	**94**	**109**	**123**	134	147	158	171	183	194	205	218	229	240	251	262	273	283	294	304	314	324	334	344	355	365
19	**99**	**115**	**129**	141	155	166	179	192	204	216	228	239	251	263	274	285	296	307	318	328	339	349	360	371	381
20	**104**	**121**	**135**	148	162	174	187	201	213	226	239	250	262	275	286	298	309	320	331	342	354	365	376	388	398
21	**109**	**127**	**141**	155	170	181	195	210	223	236	250	262	273	286	298	310	322	334	346	357	369	380	392	403	415
22	**114**	**133**	**147**	161	178	189	203	218	232	246	260	272	285	297	311	323	335	347	360	372	383	395	407	419	432
23	**119**	**138**	**152**	168	185	197	212	227	241	255	270	282	296	309	323	335	348	360	373	386	398	411	423	435	449
24	**124**	**144**	**158**	175	192	205	220	236	250	264	279	293	306	320	334	347	360	374	387	400	413	426	439	452	465
25	**129**	**150**	**164**	**182**	200	212	229	244	259	274	289	303	317	331	345	359	373	387	401	414	428	441	455	469	481
26	**134**	**156**	**170**	188	207	220	238	252	268	283	299	314	328	343	357	372	386	401	415	429	442	456	470	484	497
27	**139**	**162**	**176**	195	214	228	247	261	277	292	308	325	339	354	369	384	399	414	429	443	458	471	486	499	513
28	**144**	**168**	**182**	**202**	221	235	254	270	286	302	318	335	350	365	381	396	411	427	442	456	472	486	501	516	530
29	**149**	**173**	**187**	208	228	243	262	279	295	311	328	345	361	377	392	408	424	439	455	470	486	501	516	531	546
30	**154**	**178**	**193**	215	235	251	270	287	304	321	338	355	372	388	404	420	436	452	469	484	501	516	531	546	562

25. táblázat. $Z_{5,6}(m,n)$ alsó becslések

F.2. Nyilatkozat generatív mesterséges intelligencia alkalmazásáról

☐ **Nem használtam** semmilyen generatív MI segédeszközt.

☒ **Használtam** generatív MI segédeszközt. Az MI-vel generált tartalmakat ellenőriztem, a generált kimenetek valóságtartalmáról meggyőződtem, az alábbi táblázatban megfelelően jelöltem minden használatot.

Felhasználási módok	**Generatív MI eszköz(ök) neve**	**Érintett részek** (fejezet, oldalszám, hivatkozás)	**Használat becsült aránya** (felhasználási módonként)
Irodalomkutatás	Gemini 3 pro	Irodalomjegyzék: 55. old.	5%
Prompt lényegi része	Find this: Andrew Kay Efficient algorithms Zarankiewicz		
Programkód generálása	Gemini 3 pro	6.3., 52. old.	5%
Prompt lényegi része	Kisebb segédprogramok, scriptek létrehozása, táblázat formázás.		
Új ötletek, megoldási javaslatok generálása			
Prompt lényegi része			
Vázlat létrehozása (szövegstruktúra, vázlatpontok)			
Prompt lényegi része			
Szövegblokkok létrehozása	Gemini 3 pro	Abstract: 4. old.	90%
Prompt lényegi része	Please translate to English suitable for a Master's thesis.		
Képek generálása illusztrációs célból			
Prompt lényegi része			
Adatvizualizáció, grafikonok generálása adatpontok alapján	Gemini 3 pro	5.1., 42. old., 8. ábra, 5.2., 45. old., 9. ábra, 5.3., 47. old., 10. ábra	60%
Prompt lényegi része	Generate a mermaid class diagram from this codebase.		
Prezentáció készítése			
Prompt lényegi része			
Egyéb (nevezze meg)			
Prompt lényegi része			
Összesített százalékos érték (a feladat érdemi részére nézve):			5%
Összesített érték rövid, szöveges indoklása: A teljes programkódot magamtól írtam, kiegészítő script-ekre használtam AI-t. Például könyvtár generálás, eredmény összehasonlítás. A dolgozatban az Abstract rész a dokumentum kevesebb, mint 2%-át teszi ki. Az osztálydiagram generálása, nem adott kellően jó megoldást, ezért kézzel kellet javítanom. A felhasználás a dolgozat töredékét adja, ezért az AI-val készített tartalom becsült aránya 5%.			