ERDŐS SZILVIA[9] - KŐVÁRI BENCE[10]

BEOSZTÁSTERVEZÉS LINEÁRIS PROGRAMOZÁSSAL

Absztrakt

Az automatikus beosztástervezés évtizedek óta kutatott téma az irodalomban, ahol eddig elsősorban heurisztikus és mesterséges intelligencia alapú módszerek hoztak sikereket. A záróvizsga beosztások készítése a témakör egy speciális részfeladata, ahol különleges követelmények korlátozzák az állapotteret.

Korábbi kutatásaink során a probléma automatizálására két módszert készítettünk, melyeket a Budapesti Műszaki és Gazdaságtudományi Egyetem (BME) 100 hallgatójának záróvizsga beosztásának elkészítésével teszteltük. Mivel egyik sem hozott kielégítő eredményt, kidolgoztuk a probléma lineáris programozás alapú megoldását, mely a korábbiaknál jobb eredményt ad.

A nemzetközi irodalmat áttekintve megoldásunk egyedi módon átfogó választ ad a záróvizsgabeosztások tervezésének több kérdésére is.

Kulcsszavak: beosztástervezés, egészértékű lineáris programozás, záróvizsgabeosztás

SCHEDULE PLANNING WITH LINEAR PROGRAMMING

Abstract

The automatic generation of schedules has been in focus of researches for decades, where methods based on heuristics and artificial intelligence are the most successful. Final exam scheduling is a special subtask of this, where special requirements restrict the state space.

Along our researches we elaborated the algorithms for the problem, which were tested on a real test set, which contained the registration of 100 students from Budapest University of

[9] Budapesti Műszaki és Gazdaságtudományi Egyetem, doktorandusz

[10] Budapesti Műszaki és Gazdaságtudományi Egyetem, Villamosmérnöki és Informatikai Kar, Automatizálási és Alkalmazott Informatikai Tanszék, egyetemi docens

Technology and Economics. None of them gave a good solution, so an integer linear programming approach was invented. The results show that for this complexity there is an optimal solution and it covers more abilities of final exam scheduling than the algorithms discussed before in the literature for similar problems.

Keywords: Scheduling, Integer linear programming, Final exam scheduling

1. Bevezetés

A felgyorsult digitalizálódó társadalmunkban egyre fontosabb szerepet kap a korábban manuálisan végzett tevékenységek automatizációjára való törekvés. Különböző intelligens és mesterséges intelligencia alapú információs rendszereket készítünk az adaptív digitalizálás érdekében.

Kutatásaink során mi is egy ilyen információs rendszer elkészítését tűztük ki célul, mégpedig az automatikus beosztástervezés témakörében tevékenykedünk. Beosztások készítésére az élet minden területén szükség van, lehet szó akár oktatási intézmények órarendjéről, napirendi beosztásokról, bármilyen területen dolgozók munkájának beosztásáról, vagy akár projektek folyamatainak beütemezéséről.

Évtizedek óta kutatott téma az irodalomban a beosztások automatikus elkészítése. Mivel a vizsgálandó állapottérre kisebb bementi változások is exponenciális hatással vannak, elsősorban heurisztikus és mesterséges intelligencia alapú módszerek voltak sikeresek.

A kutatásaink alatt korábban két különböző algoritmust dolgoztunk ki beosztástervezési folyamatok megoldására: egy genetikus algoritmus alapú megoldást és egy heurisztikus magyar módszer alapú algoritmust. Mivel egyik sem hozott teljesen kielégítő eredményt, kidolgoztuk a probléma lineáris programozás alapú megoldását.

Felépítettünk egy lineáris programozás alapú saját modellt a záróvizsgabeosztás problémájára, mely minden korábbi algoritmusunknál hatékonyabb és igazságosabb eredményt ad a BME adatbázison.

Eredményeink jól mutatják, hogy erre a komplexitású feladatra lehetséges az adott kritériumrendszer mellett optimális megoldást adni, mely minden szigorú követelményt teljesít.

Hosszú távon közvetlenül az egyetemi vizsgabeosztások elkészítésének teljes automatizálását oldhatja meg az elkészült algoritmusunk, mely a megfelelő követelmények meghatározásával kiterjeszthető az élet bármely más területeinek beosztástervezési problémáira.

2. Irodalmi háttér

Évtizedek óta próbálkoznak bizonyos különböző feladatokra minél hatékonyabb, minél gyorsabb és minél jobb eredményt adó megoldásokat keresni. Ezt bizonyítja, hogy évről évre megrendezésre kerül a nemzetközi beosztástervezési verseny *(ITC 2019 Discussion Forum*

2019), valamint már nagy hagyományra visszatekintő, két évente megrendezésre kerülő nemzetközi konferencia, ami az automatikus beosztáskészítés elméleti és gyakorlati szempontjait taglalja *(PATAT Conferences 1995)*.

A beosztástervezésnek több különböző alap fajtáját különböztetik meg az irodalomban *(Schaerf 1999)*, amelyek mentén a kutatások fő vonala zajlik napjaink során is. Az egyik az Examination timetabling, vagyis a vizsgarendek beosztása egyetemi vagy főiskolai környezetben, mely probléma áll a legközelebb az általam vizsgált témakörhöz.

Ez a terület azt a problémát hivatott tárgyalni, hogy egy felsőoktatási intézmény vizsgaidőszakának beosztását hogyan készítsük el. Itt több célunk is lehet a beosztás elkészítése során. Lehet például az, hogy azoknak a vizsgáknak, amelyeket vélhetően ugyanazoknak a hallgatóknak kell teljesíteni, ne legyenek egy időpontban. Ezenkívül a hallgatók vizsgáit amennyire csak lehet, „szét kell szórni”, ami azt jelenti, hogy minél több idejük legyen egy-egy vizsga között, ezzel elősegítve a vizsgák sikerességét. A termekre is lehet figyelni, ha a maximális befogadóképességet nem lépheti túl a vizsgázók száma, de érdekesség, hogy több kisebb létszámú, azonos időtartamú vizsgát meg is lehet tartani akár egy teremben.

Az egyetemi vizsgák beosztásának megoldása során több különböző szempontot vehetünk figyelembe attól függően, hogy az adott algoritmussal a vizsgák mely tulajdonságaira szeretnénk jobban koncentrálni. Az irodalomban is eltérő megoldásokat találunk, attól függően, hogy ki milyen feltételeket tartotta szükségesnek a saját algoritmusa elkészítése során.

Wijgers, és Hoogeveen 2007-ben írt tanulmányában például előre meghatározott napokon belülre osztottak be vizsgákat. A hallgatóknak itt több vizsgájuk is lehetett, viszont megadták feltételnek, hogy egy hallgató egy napon csak egy vizsgán vegyen részt. Figyelembe vették az oktatók elérhetőségét, azonban azzal már nem számoltak, hogy mennyire terhelhető egy-egy oktató.

A 2010-ben Al-Yakoob, Sherali és Al-Jazzaf által írt cikkben figyelembe vették a termek elérhetőségét, sőt, még bizonyos nemhez kötött feltételeket is bevezettek. Azonban nem kezelték az egyedi eseteket, ha bizonyos oktatókat csak meghatározott vizsgához lehet hozzárendelni.

Az egyetemi szóbeli vizsgák egy szűkített problémájára adott megoldást Kochaniková és Rudová egy 2013-ban írt cikkében. Itt minden egyes vizsgához egy vizsgáztatót és egy hallgatót rendeltek, miközben figyelték a szereplők elérhetőségeit. Az oktatók viszont sokszor vizsgáztattak, a köztük lévő terheléseloszlást nem vették figyelembe.

Az egyetemi vizsgabeosztás témakörében írt Bergmann, Fischer és Zurheide 2014-es cikkében figyeltek a különböző erőforrások, termek, hallgatók egyenletes eloszlására, valamint lehetőséget adtak arra is, hogy az egyes vizsgák egymással párhuzamosan kerültjenek megtartásra. Azonban nem lehetett speciális szerepköröket kezelni.

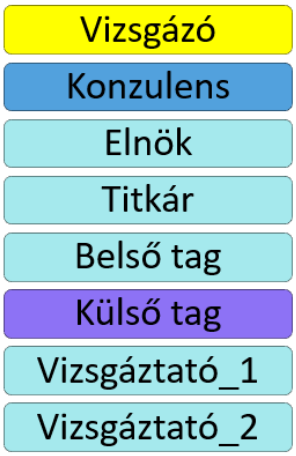
Ivancevic, Knezevic és Lukovic 2014-es cikkében nem vette figyelembe az oktatók túlterheltségét, és elérhetőséget sem kért be az oktatóktól. Azonban az általuk leírt algoritmus támogatta a párhuzamos vizsgákat, és figyelt a köztük lévő ütközésekre.

Aslan, Şîmşek és Karkacier 2017-es írásában hangsúlyt fektetett a terhelések egyenletes eloszlására az oktatók esetében, valamint az esetleges idő- és térbeli ütközéseket is kezelték. Mindezek mellett a felállított követelményeik között nem tettek fontossági különbséget, azokat egyformán fontosnak kezelték.

Látható az előző példákból is, hogy hiába beszélünk az irodalomban ugyanarról a problémáról (*Examination timetabling*), mégis más-más kutatók más rendszer szerint építik fel azt, és így az elkészített modellük, illetve algoritmusuk csak bizonyos feltételek mentén ad megfelelő beosztást.

3. Problémakör

Először is bemutatásra kerül a szóbeli záróvizsga beosztás működését, azon belül is a Budapesti Műszaki és Gazdaságtudományi Egyetem (BME) Villamosmérnöki és Informatikai Karára (VIK) koncentrálva. Ennek pontos leírását az egyetemi Tanulmányi és Vizsgaszabályzat *(Rektori Kabinet Oktatási Igazgatóság, A Szenátus X./10./2015-2016. (2016. VII. 11.) számú határozata A BME TANULMÁNYI ÉS VIZSGASZABÁLYZATÁRÓL, 2016.)*, valamint annak kari kiegészítése *(BME VIK BSc szakdolgozat, záróvizsga, oklevél szabályzat a BME, 2017)* tartalmazza. Az 1. ábra tartalmazza egy záróvizsga számunkra releváns szereplőit.



1. ábra: Egy záróvizsga résztvevői

2. *Záróvizsgabeosztás*

Természetesen a résztvevők között szerepel egy vizsgázó hallgató, akinek a vizsgája esedékes. Hozzá tartozik a konzulense (témavezetője), aki segítette a szakdolgozat vagy a diplomamunka elkészítésében. Minden vizsga levezetéséhez szükség van egy elnökre, egy titkárra és egy belső tagra, amely szerepköröket csak bizonyos feltételeket teljesítő oktatók tölthetnek be. Látható még a külső tag, aki nem áll jogviszonyban a szervező karral, nem függ a többi szereplőtől, csupán a saját elérhetősége a meghatározó. A vizsgázó képzési szintje, szakja és a választott vizsgatárgyai határozzák meg, hogy ki(k) lehet(nek) vizsgáztató(k) egy vizsgán.

A szabályoknak megfelelő intervallumon belül meghatározzák a vizsgázók száma és az elérhető termek alapján, hogy legalább hány nap szükséges a vizsgák lebonyolításához, valamint, hogy pontosan mely napok lesznek ezek. Az oktatók megadják elérhetőségeiket, valamint az is befolyásoló tényező, hogy melyik oktató melyik képzési szinten, melyik szakon lehet a vizsgáztató bizottság tagja.

A záróvizsgák különlegessége, hogy nem szükséges minden szerepkör betöltéséhez külön-külön oktatónak megjelenni, lehetséges bizonyos feladatok összevonása. Erre egy példa, hogy az elnök lehet egyben a hallgató konzulense és egy vizsgatárgyból vizsgáztató is, a titkár szintén lehet konzulens és vizsgáztató. Viszont fontos, hogy az elnök és a titkár nem összevonható szerepkörök.

3. *Komplexitás*

A beosztástervezés bizonyítottan NP-teljes probléma *(Pinedo–Michael 2016),* ez adja a nehézségének egy részét. Ezen felül a záróvizsgabeosztás bonyolultságát két részre lehet bontani: először is az állapottér nagyságát mindössze egy 100 fős hallgatói csoport esetében hihetetlenül nagy, nem lehet legenerálni minden lehetséges beosztást. 100 fős hallgatói csoport esetében a lehetséges beosztások száma 10^{462} nagyságrendű.

Másrészt bizonyos feltételek, melyeket teljesíteni kell egy beosztásnak, sokszor ellentmondásosak. A leginkább különleges szempont az oktatók terhelésének eloszlása, hiszen erre nagyon különböző feltételeket lehet megfogalmazni. Az egyenletes eloszlás fogalma már nevében is ellentmondásos, hiszen, ha jobban belegondolunk, egyes oktatóknak biztosan sokkal több konzultált hallgatója van, mint másoknak, ami már önmagában ellehetetleníti a feladat maximális teljesítését. Ezenkívül bizonyos szerepköröket csak bizonyos oktatók tölthetnek be (pl. elnök, titkár), ami már önmagában egyenlőtlenné teszi a beosztást.

4. *Formalizálás*

A probléma megoldásához szükség volt először is a záróvizsgabeosztás problémájának formális megfogalmazására. A záróvizsgabeosztás szabályokban meghatározott kereteit és a gyakorlatban alkalmazott metodikákat közös nevezőre hozva meghatároztunk bizonyos követelményeket, melyeket a beosztásnak teljesíteni kell.

Ezeknek a feltételeknek két típusa van. Az egyik csoportba azokat soroltuk, melyek teljesülése feltétlenül szükségszerű a beosztás megvalósíthatósága szempontjából. Ezeket szigorú, azaz *hard* követelményeknek neveztük el.

A másik típusba azok a feltételek tartoznak, melyek teljesülése nem feltétlenül szükségszerű, de minél több teljesül belőlük, annál jobbnak tekinthető a beosztás. Ezeket gyenge, vagy *soft* követelményeknek nevezzük.

A teljesség igénye nélkül az 1. táblázat bemutat néhány gyenge, míg a 2. táblázat pár szigorú követelményt.

1. táblázat: A záróvizsgabeosztás gyenge követelményei és a hozzájuk tartozó pontszámok

Követelmény	*Pontszám*
Elnökök saját konzultált hallgatója nem az elnök saját blokkjában vizsgázik	*2 pont máshol vizsgázó hallgatónként*
Titkárok saját konzultált hallgatója nem a titkár saját blokkjában vizsgázik	*1 pont máshol vizsgázó hallgatónként*
Vizsgáztató nem az elnök, pedig más napon elnök	*1 pont*
Az elnök képzési szintje nem egyezik meg a vizsgázó képzésével	*1 pont*
A belső tag nem elérhető a vizsga alatt	*5 pont*
A konzulens nem elérhető a vizsga alatt	*5 pont*
Az elnök terhelése a többi elnöktől eltérő	*30-0: az eltéréstől függ*
A titkár terhelése a többi titkárétól eltérő	*30-0: az eltéréstől függ*
A belső tag terhelése a többi tagtól eltérő	*30-0: az eltéréstől függ*

2. táblázat: A záróvizsgabeosztás szigorú követelményei és a hozzájuk tartozó pontszámok

Követelmény	*Pontszám*
Az elnök nem elérhető a vizsga alatt	*1000 pont*
A titkár nem elérhető a vizsga alatt	*1000 pont*
A vizsgáztató nem elérhető a vizsga alatt	*1000 pont*
Az elnök megváltozik a blokkban (változásonként)	*1000 pont*
A titkár megváltozik a blokkban (változásonként)	*1000 pont*
A vizsga megkezdődik 8:00 előtt	*140-0: kezdés időpontjától függ*
A vizsga 18:00 után végződik	*140-0: zárás időpontjától függ*

4. Egészértékű lineáris programozás alapú modell

Korábban egy genetikus *(Erdős–Kővári 2019)* és egy magyar módszer alapú heurisztikus algoritmus *(Erdős–Kővári 2019)* elkészítésével próbálkoztunk, melyek mindegyikével sikerült az összes fontos követelményt teljesíteni, azonban a terhelések eloszlása és a szerepkörök összevonásai nem teljesültek helyesen. Ezért egy lineáris programozás alapú algoritmust is megalkottunk, melyben több keresett változó lineáris függvényének szélsőértékét kell meghatározni, bizonyos korlátozó feltételek mellett.

5. *Döntési változók*

A döntési változóinkat x jelöli, melyek binárisak, és egyben azt jelölik, hogy egy adott oktató egy adott vizsgára be van osztva vagy nincs. A 3. táblázat szerint kell elképzelni őket, ahol az egyes sorokban és oszlopokban az található, hogy az adott oktató az adott időpontban elérhető-e vagy sem.

3. táblázat: Oktatók bináris döntési változóinak reprezentációja

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
Ács Judit	1	1	1	0	0	0	0	1	1	1
Albert István	0	0	0	0	1	1	1	1	1	1
Asztalos Márk	0	0	0	0	0	1	1	1	1	1
Benedek Zoltán	0	1	1	1	1	1	0	0	0	0
Blázovics László	1	1	1	1	1	0	0	0	0	0
Charaf Hassan	0	0	0	0	0	1	1	1	1	1
Cserkúti Péter	0	0	0	1	1	0	0	0	0	0
Csorba Kristóf	1	0	0	1	0	1	1	0	0	0
Dudás Ákos	0	0	0	0	0	0	1	1	1	1
Dunaev Dmitriy	1	1	1	0	0	0	0	0	0	0
Ekler Péter	1	1	1	1	1	0	0	0	0	0

6. *Célfüggvény*

Alább az algoritmusunk célfüggvénye látható, mely valójában az egyes követelmények teljesülését vizsgálja, és azok alapján keresi a legmegfelelőbb beosztást.

$$min \sum_{i}^{i\in I}\sum_{t}^{t\in T}\left(x_{i,t} * Cost_{i,t}\right) + \sum_{p}^{p\in P}\left(x_p^{\alpha} + x_p^{\beta}\right) + \sum_{r}^{r\in R}\left(x_r^{\alpha} + x_r^{\beta}\right) + \sum_{m}^{m\in M}\left(x_m^{\alpha} + x_m^{\beta}\right)$$

A célfüggvény elején a $Cost_{i,t}$ egy pozitív egész konstans, mely megfelel annak a büntetőpontnak, mikor az $i$oktató a t idősávban nem elérhető. Ezt szorozzuk össze az adott i oktató t időszeletbeli beosztásához kapcsolódó döntési változóval, és ezt megtesszük minden oktatóval minden időszeletben, melyeket mind összegzünk Ennek az összegnek a minimalizálása által elérjük, hogy lehetőleg akkor legyenek beosztva oktatók, amikor rá is érnek.

A következő három tagból álló forma sorban az elnökök, titkárok, valamint a belső tagok terhelésének eloszlására vonatkoznak. Ezek a változó értékek valójában azt jelentik az egyes résztvevők esetében, hogy mekkora az eltérés az ideális beosztási mennyiséghez képest, amit szintén minimalizálni szeretnénk.

7. *Korlátozó feltételek*

Összesen 260 lineáris korlátozó feltételt kellett meghatároznunk, hogy teljesüljenek az egyes követelmények. Ezek mindegyike a döntési változókból alkotott lineáris egyenlőség

vagy egyenlőtlenség. A következőkben ezek közül emelünk ki néhányat, hogy jobban el lehessen képzelni, miről is van szó.

Vannak a beosztás alapjaira vonatkozó feltételek is, ilyen például a következő, ami annak érdekében született, minden idősávba legyen beosztva hallgató, viszont ne is legyen egynél több, így a hallgatók beosztására vonatkozó döntési változók ($x_{s,t}$) összege minden időszeletre vonatkozóan ($t \in T$) legyen pontosan 1, melyet az alábbi feltétel ír le.

$$\sum_{s} x_{s,t} = 1 \quad s \in S, \forall t \in T$$

Emellett fontos, hogy minden hallgató vizsgázzon le, de ne vizsgázzon egy hallgató többször. Így arra is kellett egy feltételt megfogalmazni, hogy minden hallgató pontosan egy időszeletre legyen beosztva. Így a hallgatók döntési változóit ($x_{s,t}$) hallgatókként külön adjuk össze, tehát minden hallgatóra megvizsgáljuk külön-külön, hogy az összes időszeletre a döntési változóik összege 1 legyen.

$$\sum_{t} x_{s,t} = 1 \quad t \in T, \forall s \in S$$

A vizsgáztatók beosztása egy bonyolult feladatnak adódott amiatt, hogy csak annyit tudunk, hogy amikor a hallgató be van osztva ($x_{s,t} = 1$) egy $t \in T$ időszeletben, akkor legyen a vizsgatárgyából vizsgáztatható oktatók közül legalább egy beosztva, viszont egy vizsgáztató több hallgatót is vizsgáztat, így ez a viszony nem kölcsönös. Ezt úgy fogalmaztuk meg, hogy az összes lehetséges vizsgáztató döntési változóit, ha összeadjuk az adott t időszeletben, akkor az összegnek legalább 1-et kell kiadnia. (Az összeg $\sum x_{a_{c_s},t}$ ahol c_s jelenti az adott s hallgató vizsgatárgyát, az ebből a tárgyból vizsgáztatható oktatókat a_{c_s} jelöli, azokat az oktatói döntési változókat pedig $x_{a_{c_s},t} \subseteq x_{i,t}$ jelöli, amely ezekhez az oktatókhoz tartoznak.)

Így, ha kivonjuk a hallgató döntési változójából a hallgató vizsgatárgyából vizsgáztatható oktatók döntési változóinak az összegét, akkor biztosan legfeljebb 0-t kapunk eredményül. Mindez a következőképpen lett formalizálva:

$$x_{s,t} - \sum x_{a_{c_s},t} \leq 0 \quad \forall t \in T, \forall s \in S$$

Egy további fontos követelmény, hogy az elnökök teljes blokkokra legyenek beosztva, vagyis egy teljes délelőttre/délutánra, és ne csak 1-1 vizsgára a nap során. Ehhez számított döntési változókat hoztunk létre, amelyek jelzik, hogy egy teljes blokkra van-e beosztva, vagy sem az adott oktató. Ezt legegyszerűbben az adott blokkon belüli az időszeletekre vonatkozó döntési változókon végzett *és* művelettel lehet meghatározni, melyet az alábbi feltétel felvételével oldottunk meg minden egyes blokkban minden elnöki szerepet betölthető oktatóra:

$$x_{p,b} = \bigwedge_{t \in b} x_{p,t} \qquad \forall p \in P, \forall b \in B$$

Szükség van arra, hogy ennek a segítségével meg is fogalmazzuk azt a feltételt, hogy az elnökök teljes blokkra legyenek beosztva, mégpedig mindegyik blokkban pontosan egy elnök legyen. Mindezt úgy értük el, hogy az elnökökre vonatkozó számított döntési változókat ($x_{p,b}$) összeadjuk egy-egy blokkban, és ennek az összegnek meghatározzuk, hogy 1 értéket vegyen fel, így fixen egy elnök szerepét betölthető oktató lesz beosztva az adott teljes blokkban. Ezt a feltételt meghatározzuk minden blokkban, így végül formálisan az alábbi feltételt építettük fel:

$$\sum_{p} x_{p,b} = 1 \qquad p \in P, \forall b \in B$$

A többi feltétel definiálása után felépült a teljes modellünk a záróvizsgabeosztás problémájára, melyben meghatároztunk minden általunk elvárt követelményt, és így az algoritmus futtatásával elkészült a beosztás.

5. Eredmények

Az elkészült lineáris programozás alapú beosztás különösen jónak mondható, igazságosabb és használhatóbb megoldást adott a korábbiaknál.

Az elkészült beosztás vizsgálatára egyrészt felhasználtuk a Gurobi *(https://www.gurobi.com/ 2007)* adta lehetőségeket, ugyanis a modell futtatása során automatikusan megjelenít bizonyos információkat. Ezeket a 2. ábra tartalmazza. Az információk között található, hogy 260 korlátozó feltételt adtam meg összesen, 15106 változót definiáltam a modellemben, melyek közül 15060 bináris. Látható továbbá, hogy

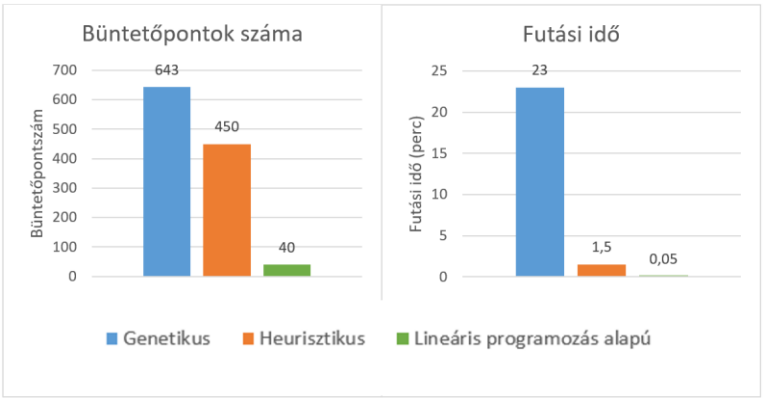
összesen 2.66 másodpercig tartott a teljes algoritmus lefutása 8 szálon futva, valamint az is, hogy ezalatt talált megoldást a megadott feltételek mellett.

```
Optimize a model with 20801 rows, 15106 columns and 94831 nonzeros
Model has 260 general constraints
Variable types: 0 continuous, 15106 integer (15060 binary)
Coefficient statistics:
  Matrix range     [1e+00, 5e+00]
  Objective range  [1e+00, 5e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+01]
Presolve removed 12029 rows and 3239 columns
Presolve time: 0.71s
Presolved: 8772 rows, 11867 columns, 63706 nonzeros
Variable types: 0 continuous, 11867 integer (11837 binary)

Root relaxation: objective 5.000000e+00, 7008 iterations, 1.40 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

H    0     0                    5.0000000 -106.00000  2220%     -    2s
     0     0     cutoff    0         5.00000    5.00000  0.00%     -    2s

Explored 0 nodes (9287 simplex iterations) in 2.66 seconds
Thread count was 8 (of 8 available processors)

Solution count 1: 5

Optimal solution found (tolerance 1.00e-04)
Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0000%
Obj: 5
```

2. ábra: Gurobi megoldó által készített összefoglaló az LP alapú beosztásról

Emellett az elkészült beosztást kézi elemzés alá vetettük, még pedig megállapítottuk, hogy csak és kizárólag terhelésekre vonatkozó követelmények nem teljesültek. A vizsgálat során kiderült, hogy azért történt ez, mert ezeknek az oktatóknak sokkal több hallgatójuk van másoknál. Ezért azokon a vizsgákon, mikor a hallgató vizsgázik, mindenképp jelen kell lenniük, és a szerepkörök összevonása miatt az algoritmus a belső tag szerepébe is őket osztotta be, ami megfelelő eljárás, hiszen így nem kell plusz oktatót behívni az adott vizsgára.

Összehasonlítottuk a két korábbi és a mostani LP algoritmus által elért legjobb beosztásokat, melyeket a 3. ábra tartalmaz. Látható, hogy messze a legjobb eredményt érte el mind büntetőpontszám tekintetében, mind pedig futási időben az új lineáris programozás alapú megoldás.



3. ábra: Záróvizsgabeosztási algoritmusok összehasonlítása

Az algoritmusok átfogóbb tulajdonságai mentén összehasonlítást végeztünk a saját lineáris programozás alapú algoritmusunk, valamint az irodalmi áttekintésben is taglalt cikkek között. Ennek eredményét a 4. táblázat mutatja be, ahol ✓ szimbólum jelzi azt, ha az algoritmus az adott képességet teljesíti, × jel pedig azt jelzi, hogy nem veszi figyelembe az adott tulajdonságot, nem teljesíti azt. A ✓/× jelentése, hogy az adott funkció ugyan elméletben már ki lett találva és az algoritmus fel is van rá készítve, tényleges leimplementáció még nem történt.

4. táblázat: Saját algoritmus képességeinek összehasonlítása a releváns irodalommal

	hard és soft követelmények megkülönböztetve	terheléseloszlás	elérhetőség	blokkok kezelése	párhuzamos vizsgák támogatása	specifikus oktatók a vizsgákhoz
saját lineáris programozás alapú algoritmus	✓	✓	✓	✓	✓/×	✓
Wijgers, és Hoogeveen (2007)	×	×	✓	✓	×	×
Al-Yakoob, Sherali és Al-Jazzaf (2010)	✓	✓	✓	×	✓	×
Kochaniková és Rudová (2013)	✓	×	✓	×	✓	✓
Bergmann, Fischer és Zurheide (2014)	✓	✓	✓	×	✓	×
Ivancevic, Knezevic és Lukovic (2014)	×	×	×	✓	✓	✓
Aslan, Şîmşek és Karkacier (2017)	×	✓	×	×	✓	×

Mindezek alapján látható, hogy algoritmusunk a nemzetközi irodalmat áttekintve az eddigieknél egyedibb módon átfogó megoldást ad a záróvizsgabeosztások tervezésének több kérdésére is, több különböző képességet ötvöz, mint amennyit az irodalomban taglalt hasonló témakörű tanulmányok.

6. Összefoglalás

Eredményeink jól mutatják, hogy erre a komplexitású feladatra lehetséges az adott kritériumrendszer mellett optimális megoldást adni. A nemzetközi irodalmat áttekintve megoldásunk egyedi módon átfogó választ ad a beosztások tervezésének több kérdésére.

Hosszú távon az egyetemi vizsgabeosztások elkészítésének teljes automatizálását oldhatja meg az elkészült algoritmusunk némi bővítés után, mely a megfelelő követelmények meghatározásával kiterjeszthető az élet bármely más területeinek beosztástervezési problémáira is.

Köszönetnyilvánítás

Project no. FIEK_16-1-2016-0007 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Centre for Higher Education and Industrial Cooperation - Research infrastructure development (FIEK_16) funding scheme.

Irodalomjegyzék

AL-YAKOOB, S. M., SHERALI, H. D.–AL-JAZZAF, M. (2010) A mixed-integer mathematical modeling approach to exam timetabling. *Computational Management Science*.

ASLAN, E., ŞIMSEK, T.–KARKACIER, A. (2017) A Binary Integer Programming Model For Exam Scheduling Problem With Several Departments. *13th International Conference on Knowledge, Economoy and Management*.

BERGMANN, L. K., FISCHER, K.–ZURHEIDE, S. (2014) A linear mixed-integer model for realistic examination timetabling problems. *10th International Conference of the Practice and Theory of Automated Timetabling*.

BME VIK BSC SZAKDOLGOZAT, ZÁRÓVIZSGA, OKLEVÉL SZABÁLYZAT (2017. 06 07). https://www.vik.bme.hu/document/1343/original/BSc-ZV-170607.pdf. [Letöltve: 2018. 10. 23.].

ERDŐS S., KŐVÁRI B. (2019) Algorithm based on Hungarian Method for Final Exam Scheduling. *Automation and Applied Computer Science Workshop.* Budapest.

ERDŐS S., KŐVÁRI, B. (2019) Genetic algorithm based solution for final exam scheduling. *MultiScience - microCAD International Multidisciplinary Scientific Conference*.

GUROBI (2007) Gurobi Optimization, LLC. *https://www.gurobi.com/*. [Letöltve: 2019].

ITC 2019 DISCUSSION FORUM (2019.) *ITC 2019: International Timetabling Competition*. https://www.itc2019.org/home. [Letöltve: 2018. október 24.].

IVANCEVIC, V., KNEZEVIC, M.–LUKOVIC, I. (2014) A Course Exam Scheduling Approach based on Data Mining. *Frontiers in Artificial Intelligence and Applications*.

KÄLLMAN, J. (2017) *EPPlus*. https://www.nuget.org/packages/EPPlus/4.5.0.1-beta. [Letöltve: 2017. december 3.].

KOCHANIKOVÁ, B., RUDOVÁ, H. (2013) Student Scheduling for Bachelor State Examinations.

PATAT CONFERENCES (1995) http://patatconference.org/index.html. [Letöltve: 2018. október 24.].

PINEDO, MICHAEL, L. (2016) *Scheduling: Theory, Algorithms, and Systems.*

REKTORI KABINET OKTATÁSI IGAZGATÓSÁG (2016. szeptember 1) *A Szenátus X./10./2015-2016. (2016. VII. 11.) számú határozata A BME TANULMÁNYI ÉS VIZSGASZABÁLYZATÁRÓL.* http://www.kth.bme.hu/document/2061/original/BME_TVSZ_2016%20elfogadott_mod_20180801_web.pdf. [Letöltve: 2018. október 2.].

SCHAERF, A. (1999) A Survey of Automated Timetabling. 87.

WIJGERS, R., HOOGEVEEN, J. (2007) Solving the examination timetabling problem.