

Eőri Dániel

VALÓS IDEJŰ ÜLŐHELYKIOSZTÁS TÁMOGATÁSA MESTERSÉGES INTELLIGENCIA SEGÍTSÉGÉVEL

Témavezető: Dr. Fogarassyné dr. Vathy Ágnes

Műszaki Informatikai Kar, Rendszer- és Számítástudományi Tanszék

Pannon Egyetem
Intézményi Tudományos Diákköri Konferencia
2022



TARTALOMJEGYZÉK

1. Bevezetés.....	1
2. Elméleti háttér.....	3
2.1. A probléma bemutatása	3
2.1.1. Teremkonfigurációk	4
2.1.2. Ültetési szabályok.....	6
2.2. Irodalmi áttekintés	7
2.3. A probléma lehetséges megoldása klasszikus allokációs módszerekkel.....	8
2.4. A probléma mesterséges intelligencia alapú megoldása	9
2.4.1. Gépi tanulás	9
2.4.2. Passzív megerősítéses tanulás az online ülőhelykiosztás optimalizálására	11
2.4.2.1. Az ülőhelyek minősítése	11
2.4.2.2. Ültetési stratégia.....	14
2.5. Implementáció.....	14
3. Eredmények	16
3.1. Tesztesetek.....	16
3.2. Ülőhelyfoglalás Best Fit, Worst Fit, Next Fit és First Fit algoritmusokkal.....	16
3.3. Az ülőhelyek minősítésének eredménye	21
3.3.1. Alapalgoritmusok használatával	21
3.3.2. Korrekciós technikák és szimulációk száma	26
3.4. Ülőhelyfoglalás passzív megerősítéses tanulást alkalmazó mesterséges intelligencia algoritmussal	31
4. Összefoglalás	38
Irodalomjegyzék	40

ÁBRAJEGYZÉK

1. ÁBRA: AZ A) A NEM ELTOLT TEREMKONFIGURÁCIÓRA, MÍG A B) AZ ELTOLT TEREMKONFIGURÁCIÓRA, C) A FOLYOSÓS NEM ELTOLT TEREMKONFIGURÁCIÓRA, D) A FOLYOSÓS ELTOLT TEREMKONFIGURÁCIÓRA ÉS E) A SZOKNYA TEREMKONFIGURÁCIÓRA MUTAT EGY-EGY PÉLDÁT.	5
2. ÁBRA: AZ ELSŐ ÁBRÁN LÁTHATÓ TEREMKONFIGURÁCIÓK MÁTRIXOS FELÍRÁSA.....	6
3. ÁBRA: A BEST FIT, WORST FIT, FIRST FIT, NEXT FIT ALGORITMUSOK ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT.....	18
4. ÁBRA: A BEST FIT, WORST FIT, FIRST FIT, NEXT FIT ALGORITMUSOK ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN FOLYOSÓS NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT.	19
5. ÁBRA: A BEST FIT, WORST FIT, FIRST FIT, NEXT FIT ALGORITMUSOK ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN ELTOLT TEREMKONFIGURÁCIÓ MELLETT.....	19
6. ÁBRA: A BEST FIT, WORST FIT, FIRST FIT, NEXT FIT algoritmusok ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN FOLYOSÓS ELTOLT TEREMKONFIGURÁCIÓ MELLETT.....	20
7. ÁBRA: A BEST FIT, WORST FIT, FIRST FIT, NEXT FI, ALGORITMUSOK ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN SZOKNYA TEREMKONFIGURÁCIÓ MELLETT.	21
8. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ ÜLŐHELYEINEK HASZNOSSÁGI ÉRTÉKE RANDOM ALGORITMUSVAL SZÁMOLVA AZ EGYES CSOPORTMÉRETEKRE.....	22
9. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ ÜLŐHELYEINEK HASZNOSSÁGI ÉRTÉKE GENETIKUS ALGORITMUSVAL SZÁMOLVA AZ EGYES CSOPORTMÉRETEKRE.....	23
10. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ ÜLŐHELYEINEK HASZNOSSÁGI ÉRTÉKE BEST FIT ALGORITMUSVAL SZÁMOLVA AZ EGYES CSOPORTMÉRETEKRE.....	24
11. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ ÜLŐHELYEINEK HASZNOSSÁGI ÉRTÉKE FIT MÓDSZERREL SZÁMOLVA AZ EGYES CSOPORTMÉRETEKRE.....	25
12. ÁBRA: NAGY TEREM ELTOLT, FOLYOSÓS NEM ELTOLT, FOLYOSÓS ELTOLT ÉS SZOKNYA TEREMKONFIGURÁCIÓ ÜLŐHELYEINEK HASZNOSSÁGI ÉRTÉKE FIT MÓDSZERREL SZÁMOLVA 4-ES CSOPORTMÉRETEKRE.....	26
13. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT, KORREKCIÓ NÉLKÜL VÉGZETT TESZTEK EREDMÉNYE BOXPLOT DIAGRAMON.....	27
14. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT, HORIZONTÁLIS KORREKCIÓVAL VÉGZETT TESZTEK EREDMÉNYE BOXPLOT DIAGRAMON.	28
15. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT, VERTIKÁLIS KORREKCIÓVAL VÉGZETT TESZTEK EREDMÉNYE BOXPLOT DIAGRAMON.....	29
16. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT, VERTIKÁLIS & HORIZONTÁLIS KORREKCIÓ VÉGZETT TESZTEK EREDMÉNYE BOXPLOT DIAGRAMON.....	30
17. ÁBRA: KÖZEPES TEREM NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT, CENTRÁLIS KORREKCIÓ VÉGZETT TESZTEK EREDMÉNYE BOXPLOT DIAGRAMON.....	31
18. ÁBRA: A BEST FIT, MI HEURISZTIKA NÉLKÜL ÉS AZ MI HEURISZTIKÁVAL ALGORITMUSOK ÁLTAL SIKERESEN LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT	33
19. ÁBRA: A BEST FIT, MI HEURISZTIKA NÉLKÜL ÉS AZ MI HEURISZTIKÁVAL ÁLTAL SIKERESEN LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN FOLYOSÓS NEM ELTOLT TEREMKONFIGURÁCIÓ MELLETT.	34
20. ÁBRA: A BEST FIT, MI HEURISZTIKA NÉLKÜL ÉS AZ MI HEURISZTIKÁVAL ÁLTAL SIKERESEN LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN ELTOLT TEREMKONFIGURÁCIÓ MELLETT.....	35
21. ÁBRA: A BEST FIT, MI HEURISZTIKA NÉLKÜL ÉS AZ MI HEURISZTIKÁVAL ÁLTAL SIKERESEN LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN FOLYOSÓS ELTOLT TEREMKONFIGURÁCIÓ MELLETT.	36
22. ÁBRA: A BEST FIT, MI HEURISZTIKA NÉLKÜL ÉS AZ MI HEURISZTIKÁVAL ÁLTAL SIKERESEN LEÜLTETETT SZEMÉLYEK SZÁMA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN SZOKNYA TEREMKONFIGURÁCIÓ MELLETT. .	37

TÁBLÁZATJEGYZÉK

1. TÁBLÁZAT: A BEST FIT, WORST FIT, FIRST FIT, NEXT FIT, ALGORITMUSOK ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMÁNAK ÁTLAGA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN NEM ELTOL, ELTOL, FOLYOSÓS ELTOLT, FOLYOSÓS NEM ELTOLT ÉS SZOKNYA TEREMKONFIGURÁCIÓ MELLETT. 17
2. TÁBLÁZAT: A BEST FIT, WORST FIT, FIRST FIT, NEXT FIT ALGORITMUSOK, ILLETVE A KÉT MESTERSÉGES INTELLIGENCIA ÁLTAL LEÜLTETETT SZEMÉLYEK SZÁMÁNAK ÁTLAGA KIS, KÖZEPES ÉS NAGY TEREM ESETÉN NEM ELTOL, ELTOL, FOLYOSÓS ELTOLT, FOLYOSÓS NEM ELTOLT ÉS SZOKNYA TEREMKONFIG.... 32

1. Bevezetés

Az elmúlt években a COVID-19 járvány miatt számos korlátozás került bevezetésre, és ennek folyamányaként az életünk is nagymértékben megváltozott. Bezártak a színházak, mozik, szórakoztató központok, sok munkahelyen a jelenléti munkát felváltotta az online munkavégzés. Ezek a korlátozások mostanra enyhültek, közülük néhány teljesen meg is szűnt. A korlátozásoknak voltak pozitív és negatív hatásai is. Rájöttünk, hogy nagyon sok munkát, amit eddig a munkahelyünkről végeztünk, akár otthonunk kényelméből is elvégezhetünk oly módon, hogy az nem megy a minőség rovására, s esetenként még a produktivitás is javítható. Ugyanakkor a kötelező maszkhasználat, a személyes távolságtartás, a betegségtől való félelem az emberek kismértékű elhidegülését hozta magával, s kevesebb személyes kapcsolat alakult ki.

A megváltozott életünk során számos olyan feladattal és problémával szembesültünk, melyek megoldása újszerű eszközöket, megoldásokat kíván. A személyes távolságtartás biztosítása a járvány elleni védekezés egyik alappillére. A színházi és mozi előadások, valamint a zárt térben rendezett sportesemények esetén kiemelt figyelmet kell fordítani a látogatók nézőtérén való elhelyezésére. Bár az ültetésre vonatkozó egységes szabályozás nem létezik, általánosan igaz, hogy a rendezvényszervezők az együtt érkező személyek csoportját nem bontják meg, viszont a különböző csoportokat igyekeznek egymástól kis mértékben szeparálni azért, hogy biztosítsák a külön érkező csoportok egymástól való biztonságos távolságban történő elhelyezését.

Az eseményen résztvevők leültetése különösen érdekes kihívást jelent abban az esetben, ha a leültetendő csoportok száma és mérete előre nem ismert, és a foglalási igények folyamatosan érkeznek. A korlátozásokat figyelembe vevő **online ülőhely kiosztás-optimalizálási probléma** érdekes kihívás elé állítja a teremkiosztást végző személyeket, melyet véleményem szerint érdemes mesterséges intelligencia fejlesztésével is támogatni.

Dolgozatomban egy új, feladat-specifikus, passzív megerősítéses tanuláson alapuló online optimalizálási módszert fejlesztettem a probléma megoldásának támogatására. Az új mesterséges intelligencia módszer hatékonyságát klasszikus allokációs módszerek (First Fit, Next Fit, Best Fit, Worst Fit) hatékonyságával hasonlítottam össze, illetve javaslatot tettem ezen módszerek ötvöztetésére is. A módszereket eltérő teremméretek és teremkonfigurációk mellett értékeltem ki és hasonlítottam össze. Az eredmények azt mutatják, hogy a korlátozásokat figyelembe vevő online ülőhelykiosztás-optimalizációs probléma hatékonyan támogatható a kidolgozott mesterséges intelligencia alkalmazásával, s a javasolt módszer

hatékonyan alkalmazható a folyamatosan érkező foglalási igényekhez kapcsolódó helykiosztásra.

2. Elméleti háttér

A korlátozásokat figyelembe vevő online ülőhely-foglalási feladat megoldását támogató mesterséges intelligencia tervezésének fontos lépései a feladat pontos megfogalmazása, a kapcsolódó szakirodalom áttekintése, a megoldáshoz használható eszköztárak és működési módok áttekintése, a megoldási javaslat kidolgozása, majd az implementációs eszközök kiválasztása. Jelen fejezetben munkám ezen lépéseit mutatom be részletesen.

2.1. A probléma bemutatása

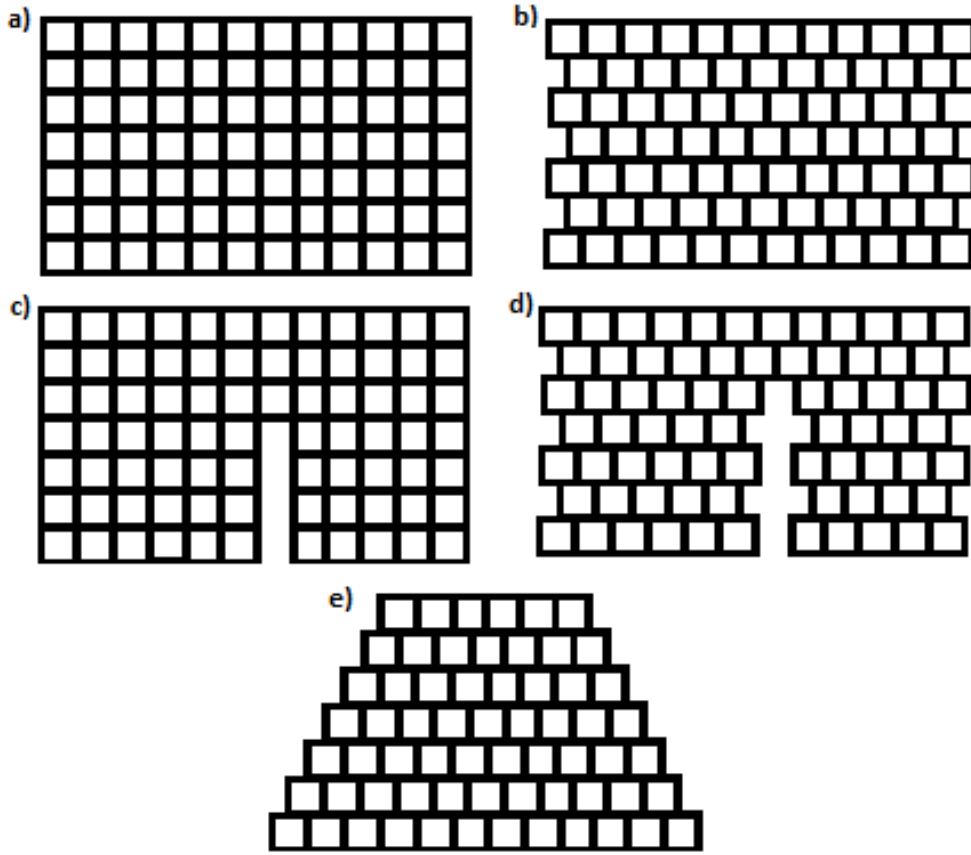
Mint a bevezetőben is említettem, munkám célja egy olyan mesterséges intelligencia kidolgozása, amely egy tetszőleges, ülőhelyekkel rendelkező nézőtér (pl. mozi, színház, sportcsarnok) ülőhelyeinek kiosztását támogatja abban az esetben, ha a rendezvényre érkező személyek száma és azok csoportjai előre nem ismertek. A feladat során adott tehát egy nézőtér, amiről feltételezzük, hogy fix elrendezésű, vagyis az ülőhelyek kiosztása során a teremben elhelyezett székek száma és helyzete nem változtatható. A rendezvényre folyamatosan érkeznek a helyfoglalások, azonban a rendezvény látogatása iránti igény előre nem ismert. Minden foglalás egy, vagy annál több személy együttes helyfoglalási igényét jelenti. Munkám során azzal a feltételezéssel éltem, hogy a foglalásnál egyszerre lefoglalható helyek száma felülről korlátos. Az egyes foglalásokhoz kapcsolódó személyek száma tehát maximalizálva van, s feltételezzük, hogy az egy foglalásban megjelenő igény esetében a csoportba tartozó személyek egymás mellé szeretnének ülni. A csoport tehát az ülőhelyek kiosztása során nem bontható, tagjait sorfolytonosan kell leültetni. Feladatunk, hogy ezeket a csoportokat az ültetési szabályokat betartva, a többi csoporttól szeparáltan, biztonságos távolságra helyezzük el. Az ültetési szabályok a székek egymáshoz viszonyított helyzetére vonatkozóan vannak meghatározva, és olyan előírások, amelyek azt hivatottak biztosítani, hogy az eltérő csoportok tagjai “székszomszédságban” ne érintkezzenek egymással. Az ültetési szabályokat a későbbiek során mutatom be részletesen.

A beérkező igényeket azonnal ki kell szolgálnunk, vagyis a csoport tagjainak ki kell osztani 1-1 ülőhelyet minden esetben, ha a csoport az ültetési szabályokat betartva szabályos módon leültethető oly módon, hogy a csoport tagja folytatólagosan egymás mellett foglalnak helyett. A foglalási igény elutasítható, ha az előbb megadott feltételek nem teljesülnek. Egy csoport elutasítása után amennyiben a későbbiekben olyan foglalási igény érkezik, amit ki tudunk szolgálni, akkor az új igényt teljesíteni kell.

A jelenlegi feladat kitűzésekor feltételeztük, hogy a székek azonos értékűek és az optimalizáció célja a minél több személy leültetése. Munkám során nem foglalkoztam azzal az esettel, ha a székenkénti jegyárak eltérőek. A feladat során tehát célunk a minél több személy leültetése oly módon, hogy az ültetési szabályokat betartsuk.

2.1.1. Teremkonfigurációk

Munkám során többféle teremkonfigurációt különböztettem meg. Az első lehetséges teremkonfiguráció szerint a sorok egymáshoz képest fél székkal el vannak tolva. Ezt a teremkonfigurációt a továbbiakban *eltolt teremkonfigurációnak* nevezem. A második lehetséges teremkonfiguráció esetén az eltérő sorokban elhelyezett székek nincsenek eltolva egymáshoz viszonyítva, vagyis közvetlen egymás mögött helyezkednek el. Ezen teremkonfigurációt a továbbiakban *nem eltolt teremkonfigurációnak* nevezem. Ezen két teremkonfigurációnak létezik egy úgynevezett “folyosós” verziója is. Ez azt jelenti, hogy a teremben valahol van egy vagy több folyosó. Ezen teremkonfigurációkat innentől *folyosós eltolt teremkonfigurációnak* és *folyosós nem eltolt teremkonfigurációnak* fogom nevezni. Illetve létezik még egy ötödik teremkonfiguráció is, ami úgy néz ki, hogy a teremben lévő sorok fokozatosan rövidülnek. Ezen teremkonfigurációt a továbbiakban *szoknya teremkonfigurációnak* fogom nevezni. Az eltérő teremkonfigurációkat az 1. ábra szemlélteti.



1. ábra: Az a) a nem eltolt teremkonfigurációra, míg a b) az eltolt teremkonfigurációra, c) a folyosós nem eltolt teremkonfigurációra, d) a folyosós eltolt teremkonfigurációra és e) a szoknya teremkonfigurációra mutat egy-egy példát.

A teremkonfigurációk formálisan a következőképpen írhatók le. Legyen $A_{n \times m}$ mátrix a terem reprezentációja, ahol n a sorok, m pedig az oszlopok számát jelöli ($n, m \in N$). $a_{i,j} \in A$ a terem i -edik sorának j -edik székét jelöli ($1 \leq i \leq n, 1 \leq j \leq m$). $a_{i,j}$ 0, 1 értékeket vehet fel, vagy egyáltalán nem vesz fel értéket ($\emptyset$). Ha $a_{i,j}$ értéke 1 ($a_{i,j} = 1$), akkor az adott hely foglalt, ha $a_{i,j}$ értéke 0 ($a_{i,j} = 0$), akkor szabad. Ha $a_{i,j}$ nem vesz fel értéket ($a_{i,j} = \emptyset$), akkor ott nincs ülőhely (pl. folyosó esetén). Az eltolat teremkonfiguráció reprezentációjához szükséges egy S oszlopvektort bevezetni, amely tartalmazza, hogy mely sorok vannak eltolva a teremben. $s_i \in S$ az S oszlopvektor i -dik eleme ($1 \leq i \leq n$), és 0 vagy 1 értékeket vehet fel. Ha s_i értéke 1 ($s_i = 1$), akkor a terem i -edik sora el van tolva, ha s_i értéke 0 ($s_i = 0$), akkor a terem i -edik sora nincs eltolva. A konzekvens és általános érvényű formalizáció és implementáció végett ezen plusz oszlopvektort a nem eltolat teremkonfiguráció esetén is bevezettem, ahol S minden eleme 0 értéket vesz fel.

Az 1. ábrán látható teremkonfigurációknak megfelelő matematikai reprezentációt a 2. ábra szemlélteti.

a) S A

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

b) S A

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 \end{bmatrix}$$

c) S A

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

d) S A

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & 0 & 0 & 0 & 0 \end{bmatrix}$$

e) S A

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} \emptyset & \emptyset & \emptyset & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset \\ \emptyset & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset \\ \emptyset & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \emptyset \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2. ábra: Az első ábrán látható teremkonfigurációk mátrixos felírása

2.1.2. Ültetési szabályok

A csoportok szabályos leültetésére vonatkozóan a következő ültetési szabályokat fogalmaztam meg. A legfontosabb kritérium, hogy a lefoglalni kívánt helyek még szabadok legyenek, vagyis ne legyenek lefoglalva mások számára. Azért, hogy biztosítsuk a különböző csoportok szeparációját fontos, hogy ezen székek mellett nem foglalhat helyet más csoportban lévő személy. Ezt kétféle módon biztosítható. Egyrészt, amennyiben az aktuálisan lefoglalandó székek mellett létezik szék, akkor azok a helyek nem lehetnek foglaltak, másrészt pedig, ha a lefoglalandó szék/székek szélső székek (akár oldalt, akár első, vagy hátsó sor révén), akkor ezen elvárás eleve biztosított. Fontos, hogy azon helyek sem lehetnek foglaltak, amelyek valamilyen módon fedésben vannak a lefoglalni kívánt székekkel, vagyis nem lehet közvetlenül a lefoglalni kívánt helyek előtt és mögött foglalt hely.

Az ültetési szabályok a következőképpen adhatók meg. Legyen $G = \{g_1, g_2, \dots, g_p\}$ a leültetendő személyek csoportja, és jelölje q a maximális csoportméretet ($q \in N, q = \max\{|g_1|, |g_2|, \dots, |g_p|\}$, ahol $|\cdot|$ a számosságot jelöli). Tekintsünk egy g_r aktuálisan beérkező csoportot, amely k személyt tartalmaz ($|g_r| = k, 1 \leq k \leq q$).

A g_r csoport leültethető, ha

$$\exists a_{i,j}, \dots, a_{i,j+k-1}, \text{ ahol } 1 \leq i \leq n, 1 \leq j \leq m - (k - 1), \quad (1)$$

és

$$a_{i,j} = 0 \wedge \dots \wedge a_{i,j+k-1} = 0, \quad (2)$$

és

$$\text{ha } j \geq 2, \text{ akkor } a_{i,j-1} = 0, \quad (3)$$

és

$$\text{ha } j + k - 1 < m, \text{ akkor } a_{i,j+k} = 0, \quad (4)$$

valamint a teremrendezéstől függően

- *nem eltolt teremkonfiguráció és folyosós nem eltolt teremkonfiguráció esetén teljesülnek továbbá:*

$$\text{ha } i \geq 2, \text{ akkor } a_{i-1,j} = 0 \wedge \dots \wedge a_{i-1,j+k-1} = 0 \quad (5)$$

$$\text{ha } i < n, \text{ akkor } a_{i+1,j} = 0 \wedge \dots \wedge a_{i+1,j+k-1} = 0 \quad (6)$$

- *eltolt, folyosós eltolt, és szoknya teremkonfiguráció esetén teljesülnek továbbá:*

$$\text{ha } i \geq 2 \text{ és } s_i = 1, \text{ akkor } a_{i-1,j} = 0 \wedge \dots \wedge a_{i-1,j+k} = 0 \quad (7)$$

$$\text{ha } i < n \text{ és } s_i = 1, \text{ akkor } a_{i+1,j} = 0 \wedge \dots \wedge a_{i+1,j+k} = 0 \quad (8)$$

$$\text{ha } i \geq 2 \text{ és } j \geq 2 \text{ és } s_i = 0, \text{ akkor } a_{i-1,j-1} = 0 \wedge \dots \wedge a_{i-1,j+k-1} = 0 \quad (9)$$

$$\text{ha } i < n \text{ és } j \geq 2 \text{ és } s_i = 0, \text{ akkor } a_{i+1,j-1} = 0 \wedge \dots \wedge a_{i+1,j+k-1} = 0 \quad (10)$$

Az (1) és (2) egyenletben az ültetési szabályok definiálásakor azt határoztuk meg, hogy a leültetendő csoport tagjai egymás melletti szabad ülőhelyeken foglaljanak helyet. A (3)-(4) szabályok azt határozzák meg, hogy az aktuálisan leültetendő csoport közvetlen jobb és bal oldalán nem ülhet más csoportbeli személy. Az (5)-(6) és (7)-(10) egyenletek pedig azt biztosítják az eltérő teremkonfigurációk esetén, hogy a leültetendő csoport előtt és mögött közvetlen módon ne legyen leültetve más csoportba tartozó személy.

2.2. Irodalmi áttekintés

A szakirodalmat tanulmányozva azt találtam, hogy a dolgozatomban felvázolt problémához hasonló probléma elsősorban a repülőjáratok ülőhelykiosztása és a vonatok

ülőhelyfoglalása során merült fel a tudományos közösség munkáiban. Az általam felvázolt problémával azonos, vagy nagy rokonságot mutató tudományos szakirodalmat sajnos nem sikerült fellelnem, ezért a rokon problémákat és az azokhoz kapcsolódó megoldásokat tanulmányoztam át.

A repülőjáratok ülőhelykiosztása kapcsán a kutatók azzal a kérdéssel foglalkoznak, hogy egy repülőgépen hány helyet tartsanak fenn az egyes utazási osztályoknak abban az esetben ha nincs átszállás, vagyis a repülő nem áll meg, illetve ezen probléma komplexebb változatában azt vizsgálják, hogy hány helyet érdemes fenntartani az egyes utazási osztályoknak ha a járat több helyen is megáll. Ugyanezen probléma felmerült vonatok esetében is, amely kérdéskörrel Xinchang Wang és munkatársai foglalkoztak elsősorban [1].

A repülőgépen történő helyfoglalás esetén Moshe Dror és munkatársai gráf algoritmusokkal igyekeztek megoldani a problémát oly módon, hogy a problémát maximális folyam problémára vetítették le [2]. Wen Zhao és Yu-Sheng Zheng tanulmányukban egy kétosztályos dinamikus ülés kiosztási modellt vizsgáltak [3]. Modelljük megmagyarázza, hogy a légitársaságok miért zárják le az akciós viteldíjakat az indulási idő közeledtével. Szoros kapcsolatot mutattak ki az ehhez tartozó optimális politika és a meglévő statikus modellek által javasolt politikák között is. Zhenying Yan és munkatársai egy valószínűségi nemlineáris programozási modellt javasolnak több személyvagonból álló nagysebességű vasúti személyszállítási hálózatra [4]. Kutatásaik során azt találták, hogy az általuk javasolt modell jobb bevételeket eredményez, mintha egy utazási osztályból fix számú hely lenne fenntartva. Xinlei Hu és társai egy nemlineáris programozási modellt javasoltak, illetve a Davidon–Fletcher–Powell módszert követve hatékony megoldási algoritmust terveztek nagysebességű vasúti hálózatok árképzési és ülőhelykiosztási problémáira [6]. Kutatásuk során azt találták, hogy a javasolt modell és megoldási algoritmus döntési alapot adhat a kedvezményes jegyértékesítéshez. Claire Cizaire és Peter Belobaba egy az előző két problémához hasonló oldottak meg repülőgépjáratok esetén [5]. Kutatásuk során hasonló eredményeket értek el.

2.3. A probléma lehetséges megoldása klasszikus allokációs módszerekkel

Dolgozatom ezen alfejezetében néhány olyan fontos megoldási módszert mutatok be, amelyek véleményem szerint figyelemre méltó megoldást adhatnak a problémára. Ezt követően a 2.4. fejezetben az általam fejlesztett mesterséges intelligencia algoritmus részletes elméleti háttérét mutatom majd be.

A ládapakolási feladatok megoldása során [7,8] gyakran alkalmazzák az operációs rendszerek témaköréből ismert *Best Fit*, *Worst Fit*, *First Fit* és *Next Fit* klasszikus allokációs

módszereket [9]. E négy algoritmust eredetileg memória allokációs feladatok kiszolgálására alkalmazták, vagyis ezen algoritmusok valamelyikével foglalta le az operációs rendszer az egyes programoknak a megfelelő mennyiségű memóriát. Ezen algoritmusok ugyan várhatóan nem képesek optimális megoldást adni az általam felvetett problémára, viszont mint a későbbiekben látjuk, majd van köztük olyan, amely egyes esetekben meglehetősen jól közelíti a probléma optimális megoldását.

A *First Fit*, *Next Fit*, *Best Fit* és *Worst Fit* algoritmusok az általam megfogalmazott problémára a következőképpen adaptálhatók:

- **First Fit:** A *First Fit* algoritmus az első olyan helyre ülteti le a csoportot, ahova a megadott ültetési szabályokat betartva el lehet helyezni. Az algoritmus a keresést minden esetben a terem $a_{1,1}$ ülőhelyétől (bal felső sarok) indítja.
- **Next Fit:** A *Next Fit* algoritmus hasonlóan működik, mint a *First Fit* algoritmus, annyi különbséggel, hogy ha új csoport számára kell helyet foglalni, akkor az algoritmus onnan folytatja a megfelelő hely/helyek keresését, ahol az előző csoport helyfoglalása során a keresést abbahagyta.
- **Best Fit:** A *Best Fit* algoritmus működésének célja, hogy olyan helyre ültesse le a csoportot, ahova “leginkább beillik”. Az algoritmus működése során minden esetben az $a_{1,1}$ ülőhelytől indulva megkeresi azt a legrövidebb szabad széksorozatot, ahova az aktuálisan beérkező csoport még befér, és a leültetés nem sérti meg az ültetési szabályokat.
- **Worst Fit:** A *Worst Fit* algoritmus az $a_{1,1}$ ülőhelytől indulva megkeresi azt az első egymás mellett előforduló leghosszabb szabad széksorozatot, ahova az aktuálisan beérkező csoport befér, s a csoport ültetése nem sérti meg a leültetési szabályokat.

Mint láthatjuk, a klasszikus memóriaallokációs algoritmusok könnyen értelmezhetők az online ülőhelyfoglalási probléma esetére is, azonban érdemes megvizsgálni, hogy a mesterséges intelligencia eszköztára miként járulhat hozzá egy új módszer kidolgozásához, illetve ezen módszerek továbbfejlesztéséhez.

2.4. A probléma mesterséges intelligencia alapú megoldása

2.4.1. Gépi tanulás

Mesterséges intelligenciát (Artificial Intelligence) napjainkban a mindennapi élet számos területén használunk. Megjelenik az önvezető autókban [10], az alakfelismerésben [11], de felhasználják különböző cégek az ügyfélszolgálat támogatásában is [12], hogy

tehermentesítsék az alkalmazottakat, vagy gondoljunk a Google Assistant-re, és még sok egyéb más mindennapi dologra. De mi is az a mesterséges intelligencia? A mesterséges intelligencia egy olyan szoftver, amely képes tapasztalni és/vagy tanulni, a tapasztalatok/tanultak alapján a lehető legjobb döntést hozni, kommunikálni a külvilággal, vagyis képes az információcserére és az információ feldolgozására.

A mesterséges intelligenciák egy területe a gépi tanulás (Machine Learning), melybe olyan mesterséges intelligencia algoritmusok tartoznak, amelyek képesek tapasztalni, és a tapasztalataikból tudást generálni, majd ezen tudás felhasználásával a lehető legjobb döntéseket hozni a probléma megoldása során. Ezen gépi tanuló algoritmusoknak három fajtája van: a *megerősítéses tanulás* (Reinforcement Learning), *felügyelt tanulás* (Supervised Learning), és *nem felügyelt tanulás* (Unsupervised Learning). Ezen három tanulási módszert két csoportba lehet sorolni: az egyik a tapasztalatokból való tanulás csoportja, amibe a megerősítéses tanulás tartozik, a másik az adatokból való tanulás csoportja, melyben a felügyelt és nem felügyelt tanulás helyezkedik el.

A *felügyelt tanulás* két fő feladatköre az *osztályozás* és az *előrejelzés* (regresszió). Osztályozási feladat megoldása során a mesterséges intelligencia algoritmus olyan osztályozó modellt tanul meg, amely az objektumokat diszkrét csoportokba sorolja. Előrejelzés során nem kategorikus osztálycímke megbecslése a cél, hanem folytonos értékű output változó értékét szeretnénk minél pontosabban megbecsülni. A felügyelt tanulás során a modell kialakítása egy tréning halmaz adatai alapján történik, amely adathalmaz tartalmazza a probléma bemeneti adatait és egyben az elvárt eredményt, az osztálycímket is. Az input adatokat objektumonként egy-egy tulajdonságvektorral adjuk meg. A tréning halmaz által tartalmazott input és elvárt output adatok alapján a mesterséges intelligencia algoritmus felépít egy prediktív modellt, amely alapján képes megoldani a jövőben előforduló hasonló feladatokat. A felügyelt tanulás tehát két lépésre bontható: (1) az első lépésben a mesterséges intelligencia megtanulja a megfelelő modellt, (2) míg a második lépésben a modell felhasználására kerül sor a későbbi problémák megoldása során. Felügyelt tanulás során két súlyos probléma merülhet fel, melyek az *alultanulás*, és a *túltanulás*. Az *alultanulás* esetében a modell túl "laza" lesz, és nem tartalmazza megfelelő mértékben az input adatokban rejlő releváns összefüggéseket. *Túltanulás* esetén a modell túlságosan "szigorú" lesz, a mesterséges intelligencia túlzó mértékben megtanulja a tréning halmaz specialitásait, hibáit, a benne rejlő zajokat, s nem rendelkezik megfelelő mértékű általánosítási képességgel. Mindkét esetben a végeredmény az, hogy az algoritmus nem működik megfelelően.

A *nem felügyelt tanulás* főként a csoportosítás, dimenziócsökkentés, és az asszociációs szabályok keresése problémakörében használatos. A felügyelt tanulással ellentétben itt nincs tréning halmaz, a mintázatok feltárását az algoritmusoknak az elvárt eredmény ismerete nélkül kell megtenniük.

A *megerősítéses tanulás* során a mesterséges intelligencia algoritmust elhelyezzük egy környezetben, amely környezetről az ágens érzékelőin keresztül információt szerez, s beavatkozási révén képes hatni rá. A mesterséges intelligencia a működése során döntéseket hoz, a döntéseiért jutalmat kap. A jutalom egy számérték, melynek nagysága függ a döntés "jóságától". Értelemszerűen minél jobb döntést hozott az ágens, annál nagyobb jutalmat kap, s minél rosszabb a döntés, annál kisebb a jutalom (amely akár negatív is lehet). A megerősítéses tanulásnak két fajtája van, a *passzív megerősítéses tanulás*, és az *aktív megerősítéses tanulás*. A passzív megerősítéses tanulás során a döntési stratégia fix, vagyis ugyanarra az állapotra minden esetben ugyanazt a döntést hozza az ágens. A passzív megerősítéses tanulás célja az egyes állapotok hasznosságának megtanulása. Aktív megerősítéses tanulás során nincs fix döntési stratégia, az algoritmusnak meg kell tanulnia, hogy melyik állapotban mi a helyes döntési stratégia. Az aktív megerősítéses tanulás célja egy optimális vagy majdnem optimális döntési stratégia felépítése az adott környezetben. Az optimális döntési stratégia az a stratégia, amely maximalizálja a várható jutalmat.

2.4.2. Passzív megerősítéses tanulás az online ülőhelykiosztás optimalizálására

Dolgozatomban a 2.1. fejezetben bemutatott korlátozásokat figyelembe vevő online ülőhelykiosztás-optimalizálási probléma megoldására passzív megerősítéses tanulást alkalmazok. A feladatban a környezet nem más, mint a terem és a beérkező csoportok, az algoritmusnak pedig döntést kell hoznia, hogy az egymás után beérkező csoportokat hova ültesse le.

2.4.2.1. Az ülőhelyek minősítése

A tanulás során Monte Carlo szimulációkat alkalmaztam, melynek lényege, hogy a probléma megoldására sok lehetséges esetet modellez és értékel ki. A lehetséges teremkiosztások modellezésére többféle algoritmust használtam, melyek a korábban tárgyalt *Best Fit*, *Worst Fit*, *First Fit* és *Next Fit* algoritmusok, továbbá a *genetikus algoritmus* és egy úgynevezett *random algoritmus*, minek lényege hogy teljesen véletlenszerű helyekre ülteti le a beérkező csoportokat. A Monte Carlo szimulációk célja az volt, hogy a teremkiosztás eredménye alapján a teremben elhelyezett ülőhelyeket kvantitatív módon minősíteni tudjam.

Az ülőhelyeket nem általános módon szerettem volna jellemezni, hanem annak függvényében, hogy milyen hasznos abban az esetben, ha $1, \dots, q$ -s fős csoport egyik tagja foglal helyet rajta. A szimulációk kiértékelése révén tehát minden $a_{i,j} \in \mathbf{A}$ ($1 \leq i \leq n, 1 \leq j \leq m$) ülőhelyre vonatkozóan meg szeretnénk tanulni a $p_{i,j}^{(1)}, \dots, p_{i,j}^{(q)}$, **hasznossági értékeket**, ahol $p_{i,j}^{(k)}$ az $a_{i,j} \in \mathbf{A}$ ülőhely hasznosságát jellemzi abban az esetben, ha egy k számosságú csoport tagja ül rajta.

Az ülőhelyek minősítése oly módon határozható meg, hogy minden egyes szimulációhoz hozzárendeljük a sikeresen leültetett személyek számát, illetve minden szimulációban minden székhez azt a k értéket, amekkora csoport tagja ült azon a helyen. A szimulációk alapján minősíthetők az egyes székek oly módon, hogy kiszámítjuk, hogy átlagosan hány embert sikerült leültetni a teremben azokban az esetekben, ha $a_{i,j}$ széken k létszámú csoport tagja ült. Ezen értékeket k csoportméretre vonatkozóan a $\mathbf{P}_{n \times m}^{(k)}$ mátrix tartalmazza ($p_{i,j}^{(k)} \in \mathbf{P}_{n \times m}^{(k)}$), és $\mathbf{M}_{i,j}^{(k)}$ azon megoldások vektora, ahol az i, j helyen k méretű csoport ült. Továbbá, $seated(u)$ az u megoldásban ($u \in \mathbf{M}_{i,j}^{(k)}$) leültetett személyek száma. Ekkor

$$p_{i,j}^{(k)} = \frac{\sum_{u \in \mathbf{M}_{i,j}^{(k)}} seated(u)}{|\mathbf{M}_{i,j}^{(k)}|} \quad (11)$$

Eredményképpen az egyes székek minősítése q darab mátrixal ($\mathbf{P}_{n \times m}^{(1)}, \dots, \mathbf{P}_{n \times m}^{(q)}$) határozható meg, amelyek tartalmazzák az egyes csoportméretekre nézve a teremben lévő helyek hasznosságát.

Mint említettem, az egyes szimulációk során *First Fit*, *Next Fit*, *Best Fit*, *Worst Fit*, *genetikus algoritmus* és *random* ültetési stratégia alkalmazásával ültettem le a csoportokat a beérkező igények alapján. Mivel a *Best Fit*, *Worst Fit*, *First Fit* algoritmusok a megfelelő hely keresését a terem bal felső sarkából kezdik, ezért az ülőhelyek minősítése torzulhat, vagyis szimmetrikus termék esetén sem lesz szimmetrikus az ülőhelyek minősítése. Analóg igaz ez a *Next Fit* algoritmusra is, habár ezen esetben a torzulás mértéke várhatóan kisebb. Ezen okból kifolyólag szükségessé válhat a hasznossági értékek korrigálása. Az ülőhelyek minősítésének korrigálásához négy különféle módszer határoztam meg. Az első módszer során a q darab mátrixot úgy módosítjuk, hogy minden egyes mátrix esetében vertikális tükrözést alkalmazunk a terem vertikális középtengelyére vonatkozóan, és a középvonaltól jobbra elhelyezkedő székek esetében székenként átlagoljuk az adott székre kiszámolt $p_{i,j}$ értékeket a rá tükrözött értékekkel. Ez a módszert *vertikális korrekciónak* nevezem. A második módszer ugyanaz, mint az első csak 90 fokkal elforgatva, vagyis a q darab mátrixot úgy módosítjuk, hogy minden

egyes mátrix esetében a horizontális középvonalra tükrözzük a mátrixot és a terem alsó felében elhelyezkedő székek esetében átlagoljuk az eredeti értékeket a tükrözött értékekkel. Ezt a módszert *horizontális korrekciónak* nevezem. A harmadik módszer az első kettő ötvözete, vagyis először végrehajtjuk a *vertikális korrekciót* majd az így előálló mátrixokon végrehajtjuk a *horizontális korrekciót*. Ezt a módszert innentől *vertikális & horizontális korrekciónak* nevezem. A negyedik módszer során a q darab mátrixot úgy módosítjuk, hogy minden egyes mátrix esetében középpontosan tükrözzük a mátrix felső felét páratlan számú sorok esetén a középső sor első felét is beleértve a mátrix alsó felére. Ezt a módszert a továbbiakban *centrális korrekciónak* nevezem.

Jelölje $\mathbf{C}_{n \times m}^{(k)}$ mátrix a k csoportméretre vonatkozó korrigált hasznossági értékeket. Ekkor a négy korrekciós módszer a következőképpen írható fel formálisan:

- *vertikális korrekció:*

$$\text{ha } j > \lfloor m/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = \frac{p_{i,j}^{(k)} + p_{i,m-j+1}^{(k)}}{2} \quad (12)$$

$$\text{egyébként } c_{i,j}^{(k)} = p_{i,j}^{(k)} \quad (13)$$

- *horizontális korrekció:*

$$\text{ha } i > \lfloor n/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = \frac{p_{i,j}^{(k)} + p_{n-i+1,j}^{(k)}}{2} \quad (14)$$

$$\text{egyébként } c_{i,j}^{(k)} = p_{i,j}^{(k)} \quad (15)$$

- *vertikális & horizontális korrekció:*

$$\text{ha } i > \lfloor n/2 \rfloor \text{ és } j > \lfloor m/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = \frac{\frac{p_{i,j}^{(k)} + p_{i,m-j+1}^{(k)}}{2} + \frac{p_{n-i+1,m-j+1}^{(k)} + p_{n-i+1,j}^{(k)}}{2}}{2} \quad (16)$$

$$\text{ha } i > \lfloor n/2 \rfloor \text{ és } j \leq \lfloor m/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = \frac{p_{i,j}^{(k)} + p_{n-i+1,j}^{(k)}}{2} \quad (17)$$

$$\text{ha } i \leq \lfloor n/2 \rfloor \text{ és } j > \lfloor m/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = \frac{p_{i,j}^{(k)} + p_{i,m-j+1}^{(k)}}{2} \quad (18)$$

$$\text{egyébként } c_{i,j}^{(k)} = p_{i,j}^{(k)} \quad (19)$$

- *centrális korrekció:*

$$\text{ha } i > \lfloor n/2 \rfloor \text{ és } j \leq \lfloor m/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = p_{n-i+1,m-j+1}^{(k)} \quad (20)$$

$$\text{ha } i > \lfloor n/2 \rfloor \text{ és } j > \lfloor m/2 \rfloor, \text{ akkor } c_{i,j}^{(k)} = p_{n-i+1,m-j+1}^{(k)} \quad (21)$$

$$\text{egyébként } c_{i,j}^{(k)} = p_{i,j}^{(k)} \quad (22)$$

ahol $(c_{i,j}^{(k)} \in \mathbf{C}_{n \times m}^{(k)})$ és $\lceil \cdot \rceil$ a felső egészt, míg $\lfloor \cdot \rfloor$ az alsó egészet jelöli.

2.4.2.2. Ültetési stratégia

A mesterséges intelligencia megalkotásához az üléshelyek hasznossági értékének megtanulása mellett fontos feladat a döntési stratégia kialakítása. Egy adott állapothoz tartozó döntés az üléshelyek minősítése és a beérkező foglalási igény alapján kerül meghatározásra.

Online üléshelyfoglalás során folyamatosan érkeznek a foglalási igények. Ha érkezik egy k méretű csoportra vonatkozó foglalási igény, akkor mindenekelőtt megvizsgálásra kerül, hogy van-e olyan szabad széksorozat a teremben ahova az adott csoport éppen befér. Ha van ilyen, akkor oda beültetésre kerül az adott csoport. Ha több ilyen szabad széksorozat is van, akkor azon szabad széksorozat kerül lefoglalásra a csoport számára, amelyiknél a $C_{n \times m}^{(k)}$ mátrix alapján az üléshelyek hasznossági értékének összege a legnagyobb. Amennyiben nincs a csoport méretének pontosan megfelelő (k méretű) hely, akkor az algoritmus megnézi, hogy a $C_{n \times m}^{(k)}$ mátrixban a még le nem foglalt helyek közül mely széksorozatok kapták a legjobb minősítést (a székek minősített értékének összege alapján), majd a beérkező csoportot leülteti azon folytonos szabad széksorozatra, ahol hasznossági értékek összege a legnagyobb. Ha több egyforma minősítésű hely is van a csoport számára, akkor ezek közül a legelső hely kerül lefoglalásra (bal felső sarokból indulva).

Mint láthatjuk, az alkalmazott stratégia nem feltétlen azon szabad széksorozatra ülteti le a csoportot, ahol a székek hasznossági értékének összege a legnagyobb. Az algoritmus a *Best Fit* algoritmushoz hasonlóan megkeresi, hogy létezik-e olyan szabad széksorozat, ahova a csoport pontosan beültethető, s ezek közül választja ki a leginkább megfelelőt. Így előfordulhat, hogy létezett volna egy jobb minőségi értékkel rendelkező széksorozat is, azonban ott nem lett volna ideális a térkitöltés. Az algoritmus tehát elsősorban azt preferálja, hogy ne pazaroljunk el feleslegesen helyeket.

2.5. Implementáció

Munkám során a fejlesztést Python-ban végeztem objektumorientált módon. A Python egy nagyon magasszintű programozási nyelv, ezért jóval gyorsabb munkát tesz lehetővé, mint egy alacsonyabb szintű programozási nyelv (pl.: C, C++). A Python többek közt támogatja a funkcionális, az objektumorientált, az imperatív és a procedurális programozási paradigmákat. A Python egy interpreteres nyelv, ami azt jelenti, hogy a kód fordítása gépi nyelvre futási időben történik. Munkámat a következő Python könyvtárak segítették: *numpy*, *pandas*, *pyodbc*, *multiprocessing*, *matplotlib*, *seaborn*, *geneticalgorithm*. A fejlesztéshez a PyCharm környezetet használtam, mivel korábbi munkám során használtam már egy másik a JetBrains

által fejlesztett PhpStorm nevű környezetet is, amellyel jó tapasztalataim voltak. Az adatok tárolásáért egy erre a célra létrehozott MSSQL adatbázis felelt.

3. Eredmények

Dolgozatom ezen fejezetében azt mutatom be, hogy milyen tesztek véghezvitelének érdekében, hogy a lehető legjobb mesterséges intelligencia algoritmust tudjam létrehozni az általam támogatott probléma megoldására. Illetve, hogy ezen tesztek milyen eredményeket hoztak és ezek alapján milyen továbbfejlesztési lépéseket valósítottam meg az algoritmus fejlesztésének érdekében.

3.1. Tesztesetek

A tesztelés során 3 alap teremméretet használtam. Ezen teremméretek a következők:

- **Kis terem:** sorok száma 7, az egy sorban lévő székek száma 12 (7×12).
- **Közepes terem:** sorok száma 16, az egy sorban lévő székek száma 20 (16×20).
- **Nagy terem:** sorok száma 35, az egy sorban lévő székek száma 35 (35×35).

A tesztek során mind a három teremméret esetén az 1. ábrán bemutatott teremkonfigurációkat használtam. A kis teremméret esetén a pontos teremalaprajzok megegyeznek az 1. ábrán szemléltetett termék alaprajzaival. A közepes és nagy termék esetén a különböző teremkonfigurációk hasonlítanak a kisméretű termékekre annyi különbséggel, hogy a *folyosós nem eltolt teremkonfiguráció* és a *folyosós eltolt teremkonfigurációk* esetén egy folyosó helyett kettő van a nagyobb termekben. Közepes terem esetén a két folyosó az 5. sor 7. és 14. oszlopától kezdődően az utolsó sor 7. és 14. oszlopáig tart. Nagy terem esetén a két folyosó a 8. sor 13. és 24. oszlopától kezdődően az utolsó sor 13. és 24. oszlopáig tart. *Szoknya teremkonfiguráció* esetén nagy teremméret mellett az 1-20 sorokban nem csökken a székek száma, hanem váltakozva 15 és 14 ülőhely található.

A tesztek során a maximális csoportméretet 4 főben határoztam meg ($q = 4$). A Monte Carlo szimulációkhoz véletlenszerűen generáltam 100-100 székfoglalási igénysorozatot minden teremmérethez. Ezen véletlenszerű székfoglalási igénysorozatokat rögzítettem, így az azonos méretű, de eltérő teremkonfigurációk esetén az algoritmusokat ugyanazon bemeneti foglalási igények mellett teszteltem és hasonlítottam össze.

3.2. Ülőhelyfoglalás Best Fit, Worst Fit, Next Fit és First Fit algoritmusokkal

Mint ahogy azt már korábban említettem, az általam támogatott problémát a *Best Fit*, *Worst Fit*, *First Fit* és *Next Fit* algoritmusokkal is meg lehet oldani. Az 1. táblázat ezen algoritmusok átlagos teljesítményét mutatja kis, közepes és nagy teremméret mellett *nem eltolt*,

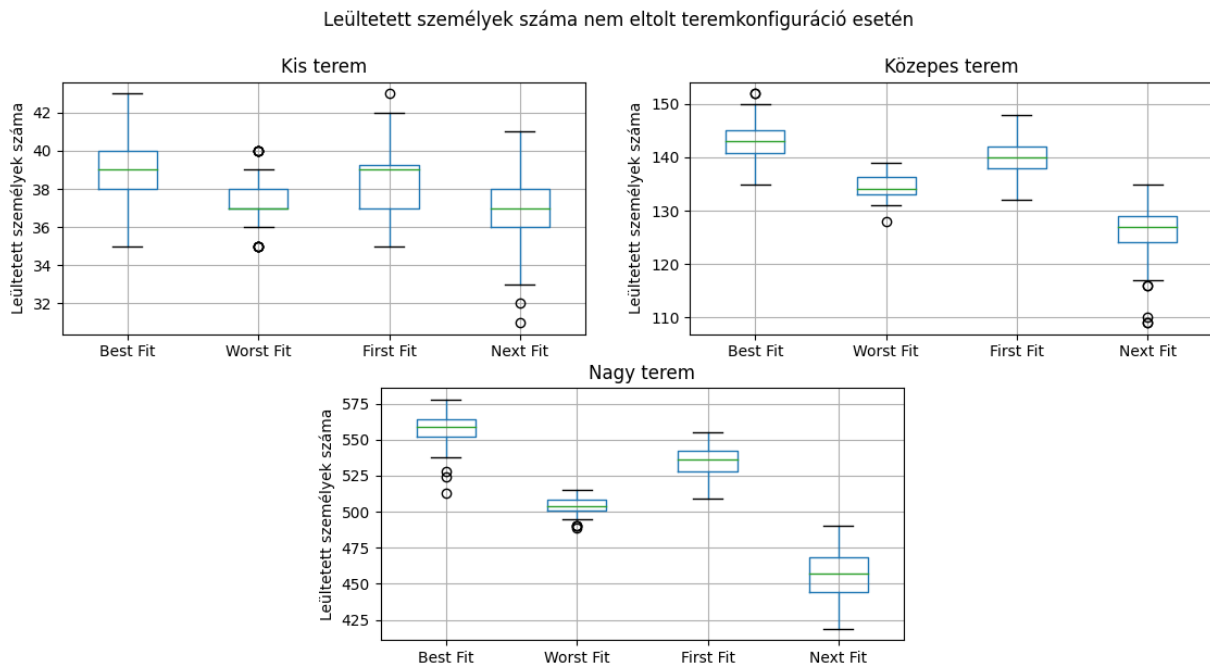
eltolt, folyosós eltolt, folyosós nem eltolt és szoknya teremkonfigurációra. A táblázat az egyes teremkonfigurációkra vonatkoztatva tartalmazza az algoritmusok által a 100 szimuláció során leültetett személyek számának átlagát.

1. táblázat: A *Best Fit*, *Worst Fit*, *First Fit*, *Next Fit*, algoritmusok által leültetett személyek számának átlaga kis, közepes és nagy terem esetén *nem eltolt, eltolt, folyosós eltolt, folyosós nem eltolt és szoknya teremkonfiguráció* mellett.

Nagy terem				
	Best Fit	Worst Fit	First Fit	Next Fit
Nem eltolt	557,99	503,94	535,33	456,37
Folyosós nem eltolt	512,79	487,95	517,33	426,23
Eltolt	433,51	452,73	457,50	439,68
Folyosós eltolt	414,93	438,32	450,09	407,54
Szoknya	272,39	253,51	264,75	43,99
Közepes terem				
	Best Fit	Worst Fit	First Fit	Next Fit
Nem eltolt	142,85	134,64	139,82	126,32
Folyosós nem eltolt	139,60	125,31	133,32	115,42
Eltolt	111,08	115,42	117,44	112,46
Folyosós eltolt	112,77	109,39	114,89	94,99
Szoknya	69,95	68,95	69,96	23,02
Kis terem				
	Best Fit	Worst Fit	First Fit	Next Fit
Nem eltolt	39,11	37,36	38,63	36,85
Folyosós nem eltolt	37,74	34,62	36,78	34,00
Eltolt	27,69	34,95	35,62	34,51
Folyosós eltolt	27,13	33,21	34,41	32,64
Szoknya	25,75	25,62	25,56	19,39

Az 1. táblázat adatai alapján láthatjuk, hogy *nem eltolt* alap *teremkonfigurációk* esetében a legtöbb esetben a *Best Fit* algoritmus tűnik a leghatékonyabbnak, *eltolt teremkonfiguráció* esetében azonban a *First Fit* algoritmus rendre jobb eredményt ér el. Mindemellett feltűnő, hogy szoknya teremkonfigurációnál a *Next Fit* algoritmus sokkal rosszabb eredményt ér el mint a másik három algoritmus.

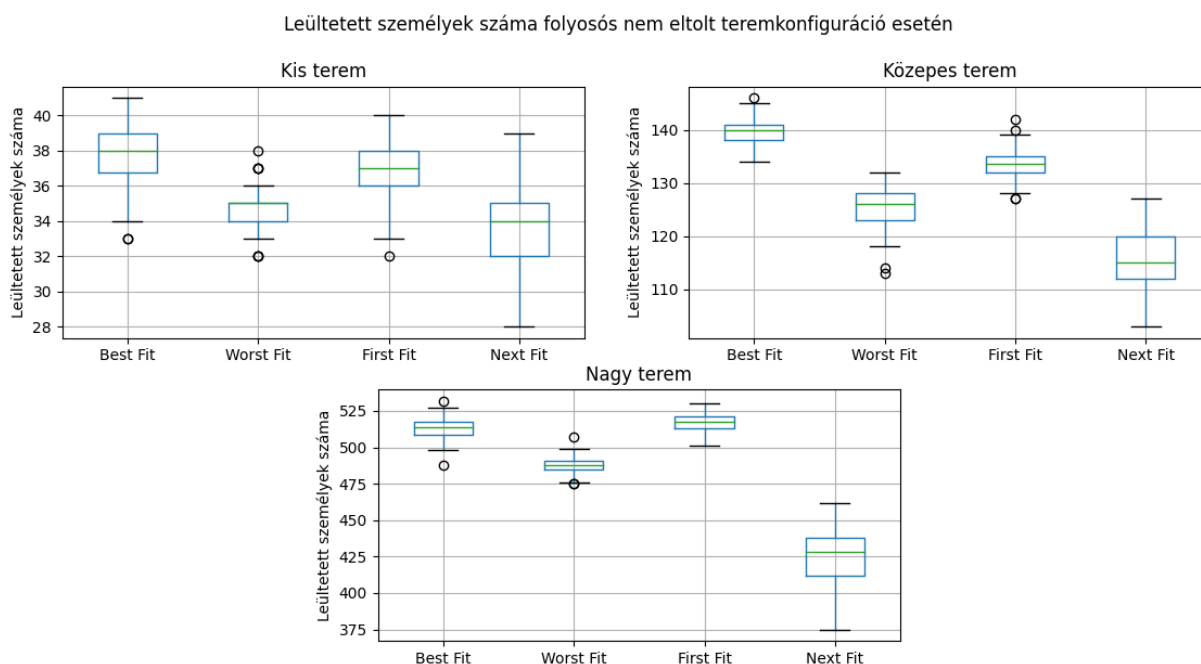
Az átlagosan leültetett nézőszám kiértékelése mellett érdemes megvizsgálni az eredmények szóródását is. A 3. ábra ezen négy algoritmus által sikeresen lefoglalt helyek boxplot diagramját mutatja kis, közepes és nagy teremmellet *nem eltolt teremkonfiguráció* esetén. Az ábrán a 100 Monte Carlo szimuláció eredményének eloszlása tekinthető meg.



3. ábra: A Best Fit, Worst Fit, First Fit, Next Fit algoritmusok által leültetett személyek száma kis, közepes és nagy terem esetén nem eltolt teremkonfiguráció mellett.

Az ábrákon látszik, hogy minél nagyobb teremről van szó, annál inkább kijön az egyes algoritmusok hatékonyságbeli különbsége. Ahogyan az is látszik, hogy a *Next Fit* algoritmus teljesítménye *nem eltolt teremkonfiguráció* esetén minden más algoritmushoz képes nagyban elmarad, és nagy szórással dolgozik. Ez alapján kijelenthetjük, hogy ezen probléma megoldására alkalmatlan. A *Worst Fit* jobb megoldást képes adni a problémára, és mind a négy algoritmus közül ez dolgozik a legkisebb szórással, de még mindig rosszabb, mint a *Best Fit* és *First Fit* algoritmusok. A négy algoritmus közül a *Best Fit* teljesít a legjobban, így kijelenthető, hogy a négy algoritmus közül ez a legalkalmasabb az általam felvetett probléma megoldására *nem eltolt teremkonfiguráció* mellett.

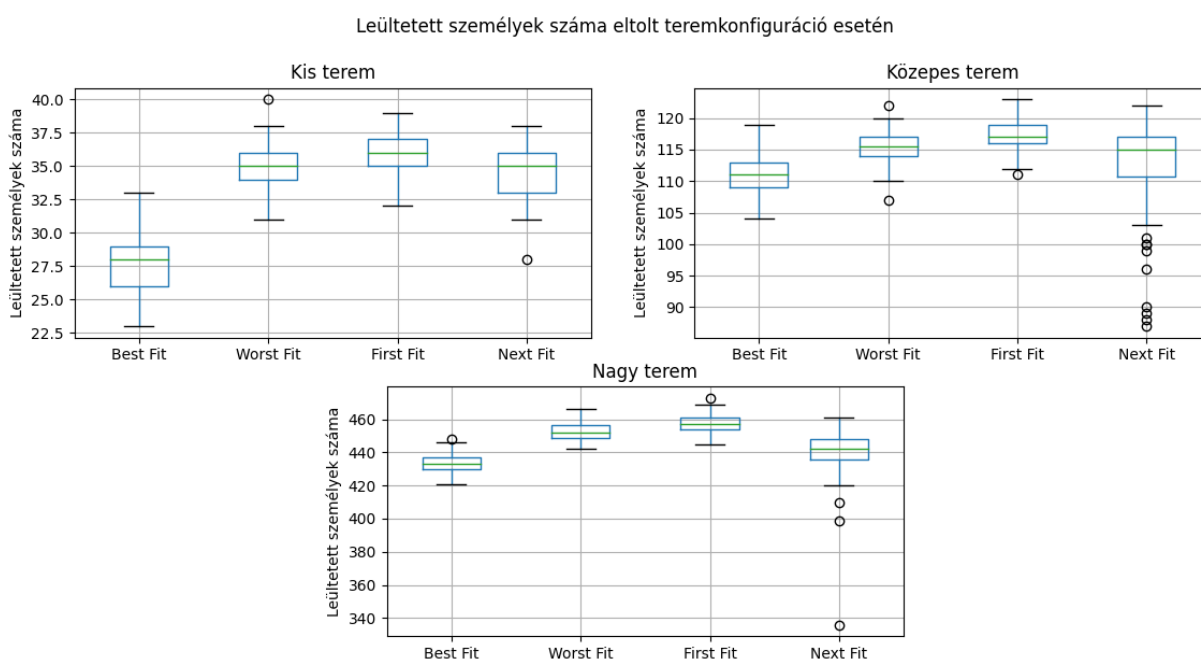
Az algoritmusok teljesítményét a *folyosós nem eltolt teremkonfigurációk* esetén a 4. ábra szemlélteti.



4. ábra: A Best Fit, Worst Fit, First Fit, Next Fit algoritmusok által leültetett személyek száma kis, közepes és nagy terem esetén folyósós nem eltolt teremkonfiguráció mellett.

Az ábrákon ugyanazt figyelhetjük meg, mint a folyósóval nem rendelkező *nem eltolt teremkonfigurációk* esetén. Kijelenthetjük, hogy ebben az esetben is a *Best Fit* algoritmus a legalkalmasabb a probléma megoldására a négy algoritmus közül.

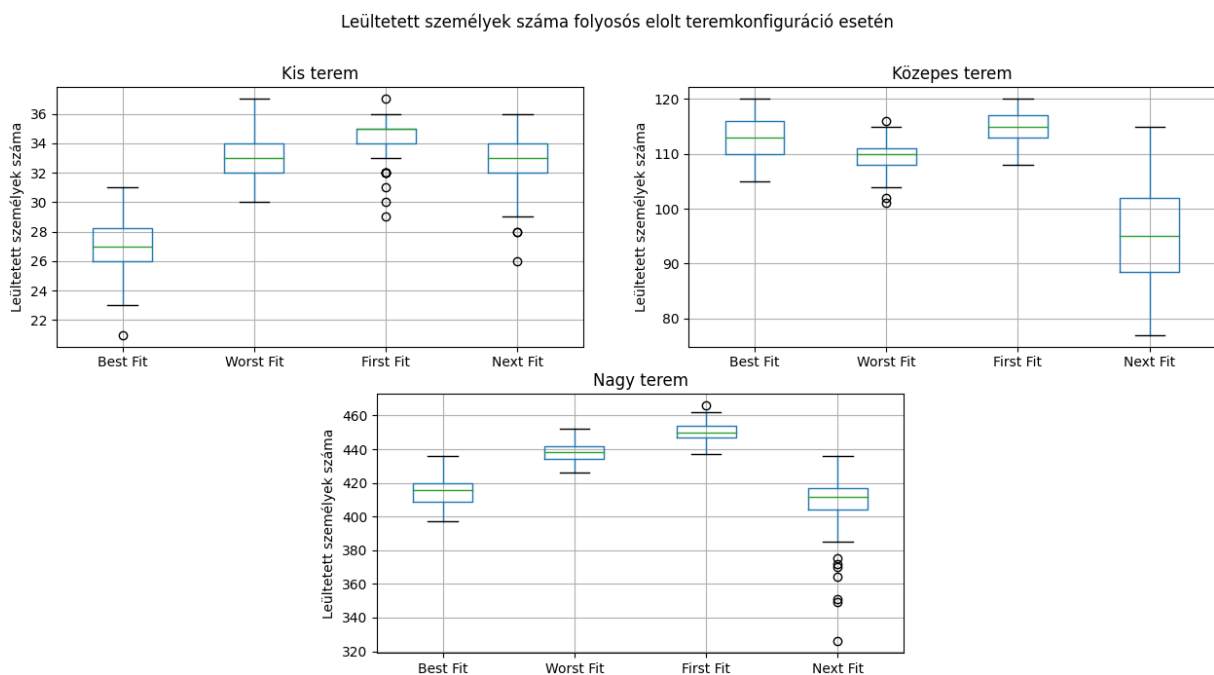
Az *eltolt teremkonfigurációk* esetén kis, közepes és nagy termeken a négy algoritmus az 5. ábrán látható módon teljesít.



5. ábra: A Best Fit, Worst Fit, First Fit, Next Fit algoritmusok által leültetett személyek száma kis, közepes és nagy terem esetén eltolt teremkonfiguráció mellett.

Az ábrákon jól látszik, hogy ebben az esetben a *Best Fit* algoritmus teljesít a legrosszabbul. Ennél egy fokkal jobb eredményeket ad a *Next Fit* algoritmus, viszont ennek a legnagyobb a szórása mind a négy algoritmus közül ezen tesztesetekben. A *Worst Fit* és *First Fit* algoritmusok közel egyformán jól képesek megoldani a problémát, de a *First Fit* algoritmus kicsivel hatékonyabb. Ez alapján azt lehet mondani, hogy *eltolt teremkonfiguráció* esetén a *First Fit* algoritmus nyújtja a legjobb megoldást a négy algoritmus közül.

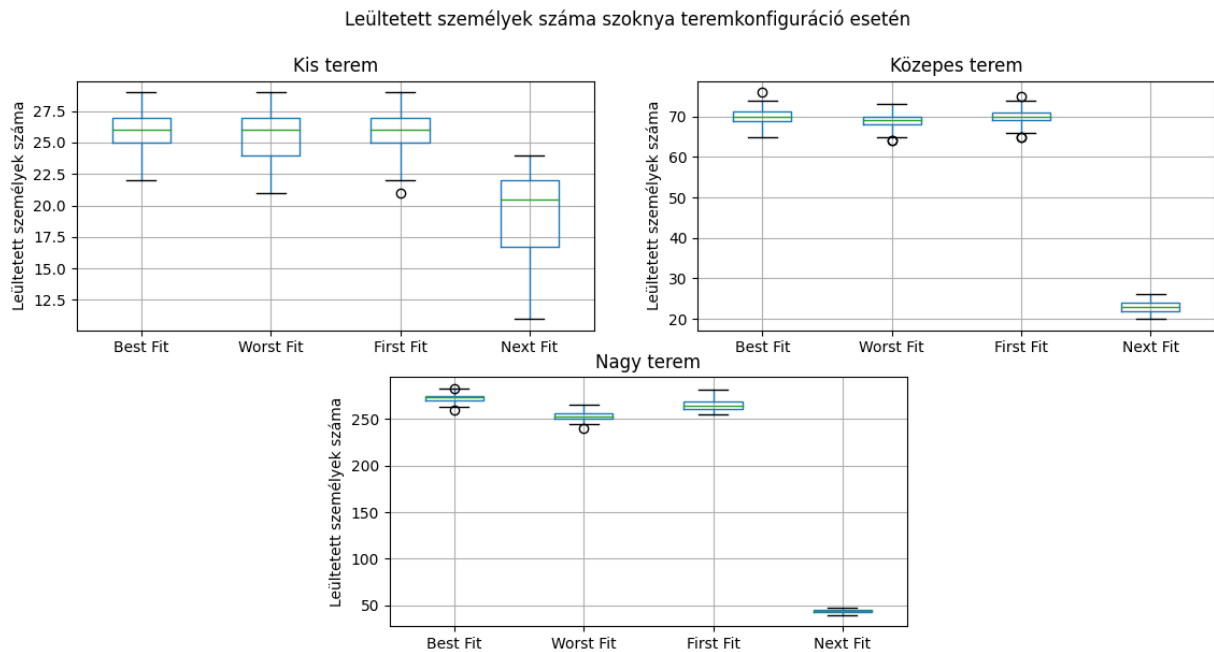
Folyosós eltolt teremkonfigurációk esetén kis, közepes és nagy termeken a négy algoritmus a 6. ábrán látható módon teljesít.



6. ábra: A *Best Fit*, *Worst Fit*, *First Fit*, *Next Fit* algoritmusok által leültetett személyek száma kis, közepes és nagy terem esetén folyosós eltolt teremkonfiguráció mellett.

Az ábrán láthatjuk, hogy *folyosós eltolt teremkonfiguráció* esetén is hasonlóan teljesít a négy algoritmus mint *eltolt teremkonfiguráció* esetén. Itt is azt a következtetést vonhatjuk le, hogy a probléma megoldására ezen teremkonfiguráció mellett a *First Fit* algoritmus a leghatékonyabb.

Szoknya teremkonfiguráció esetén kis, közepes és nagy termeken a négy algoritmus a 7. ábrán látható módon teljesít.



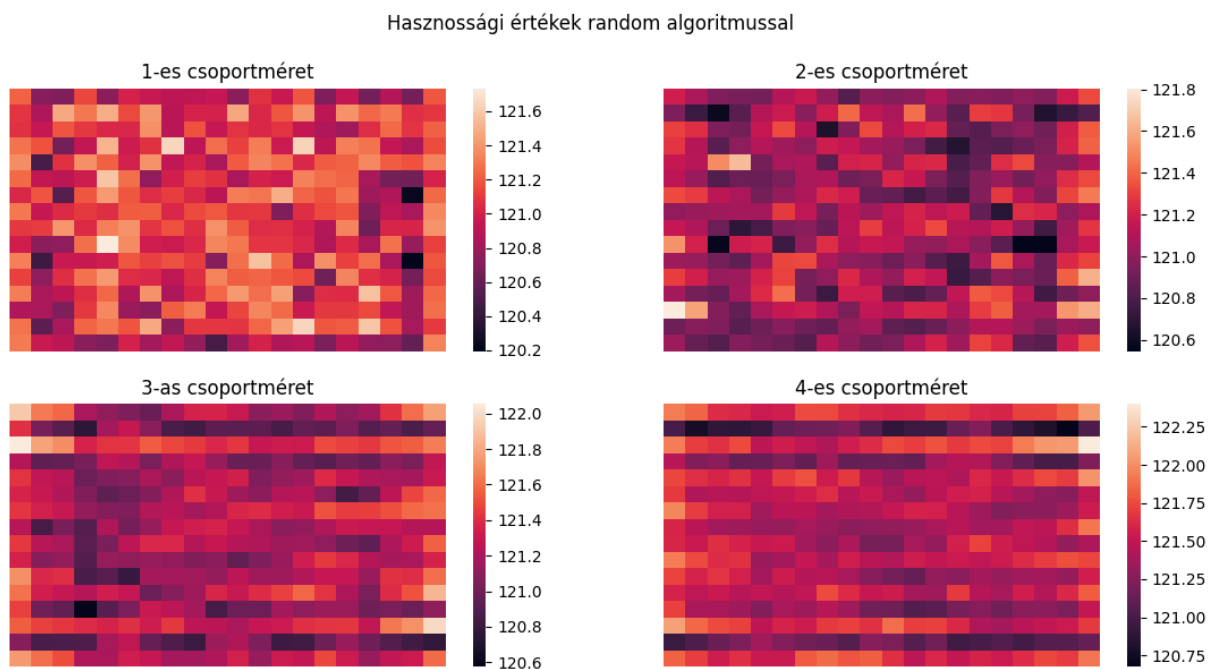
7. ábra: A Best Fit, Worst Fit, First Fit, Next Fi, algoritmusok által leültetett személyek száma kis, közepes és nagy terem esetén szoknya teremkonfiguráció mellett.

Az ábrán jól látható, hogy a *Next Fit* algoritmus teljes mértékben alkalmatlan a probléma megoldására *szoknya teremkonfiguráció* mellett. Ezen kívül a másik három algoritmus nagyon hasonlóan teljesít. Közülük a *Worst Fit* érte el a legkevésbé jó eredményeket, és a szórása is ezen módszernek volt a legnagyobb. A *Best Fit* és *First Fit* algoritmusokat összehasonlítva a *Best Fit* módszer kissé talán meggyőzőbb eredményt ért el, amely főként nagy teremméret mellett látható.

3.3. Az ülőhelyek minősítésének eredménye

3.3.1. Alapalgoritmusok használatával

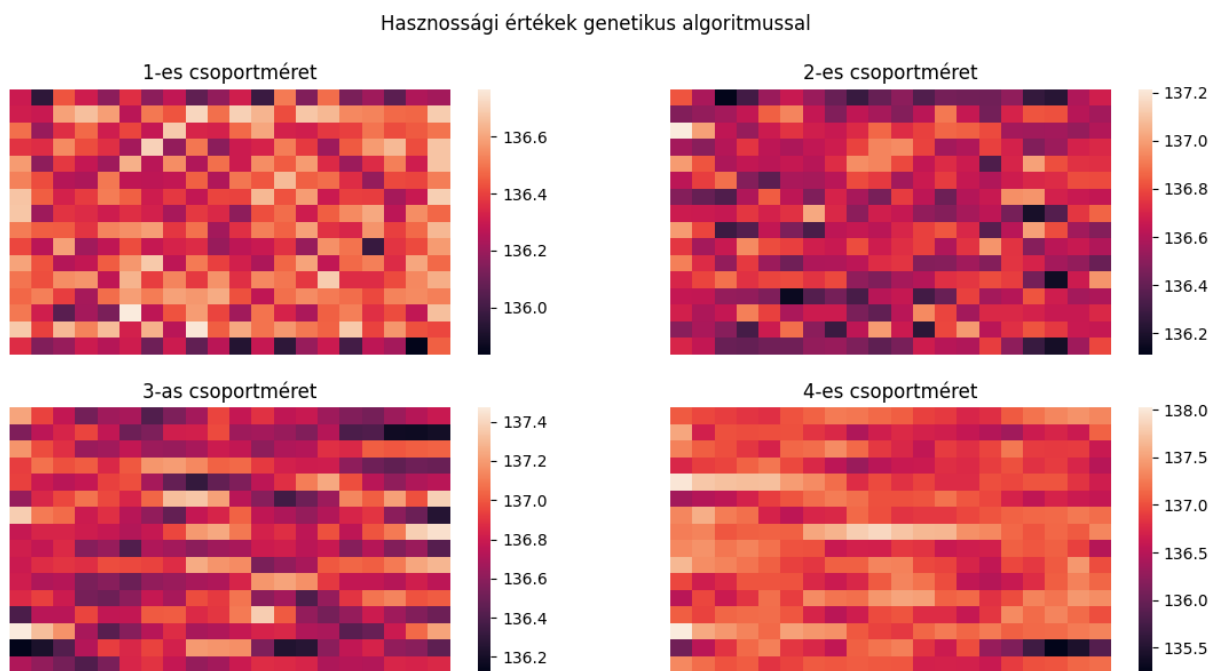
Mint ahogy azt már dolgozatom 2.4.2.1-es fejezetében említettem, az ülőhelyek minősítését a teremben Monte Carlo szimulációk segítségével határoztam meg a (11)-es képlet alapján. Korábban azt is említettem, hogy többféle algoritmust használtam az ülőhelyek minősítésére. Az első ilyen algoritmus a *random algoritmus*, amivel azt igyekeztem szimulálni mintha egy átlagos ember próbálná lefoglalni a helyeket a beérkező csoportok számára a korábban megfogalmazott szabályok alapján, pusztán ránézésre. Az így kapott hasznossági értékek hő térképes megjelenítését közepes teremméretre *nem eltolt teremkonfiguráció* esetén a 8. ábra mutatja.



8. ábra: Közepes terem nem eltolt teremkonfiguráció ülőhelyeinek hasznossági értéke random algoritmussal számolva az egyes csoportméretekre.

Az ábrán láthatjuk, hogy a székek minősítése viszonylag kis tartományban mozog, valamint minél nagyobb csoportméretre vonatkoztatva szeretnénk minősíteni a székeket, annál inkább meghatározó a sor a csoport leültetésére vonatkozóan.

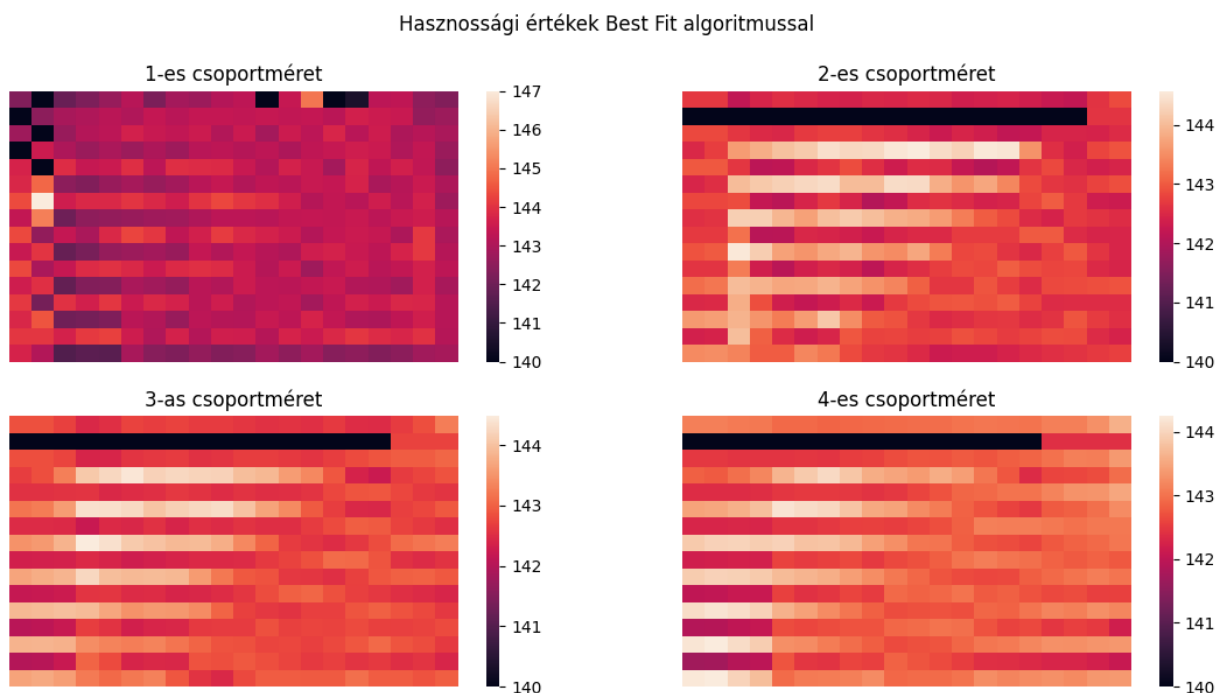
A második ilyen algoritmus a *genetikus algoritmus* volt. Mivel ez az algoritmus képes tökéletes megoldást is találni egy-egy problémára, ezért elképzelhetőnek tűnt, hogy ha a szimulációk során ezt használom egy közel tökéletes mesterséges intelligenciát kaphatok. Az így kapott hasznossági értékek hőterképét közepes teremméretre *nem eltolt teremkonfiguráció* esetén a 9. ábra szemlélteti.



9. ábra: Közepes terem nem eltolt teremkonfiguráció ülőhelyeinek hasznossági értéke genetikus algoritmussal számolva az egyes csoportméretekre.

A 8. és a 9. ábrákat összehasonlítva hasonló mintázatokat láthatunk, de a genetikus algoritmus esetében a nagyobb csoportméreteknel kevésbé rajzolódnak ki karakterisztikusan a sorok. Itt jegyezném meg, hogy munkám során felmerült a minőségi mutatók normálásának a gondolata is a $[0,1]$ tartományra, azonban ezen normálás esetén elveszne azon információ, hogy milyen sikeres volt az adott módszer. Ezért végül nem alkalmaztam normálást az ülőhelyek minősítése során.

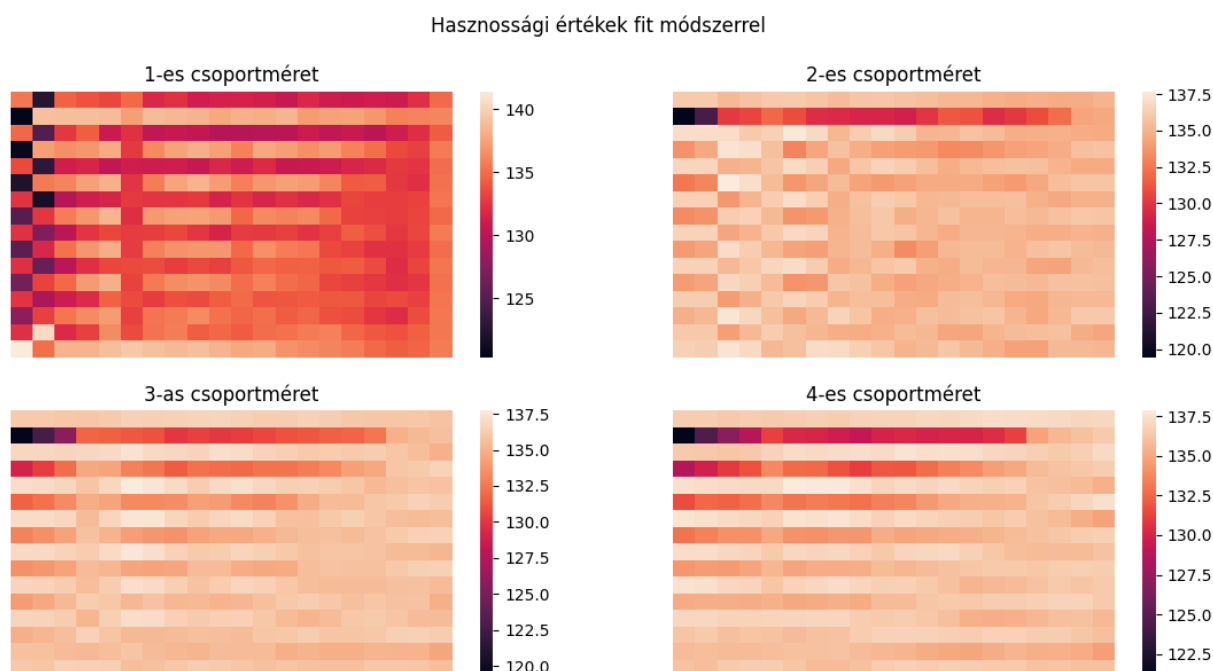
Az ülőhelyek minősítésére alkalmazott harmadik algoritmus a *Best Fit* algoritmus volt. Korábbi tesztek alapján az előzőekben említett *Best Fit*, *First Fit*, *Worst Fit* és *Next Fit* algoritmusok közül ez bizonyult a probléma megoldása során a leghatékonyabbnak. Az így kapott hasznossági értékek hő térképét közepes terem méretre *nem eltolt teremkonfiguráció* esetén a 10. ábra mutatja.



10. ábra: Közepes terem nem eltolt teremkonfiguráció üléshelyeinek hasznossági értéke Best Fit algoritmussal számolva az egyes csoportmértetekre

Az ábrán érdekes megfigyelni, hogy nagyobb csoportmértet esetén mennyire elkülönülnek a páros és páratlan sorok egymástól. Mindemellett az is jól megfigyelhető, hogy az algoritmus működéséből adódóan a terem bal felső mátrixában található székek minősítése karakterisztikusabb (nagyobb eltérések láthatók), mint a jobb alsó mátrixban.

A *Best Fit* algoritmus hibája, hogy a szimulációk jelentős részében a terem egyes részeire nem ültet csoportokat, ebből adódóan előfordulhatnak kihasználatlan helyek. Ezt orvosolja a negyedik módszer, amelyben a *Best Fit* algoritmus csak a beérkező foglalási igények negyedét szolgálja ki, a maradék háromnegyed részből pedig további egy-egy harmadot a *First Fit*, *Next Fit* és *Worst Fit* algoritmusok szolgálnak ki. Továbbiakban ezt a módszer *fit módszernek* nevezem. Az így kapott hasznossági értékek hőtérképes ábrázolását közepes teremméretre *nem eltolt teremkonfiguráció* esetén a 11. ábra szemlélteti.

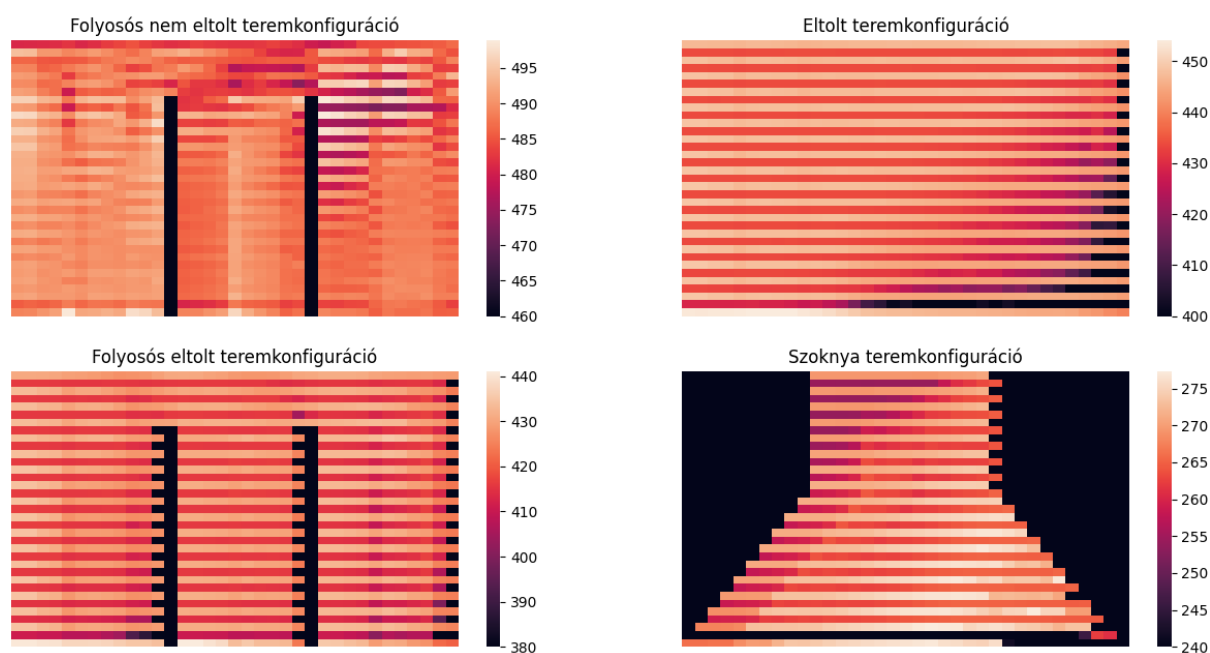


11. ábra: Közepes terem nem eltolt teremkonfiguráció ülőhelyeinek hasznossági értéke fit módszerrel számolva az egyes csoportméretekre.

Az ábrán látható, hogy az első sorokban található székek minősítése továbbra is nagyon eltér annak megfelelően, hogy páros, vagy páratlan sorról van-e szó, azonban ezen különbségek a terem nagyobb részében nem karakterisztikusak.

A kis és nagy teremmérethez az előzőekben bemutatott módon szintén elkészítettem az ülőhelyek minősítését. Az eredmények arányaiban nagyon hasonlóak.

A következőkben szeretném szemléltetni, hogy ugyanazon módszer eltérő teremkonfigurációk esetén milyen eredményeket hozott. A 12. ábra 4-es csoport méretre mutatja a hasznossági értékeket *fit módszerrel* számolva *folyosós nem eltolt*, *eltolt*, *folyosós eltolt* és *szoknya teremkonfigurációkra* nagy teremméret esetén. Az ábrán látható, hogy azon teremkonfigurációk esetében ahol nincs folyosó, ott a hasznossági értékek minden második sorban vesznek fel magasabb értéket. A soron belül ezek az értékek meglehetősen közel vannak egymáshoz. Azon teremkonfigurációk esetén ahol van folyosó és nem alkalmaztunk eltolást, ezen nagyobb hasznossági értékek 4 székenként elcsúsznak egy sorral azokhoz a teremkonfigurációkhoz képest, amelyekben nincs folyosó. Azon teremkonfigurációkban, amelyekben alkalmaztunk eltolást, minél hátrébb lévő sort nézünk, a kevésbé értékes sorok egyre több széke elértéktelenedik.



12. ábra: Nagy terem eltolt, folyosós nem eltolt, folyosós eltolt és szoknya teremkonfigurációk ülőhelyeinek hasznossági értéke fit módszerrel számolva 4-es csoportméretre.

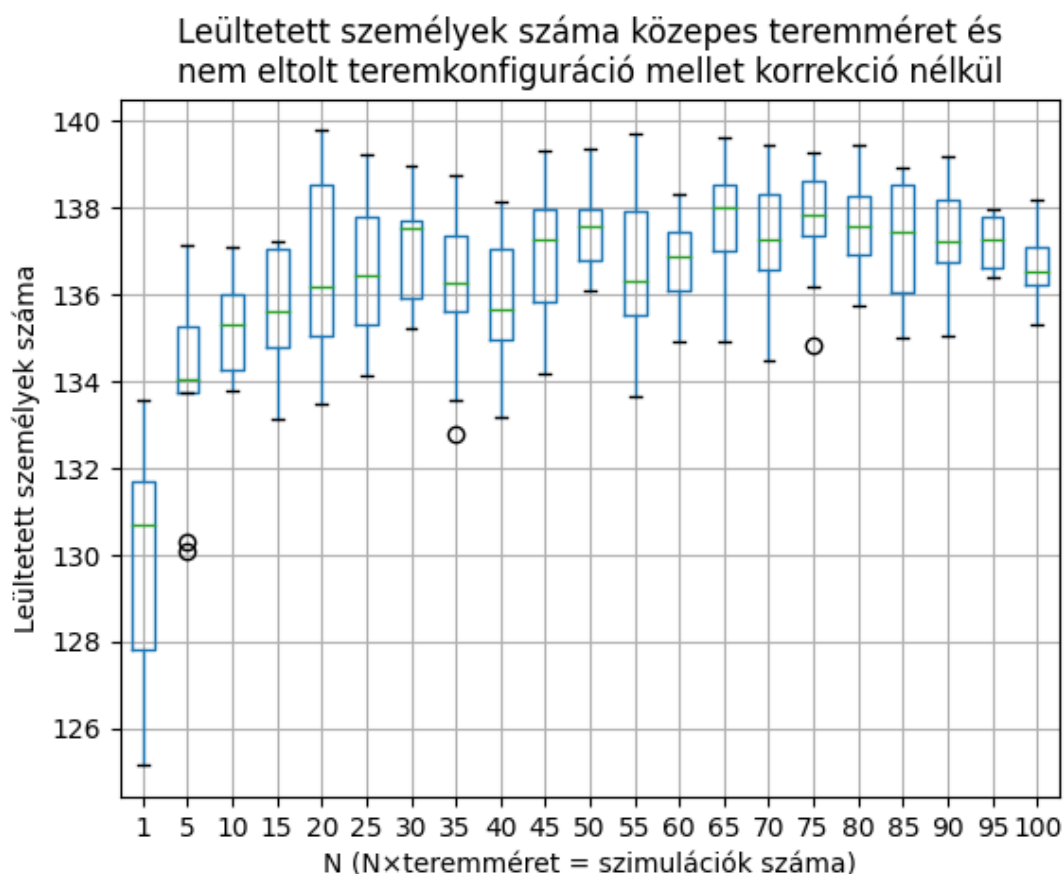
Miután a *fit* módszerrel számolt hasznossági értékek mutatják a legnagyobb strukturáltságot, és közel a legjobb hasznossági értékeket, illetve a futtatott tesztek során ez a módszer bizonyult a leghatékonyabbnak, ezért úgy döntöttem, hogy a továbbiakban ezt a technikát használok az ülőhelyek minősítésére. Viszont a hőterképeken jól látható, hogy a struktúra nem teljes, így indokolt lehet a hasznossági értékek korrekciója.

3.3.2. Korrekciós technikák és szimulációk száma

Miután a *fit* módszerrel számolt hasznossági értékek hőterképe nem mutatott teljes strukturáltságot, ezért indokoltnak láttam a hasznossági értékek korrekcióját. Ezenkívül még nem ismert az a szimulációs szám, amely mellett az algoritmus a legjobb eredményeket adja. Mivel ezen két tényező szorosan összefügg, ezért munkám során őket együttesen vizsgáltam.

Mivel a székek megfelelő minősítéséhez nagyobb teremméret esetén több szimuláció szükséges, ezért munkám során a szimulációs számot a teremméret többszörösének választottam, és azt vizsgáltam, hogy hány-szoros szorzót kell alkalmaznom ahhoz, hogy a lehető legjobb eredményt kapjam a székek minősítésének tekintetében. A tesztek során minden esetben egy új mesterséges intelligencia próbálta megoldani egyenként 100 inputtal a problémát úgy, hogy először csak a teremméret egyszerese volt a szimulációk száma, második körben 5-szöröse, harmadik körben 10-szerese, majd 5-ös lépésközzel egészen 100-szoros teremméret szimulációs számig. Ezután a 100 input ültetési kérésorozatra kapott megoldásait

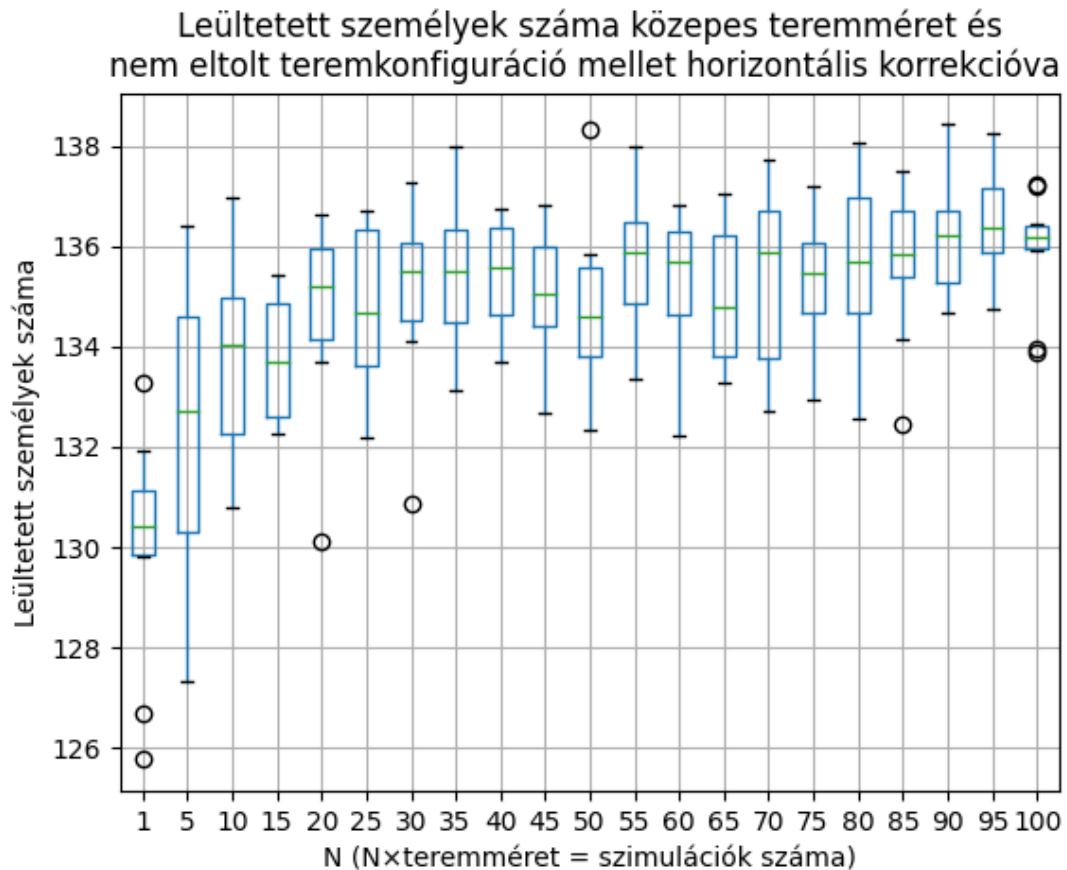
átlagoltam, majd ezt az egész műveletet még 9-szer ismételt meg. A 13-17 ábrák az így kapott eredményeket, vagyis a leültetett személyek számának eloszlását mutatják a szimulációs szám függvényében. A 13. ábrán a *fit módszerrel* számolt ülőhelyminősítés alapján leültetett személyek számának eloszlása látható közepes teremméretű *nem eltolt teremkonfiguráció* esetén.



13. ábra: Közepes terem nem eltolt teremkonfiguráció mellett, korrekció nélkül végzett tesztek eredménye boxplot diagramon.

Az ábrán jól látható, hogy a sikeresen leültetett személyek száma gyorsan növekedésnek indul, majd e növekedés elkezd lassulni. 50-szeres szimulációs számtól pedig az eredmények szórása kezd el inkább csökkenni, mintsem hogy a sikeresen leültetett személyek száma növekedjen.

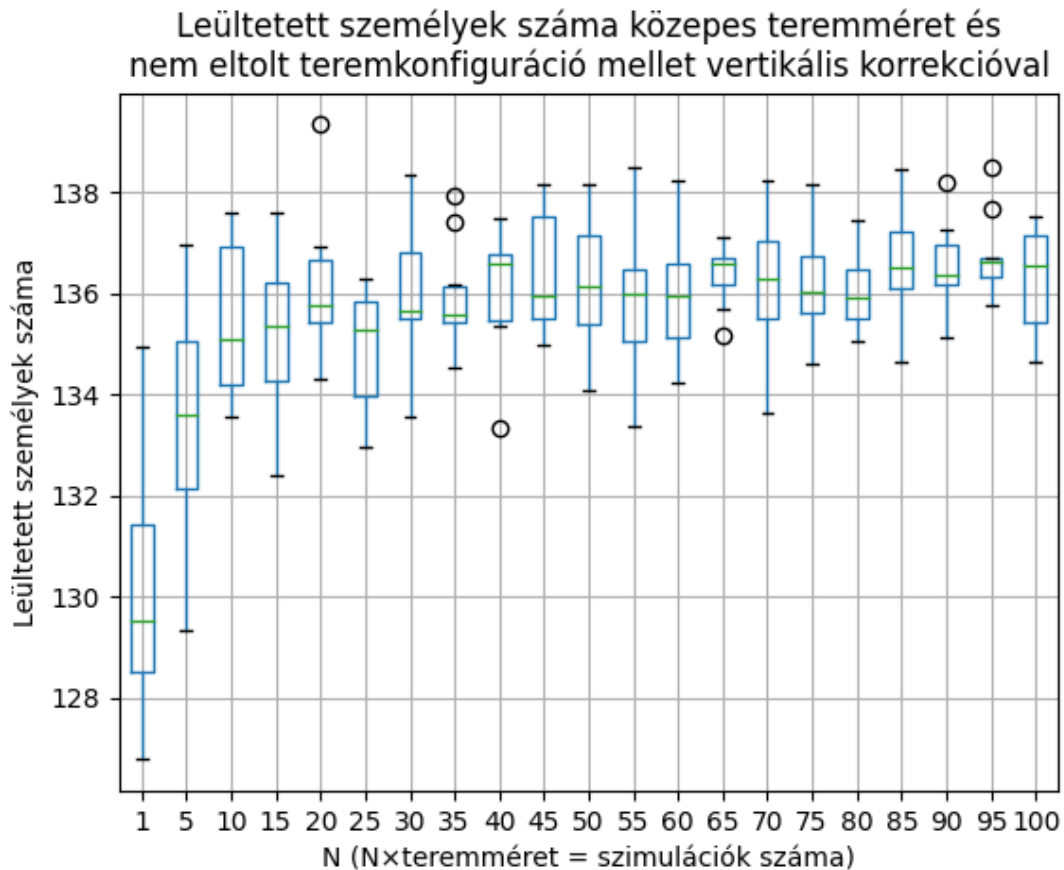
A 14. ábra a *horizontális korrekcióval* (lásd. 14. és 15. képletek) végzett tesztek eredményét mutatja ugyanazon terem esetében.



14. ábra: Közepes terem nem eltolt teremkonfiguráció mellett, horizontális korrekcióval végzett tesztek eredménye boxplot diagramon.

Az ábrán nagyjából ugyanolyan jellegű eredmény látható, mint a korrekció nélküli eseten (13. ábra) annyi különbséggel, hogy összességében kevesebb személyt képes leültetni az algoritmus, mint korrekció nélkül. Ez alapján kimondható, hogy ez a *horizontális korrekció* nem megfelelő a probléma megoldására.

A 15. ábra a *vertikális korrekcióval* (lásd 12. és 13. egyenletek) végzett tesztek eredményét mutatja, szintén ugyanazon teremre vonatkoztatva.

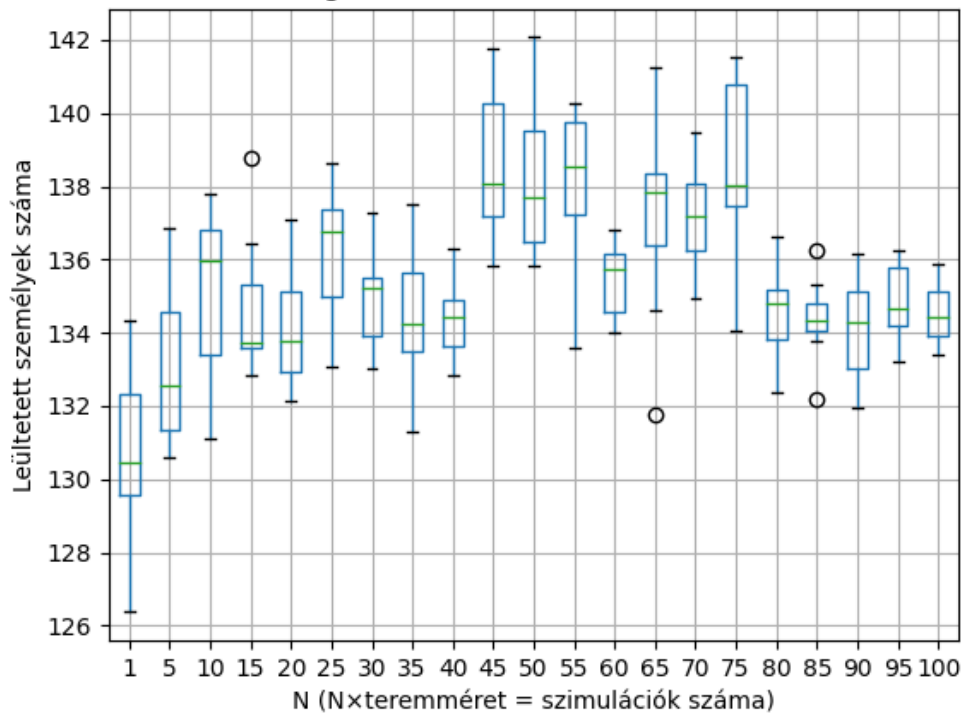


15. ábra: Közepes terem nem eltolt teremkonfiguráció mellett, vertikális korrekcióval végzett tesztek eredménye boxplot diagramon.

Az ábrán látható, hogy *vertikális korrekció* esetén a mesterséges intelligencia által leültetett személyek száma hamar felszökik, de ezután nem látható érdemi javulás. Körülbelül ugyanannyi helyet tud lefoglalni, mint a horizontális korrekció, így ez sem bizonyult megfelelő módszernek.

A 16. ábra a *vertikális & horizontális korrekcióval* (16.-19. egyenletek) végzett tesztek eredményét mutatja, szintén ugyanazon terem esetén.

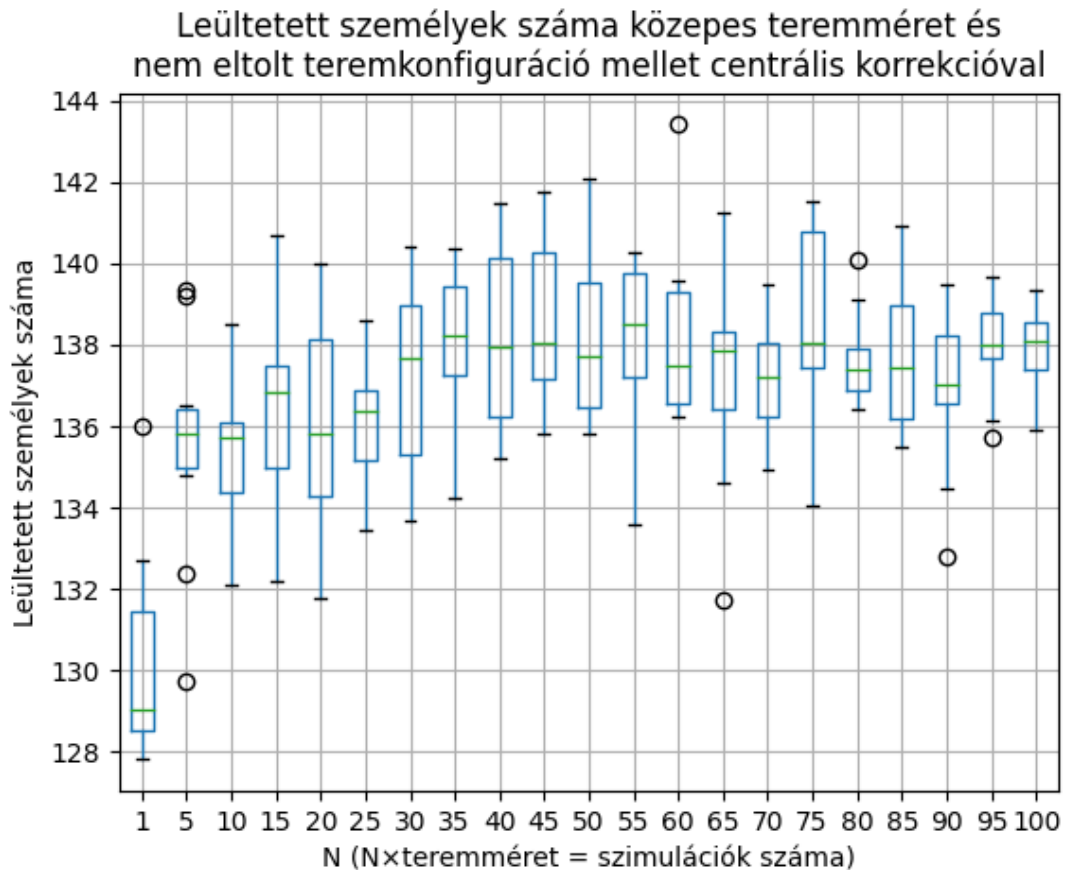
Leültetett személyek száma közepes teremméret és nem eltolt teremkonfiguráció mellett vertikális & horizontális korrekcióval



16. ábra: Közepes terem nem eltolt teremkonfiguráció mellett, vertikális & horizontális korrekció végzett tesztek eredménye boxplot diagramon.

Mint ahogy az előzőekben is látszott, kezdetben a szimulációs számok növelésével a sikeresen lefoglalt helyek száma nagy növekedésnek indul, majd jóval lassabban folytatódik. Középen egy nagy kiugrást láthatunk, mely után egy hasonló eredmény, mint a kiugrás előtt, majd egy újabb kiugrás következik. Ezt követően az eredmények újra beállnak egy érték köré. Ezen érték meglehetősen alacsony, így azt sem tekinthetjük jó megoldásnak.

A 17. ábra a *centrális korrekcióval* (20.-22. egyenletek) végzett tesztek eredményét mutatja ugyanazon teremre vonatkoztatva.



17. ábra: Közepes terem nem eltolt teremkonfiguráció mellett, centrális korrekció végzett tesztek eredménye boxplot diagramon.

Ahogy az az ábrán látható, a leültetett személyek száma gyorsan megugrik majd lassú, de biztos növekedést mutat. 50-szeres szimulációs számnál eléri a maximum teljesítményt és innentől csak az eredmények szórása lesz egyre kisebb és kisebb. Ugyanezt a jellegű eredményt tapasztaltam abban az esetben is, ha nem használtam korrekciót (lásd 13. ábra). Ezek alapján kijelenthető, hogy a szükséges szimulációs szám a teremméret 50-szerese, de annak érdekében, hogy stabil eredményt kapjunk érdemes ezt a számot akár 100-ra is felemelni. Mindemellett mivel a *centrális korrekciót* alkalmazó módszer adja a legjobb eredményeket a probléma megoldására, ezért a továbbiakban a *centrális korrekciós* technikát fogom használni 100 szoros szimulációs szám mellett.

3.4. Ülőhelyfoglalás passzív megerősítéses tanulást alkalmazó mesterséges intelligencia algoritmussal

Miután sikeresen minősítettük az ülőhelyeket és a megfelelő szimulációs számot is sikerült meghatároznunk, ideje megtekinteni, hogy az így sikeresen betanított mesterséges intelligencia milyen eredményeket ad, és ezen eredmények hogyan viszonyulnak a korábban már említett

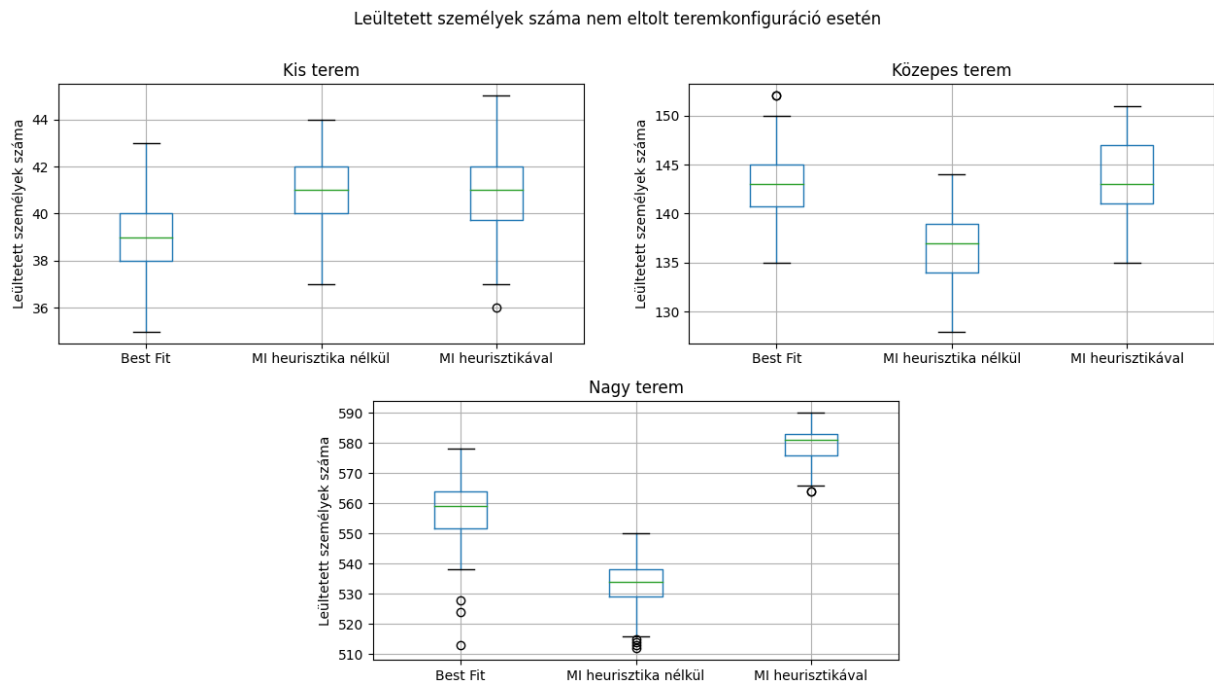
algoritmusokhoz. Az elkészült mesterséges intelligenciát kétféle módon teszteltem. Az első mód az, hogy csak a székek minősítése alapján ülteti le az embereket. Ezt innentől úgy nevezem, hogy *MI heurisztika nélkül*. A második eset megegyezik a 2.4.2.2. fejezetben tárgyalt ültetési stratégiával. Ezt innentől *MI heurisztikával* néven említem. A 2. táblázat a különféle algoritmusok által elért eredményeket tartalmazza különféle teremméretekre és teremkonfigurációkra vonatkoztatva.

2. táblázat: A Best Fit, Worst Fit, First Fit, Next Fit algoritmusok, illetve a két mesterséges intelligencia által leültetett személyek számának átlaga kis, közepes és nagy terem esetén nem eltol, eltol, folyosós eltol, folyosós nem eltol és szoknya teremkonfig

Nagy terem								
	Best Fit	Worst Fit	First Fit	Next Fit	MI heurisztika nélkül	MI heurisztikával	Maximum	Megoldási módszer
Nem eltol	557,99	503,94	535,33	456,37	532,69	579,41	579,41	MI heurisztikával
Folyosós nem eltol	512,79	487,95	517,33	426,23	508,36	549,44	549,44	MI heurisztikával
Eltol	433,51	452,73	457,50	439,68	442,69	454,95	457,50	First Fit
Folyosós eltol	414,93	438,32	450,09	407,54	439,33	461,06	461,06	MI heurisztikával
Szoknya	272,39	253,51	264,75	43,99	270,09	275,68	275,68	MI heurisztikával
Közepes terem								
	Best Fit	Worst Fit	First Fit	Next Fit	MI heurisztika nélkül	MI heurisztikával	Maximum	Megoldási módszer
Nem eltol	142,85	134,64	139,82	126,32	136,86	143,69	143,69	MI heurisztikával
Folyosós nem eltol	139,60	125,31	133,32	115,42	138,84	140,82	140,82	MI heurisztikával
Eltol	111,08	115,42	117,44	112,46	114,87	117,36	117,44	First Fit
Folyosós eltol	112,77	109,39	114,89	94,99	109,53	113,48	114,89	First Fit
Szoknya	69,95	68,95	69,96	23,02	68,64	70,30	70,30	MI heurisztikával
Kis terem								
	Best Fit	Worst Fit	First Fit	Next Fit	MI heurisztika nélkül	MI heurisztikával	Maximum	Megoldási módszer
Nem eltol	39,11	37,36	38,63	36,85	40,77	40,69	40,77	MI heurisztika nélkül
Folyosós nem eltol	37,74	34,62	36,78	34,00	37,49	37,50	37,74	Best Fit
Eltol	27,69	34,95	35,62	34,51	33,74	35,01	35,62	First Fit
Folyosós eltol	27,13	33,21	34,41	32,64	32,67	31,58	34,41	First Fit
Szoknya	25,75	25,62	25,56	19,39	25,04	25,49	25,75	Best Fit

A táblázatban jól látszik, hogy minél nagyobb teremről van szó, annál jobban teljesít a mesterséges intelligencia, s ezen belül is annak a heurisztikával kiegészített (*MI heurisztikával*) verziója. Emellett az is látszik, hogy kis terem esetén talán jobb, ha nem használunk heurisztikát, mivel ez esetben a hagyományos algoritmusok jobban teljesítenek a mesterséges intelligenciánál.

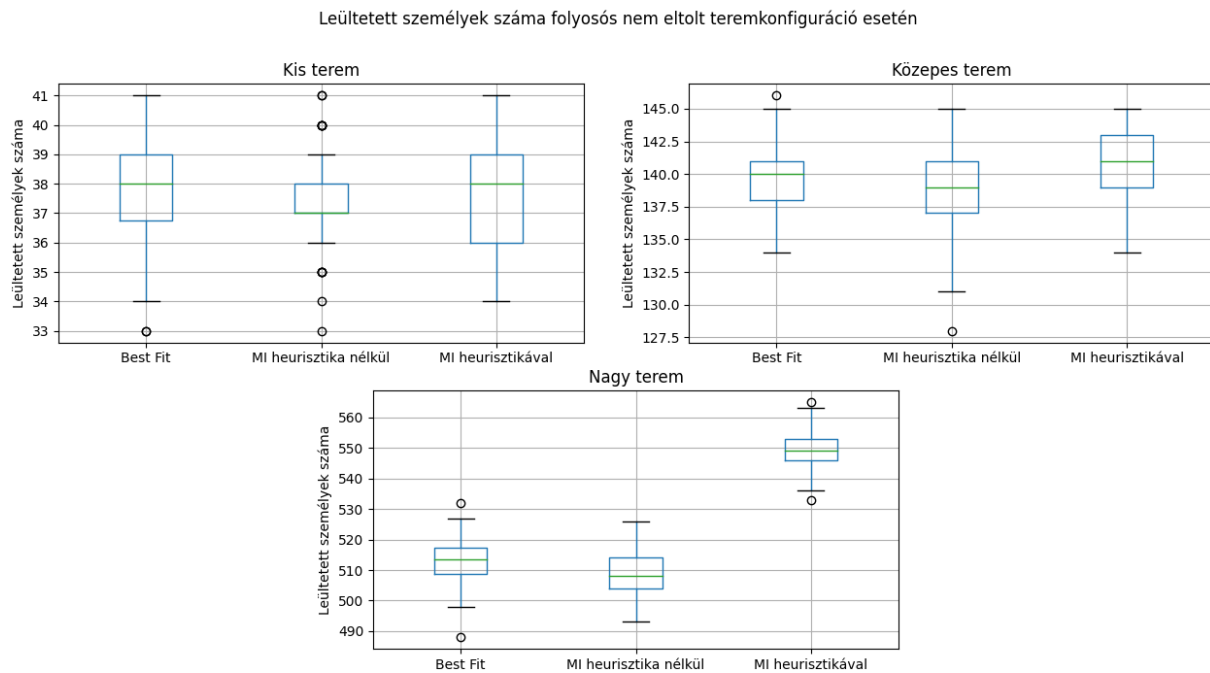
A fenti táblázatból bár le lehet vonni néhány következtetést, érdekesebb mégis részletesebben is megvizsgálni az eredményeket. A részletesebb rálátás érdekében a következőkben boxplot diagramokon mutatom be az általam fejlesztett mesterséges intelligencia teljesítményét. Mivel korábban már bemutattam a *Best Fit*, *Worst Fit*, *First Fit*, és *Next Fit* algoritmusok teljesítményét, ezért az alábbi ábrákon csak az adott teremméret és teremkonfiguráció esetén legjobban teljesítő algoritmussal hasonlítom össze a mesterséges intelligencia eredményét. A 18. ábra *nem eltol teremkonfiguráció* esetén mutatja a mesterséges intelligencia által sikeresen leültetett személyek számát.



18. ábra: A Best Fit, MI heurisztika nélkül és az MI heurisztikával algoritmusok által sikeresen leültetett személyek száma kis, közepes és nagy terem esetén nem eltolt teremkonfiguráció mellett

Az ábrán jól látszik, hogy kis terem esetén a mesterséges intelligencia mindkét verziója jobban teljesít, mint a *Best Fit*, az *MI heurisztika nélkül* pedig talán kicsit jobban is, mint az *MI heurisztikával*. Közepes és nagy terem esetén ez már nem mondható el, mivel az *MI heurisztika nélkül* eredménye a leggyengébb, ezt követi a *Best Fit*, legjobb teljesítményt pedig az *MI heurisztikával* módszer éri el.

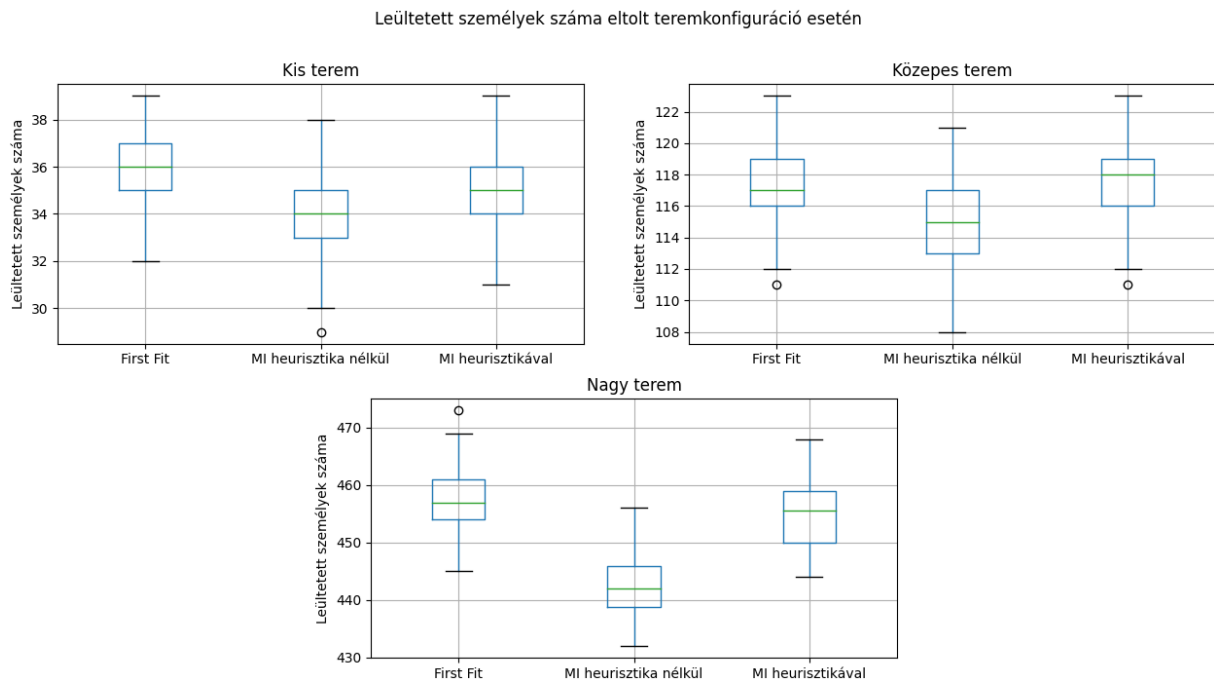
A 19. ábra *folyosós nem eltolt teremkonfiguráció* esetén mutatja a mesterséges intelligencia által sikeresen leültetett személyek számát.



19. ábra: A Best Fit, MI heurisztika nélkül és az MI heurisztikával által sikeresen leültetett személyek száma kis, közepes és nagy terem esetén folyosós nem eltolt teremkonfiguráció mellett.

Az ábrán látszik, hogy *folyosós nem eltolt teremkonfiguráció* mellett kis terem esetén nem igazán érdemes alkalmazni a mesterséges intelligenciát, mivel nagyjából ugyanolyan eredményeket ad, mint a *Best Fit*, csak jóval nagyobb szórással. Közepes terem méret esetén már érdemes lehet használni erre a teremkonfigurációra az *MI heurisztikával* módszert, mivel ebben az esetben már hatékonyabb, mint a *Best Fit*, viszont az *MI heurisztika nélkül* algoritmus körülbelül ugyanúgy teljesít, illetve egy kicsit rosszabbul is. Nagy terem esetén még jobban kijön a különbség. Az *MI heurisztikával* már messze megelőzi a *Best Fit*-et, míg az *MI heurisztikával* arányaiban ugyanúgy teljesít, mint közepes terem esetén. Így megállapítható, hogy nagyobb teremméretek esetén *folyosós nem eltolt teremkonfiguráció* mellett az *MI heurisztikával* a megfelelő módszer.

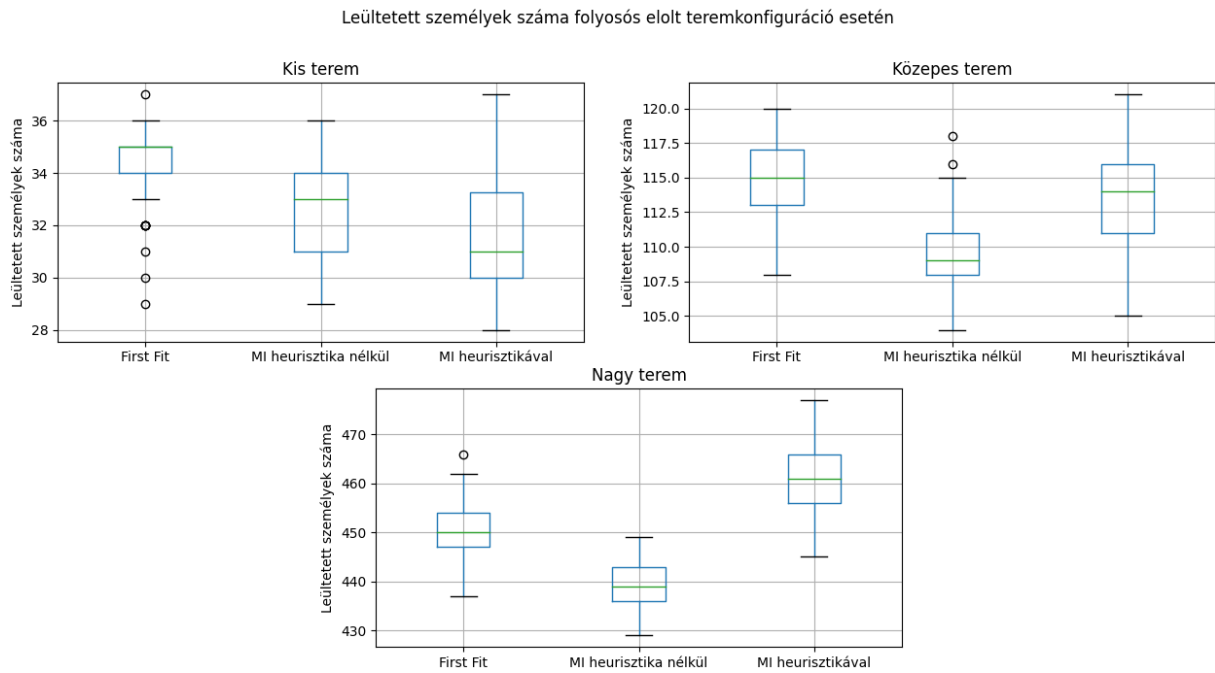
A 20. ábra *eltolt teremkonfiguráció* esetén mutatja a mesterséges intelligencia által sikeresen leültetett személyek számát.



20. ábra: A Best Fit, MI heurisztika nélkül és az MI heurisztikával által sikeresen leültetett személyek száma kis, közepes és nagy terem esetén eltolt teremkonfiguráció mellett.

Az ábrán láthatjuk, hogy kis terem esetén *eltolt teremkonfiguráció* mellett egyáltalán nem érdemes mesterséges intelligenciát használni, ez esetben a *First Fit* jobban teljesít, mint a mesterséges intelligencia mindkét verziója. Viszont közepes és nagy terem esetén az *MI heurisztikával* már nagyon hasonlóan teljesít, mint a *First Fit* metódus, sőt közepes terem esetén egy kicsit jobban. Ezek alapján nem mondható biztosra, de véleményem szerint nagy termék esetén érdemes lehet a mesterséges intelligenciát alkalmazni.

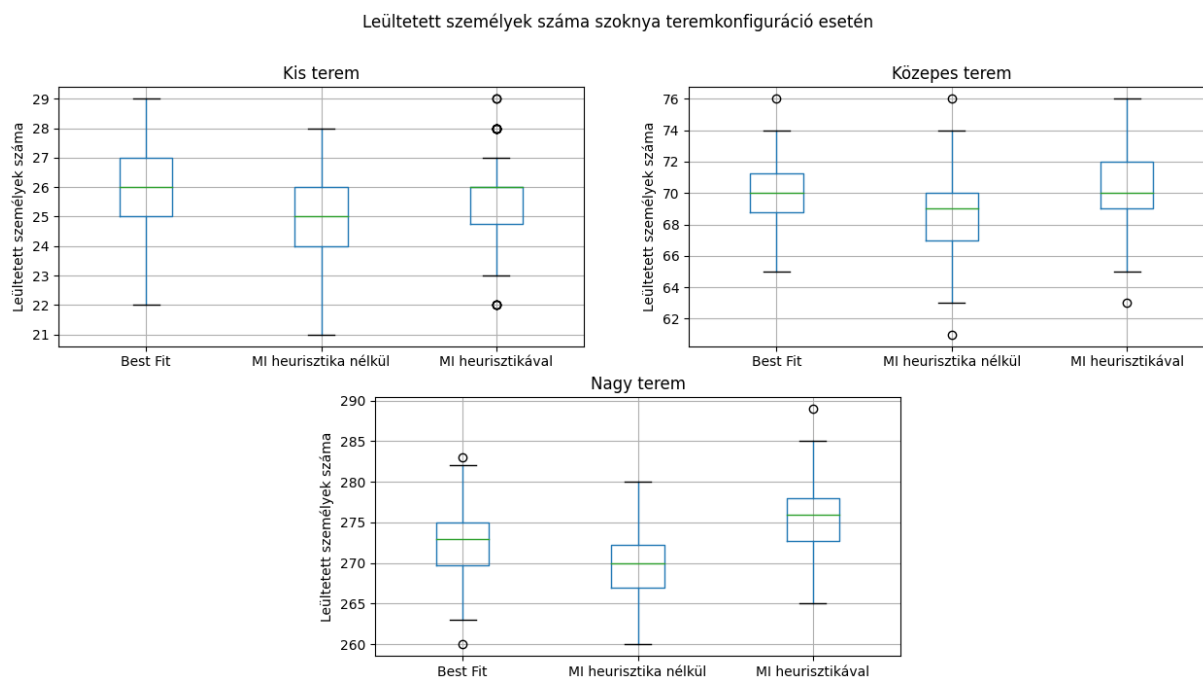
A 21. ábra *folyosós eltolt teremkonfiguráció* esetén mutatja a mesterséges intelligencia által sikeresen leültetett személyek számának eloszlását.



21. ábra: A Best Fit, MI heurisztika nélkül és az MI heurisztikával által sikeresen leültetett személyek száma kis, közepes és nagy terem esetén folyosós elolt teremkonfiguráció mellett.

Az ábrán jól látszik, hogy kis teremméret esetén a *First Fit* teljesít a legjobban, ezt követi az *MI heurisztika nélkül* és a legutolsó helyen végez az *MI heurisztikával*. Közepes teremméret esetén viszont már az *MI heurisztikával* jobban teljesít, mint az *MI heurisztika nélkül*. Nagy terem esetén pedig az *MI heurisztikával* már mindegyik módszernél jobban teljesít. Ezek alapján az mondható, hogy nagy termék esetén *folyosós elolt teremkonfiguráció* mellett érdemes alkalmazni az *MI heurisztikával* elnevezésű módszert.

A 22. ábra a *szoknya teremkonfiguráció* esetén mutatja a mesterséges intelligencia által sikeresen leültetett személyek számának eloszlását.



22. ábra: A Best Fit, MI heurisztika nélkül és az MI heurisztikával által sikeresen leültetett személyek száma kis, közepes és nagy terem esetén szoknya teremkonfiguráció mellett.

Mint ahogy az az ábrán is látszik, szoknya teremkonfiguráció mellett kis terem esetén a *Best Fit* teljesít a legjobban, viszont nagyon hasonló eredményeket ad az *MI heurisztikával* és csak kicsivel marad le az *MI heurisztika* nélkül. Közepes terem esetén is ugyanez mondható el. Nagy terem esetén viszont az *MI heurisztikával* már valamivel jobban teljesít a *Best Fit*-nél. Ezek alapján kimondható, hogy nagy termék esetén érdemes használni az *MI heurisztikával*-t szoknya teremkonfiguráció esetén is.

Összességében a legtöbb teremkonfiguráció esetén elmondható, hogy nagy termék esetén érdemes mesterséges intelligenciát használni annak is a *MI heurisztikával* verzióját. Kis termék esetén viszont egyáltalán nem érdemes mesterséges intelligenciát használni, s eltérő teremkonfigurációk esetén eltérő algoritmusokat érdemes használni. Közepes termék esetén az optimális algoritmus kiválasztása függ a teremkonfigurációtól.

4. Összefoglalás

Napjainkban érdekes kihívást jelent a személyek megadott szabályoknak megfelelő leültetése egy rendezvényen, abban az esetben, ha ezen személyek csoportja előre nem ismert, és az érkezés/foglalás pillanatában kell helyet biztosítani számunkra oly módon, hogy a biztonságos távolság meglegyen a többi csoporthoz képest, és a csoport tagjai egymás mellett foglaljanak helyet. Miután nagy teremméret esetén ezt emberi erővel nehéz hatékonyan elvégezni, ezért indokoltnak látom a probléma támogatását egy mesterséges intelligenciával. Dolgozatomban egy új, feladatspecifikus, passzív megerősítéses tanuláson alapuló online optimalizálási módszert fejlesztettem a probléma megoldásának támogatására.

A fejlesztés során több teremkonfigurációt határoztam meg. Minden teremkonfigurációt implementáltam kis, közepes és nagy méretben is. Első lépésben megvizsgáltam a probléma megoldására már létező klasszikus allokációs módszereket. Teremkonfigurációtól függően a *Best Fit* és a *First Fit* bizonyult a leghatékonyabbnak ezen algoritmusok közül.

A passzív megerősítéses tanulást alkalmazó mesterséges intelligencia betanítása során négyféle módszert alkalmaztam. Ezek a *random algoritmus*, *genetikus algoritmus*, *Best Fit* és *fit módszer*. A tesztek azt mutatták, hogy a leghatékonyabb a *fit módszer*, így ezen módszert használtam az ülőhelyek minősítésre vonatkozóan.

Miután a *fit módszerrel* számolt hasznossági értékek valamilyen struktúrát mutattak, de nem mindenhol, így indokoltnak láttam az ülőhelyek hasznossági értékének korrekcióját. Ehhez négyféle módszert határoztam meg, ezek a *vertikális korrekció*, *horizontális korrekció*, *vertikális & horizontális korrekció* és a *centrális korrekció*. Ezen négy korrekciós módszer közül a *centrális korrekció* bizonyult a leghatékonyabbnak. Emellett arra is sikerült fényt derítenem, hogy a megfelelő szimulációszám legalább a terem méretének 50-szerese. Annak érdekében, hogy minél stabilabb eredményeket adjon a mesterséges intelligencia algoritmus, a szimulációs számot érdemes a teremméret 100-szorosára növelni.

Az így megalkotott mesterséges intelligencia közepes és nagy terem esetén hatékonyan működik, közepes terem esetén azonban az eredmények függenek a terem konfigurációjától is. Nagy teremméretek esetén már meglehetősen jobb eredményeket produkál a mesterséges intelligencia, mint a klasszikus allokációs módszerek. Kis terem esetén a klasszikus allokációs módszerek hatékonyabb megoldást adnak.

Egy lehetséges továbbfejlesztési lehetőség, hogy *fit módszerrel* tanítjuk a mesterséges intelligenciát, de kivesszük belőle a *Next Fit* algoritmust. Egy másik lehetséges továbbfejlesztési mód a korrekciós technikákra térne ki, melynek lényege, hogy a hasznossági

értékek bal felső háromszögmátrixát tükrözzük középpontosan a jobb alsó háromszögmátrixra. Egy harmadik lehetséges továbbfejlesztési módszer egy olyan heurisztika bevezetése mely az egyforma méretű csoportokat egymás mögé ültetné, vagy már a tanulási fázisban egy olyan algoritmussal tanítanánk a mesterséges intelligenciát, amely egymás mögé ültetné ezen csoportokat.

Irodalomjegyzék

- [1] Xinchang Wang, Hua ,Wang, Xiaoning Zhang. Stochastic seat allocation models for passenger rail transportation under customer choice. In: *Transportation Research Part E: Logistics and Transportation Review* Volume 96, December 2016, Pages 95-112
- [2] Moshe Dror, Pierre Trudeau Shaul, P. Ladany. Network models for seat allocation on flights. In: *Transportation Research Part B: Methodological* Volume 22, Issue 4, August 1988, Pages 239-250
- [3] Wen Zhao, Yu-Sheng Zheng. A Dynamic Model for Airline Seat Allocation with Passenger Diversion and No-Shows In: *Transportation Science* Volume 35, Issue 1, February 2001, Pages 1-105
- [4] Zhenying Yan, Xiaojuan Li, Qi Zhang, Baoming Han. Seat allocation model for high-speed railway passenger transportation based on flexible train composition. In: *Computers & Industrial Engineering* Volume 142, April 2020, 106383
- [5] Claire Cizaire, Peter Belobaba. Joint optimization of airline pricing and fare class seat allocation. In: *Journal of Revenue and Pricing Management* Volume 12, 21 December 2012, p. 83–93
- [6] Xinlei Hu, Feng Shi, Guangming Xu, Jin Qin. Joint optimization of pricing and seat allocation with multistage and discriminatory strategies in high-speed rail networks. In: *Computers & Industrial Engineering* Volume 148, October 2020, 106690
- [7] Dósa, György, and Jiri Sgall. First Fit bin packing: A tight analysis. In *30th International Symposium on Theoretical Aspects of Computer Science (STACS 2013)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.
- [8] Dósa, György; Sgall, Jiří. Optimal analysis of best fit bin packing. In: *International Colloquium on Automata, Languages, and Programming*. Springer, Berlin, Heidelberg, 2014. p. 429-441.
- [9] Bays, Carter. A comparison of next-fit, first-fit, and best-fit. *Communications of the ACM*, 1977, 20.3: 191-192.

- [10] Rao, Qing; Frtunikj, Jelena. Deep learning for self-driving cars: Chances and challenges. In: *Proceedings of the 1st International Workshop on Software Engineering for AI in Autonomous Systems*, 2018, 35-38.
- [11] Zhao, Zhong-Qiu, et al. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 2019, 30.11: 3212-3232
- [12] Gacanin, Haris; Wagner, Mark. Artificial intelligence paradigm for customer experience management in next-generation networks: Challenges and perspectives. *IEEE Network*, 2019, 33.2: 188-194.