



Budapesti Műszaki és Gazdaságtudományi Egyetem
Közlekedésmérnöki és Járműmérnöki Kar
Közlekedésüzemi Tanszék

***Alkalmazás fejlesztése különböző
cikkelemek raktári tárolótéren történő
optimális elhelyezéséhez***

TDK dolgozat

Készítette:

Fésüs Norbert, BSc képzés, Közlekedésmérnöki szak,
(V. évfolyam)

Konzulens:

Kardos Ádám, PhD hallgató, Közlekedésüzemi Tanszék

2012

A dolgozat bemutatását a TÁMOP - 4.2.2.B-10/1--2010-0009 "Új tehetséggondozó programok és kutatások a Műegyetem tudományos műhelyeiben" című program támogatta.

Tartalomjegyzék

BEVEZETÉS	4
1. A RAKTÁRI TÁROLÓTEREK TECHNOLÓGIAI TERVEZÉSÉNEK ÁTTEKINTÉSE	6
1.1. A raktározás jelentősége napjainkban	6
1.2. A raktári tárolótér előtervezése	7
1.3. A tárolótéri áruelrendezés kérdése	8
2. A GENETIKUS ALGORITMUS, MINT LÁGY SZÁMÍTÁS.....	9
2.1. A genetikus algoritmus, mint optimum-kereső eljárás.....	10
2.2. Egy általános genetikus algoritmus működése.....	12
3. AZ OPTIMÁLIS ÁRUELRENDEZÉS KIALAKÍTÁSÁHOZ FEJLESZTETT ALKALMAZÁS	17
3.1. Az alkalmazás célja	17
3.2. A feladat lehatárolása.....	17
3.3. A szükséges kiindulási paraméterek.....	18
3.4. Inicializáció	20
3.4.1. A tárolótéri lokációk felkeresési idejének meghatározása	20
3.4.2. Az egyes árucikkekhez tartozó felkeresési gyakoriságok meghatározása	22
3.4.3. Az áruelrendezést jellemző paraméterek	23
3.5. Az első részeredmény: a legjobb elérési idők meghatározása	24
3.6. A probléma és a genetikus algoritmus kapcsolata	25
3.6.1. A fitness függvény meghatározása	26
3.6.2. Az áruelrendezés technikai leképezése a genetikus algoritmushoz	29
3.6.3. A reprodukciós operátor: rulett-kerék kiválasztás	31
3.6.4. A keresztezési operátor: Partially Mapped Crossover	32
3.6.5. A mutációs operátor: Inverted Displacement	34
3.7. A genetikus algoritmus futtatási eredményei	35
3.7.1. A teszteléshez használt paraméterek és beállítások	35
3.7.2. A tesztfuttatás eredményeinek értékelése	35
4. TOVÁBBI FEJLESZTÉS IRÁNYVONALAI, A KORLÁTOK FEOLDÁSÁNAK LEHETŐSÉGEI.....	40

5. AIM SEGÉDPROGRAM BEMUTATÁSA.....	42
5.1. Az AIM módszer és használata	42
5.2. Az AIMaster program célja.....	44
5.3. A program funkciói és használata	45
5.4. A program további fejlesztési lehetőségei	50
ÖSSZEFOGLALÁS	51
ÁBRAJEGYZÉK.....	53
IRODALOMJEGYZÉK.....	54

BEVEZETÉS

A raktárak funkciójukat tekintve puffer-tárolóként támogatják az ellátási lánc működését. A raktárak legnagyobb hátránya, hogy alapesetben nem teremtenek értéket, viszont a beruházási- és üzemeltetési költségük meglehetősen magas. A termékek az egyes felhasználási pontok közötti tárolása nélkül valósul meg, hogy az hozzáadott értéket teremtsenek, így a raktározási folyamat elsősorban a termékhez kapcsolódó logisztikai költségeket (és ezzel együtt a végső fogyasztói árat) növeli.

A logisztikai költségek jelentős részét képezik a raktári operációval kapcsolatos költségek, éppen ezért ezen költségek csökkentése a termék végfelhasználói árat csökkentheti, illetve egyes esetekben növelheti is a vevői kiszolgálás színvonalát (pl. hamarabb jut a vevő a termékhez). Az operációval kapcsolatos költségek jelentős mértékben csökkenthetők, ha a raktárak technológiai tervezésének korai fázisban – már az előzetes tervezés során – a rendszer működését tekintve az optimális layout kialakítására törekszünk.

A Közlekedésüzemi Tanszéken az Anyagmozgatási és raktározási rendszerek tervezése I. című tantárgy keretében egy hagyományos kialakítású, soros állványos, nagyemelésű gépi targoncás kiszolgálású raktári rendszer technológiai tervezése volt a feladat, amely során megismerkedtünk a tárolótér tervezésének jelentőségével.

A tárolótéren történő különböző árucsoportok árucikkeinek kiosztását, vagyis az árucikkek a tárolótér soros állványos tárolási rendszerének egyes lokációhoz való hozzárendelését intuitíven (ebből fakadóan heurisztikusan) oldottuk meg a főbb tervezési szempontok figyelembevételével. Könnyen belátható, hogy az ilyesféle „próbálgatásos” módszer rengeteg hibát rejthet magában, így az optimális tárolótéri elrendezés megtalálása is nehézségbe ütközik.

A korszerű számítástechnikai eszközök és az optimalizálási módszerek fejlődésével lehetőség nyílik a tárolótéri áruelrendezés számítógéppel támogatott tervezésére. Jelen dolgozatom célja egy olyan *alkalmazás fejlesztése*, amely a tárolótérek előzetes, technológiai tervezése során segíti az egyes árucikkek adott szempontok szerinti megfelelő tárolási lokációkhoz való hozzárendelését, amely különösen nagy cikkelem-

számok esetén lehet hasznos, hiszen ekkor igen nehézkes és lassú lenne intuitív módszerrel meghatározni az árucikkek optimális lokációit.

A dolgozatom egy további célkitűzése, hogy a raktári operációs tevékenységek időszükségletének meghatározására egy segédalkalmazást fejlesszek ki. Az anyagmozgatási időszükséglet az egyik legfontosabb jellemzője a raktári folyamatoknak. Az operációs folyamatok időszükségletének pontos ismerete elengedhetetlen: segítségével felmérhetők az anyagáramlási folyamatok hatékonysága, meghatározhatók a szűk keresztmetszetek, ezzel alapot szolgáltatva a reális jövőbeni fejlesztési célkitűzéseknek. Az alkalmazás fejlesztésével az időszükséglet meghatározását szeretném megkönnyíteni, kényelmesebbé és hatékonyabbá tenni.

1. A raktári tárolóterek technológiai tervezésének áttekintése

1.1. A raktározás jelentősége napjainkban

Napjaink felgyorsult világában a piaci verseny egyre fokozódik. A termékek ára, amelyet a különböző vállalatok kínálnak, döntően meghatározó a vevő számára, így törekedni kell a lehető legalacsonyabb költségek elérésére. A termékkel kapcsolatos költségeket az ellátási láncon történő végigfutás során sok paraméter befolyásolja, a termék fogyasztói árának jelentős hányadát a logisztikával kapcsolatos, a logisztikához köthető folyamatok korszerűsége és ezzel együtt hatékonysága döntően meghatározza. A változó (logisztikai) tényezők bonyolult szervezési feladat elé állítják az ellátási lánc szereplőit, azonban ezzel együtt számtalan (optimalizálási, kreatív feladat-megoldási, korszerűsítési, hatékonyságnövelő illetve veszteségcsökkentő) lehetőséget is rejtenek magukban.

A logisztikai szakemberek egyik legnagyobb kihívása, hogy az eltérő időben jelentkező vevői igényeket folyamatosan ki tudják szolgálni (ne jelentkezzen hiány, azaz ne kelljen visszautasítani a vevő igényét, hiszen ez jelenti a legnagyobb bevételi forrást). A terméket előállító vállalatok azonban – elsősorban a technológiai kötöttségek és sajátosságok miatt – kevésbé tudnak rugalmasak lenni, kevésbé tudnak igazodni az igények ingadozásához (gyártó vállalat esetén jó példa erre egy öntőde, de rugalmasabb gyártó esetén gondot okozhat a nagy távolságra történő eljuttatás is, amennyiben globális piaci érdekeltségek jellemzik a vállalatot). A rugalmasságot azonban a vevő kíméletlenül megköveteli, így a legtöbb ellátási lánc számos raktár közbeiktatásával próbálja biztosítani az egyenletes ellátási színvonalat.

Raktárak létesítése során fontos tisztában lenni azzal, hogy az hosszú- és rövidtávon milyen költségekkel fog járni. A raktározás, raktári folyamatok kapcsán az alábbi (legfontosabb) költségek merülnek fel:

- *beruházási költség*, amely többek között a raktár mérete és a raktári folyamatok végrehajtásához szükséges gép-, eszköz-, berendezésigény függvénye,
- *üzemeltetési költségek*, amely mértéke szintén az előbbi paraméterektől (is) függ.

A megfelelő raktár kialakításához ezért elengedhetetlen, hogy tisztában legyünk a következő tényezőkkel:

- mekkora *mennyiséget* szeretnénk tárolni,
- mekkora a tárolandó *szortiment*,
- a szortiment egyes termékfajtái fizikai jellemzőit tekintve mennyire *homogének*, illetve milyen speciális tulajdonságokkal rendelkeznek a tárolás szempontjából,

Az előbbi paraméterek egyértelműen meghatározzák, hogy mekkora raktárra van szükségünk, azon belül pedig milyen anyagmozgató gépeket, eszközöket, berendezéseket, valamint tárolási módot válasszunk. A raktár ezen paraméterei kvázi-statisztikus jellemzők, elsősorban a beruházási költségeket határozzák meg, a beruházás utáni változtatásuk nehézkes, így célszerű alaposan megfontolni, hogy valóban szükségesek-e.

A beruházási (és a későbbi üzemeltetési) költségek jelentős mértékben csökkenthetők, ha az implementálandó raktári folyamatokat és a vevői igények ingadozásának intenzitását már a tervezés korai szakaszaiban mélységében megismerjük, elemezzük és megértjük. A helyes feltárás eredményeképpen a raktári folyamatok hatékonysága és a raktár kapacitás kihasználtsága maximumhoz, az említett költségek pedig a minimumhoz közelíthetők.

A raktári rendszert alapvetően három – logikailag és területileg általában jól lehatárolható – részre bonthatjuk: a tárolótérre, az áruelőkészítő térre, valamint az egyéb operációs folyamatok megvalósítását biztosító elemekre (rakodófrontok, irodák, karbantartó helyiségek). A raktár e három részterülete meglehetősen sok és diverz folyamatot foglal magába, így a dolgozat keretei között kizárólag a tárolótérhez kapcsolható folyamatokkal foglalkozunk.

1.2. A raktári tárolótér előtervezése

A raktárak technológiai tervezési folyamatának három fázisa az előzetes-, a részletes-, illetve a kiviteli tervezés. Ennek egy igen lényeges szakasza a tárolótér tervezése, amelynek tekintetében is erre a három fázisra osztható a folyamat. Az *előtervezés* azt a célt szolgálja, hogy megalapozza a részletes tervezés folyamatát, illetve a legfontosabb szempontok figyelembevételével a belső elrendezés tekintetében az alapkoncepció világossá váljon. Az előzetes tervezés során több változatot érdemes létrehozni, hiszen

egyres megoldások bizonyos szempontok szerint jobbak, míg mások szerint gyengébbek lehetnek, mint a többi. A gyakorlati tapasztalatok azt mutatják, hogy az ideális megoldás megtalálásához célszerű ebben a fázisban minél több (egymástól lényegesen különböző) lehetséges változatot megtalálni [2].

A raktári tárolótér előtervezés megkönnyítésére már léteznek különböző megoldások, amelyek az előtervezési fázist számítástechnikai eszközökkel támogatják (pl. *SATAK* és *rAcKer*). E tárolótéri előtervezést segítő alkalmazások folyamatosan fejlődnek, funkcióspektrumuk folyamatosan bővül, amely eredményeképpen az előtervezési és részletes tervezési szakaszok határai elmosódnak.

1.3. A tárolótéri áruelrendezés kérdése

A tárolótéri előtervezés és részletes tervezés során a tárolótéri áruelrendezés kérdése – kifejezetten, ha a komissiózó téri áruelrendezési problémák egyre tágabb területére gondolunk – méltánytalanul elhanyagolt, gyakran csak a részletes vagy kiviteli tervezés utolsó szakaszaiban történnek elenyésző mértékű törekvések az optimális áruelrendezés tudatos (tervezői szintű) kialakítására.

A tárolótéri áruelrendezés kérdésére nagyobb hangsúlyt kell fektetni, a raktári forgalom növekedésével a tárolótéri folyamatok precíz meghatározása egyre fontosabbá válik, kiemelten azok a folyamatok, amelyek az tárolótéri árucikkek mozgásával kapcsolatosak (be-, ki- és áttárolási folyamatok), hiszen ezek jelentősen befolyásolják egy raktári rendszer működési költségét.

2. A genetikus algoritmus, mint lágy számítás

Az ember úgy ötven éve végre felfedezte saját zsenialitását és elkezdte annak egyre hatékonyabb gyakorlati felhasználását az élet szinte minden területén. E felismerés a közvetlen vagy közvetett alapozója a *genetikus algoritmusoknak* [5].

A kemény számításban, a klasszikus számításban az elsődleges szempontok, kívánalmak a pontosság és a szigorúság. Zadeh¹ megmutatta, hogy a konvencionális rendszertechnika a nemlineáris, bonyolult rendszerek vizsgálatára alkalmatlan. Összeférhetetlenségi elve szerint: „Ahogyan egy rendszer bonyolultsága nő, úgy csökken a képességünk arra, hogy a viselkedésről precíz és szignifikáns kijelentéseket tegyünk, míg csak elérünk egy küszöböt, amelyen túl a precizitás és a szignifikancia szinte egymást kizáró jellemvonássá válnak.” [5]

A lágy számítás alapgondolata: visszatérés a közelítő, de rendkívül hatékony, rugalmas és gyors emberi gondolkodáshoz, így az emberi következtetést, gondolkodást használjuk mintamodellként. Azáltal, hogy kihasználjuk a pontatlansággal és a határozatlansággal szembeni toleranciát, megalkuvást, azt a figyelemre méltó képességet nyerjük, hogy ebben a bizonytalan és pontatlan környezetben is racionális döntéseket tudunk hozni. Ezzel a vizsgált rendszer jobban közelítő, egyszerűbb, robusztusabb modelljét tudjuk kifejleszteni alacsonyabb áron, a kommunikáció és a számítási idő rövidítésével, amelynek eredményeképpen a közelítően megfogalmazott problémák approximatív megoldását nyerjük [5].

Fontos megjegyezni, hogy a lágy számítás ezt a pontatlan, bizonytalan, közelítő gondolatmenetre épülő „modellt” szigorú matematikai ruhába öltözteti. Ez a matematikai eljárás egybefoglalja a klasszikus analitikai és köznapi nyelvi modellezést. Ezáltal az egyedüli olyan változat, amelyik az informatika két alapforrását, nevezetesen az érzékelők és a mérőeszközök adatait, valamint az emberi tapasztalati tudást egyaránt felhasználni képes. [5]

¹ **Prof. Dr. Lofti A. Zadeh:** matematikus, villamosmérnök és mérnök informatikus, a mesterséges intelligencia kutatója, a kaliforniai egyetemen (Berkeley) a mérnöki informatika Professzor Emeritusa.

2.1. A genetikus algoritmus, mint optimum-kereső eljárás

A genetikus algoritmusok, mai felfogásban, egy szélesebb, összefoglaló tudományterületnek, az evolúciós számításnak (Evolutionary Computation, EC) részét képezik. Az összefoglalt részterületek közös elnevezését csak 1991-ben „találták fel” abból a célból, hogy egyesítsék azoknak a kutatóknak a munkásságát, akik az evolúció szimulálásának különböző megközelítését vizsgálták [5].

A név így olyan algoritmusokat foglal magába, amelyek egyedekből álló népesség – populáció – fejlődésére alapozottak problémák megoldása céljából. Az evolúció a leg fittebb túlélésének nagyhatású elve, ami bizonyos természeti alapjelenségeket modellez, így a genetikus öröklődést, és a darwini harcot a túlélésért [5].

E terület ma leghasználatosabb három fő változata az alábbi:

- **a genetikus algoritmusok,**
- az evolúciós stratégiák,
- az evolúciós programozás [5].

A genetikus algoritmusok kereső eljárások, amelyek alapja a természetes kiválasztás és az öröklés mechanizmusa. Az algoritmus alapja egy rögzített számú népesség, populáció, amelynek tagjai a kromoszómák. A populáció minden tagja, mindegyik kromoszóma a feladat egy lehetséges megoldása, így az algoritmus szélesen kutat a kereső térben [5].

A kromoszóma lehet egy bitfüzér, a gének füzére, ahol minden bit – vagy részcsoport – egy tulajdonságot jellemez. Ha 1 áll ott, akkor a kromoszóma rendelkezik az adott tulajdonsággal, ha 0, akkor nem [5]. Bonyolultabb esetben (permutációs genetikus algoritmusok esetén) a kromoszóma génjei nem bitek, hanem például valós számú értékek. Ebben az esetben a kromoszóma tulajdonságát az jellemzi, hogy ezek a gének milyen sorrendben helyezkednek el egymáshoz képest, milyen az egymáshoz való viszonyuk (innen származik a „permutációs” elnevezés).

Mindegyik kromoszómához (azaz a népesség egy tagjához) hozzárendelnek egy „jósági”, fitness értéket, ami annak a jellemzője, hogy az milyen mértékben közelíti a kitűzött feladatot, az adott kritérium függvény értékét [5].

Az algoritmus rendkívül hatásos, bár alapalakjában csak három nagyon egyszerű alpművelettel

- a reprodukcióval (vagy szelekcióval),
- a keresztezéssel és
- a mutációval dolgozik [5].

A reprodukció a legjobb fitness értékű kromoszómák kiválogatása és átvitele a következő generációba, a keresztezés két kromoszóma egy részének cseréje, a mutáció pedig egy adott kromoszóma néhány tulajdonságának megváltoztatása.

A genetikus algoritmusok (GA) kereső rendszerek, amelyek alapja a természetes kiválasztás és az öröklés mechanizmusa. Azok robusztusak és a feladatról csak kevés információt igényelnek. Munkamódjuk rendkívül eltérő jellegű a legtöbb tradicionális optimalizáló technikáétól. A genetikus algoritmusok a konvencionális eljárásoktól főleg a következő négy pontban térnek el [5]:

1. A legdöntőbb különbség, hogy a genetikus algoritmusok nem egyetlen megoldással, hanem megoldások egy populációjával (halmazával) dolgoznak, tehát nem egy egyedülálló pontot, hegycsúcsot (vagy völgyet) keresnek, nem egy optimális helyet, hanem sok csúcsot párhuzamosan, a pontok egy populációját, a megoldások egy halmazát. A fitness szempontjából ezek mindegyike lehet jó megoldás, de nem egyformán jó. A műveletek révén információkat cserélnek a csúcsok, vagy völgyek között, így csökkentve a helyi minimum csapdájának lehetőségét és jobban valószínűsítve a globális megoldást.
2. A genetikus algoritmusok a változók kódolásával (a kromoszómákkal) dolgoznak, nem a változókkal magukkal. Így az optimalizálás során nem az egyes paramétereket próbálják megtalálni, hanem az egész paraméterkészlettel dolgoznak egyszerre. (További előny, hogy diszkrétte alakítja a keresőteret még akkor is, ha a függvény folytonos.)
3. Úgynevezett „vakkereséssel” keresnek, azaz a végeredményt leíró függvényen kívül se deriváltat, se más segédinformációt nem igényelnek. Csak a kritériumfüggvény egyes részeit használják, de arról semmilyen jellemvonást sem tételeznek fel.

4. További jellegzetes eltérés, hogy a GA-k keresésük irányítására valószínűségi szabályokat, műveleteket használnak.

A legtöbb tradicionális módszernél alapprobléma, hogy rögzített átviteli szabályok szolgálnak arra, hogy egyik megoldásról áttérjenek a másikra. Ez vezet arra, hogy az eljárások általában csak a problémák speciális osztályaira alkalmazhatók, így ezek a módszerek nem robusztusak és nem jók a probléma széles körére. A robusztus GA-k ezzel szemben induláskor tetszőleges irányt választhatnak.

A valószínűség használata nem jelenti azt, hogy az eljárás egyszerű véletlenszerű keresés. A GA-k a véletlenszerű választást szerszámként használják, amely őket olyan régiók felé vezeti a kereső térben, ahol valószínű javulás áll be.

2.2. Egy általános genetikusan működése

A populáció egyes kromoszómái eltérő sikerrel oldják meg a feladatot, azaz a probléma szempontjából eltérő képességűek. A genetikusan algoritmustól azt várjuk, hogy a természetes szelekció mintájára egy látszólag spontán fejlődés során egy adott populációból, olyan egymást váltó újabb generációk jöjjenek létre, amelyekben a jó megoldást jellemző egyedek egyre nagyobb számban lesznek és a gyengék egyre ritkulnak [5].

A GA-kban, ahol mindegyik kromoszóma egy megoldást jellemez, annak „jóságát” egy fitnessz érték jellemzi. Maximalizálási feladatoknál a fitnessz rendszerint a feladatra felírt célfüggvény értékével egyezik meg. Minimalizálási problémák esetében ez gyakran a célfüggvény értékének reciprokát, vagy a célfüggvénytől való eltérését (különbségét) jelenti [5]. Tehát látható, hogy a fitnessz függvény egy matematikailag egzakt módon leírható kifejezés.

A genetikusan algoritmus működését az 1. ábra mutatja. A GA felparaméterezését követően (lásd később) első lépésben egy kiinduló populációt generálunk, amelyet célszerű egy véletlen halmazból kiválasztani. Ez a genetikusan algoritmus inicializációs lépése. Ezután olyan műveleteket kell alkalmazni (reprodukción, keresztezés, mutáción), amelyek ebből a kezdeti populációból egymást követő populációkat – generációkat –

hoznak létre, és pedig olyanokat, amelyek – remélhetőleg – javuló fitness értéket mutatnak [5]. A következőekben a genetikus algoritmus alapműveleteit mutatom be.

A reprodukción, vagy szelekción, amely rendszerint az első művelet, az egyes kromoszómákat „jóságuk”, fitnessségük mértékében válogatja ki, és másolja át a következő generációba, amelyet a keresztezés párosodási készletének lehet tekinteni. Az átlag feletti „jóságú” kromoszómákból több kópia kerül át, míg a gyengék sorsa az eliminálás, kihalás [5].

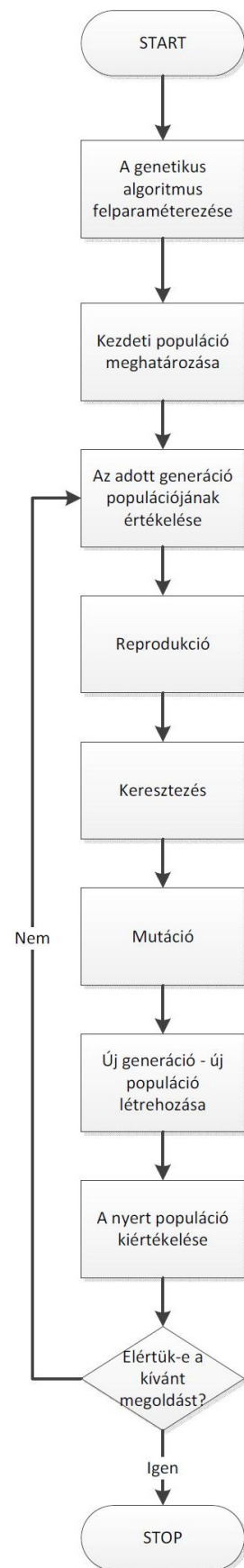
A reprodukció során egy kromoszóma a következő generáció részeként is megjelenik, vagyis az általa létesített megoldása a következő generáció megoldás-halmazának is része marad, azaz reprodukálódik [5].

Egy adott kromoszóma esetén a reprodukció, vagyis a túlélés valószínűsége annak „jóságával” van kapcsolatban. Minél nagyobb a kromoszóma „jósága” a populációhoz tartozó kromoszómák átlagos „jóságához” viszonyítva, annál nagyobb számban (eséllyel) fog reprodukálódni. A reprodukció ezen fajtáját arányos kiválasztási mechanizmusnak nevezik [5].

A fitness függvény valamilyen mértéke annak a „hasznosságnak”, vagy „jóságnak”, amit maximalizálni kívánunk.

A legismertebb és talán legegyszerűbb reprodukciós eljárás, a rulett-kerekes kiválasztás, amelynek részletezésére a 3.4.1. fejezetben kerül sor.

A keresztezés. A keresztezés során a reprodukció alkalmával kijelölt kromoszómákból kiválasztunk (legalább) kettőt, amelyek alapján a művelet végrehajtásra kerül: ezeket a kromoszómákat szülőnek nevezzük (parent). A szülő kromoszómákból ezután új kromoszómákat állítunk elő egy



1. ábra:
Egy GA működése

alkalmasan választott keresztezési operátorral (2. ábra). A keresztezés végrehajtása eredményeképpen előálló utód kromoszómák (child) várhatóan a reprodukció során kiválasztott legjobb szülői tulajdonságokat öröklik. Fontos megjegyezni, hogy az utódokkal a populáció mérete nem változik, azok a reprodukció során – az előző populáció – leggyengébbnek ítélt kromoszómáinak helyét veszik át.

A keresztezés fő célja, hogy paraméter terét felkutassa, illetve a keresés olyan módon történjék, hogy a szülőkből felhalmozódott információ maximálisan megőrződjék.

A mutáció. Míg a genetikus algoritmusok keresési aspektusáért a keresztezési művelet a fő felelős, és az első két operátor célja, hogy kihasználják és javítsák a legjobb kromoszómák felépítését a kurrens populációban, a mutáció célja, hogy változatossá tegye a keresést, és új kromoszómákat vigyen be a populációba, abból a célból, hogy teljesen felkutassa a kereső teret.

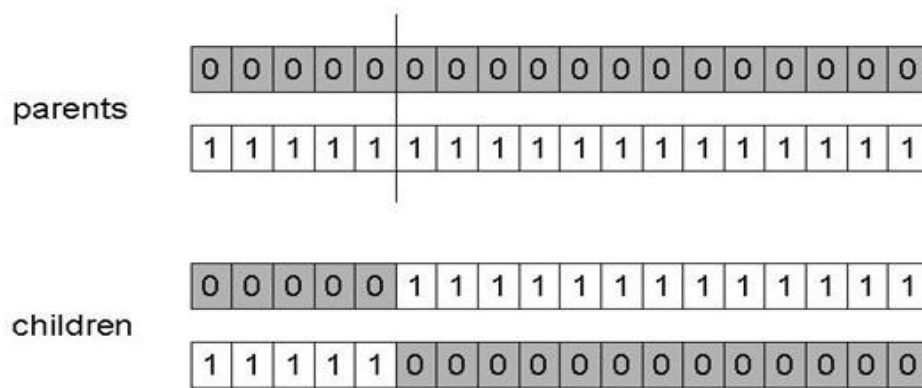
A mutáció a populáció változatosságának fenntartásához szükséges. Az új kromoszóma létesítése rendszerint a teljes keresési tér lehetséges megoldás-változatainak száma és a populáció kromoszómáinak száma közötti nagy különbségben rejlik.

A genetikus algoritmusban a mutáció határozottan másodlagos szerepet játszik. A szükségességét mégis igazolja, hogy a reprodukciós és keresztezési művelet néha túlzottan elronthatnak valamely potenciálisan hasznos genetikus anyagot (genetikus sodródás). A mutációs operátor véd az ilyen visszafordíthatatlan veszteség ellen, mivel használatával teljesen új tulajdonságú kromoszómák jöhetnek létre, ezzel megakadályozva a további fejlődésre képtelen uniform populáció létrejöttét. Az új kromoszómák ezenkívül a megoldási tér olyan területeit is feltérképezhetik, amelyekre az eddigi populációk nem terjedtek ki.

Az előbbiekkal szemben azonban a túlzott mutáció alkalmazása a populáció leg fittebb kromoszómáit is elpusztíthatják, ami a konvergenciát lassítja, sőt meg is akadályozhatja.

A keresztezési és mutációs operátorokra a szakirodalomban többféle eljárás is fellelhető függően attól, hogy milyen típusú problémával állunk szemben. Ezért, a dolgozat terjedelmi korlátai miatt most csak egy-egy módszert mutatok be ezen alapl műveletek szemléltetése és megértése céljából:

Az egy pontos keresztezési eljárás [6][7]



2. ábra: Az egyponthos keresztezési eljárás alapelve [7]

A kiválasztott két szülő kromoszómáját egy véletlenszerűen kiválasztott ponton kettéválasztjuk, majd a két szülő egyes szakaszainak keveredésével hozzuk létre a két új kromoszómát (2. ábra). Ez az eljárás elsősorban az egyszerű, bitsoros genetikai algoritmusok esetén használható.

Mutáció invertálással [6][7]

Az invertáláson alapuló mutáció során a létrejött utódon véletlenszerűen kijelölünk két pontot, majd ezen két pont közötti szakaszon megfordítjuk (invertáljuk) az egyes gének sorrendjét (3. ábra). Ezen mutációs eljárás jól alkalmazható a permutációs genetikai algoritmusok esetén.



3. ábra: Az invertáláson alapuló mutáció elve [7]

Az előbbieket alapján tehát összefoglalóan megállapítható, hogy a bemutatott három művelet szekvenciálisan és lineárisan kapcsolódik egymáshoz. A reprodukciós művelet kiválogatja a legjobb kromoszómákat, amelyek egyes részeit ezután a keresztezési operátor úgy kombinál, hogy remélhetően még jobb tulajdonságokkal rendelkező kromoszómákat kapjunk. A mutáció kromoszómákat lokálisan módosít itt is abban a reményben, hogy egy jobb kromoszómát alkosson. Bár e kívánalmak egyike sem garantálható, az várható, hogy a kialakított rossz kromoszómákat a reprodukció eliminálja, a jókat pedig hangsúlyozza, sokszorozza.

Fontos megjegyezni, hogy bár a műveletek véletlenszerűek, a kromoszómák kiválasztása, a reprodukció már nem, tehát az algoritmus az irányítottság és a véletlenszerűség egy kombinációját használva működik olyan módon, hogy a korábbi

próbálkozások legjobb megoldásaiból építenek jobb, új változatokat. A reprodukció és a keresztezés révén a GA erőteljesen kihasználja a kromoszómák információgazdagságát, hiszen az egyes kromoszómák a feladat megoldása szempontjából eltérő elképzeléseket tartalmaznak arról, hogy mi a fontos, vagy döntő.

3. Az optimális árueლrendezés kialakításához fejlesztett alkalmazás

Az előző fejezetekben megismerhettük a raktári tárolótéren történő optimális árueლrendezés fontosságát (probléma), valamint a genetikus algoritmust (eszköz). A dolgozatom fő célkitűzése, hogy a vázolt logisztikai problémára a választott eszköz (egy korszerű optimum-kereső eljárás) segítségével megoldást találjak.

Az algoritmus implementációját MS Excel VBA felületén valósítottam meg. A fejlesztői környezet kiválasztásakor törekedtem arra, hogy a fejlesztendő alkalmazást egy olyan felületen hozzam létre, amely minél szélesebb körben elérhető. A Microsoft Office programcsaládja napjainkban a vállalatok döntő többségénél rendelkezésre áll.

3.1. Az alkalmazás célja

A fejlesztett alkalmazás célja, hogy a genetikus algoritmus – mint lágy számítás eljárás – eszközszeresét felhasználva meghatározzunk egy olyan *optimális árueლrendezés tárolótéri layoutját*, amely a felkeresési gyakoriságok és felkeresési idők összehangolásán túl, megakadályozza az irracionális árueლosztást is, ezzel elérve, hogy a raktári tárolótéri anyagmozgatási teljesítményszükséglet minimális legyen amellet, hogy a szervezési-operációs feladatok ne legyenek irracionálisan nagyok.

3.2. A feladat lehatárolása

A logisztikai rendszereket tekintve elmondható, hogy az egyes szegmensekben számtalan variáció született már egyes problémák kezelésére. A felmerülő (logisztikai) igények kielégítésére szinte minden esetben egyedi megoldások születnek. Általánosságban véve ezeket ez egyedi logisztikai megoldásokat nehézkes egységesen kezelni, ezért szükséges tudni, hogy a fejlesztett alkalmazással támogatható rendszer milyen ismérvekkel rendelkezik.

A raktári tárolótéri rendszerek eltérő jellemzőkkel bírhatnak, így az algoritmus számítási módszertanának kialakítása során az alábbi korlátozási feltételeket határoztam meg:

- a *raktár hagyományos* kialakítású,
- a raktári áruelőkészítő tér és tárolótér *fej elrendezésben* kapcsolódnak egymáshoz,
- a tárolótéri anyagmozgatási feladatokat *nagyemelésű gépi targoncák* végzik,
- a tárolótéri be- és kitárolási folyamatok kizárólag *egyszerű ciklusokból* állnak (nincs kombinált ciklus),
- a tárolótéri tárolást *soros állványos tárolási rendszerrel* valósítják meg,
- a tárolótéri folyosókon *kétirányú targoncaközlekedést* engednek meg,
- a mozgatási egységek *EUR rakodólapos egységrakományok*,
- az árukészlet *polisztruktúrájú*, az áruféleségek *inhomogének*,
- a *kommissiózás a tárolótéren kívül* történik.

3.3. A szükséges kiindulási paraméterek

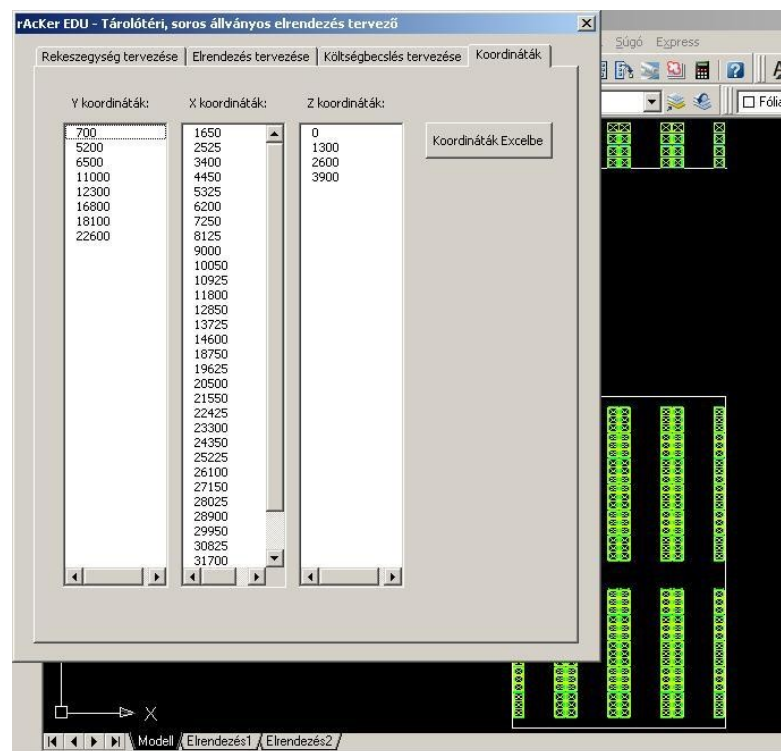
A tárolótéren történő optimális áruelrendezés kialakításához ismerni kell az egyes árucikkek forgási sebesség értékét, illetve a tárolótér geometriáját. A felhasználónak ezért a következő paramétereket szükséges megadnia:

- A tárolótér technológiai tervezése során kialakított elrendezés geometriai adottságait, különös tekintettel az egyes *tárolótéri lokációk térben való elhelyezkedésére* – az egyes lokációk (X, Y, Z) koordinátái egy közös viszonyítási ponttól (bázisponttól) számolva:
 - bázisponttól számított x koordináta értéke: $[X] = \text{mm}$,
 - bázisponttól számított y koordináta értéke: $[Y] = \text{mm}$,
 - bázisponttól számított z koordináta értéke: $[Z] = \text{mm}$,
- Az árucikkekre vonatkozó néhány fontosabb készletezési paramétert:
 - az árucikk *forgási sebességét*: $[v] = \text{nap}$,
 - az egyes árucikkekből tárolni kívánt *mennyiséget*: $[Q] = \text{ER}$ (egységrakomány rövidítése: ER).

A tárolótéri lokációk elhelyezkedését egy célszerűen kiválasztott *bázispontból* szükséges megadni, amely alapján az egyes lokációk (x, y, z koordinátái) egyértelműen meghatározhatók a térben. Ezen koordináták meghatározására kiválóan alkalmas a

rAcKer tárolótér tervező program [2]. A rAcKer lehetőséget biztosít, hogy a tárolótéri előtervezés során elkészült tárolótéri előtervek geometriai paramétereit az előzőekben megfogalmazott kívánalmak szerint megkapjuk. Az alkalmazás exportálási funkcióját (4. ábra), illetve a kinyert adathalmaz egy részletét (5. ábra) az alábbiakban láthatjuk.

Az árucikkek készletezési paramétereit közül a legkönnyebben mérhetők közé tartozik a forgási sebesség és a tárolandó mennyiség, amelyek adott esetben pl. egy vállalatirányítási rendszerből könnyedén kinyerhetők. Az árucikkek ezen paramétereinek szükséges megadási módját a 6. ábra mutatja.



4. ábra: A tárolótéri elrendezés adatainak exportálása a rAcKer-ből [2]

Lokáció azonosító	X koordináta	Z koordináta	Y koordináta
LOK0001	1750	0	700
LOK0002	2625	0	700
LOK0003	3500	0	700
LOK0004	4550	0	700
LOK0005	5425	0	700
LOK0006	6300	0	700

5. ábra: A tárolótéri lokációk adatai Excelben

Árucikkek készletezési paraméterei		
Árucikk azonosító	Forgási sebesség [nap]	Tárolási mennyiség [ER]
1	4,3256878	53
2	1,65496	87
3	3,0256	140
4	0,876	320
5	9.6311	65

6. ábra: A készletezési adatok megjelenése Excelben

3.4. Inicializáció

Az alkalmazás inicializációs része magába foglalja mindazon lépéseket, amellyel a kiinduló adatokat az elvégzendő számításokhoz, illetve a genetikus algoritmus futtatásához alkalmas alakra hozzuk.

3.4.1. A tárolótéri lokációk felkeresési idejének meghatározása

A számítások elvégzése során elengedhetetlen, hogy mindig pontosan tudjuk, minde egyes lokáció milyen elérési paramétereit és az bármikor meghívható legyen. Ezért egy indexelési szabályt szükséges létrehozni, amely alapján a lokációkat sorba lehet rendezni:

1. A bázispontból indulva először a mellékfolyosón haladunk végéig (X koordináta szerint).
2. A mellékfolyosó bejárás után, a következő magassági szinten folytatjuk a folyamatot (Z koordináta szerint).
3. A teljes mellékfolyosó bejárása után, a főfolyosó mentén a következő mellékfolyosóra lépünk át. (Y koordináta szerint).

A rAcKer fel lett készítve arra, hogy minden egyes tárolótéri lokáció (x, y, z) koordinátáit legenerálja, és azt egy adott struktúra szerint egy Excel munkafüzetbe exportálja [2].

A lokációk felkeresési (elérési) időszükségletét *AIM módszer* segítségével határoztuk meg, az anyagmozgatási feladatra *villamos gépi emelőtargoncát* választottunk. Az anyagmozgatási feladatok időszükséglete az alábbi minta szerint lett kiszámítva:

- helyváltoztatás rakottan (x és y koordinátatengelyek mentén),
- rakományelhelyezés a tárolótéren (z koordináta szerinti magasságon),
- indítás és megállás,
- irányváltás,
- helyváltoztatás üresen (x és y koordinátatengelyek mentén),
- rakományfelvétel talajszintről az áruelőkészítő téren (z koordináta nulla),
- indítás és megállás,
- irányváltás.

(Az előbbi egy általános betárolási folyamat leírása. Kitárolási folyamat esetén a rakományfelvétel a tárolótérről, a rakományelhelyezés az áruelőkészítő téren történik. A pótlékolási tényezőt a számítások során 0,2-nek választottuk.)

Az egyes felkeresési idők számításánál fontos, hogy minden esetben azonos viszonyítási ponttól legyenek értelmezve, hiszen csak így lehetséges az egyes eltérések meghatározása. A feladat lehatárolása során tett korlátozás – miszerint a raktár fej elrendezésű – különösen fontos a számítás ezen szakaszában.

Az inicializációs lépés során megadásra kerül a főfolyosó-szélesség értéke. A lokációk sorba rendezett X koordinátáinak vizsgálata során figyeljük, hogy mely szomszédos lokációk esetében nagyobb a koordinátakülönbség, mint a megadott főfolyosó-szélesség mérték. Ahol nagyobb, azt a program főfolyosónak tekinti. A tesztelés során ez az érték 2500 mm-re lett választva (1):

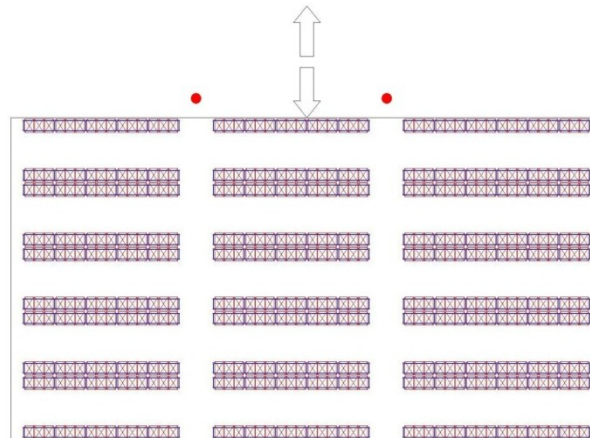
$$X_i - X_{i+1} \geq 2500 \text{ mm} \quad (1)$$

A fej elrendezés kötöttségének felhasználásával minden egyes főfolyosón történő tárolótéri belépési pontra meghatározható az összes lokáció elérési ideje, majd azzal a feltételezéssel élve, hogy a belépési pontokon a forgalom egyenletesen oszlik meg, meghatározható ezen értékek átlaga (2):

$$\frac{\sum_{j=1}^n t_{i0,j}}{n} = t_i \quad (2)$$

ahol t_i az i-edik lokáció átlagos elérési ideje [s], n a főfolyosók száma [db], $t_{i0,j}$ az i-edik lokáció j-edik főfolyosó belépési pontjától számított elérési idő [s].

A bázispontok egy elképzelését a 7. ábra mutatja.



7. ábra: A bázispontok illusztrációja a tárolótéri layouton (pirossal jelölve)

Az alkalmazás inicializációs fázisa során az összes lokáció (2) kifejezés szerint számított felkeresési ideje meghatározásra kerül, azaz $f(x,y,z) = t_i(x,y,z)$.

3.4.2. Az egyes árucikkekhez tartozó felkeresési gyakoriságok meghatározása

Az egyes lokációk felkeresési ideje mellett szükséges tudni, hogy az adott lokáció felkeresése átlagosan milyen gyakorisággal történik meg. Egy lokáció felkeresési gyakoriságát az adott lokáción elhelyezett árucikk forgási sebessége határozza meg.

Egy adott árucikk forgási sebessége azt mutatja meg, hogy az árucikkből tárolt teljes mennyiség mekkora idő alatt „fordul meg” a raktárban [4]. Tehát, egy forgási sebességnyi idő alatt az adott árucikk teljes mennyisége „kicserélődik” a tárolótéren, vagyis az árucikk teljes mennyiségét ennyi idő alatt egyszer betárolják, illetve egyszer kitárolják, így elmondható, hogy az árucikkkel kapcsolatban összesen $2Q$ alkalommal történik anyagmozgatási feladat v idő alatt (3):

$$\frac{2 \cdot Q_j}{v_j} \quad \frac{ER}{nap} \quad (3)$$

ahol Q_j a j -edik árucikkből tárolandó átlagos mennyiség, v_j pedig annak forgási sebessége.

A (3) kifejezés a fordulatszám fogalmával mutat hasonlóságot. A fordulatszám azt mutatja meg, hogy egy adott vizsgálati időszak alatt átlagosan hány készletezési ciklus történik, „átlagosan” hányszor fordul meg az „átlagos” raktárkészlet [4]. Tekintsük a vizsgálati időszakot 1 napnak, ezzel megkapjuk a napi felkeresések átlagos értékét,

amely megegyezik a (3) kifejezés az árucikk egy áruelemének átlagos napi felkeresési gyakoriságával (4):

$$p_j = \frac{1}{v_j} = \frac{Q_j}{v_j \cdot Q_j} \cdot \frac{1}{nap} \quad (4)$$

ahol Q_j a j-edik árucikkból tárolandó átlagos mennyiség, v_j annak forgási sebessége, p_j pedig az árucikk egy elemének fordulatszáma.

Az alkalmazás inicializációs fázisa során az összes árucikkhez (4) módon generált felkeresési gyakorisága meghatározásra kerül, azaz $f(v, Q) = p_i(v, Q)$.

3.4.3. Az áruelrendezést jellemző paraméterek

Az alkalmazás legfontosabb feladata, hogy a működés során az egyes árucikkelemeket az egyes tárolótéri lokációkhoz hozzárendelje. Ez technikailag azt jelenti, hogy az egyes lokációk elérési idejét, illetve az árucikkek forgási sebességét kell összevetni. Az átlagos felkeresési időt a teljes tárolótérre vonatkozóan az alábbi kifejezéssel írhatjuk le:

$$\sum_{j=1}^n \frac{2^{Q_j}}{v_j} t_i \rightarrow \text{MIN!} \quad (5)$$

ahol:

- t_i : az i-edik tárolási lokáció felkeresési ideje,
- Q_j : a j-edik árucikk tárolótéren történő tárolni kívánt mennyisége,
- v_j : a j-edik árucikk forgási sebessége,
- n : az összes árucikk-fajták száma.

Az egyes tárolótéri lokációkat a (4) szerint kifejezett gyakorisággal kell felkeresni, amely minden alkalommal egy t felkeresési idővel jár. Az (5) kifejezés így egy olyan árucikk szerint aggregált időértéket ad, amely a teljes tárolótér áruelrendezésének egy időjellemzője. Az (5) minimalizálásával, vagyis az összes felkeresési idő minimalizálásával érhető el a *legjobb idő szerinti áruelrendezés*.

Az áruelrendezés meghatározása során fontos szempont az egyes árucikkek, árucsoportok „együtt tartása”, vagyis a koherens kiosztás megvalósítása. A (6) kifejezés az áruelrendezés egy olyan jellemzője, amely azt mutatja meg, hogy az egyes árucikkek egyes elemei egymástól milyen messze helyezkednek el:

$$\sum_{j=1}^n \frac{1}{2} \sum_{k=1}^{Q_j} \sum_{i=1}^{Q_j} \Delta x_{i,k} + \Delta y_{i,k} + \Delta z_{i,k} \rightarrow \text{MIN!} \quad (6)$$

ahol:

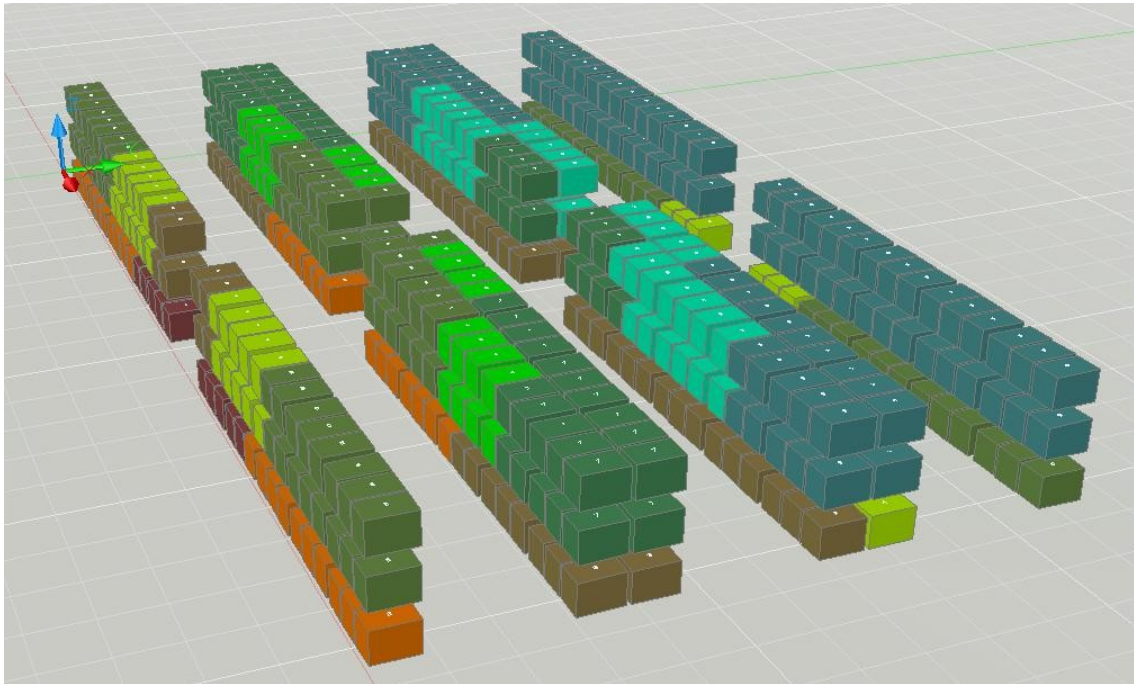
- $\Delta x_{i,k}$: az adott árucikk i-edik elemének k-adik elemétől vett „x” irányú távolsága, ami a mellékfolyosó mentén mért távolság,
- $\Delta y_{i,k}$: az adott árucikk i-edik elemének k-adik elemétől vett „y” irányú távolsága, ami a főfolyosó mentén mért távolság,
- $\Delta z_{i,k}$: az adott árucikk i-edik elemének k-adik elemétől vett „z” irányú távolsága, azaz a magasságkülönbség,
- Q_j : a j-edik árucikk tárolótéren történő tárolni kívánt mennyisége,
- n : az összes árucikk-fajták száma.

A Δx , Δy , Δz távolságkülönbségek minimalizálásával egyre inkább olyan elrendezés felé közelíthetünk, amelyben az egyes árucikkek áruelemei egyre kisebb távolságra vannak egymástól, azaz egyre inkább összefüggő mezőként foglalják el a helyüket az egyes tárolótéri lokációkon. Ezt a mutatót a teljes tárolótérre vonatkoztatva (tehát az összes árucikk ezen értékét együttesen vizsgálva) (6) egy alkalmas áruelrendezést jellemző paraméter.

3.5. Az első részeredmény: a legjobb elérési idők meghatározása

Az alkalmazás inicializálása és genetikusan algoritmus fázisa között egy jelentős eredmény már megjelenik. Ez a legjobb összes felkeresési idők által meghatározott áruelrendezés.

Az inicializációs fázisban meghatározásra kerültek az egyes lokációk felkeresési idői (3.4.1. fejezet), illetve az egyes árucikkekhez tartozó felkeresési gyakoriságok (3.4.2. fejezet). Határozzuk meg az áruelrendezést olyan módon, hogy a legkisebb felkeresési idővel rendelkező lokációk helyére a legnagyobb felkeresési gyakoriságú árucikkelemeket helyezzük, a hosszabb felkeresési idővel rendelkező lokációhoz, pedig a kisebb felkeresési gyakoriságú árucikk elemeket. Ezzel megvalósítható az (5) kifejezés minimalizálása. Az így elvégzett áruelrendezés egy (teszt adatsor alapján készített) illusztrációja a 8. ábrán látható.



8. ábra: A legjobb összes felkeresési idő szerinti változat

Ez az elrendezés az alapja az egész további futásnak, ahogyan az majd a következőkben is látható lesz, ehhez az elrendezéshez történik majd az egyes áruelrendezés-változatok hasonlítása. Az alkalmazás a futás ezen pontján meghatározza az (5) és (6) értékeket erre az elrendezésre.

3.6. A probléma és a genetikus algoritmus kapcsolata

A 2. fejezetben részletesen bemutatásra került a genetikus algoritmus – mint lágy számítási eljárás –, annak felépítése és általános működése.

De hogyan alkalmazható ez az eljárás az optimális áruelrendezés meghatározására?

Induljunk ki több – a meghatározott populáció méretétől függő –, teljesen véletlenszerű elrendezésből. Értékeljük az elrendezéseket az összes felkeresési idő és felkeresési gyakoriság összerendeltsége, valamint az egyes árucikkeken belüli elemek egymástól való távolsága alapján (fitness függvény). Az értékelések alapján rendezzük sorba az egyes áruelrendezési változatokat, majd válasszuk ki a legjobbakat, amelyek keresztezésével újabb változatokat hozunk létre. A keresztezéssel létrejövő változatok a korábban gyengébb fitnessű elrendezések helyét veszik át a populációban, így ezen gyengébb elrendezéseket egy várhatóan jobb fitness értékű variációra cseréljük,

amellyel létrejön a következő generáció. A generációk előrehaladtával így – azzal a feltételezéssel élünk, hogy – folyamatosan javuló populációkat kapunk, amelyeken belül pedig folyamatosan jobb fitnessű kromoszómák fognak szerepelni.

A genetikus sodródás és a lokális szélsőértékek elkerülése céljából véletlenszerűen mutációt alkalmazunk a keresztezés során, amellyel a korábbiaktól eltérő változatok keletkeznek, így az algoritmus új irányban tud tovább haladni.

A 2. fejezetben megismert GA alapfogalmakat az alábbiak szerint értelmezzük az adott probléma szemszögéből:

- Kromoszóma: egy adott *áruelerendezési elképzelés* (például a véletlenszerű áruelerendezés)
- Gén: a kromoszómát felépítő tulajdonságok, ami esetünkben *az egyes áruelemek adott helyen levő pozícióját jelenti* (például az egy tulajdonságnak tekinthető, hogy az egyik árucikk egyik árueleme az első mellékfolyosó első lokációjára lett kiosztva).
- Populáció: az *egyes áruelerendezések* (a teljes tárolótérre vonatkoztatott) különböző elképzelései.
- Generáció: *az áruelerendezések elképzeléseinek egyes különböző csoportjai*, amelyek kromoszómái az algoritmus előrehaladtával egyre javulnak.

3.6.1. A fitness függvény meghatározása

A genetikus algoritmus legfontosabb eleme a *fitness függvény*, amely ismeretében az algoritmus képes arra, hogy megkülönböztesse a jó és a kevésbé jó áruelerendezéseket: a fitness függvény méri az egyes elrendezések „jóságát”. Az alkalmazott fitness függvény a korábban már ismertetett (5) és (6) kifejezések egy összevont verziója (7):

$$\frac{\sum_{j=1}^n \frac{2}{v_j} \sum_{i=1}^{Q_j} t_i}{\sum_{j=1}^n \frac{2}{v_j} \sum_{i=1}^{Q_j} t_i} \cdot \frac{\sum_{j=1}^n \frac{1}{2} \sum_{k=1}^{Q_j} \sum_{i=1}^{Q_j} \Delta x_{i,k} + \Delta y_{i,k} + \Delta z_{i,k}}{\sum_{j=1}^n \frac{1}{2} \sum_{k=1}^{Q_j} \sum_{i=1}^{Q_j} \Delta x_{i,k} + \Delta y_{i,k} + \Delta z_{i,k}} \rightarrow \text{MAX!} \quad 7$$

$$Q_j > 0, \forall j = 1, 2, 3, \dots, n$$

$$\Delta x_{i,k} \geq 0, \forall i = 1, 2, 3, \dots, Q_j, \forall k = 1, 2, 3, \dots, Q_j$$

$$\begin{aligned}\Delta y_{i,k} &\geq 0, \forall i = 1, 2, 3, \dots Q_j, \forall k = 1, 2, 3, \dots Q_j \\ \Delta z_{i,k} &\geq 0, \forall i = 1, 2, 3, \dots Q_j, \forall k = 1, 2, 3, \dots Q_j \\ v_j &> 0, \forall j = 1, 2, 3, \dots n\end{aligned}$$

A fitness függvény két részből (két hányados szorzataként) tevődik össze, amely a két fontos optimalizálási szempontot képviseli és azokat egzakt matematikai nyelven írja le.

Az első hányados *a felkeresési idők és a felkeresési gyakoriságok összerendeltségét* írja le, ahol:

- t_i : az i -edik tárolási lokáció felkeresési ideje,
- Q_j : a j -edik árucikk tárolótéren történő tárolni kívánt mennyisége,
- v_j : az j -edik árucikk forgási sebessége,
- n : az összes árucikk-fajták száma.

Az első hányados számlálójában a 3.5 fejezetben részletezett legjobb összes felkeresési idő szerinti időmennyiség található, a fitness függvényben ezt tekintjük az összehasonlítás bázisának (ezt jelképezi a „0” jelölés). Ehhez az időhöz viszonyítjuk a vizsgálandó kromoszóma által képzett áruelrendezés esetén adódó összes felkeresési időt (a hányados nevezőjében). A hányados értéke az alábbiak szerint változik:

- ha az új kromoszóma által képzett elrendezés összes felkeresési ideje nagyobb (rosszabb), mint a viszonyítás alapja, akkor a hányados 1-nél kisebb értéket ad,
- ha az új kromoszóma által képzett elrendezés összes felkeresési ideje kisebb (jobb), mint a viszonyítás alap, akkor a hányados 1-nél nagyobb értéket ad.

Az előbbiek szerint az áruelrendezés felkeresési idő szerinti értéke annál jobb, minél magasabb értéket ad a hányados.

A fitness függvény másik hányadosa az *azonos cikkszámúval rendelkező áruelemek egymáshoz viszonyított távolságát* méri a teljes tárolótérre vonatkozóan. A jelölések értelmezése:

- $\Delta x_{i,k}$: az adott árucikk i -edik elemének k -edik elemétől vett „x” irányú távolsága, ami a mellékfolyosó mentén mért távolság,
- $\Delta y_{i,k}$: az adott árucikk i -edik elemének k -edik elemétől vett „y” irányú távolsága, ami a főfolyosó mentén mért távolság,

- $\Delta z_{i,k}$: az adott árucikk i -edik elemének k -adik elemétől vett „ z ” irányú távolsága, azaz a magasságkülönbség,
- Q_j : a j -edik árucikk tárolótéren történő tárolni kívánt mennyisége,
- n : az összes árucikk-fajták száma.

A hányadosban így a teljes tárolótérre vonatkozó (aggregált) összes távolságkülönbség található, amely az összes árucikken belüli elhelyezések távolságkülönbségét együttesen kezeli. A hányados számlálója ismételten a 3.5 fejezetben megismert legjobb összes felkeresési idő szerinti elrendezésre számított távolságkülönbség található, amely a viszonyítás bázisa („0” index). A hányados nevezője az aktuális kromoszóma (elrendezés-változat) esetén mért összes távolságkülönbség. A hányados tehát azt fejezi ki, hogy a bázis változathoz viszonyítva a vizsgált változatban mennyivel vannak közelebb az egyes árucikkek a tárolótéren történő elrendezésüket illetően. A hányados értéke az alábbiak szerint változik:

- ha az új kromoszóma által képzett elrendezés összes távolságkülönbsége nagyobb, mint a viszonyítás alapja – tehát az árucikkek kevésbé alkotnak összefüggő csoportokat a tárolótéren –, akkor a hányados 1-nél kisebb értéket ad,
- ha az új kromoszóma által képzett elrendezés összes távolságkülönbsége kisebb, mint a viszonyítás alapja – tehát az árucikkek összefüggőbb csoportokat képeznek a tárolótéren –, akkor a hányados 1-nél nagyobb értéket ad.

Az érthetőség kedvéért ismét ki kell emelni: az összefüggő csoportokat alkotó árucikkek esetén az árucikkhez tartozó áruelemek közötti távolságok várhatóan kisebbek, mint egy véletlen elrendezés esetén, amikor az árucikkek sztochasztikusan vannak szétszórva a teljes tárolótéren.

Az előbbieken részletezett két tulajdonság eltérő dimenziójú és eltérő nagyságrendű, ezért ha a kettőt egyszerre szeretnénk vizsgálni, akkor a (7) kifejezéssel ez könnyedén megtehető, hiszen így relatív értékekkel tudunk számolni. A két hányados közötti szorzás kifejezi, hogy nem elég az, hogy az elrendezés csak az egyik szempont szerint legyen megfelelő. A genetikus algoritmus futása során ezen két szempont szerint fognak „vetélkedni” egymással az elrendezések.

3.6.2. Az áruelrendezés technikai leképezése a genetikus algoritmushoz

Mielőtt a genetikus algoritmus bemutatására kerülne sor, fontos kitérni arra, hogy a genetikus algoritmus hogyan kezeli az információkat a problémával kapcsolatban, illetve melyek azok a paraméterek amelyekkel dolgozik.

Tekintsük egy véletlen-áruelrendezés részletének illusztrációját kiinduló állapotként (9. ábra). A tárolótér minden lokációja (9. ábra, második sor) egy felkeresési idővel rendelkezik (9. ábra, első sor), amelyek között a kapcsolat elválaszthatatlan. Az egyes tárolótéri lokációkhoz hozzárendeltük valamely árucikk egy elemét (9. ábra, harmadik sor), amelyhez pontosan meghatározott felkeresési gyakoriság tartozik (9. ábra, negyedik sor). Rögzítsük az árucikk ezen sorrendjét, amely *a teljes futás során ezután soha nem fog megváltozni*.

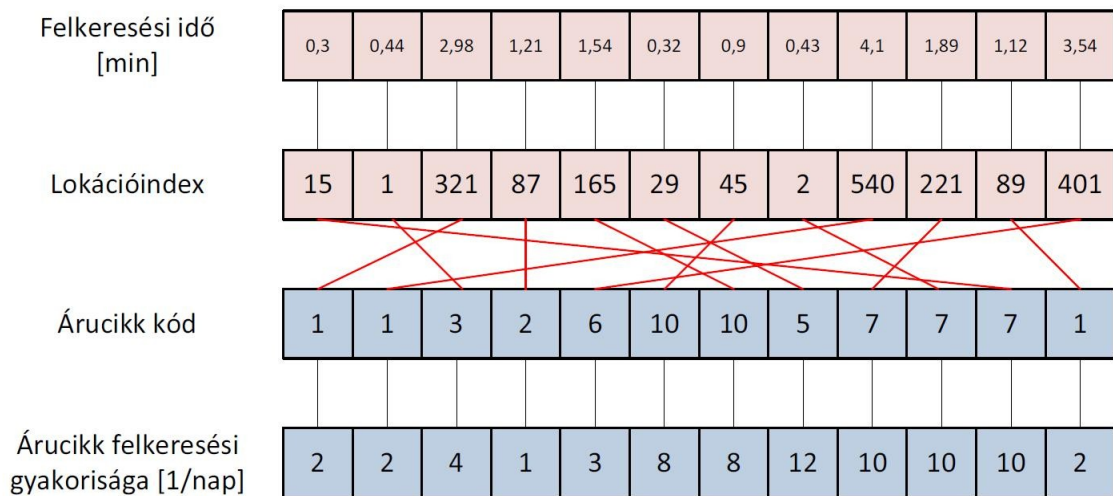
Felkeresési idő [min]	0,3	0,44	2,98	1,21	1,54	0,32	0,9	0,43	4,1	1,89	1,12	3,54
Lokációindex	15	1	321	87	165	29	45	2	540	221	89	401
Árucikk kód	1	1	3	2	6	10	10	5	7	7	7	1
Árucikk felkeresési gyakorisága [1/nap]	2	2	4	1	3	8	8	12	10	10	10	2

9. ábra: Véletlen elrendezés kiinduló állapota (részlet)

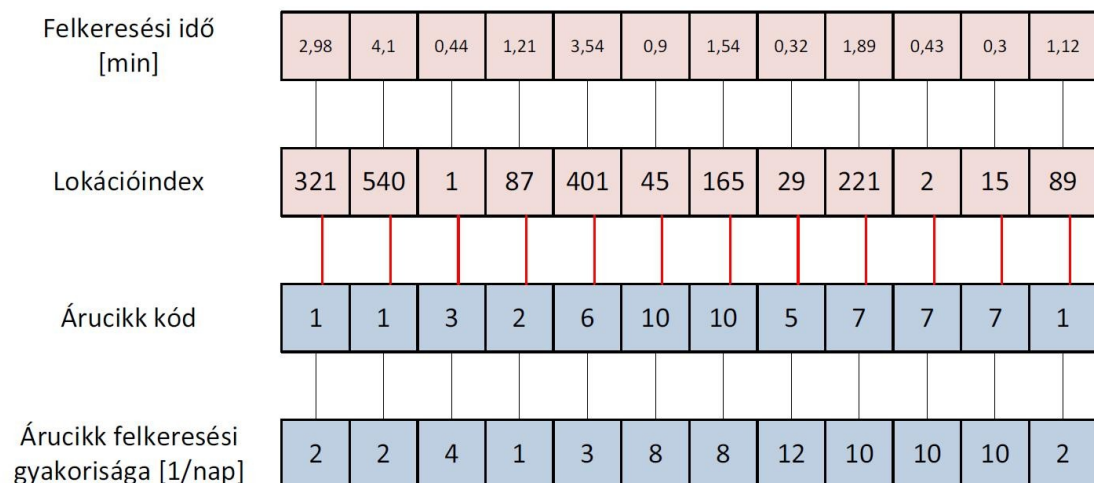
Egy új elrendezés létrehozásánál (10. ábra) az előbbi lokációkat hozzárendeljük egy új árucikkhez, amely során az egyes lokációk – technikailag – új helyre kerülnek (11. ábra).

A genetikus algoritmus futása során a – technikailag – rögzített árucikk kód sorhoz különböző variációban rendeljük hozzá az egyes lokációkat. Mivel az árucikk pozíciója kötött, ezért elegendő kizárólag a lokációk sorrendjével foglalkozni. Tehát a genetikus algoritmus egyes kromoszómái az egyes tárolótér lokációk különböző

sorrendje által meghatározott árue rendezést képviselik, a genetikus algoritmus futtatása során csak lokációk permutációját fogjuk változtatni.



10. ábra: Az új hozzárendelés kijelölése (részlet)



11. ábra: Az új hozzárendelés érvényesítése (részlet)

Ezzel a megoldással a problémát egy *permutációs genetikus algoritmushoz* hangoltuk.

Megjegyzés: Az egyes árucikk kódok annyiszor szerepelnek a sorban, amennyi a belőlük tárolandó mennyiség (egy lokációra csak egyetlen elem kerülhet), tehát az árucikk kód sor és a lokációindex sora egyenlő hosszúságú. Két eset állhat fent:

- Ha a tárolókapacitás megegyezik, vagy nagyobb, mint a tárolandó áruk összes mennyisége, akkor az árucikk kódok sorában soha nem találunk nullát.
- Ha a tárolókapacitás nagyobb, mint a tárolandó áruk mennyisége, akkor az árucikk kód sort kiegészítjük annyi nullával, hogy a lokációindex sor hosszával

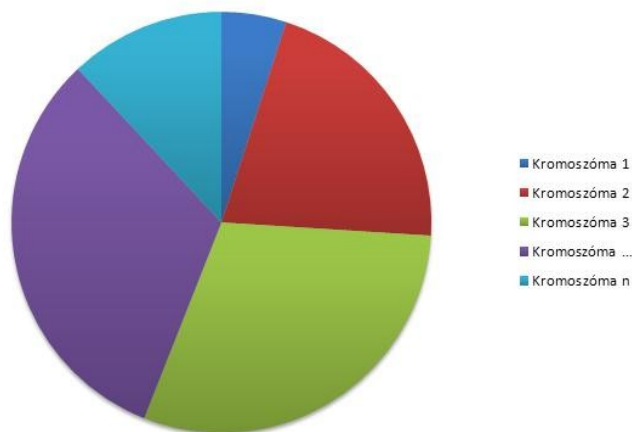
megegyezzen. (A nulla érték azt jelenti, hogy a hozzárendelt lokáción nem történik tárolás.

A következőkben ismertetésre kerülnek az implementált genetikus algoritmus alapl műveletei.

3.6.3. A reprodukciós operátor: rulett-kerék kiválasztás

A reprodukciós művelet megvalósítására az egyik legegyszerűbb sztochasztikus módszert választottam. A „befolyásolt” rulett-kerekes, sztochasztikus kiválasztás oka az, hogy minden alacsonyabb fitness értékű kromoszóma is tartalmazhat valamilyen részlegesen hasznos információt a keresés irányítására. Emiatt – bár a túlélés valószínűsége kicsi –, a kiválasztás során mégsem hagyjuk teljesen figyelmen kívül [5].

A „befolyásolt” rulett-kerék kiválasztás egy illusztrációját a 12. ábra mutatja. A rulett-kereket felosztjuk a kromoszómák egymáshoz viszonyított „jósági” értéke szerint úgy, hogy a jobb kromoszómák nagyobb területet kapnak, a gyengébbek pedig kevesebbet („befolyásolt”). Ezután egy véletlen számot generálunk, amely a felvett érték alapján kijelöli számunkra a reprodukálendő kromoszómát, tehát az előbbiek szerint a jobb kromoszómák kiválasztása így nagyobb eséllyel történik meg amellet, hogy a gyengébbek is lehetőséget kapnak.



12. ábra: A „befolyásolt” rulett-kerék illusztrációja, a kromoszómák eltérő fitnessének arányában felosztva

A reprodukciót addig végezzük, amíg a megfelelő számú kromoszómát (árue rendezés változatot) ki nem választottuk. Ezt az értéket a *keresztelési ráta* adja meg, amely egy 0 és 1 közötti érték, annak függvényében, hogy mekkora százalékban legyen az új

generáció populációjába keresztezéssel létrejött, illetve korábbi populációból származó változat.

A keresztezési ráta, a genetikus algoritmus egyik meghatározó paramétere, amely megválasztásával a keresőtér spektruma beállítható. Alacsony érték esetén a konvergálás lassú lehet, magas érték esetén pedig könnyen a genetikus sodródás állapotába juthatunk, amely lokális szélsőértékhez vezethet.

3.6.4. A keresztezési operátor: Partially Mapped Crossover

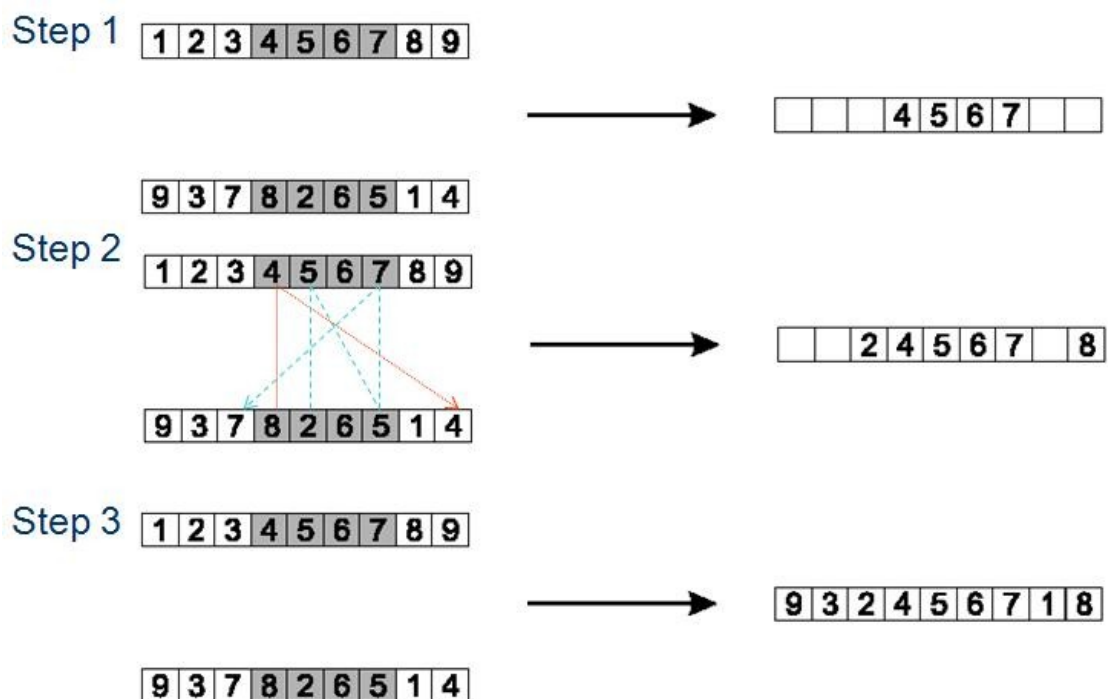
A reprodukció során kiválasztott kromoszómákból újakat hozunk létre. Az alkalmazásban erre az alapl műveletre a permutációs genetikus algoritmusok esetén leggyakrabban használt ún. *Partially Mapped Crossover* (PMX) eljárást használtam fel. Az eljárás alkalmazásának egyik fő indoka az volt, hogy a keresztezés során (például az egyponos keresztezés során ez gyakran előfordul) ne legyenek kettős értékek, hiszen a részkromoszómák kicserélése (keveredése) során – mivel azokban eltérő permutációban szerepeltek a gének, vagyis más volt az elrendezés – már meglévő érték kerülhet be a kromoszómába, ami a permutációs genetikus algoritmus és a probléma megoldásunk elvi hibáját eredményezni. A PMX ezt a hibázási lehetőséget kiszűri.

A Partially Mapped Crossover eljárás három, egymást követő lépésből áll (13. ábra).

A reprodukció során kiválasztott két szülő (13. ábra, „Step 1”) kromoszómája mentén véletlenszerűen kijelölünk két pontot és tekintjük a két pont által közrefogott intervallumot (fontos megjegyezni, hogy a két szülő ugyanazon részén jelöljük az intervallumot). Az utód kromoszóma (amely ugyanolyan hosszúságú, hiszen ugyannyi gént tartalmaz, másképpen: a tárolótéren a lokációk száma nem változik) az egyik szülőn kiválasztott intervallum génjeit közvetlenül megkapja.

A második lépésben a másik kiválasztott szülő génjeit kell az utód kromoszóma intervallumon kívüli helyeire átvinni. Megvizsgáljuk, hogy az utód kromoszómába helyezett első szülő génjei szerepelnek a második szülő kiválasztott intervallumában, amennyiben nem szerepelnek, egyszerűen átkerül az adott gén (13. ábra, „Step 2”, 9-3-1-es jelű gének). Ha szerepelnek a középső intervallum génjei a második szülő kiválasztott génjei között, akkor az átvitel a következők szerint történik:

- először megvizsgáljuk a második szülő kiválasztott intervallumának problémás értékeihez tartozó első szülőbeli géneket (13. ábra, 8-as és 4-es gén esete, piros vonal), majd ezeket a géneket megkeressük a második szülőben, ahol ez a gén megtalálható, az új kromoszómában ezen a pozíción fog a problémás, második szülőbeli gén helyet kapni.
- ha a második szülőben felkutatott első szülőbeli érték a kiválasztott intervallumon belül található (13. ábra, 2-es, 5-ös és 7-es gének, szaggatott kék vonal), akkor a keresési folyamat egy ciklussal tovább folytatódik.



13. ábra: A Partially Mapped Crossover keresztezési eljárás [7]

A lépést addig ismételjük, amíg az új kromoszóma teljesen fel nem töltődik génekkel (13. ábra, „Step3”).

A keresztezési folyamat során két szülőből két utód keletkezik, amelyből az első létrehozásának módját már ismerjük. A második utód létrehozásakor ugyanezeket a lépéseket hajtjuk végre azzal a különbséggel, hogy a szülőket felcseréljük.

(Az 13. ábra példájában a számokat tekintjük az egyes tárolóhelyek indexeinek. Így már könnyen értelmezhető két áruelrendezés-változat keresztezése.)

Fontos megemlíteni, hogy a keresztezésre az ún. *Variant Partially Mapped Crossover* (VPMX) eljárást is implementáltam. A hagyományos PMX eljárástól annyiban tér el ez

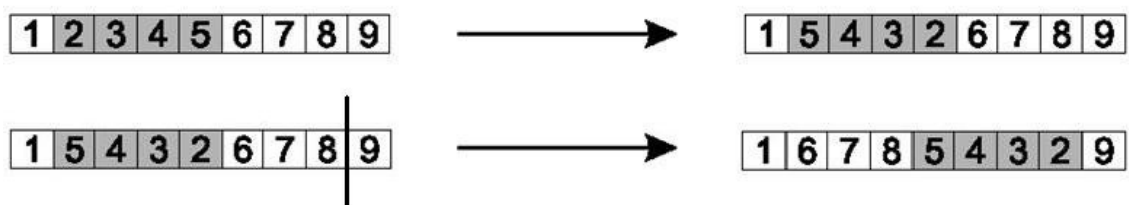
a technika, hogy a két szülőből nem azonos helyen történik (de azonos hosszúságú!) rész-kromoszómák kiválasztása. A tesztelések során ez az eljárás nem hozott megfelelő konvergálást, így a genetikus algoritmus továbbra is az általános PMX technikát használja a feladat megoldására.

Megjegyzés. A reprodukció és a keresztezés egy időben valósul meg. Mindaddig újabb és újabb reprodukció zajlik le, amíg a keresztezéssel létrejött új kromoszómák száma el nem éri a keresztezési mennyiséget. A reprodukció mindig a teljes populációt vizsgálja (rulett-kerék), időközben nem történik még meg a gyengébb kromoszómák eliminálása. Amint a keresztezett kromoszómák száma eléri a keresztezési mennyiséget, az új kromoszómák átveszik a gyengébb kromoszómák helyét! Tehát a legerősebb kromoszómák így kerülhetnek át a következő generáció populációjába.

3.6.5. A mutációs operátor: Inverted Displacement

A keresztezés során létrejött egyedeket egy adott valószínűséggel mutációnak vetem alá, amelynek esélyét a *mutációs ráta* szabja meg. A mutációra az egyik leghatékonyabb (a mutációt legjobban, de viszonylag irányítottan végrehajtó) eljárást alkalmaztam, ez az ún. *Inverted Displacement* eljárás.

Ez a mutációs eljárás két, egymás utáni lépésen alapul. Első lépésben véletlenszerűen kiválasztjuk a kromoszóma egy részét, amely elemeinek sorrendjét megfordítjuk az adott intervallumon belül (a 2.2-es fejezetben részletezett invertáló mutációs eljárás). Ezután az intervallumon kívül kiválasztunk egy ún. beszúrási pontot, amely után beillesztjük az előbbiek során invertált intervallumot (displacement), ezzel átrendezve a teljes kromoszómát. Az Inverted Displacement mutációs eljárást az 14. ábra szemlélteti.



14. ábra: Az Inverted Displacement mutációs eljárás

3.7. A genetikus algoritmus futtatási eredményei

A kemény számítási módszerekkel szemben, a lágy számítások implementációjának szerves része a tesztelési fázis. A genetikus algoritmus alapléteiben viszonylag nagy szerepet játszik a véletlen, még akkor is, ha az algoritmus irányítottan zajlik. A genetikus algoritmus a kiinduláskor és a futás során számos különböző irányba elmozdulhat.

3.7.1. A teszteléshez használt paraméterek és beállítások

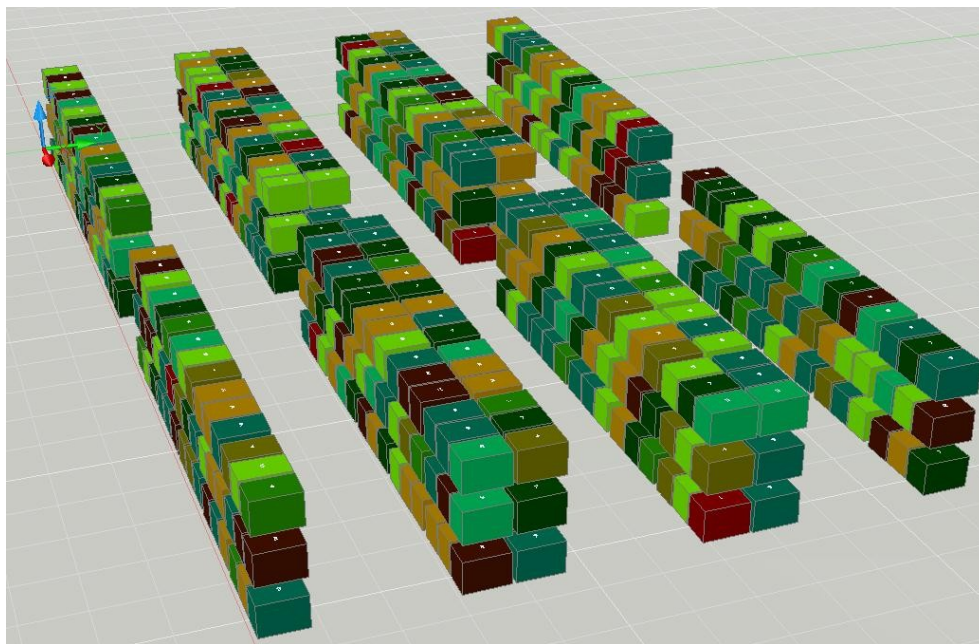
A tesztelés kiinduló adatsorát – a lokációk térbeli elhelyezkedésének értékeit – a rAcKer tárolótér tervező program szolgáltatta, a tárolandó árucikkek paraméterei tetszőlegesen lettek megválasztva. A tárolótér kapacitása *540 ER* volt, amely ugyanennyi lokációt jelent. A tárolandó szortimentben *9 árucikk* lett meghatározva eltérő, de összességében nem több mennyiségben, mint a tárolótér kapacitása. A megfelelő illusztrálás céljából az árucikkek *eltérő forgási sebességekkel* rendelkeztek.

A genetikus algoritmusra vonatkozó beállítások:

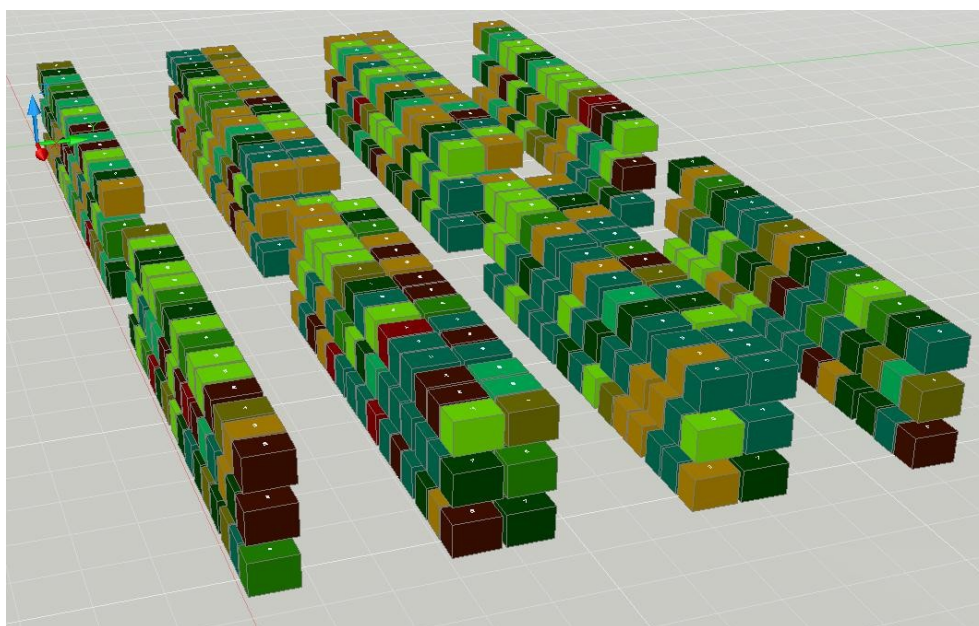
- A populáció mérete: *250 kromoszóma* volt, vagyis minden esetben ennyi különböző áruelrendezési változatot vizsgált az algoritmus.
- A keresztezési ráta *0,7-re* lett megválasztva, ami minden újabb generációban 175 új és 75 korábbi generációból származó áruelrendezést jelentett.
- A mutációs ráta *0,15-re* lett megválasztva, amely értelmében a keresztezés során létrejövő egyedek *15%* eséllyel mutálódtak.
- Az algoritmust *7500* generáción keresztül futtattuk.

3.7.2. A tesztfuttatás eredményeinek értékelése

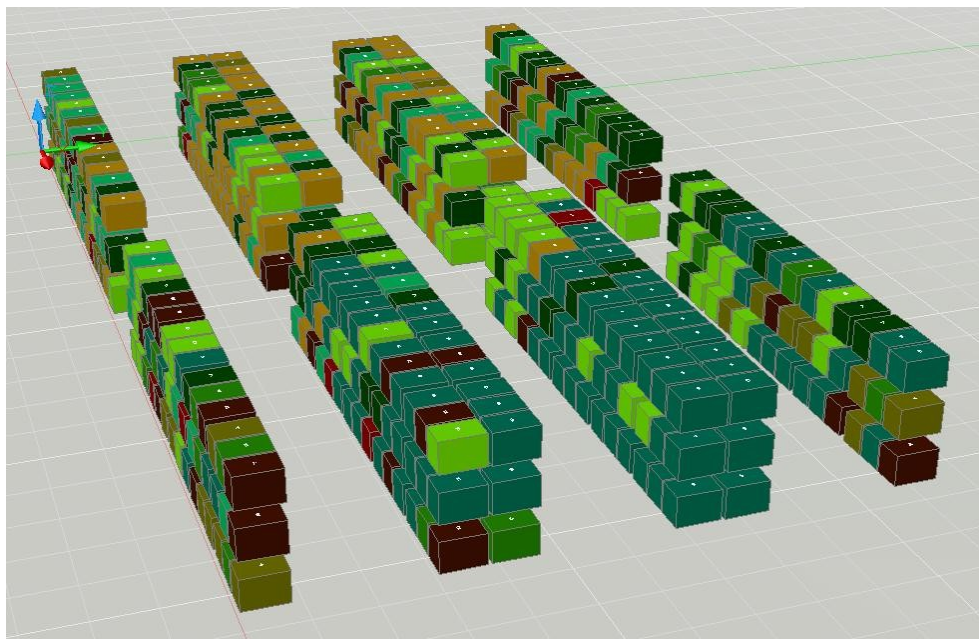
Az algoritmus futása során különböző generációk legjobb kromoszómájához tartozó áruelrendezést megjelenítettük. A vizuális megjelenítésre egy makró készült, amely segítségével AutoCad környezetben tekinthetjük meg az áruelrendezést. A tesztfuttatás eredményei az alábbi ábrákon láthatók (15., 16., 17., 18. és 19. ábra).



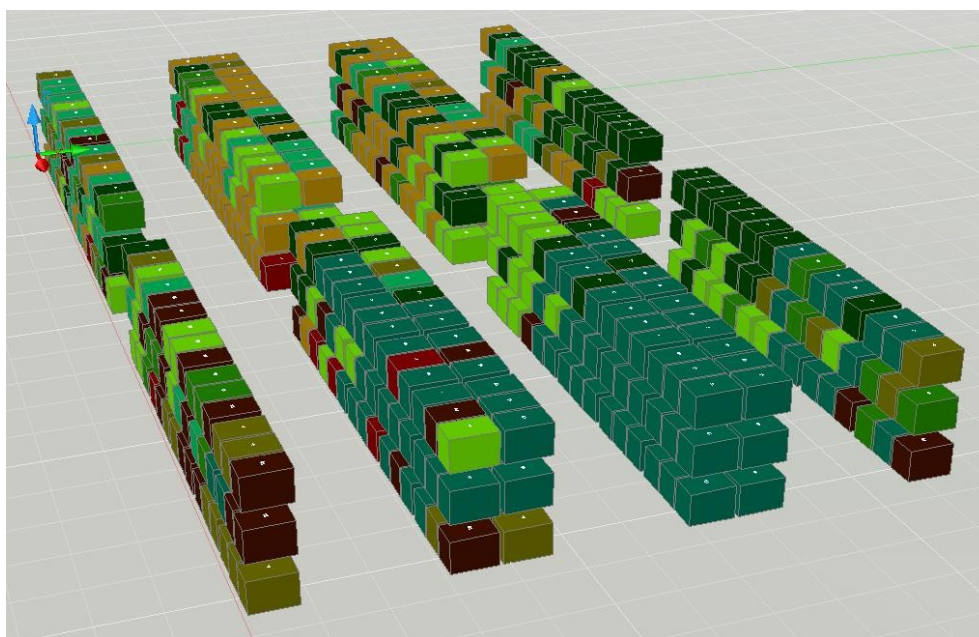
15. ábra: Az első generáció legjobb elrendezése (véletlenszerű elrendezés)



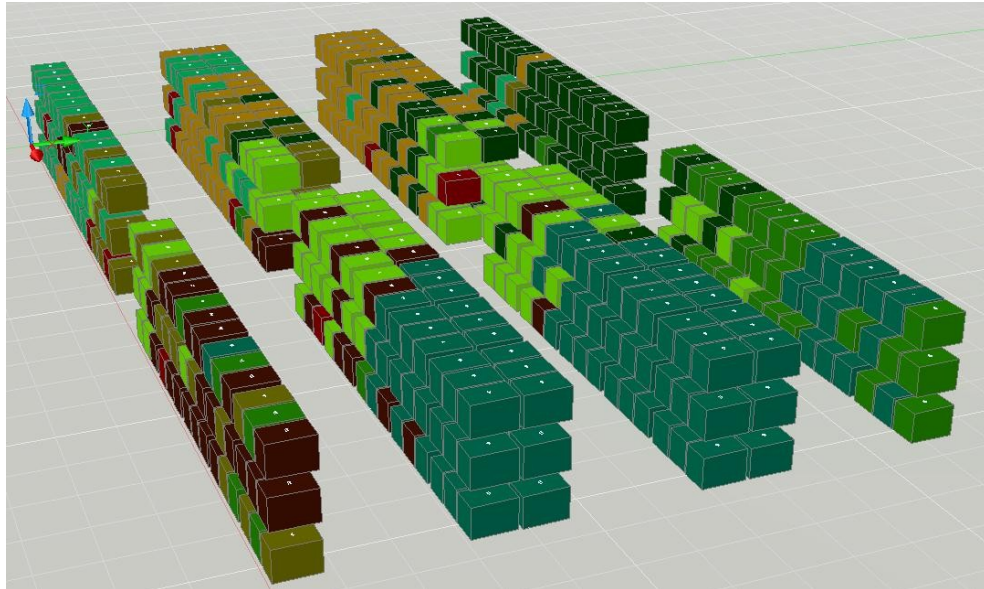
16. ábra: A 100. generáció legjobb elrendezése



17. ábra: Az 500. generáció legjobb elrendezése



18. ábra: Az 1000. generáció legjobb elrendezése



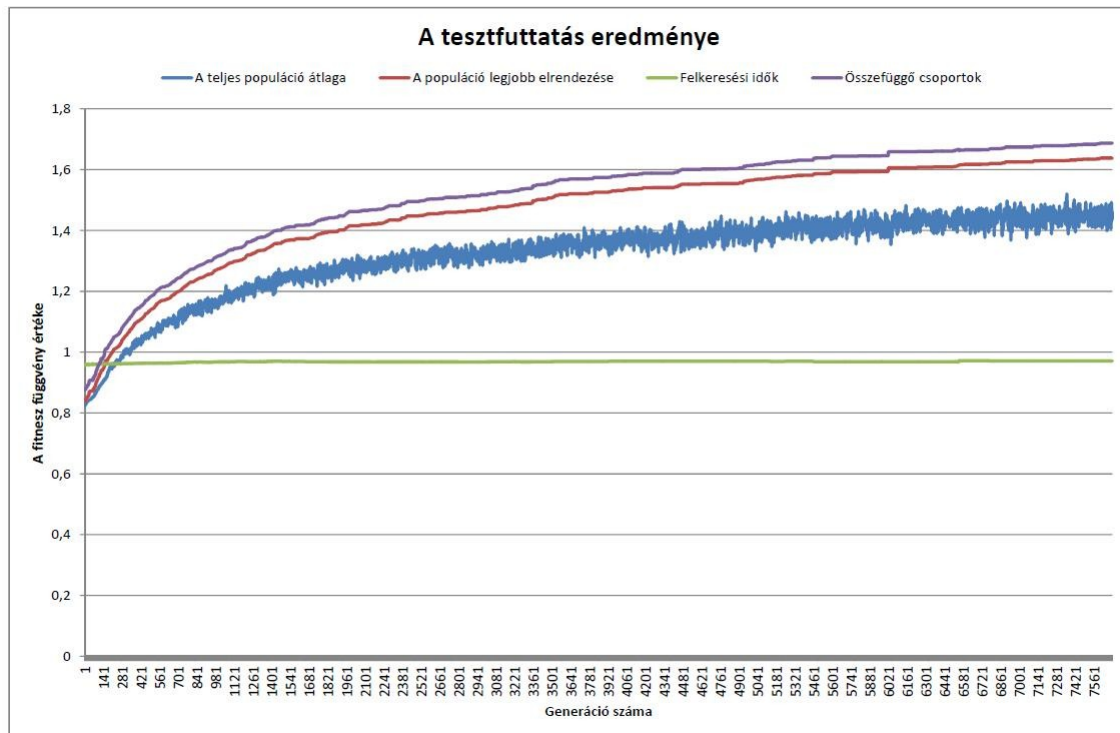
19. ábra: A 7500. generáció legjobb elrendezése

A 15. ábrán az első generáció populációjának egy áruelrendezés-változata, a véletlenszerű kiosztás látható. A genetikusan algoritmus a 250 hasonló véletlenszerű elrendezésből próbálja öröklés útján megtalálni a legjobb elrendezést. A generációk előrehaladtával az elrendezések egyre rendezettebb képet mutatnak.

A befejező – 7500. – generáció populációjának legjobb kromoszómája a 19. ábrán látható. Az ábrán jól megfigyelhető, hogy a véletlenszerű elrendezésből már egy viszonylag rendezett állapot alakult ki, azonban a tárolótér nagy részén még nagy szórás tapasztalható, ami azt jelenti, hogy a 7500 generáció kevésnek bizonyult.

Az előbbi tapasztalatokat összevetve a 20. ábra összefoglaló diagramjával látható, hogy a 7500 generáció alatt a fitness függvény értékei folyamatos konvergálást mutatnak, ami a 7500. generáció előtt sem állt meg, ami megerősíti, hogy további generációk képzésével az elrendezés tovább javulhatott volna.

A 20. ábrán feltüntetésre kerültek az egyes generációkban mért fitness értékek különböző aspektusai. A zöld görbe a fitness függvény felkeresési időre vonatkozó részét mutatja, ami a generációk során viszonylag lassan, de egyre jobban megközelíti az 1 értéket. Ez igazolja a 3.5. fejezetben kialakított legjobb felkeresési idő szerinti elrendezés helytállóságát. A fitness függvény második részének értékeit a lila színű görbe mutatja. Mivel a távolságkülönbségek viszonyítására is a legjobb összes felkeresési idővel rendelkező elrendezést választottuk (és nem a távolságkülönbség-minimumokból indultunk ki), a konvergálás erősen jelentkezik.



20. ábra: A genetikusan tesztelés eredménye

A teljes fitness függvény értékeit (amely az adott generáció populációjának legjobb áruelrendezési változatához tartozik) a bordó görbe prezentálja. A görbe erős konvergálása azt mutatja, hogy bár a felkeresési idők közel állnak a legjobb idő szerinti elrendezéshez, az árucikkek sokkal inkább egymás mellé rendelhetők és a csoportképzésre nagy tartalkékok jelentkeznék.

Az előbbi görbék nem szigorúan monoton növekednek, amely a genetikusan algoritmus lokális bizonytalanságára utal. A futás során előfordult, hogy a reprodukciós lépéssel a legjobb elrendezés közvetlenül a következő generációba jutott, azonban a keresztezések és mutációk hatására gyengébb kromoszómák is keletkeztek a keresési tér bővítése céljából.

A kék színű görbe az egyes generációk populációjának átlagos fitness értékét mutatja. Ezen a görbén a genetikusan algoritmus igazi kereső jellege látható, miszerint a keresztezések és mutációk hatására az adott populáció áruelrendezései nem minden esetben mutatnak átlagos javulást, azonban ezzel szemben mégis globális jellegű konvergálás mutatkozik. A generációk előrehaladtával a konvergálás mértéke folyamatosan csökken, ami abból fakad, hogy az egyre jobb áruelrendezések után már nehezebb javulást elérni.

4. További fejlesztés irányvonalai, a korlátok feloldásának lehetőségei

A kutatásom során sikerült az optimális áruelrendezés problematikáját egy lágy számítási módszerrel összekapcsolni, azonban még számos lehetőség rejlik az ismertetett elgondolásban. Az alkalmazás jelenleg is fejlesztés alatt áll, így bizonyos funkciók még nem működnek vagy nem elérhetők, a későbbiek során a diplomaterv keretében ezek még fejlesztésre kerülnek, amelybe a TDK keretében szerzett tapasztalataimat is beépíthetem.

A program első számú korlátai a logisztikai technológiai lehatárolásokhoz kapcsolódnak (3.2. fejezet). További célkitűzések során az alábbi korlátok feloldása lehetséges:

- a *célfüggvény vizsgálata*, kiterjesztése: egyéb lehetséges, figyelembeveendő szempontok matematikai megfogalmazása
- *átmenő és vegyes elrendezésű raktár* struktúrák figyelembevétele,
- *kombinált ciklusok és tárolótéri kommissiózási folyamatok* lehetőségének vizsgálata,
- *különböző tárolási rendszerek*, eltérő adottságainak vizsgálata a program működésének szempontjából,
- *tárolótéri forgalomkorlátozások* figyelembe vétele,

A programban működő genetikus algoritmust lehetne *finomítani*, más jellegű (keresztezési, mutációs stb.) módszerekkel is tesztelni, amely eredményeképpen meghatározhatók lennének a *leghatékonyabb eljárások* a keresztezésre és a mutációra egyaránt.

A logisztikai korlátozások feloldásán túl célszerű a genetikus algoritmus adott feladatokhoz való *megfelelő paraméterezését meghatározni*. A populáció mérete alapvetően befolyásolja a genetikus algoritmus keresési terét, azonban a populáció mérete függ a kromoszómák génjeinek számától, azaz esetünkben a tárolótér lokációinak számától. *További tesztelések* során *összefüggések keresésével* biztosítani lehetne, hogy a különböző méretű génállományokhoz (a különböző tárolótér kapacitásokhoz) melyek az ideális populációméretek, keresztezési és mutációs ráták, amellyel érdemes az algoritmust futtatni.

Jelenleg a kifejlesztett alkalmazáshoz nem tartozik egy olyan *felhasználói felület*, ahol minden – a futtatáshoz szükséges – paraméter beállítható lenne, azaz a későbbiekben cél egy olyan *űrlap létrehozása*, amelyre a felhasználó képes saját maga a genetikai algoritmust az adott feladathoz felparaméterezni (populáció mérete, generációk száma, keresztezési és mutációs ráta stb.). A *vizuális megjelenítésre* nagyobb hangsúly lehetne fektetni azáltal, hogy szervesen beépülne az alkalmazásba, továbbá a program futási eredményeiből kinyerhető információk megjelenítésére is nagyobb hangsúlyt fektetnék, azokat *standard riportok* formájában célszerű könnyen elérhetővé tenni.

5. AIM segédprogram bemutatása

Az anyagmozgatási feladatok esetében, a szükséges, becsült humán- illetve gépi erőforrások meghatározására egyik gyakran alkalmazott módszer a Közlekedésüzemi Tanszéken oktatott, úgynevezett „*Anyagmozgatási Időszükségletek Meghatározása*” (továbbiakban: AIM) módszer.

A TDK dolgozat egy másik célkitűzése az AIM módszer szerint történő anyagmozgatási időszükséglet meghatározását támogató alkalmazás kifejlesztése, amely a következő említett okok miatt szükséges.

5.1. Az AIM módszer és használata

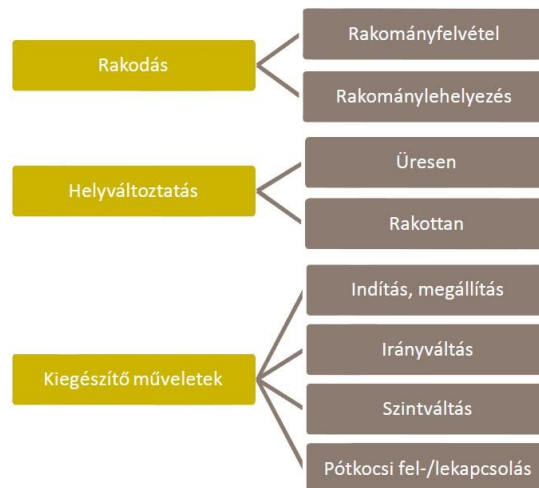
Az AIM módszer fejlesztői számba vették a leggyakrabban alkalmazott anyagmozgatási ciklusokat, eszközöket és gépeket, majd az egyes ciklusokat részfolyamataira bontották. Az ily módon kapott részfolyamatokra nagyszámú időmérési vizsgálatot végeztek el. A mérések alapján ezt követően átlagos időszükségleteket határoztak meg az egyes műveletelemekre nézve és eredményeiket táblázatos formába foglalták.

Tehát az AIM módszertan egy *adattáblázatos* időszükséglet meghatározó eljárás, amely nagyszámú mérési eredmények által létrehozott *időállandók* alapján történő időszükséglet meghatározásra épül [9].

Az AIM – módszerrel történő időszükséglet meghatározása az alábbi lépések szerint történik [9][10]:

- Az adott anyagmozgatási feladathoz tartozó, AIM szerinti műveletelemek felhasználásával felépítünk egy mozgató ciklust az 21. és 22. ábra alapján.
- A kiválasztott műveletelemeket a 22. ábra alapján a táblázatokból vett értékekkel felparaméterezzük, így megkapjuk egy mozgató ciklus idejét.
- Az így kapott ciklus időállandóját a folyamatban előforduló gyakoriságukkal megszorozva majd összegezve, megkapjuk a teljes mozgató feladat alapidejét, vagyis azt a veszteségmentes időigényt, amelyre egy átlagosan gyakorlott, kipihent dolgozónak ideális körülmények között szüksége van a feladat végrehajtásához [9].

- A 23. és 24. ábra alapján meghatározzuk a pótlékoló (pihenési és környezeti) tényezőket, amelyek a mozgatási feladat hatékonyságát befolyásolják [9].
- Az alapidőt a pótlékoló tényezőkkel korrigálva megkapjuk a teljes folyamat tervezhető idejét, amely az adott anyagmozgatási feladat időigényét a legjobban jellemzi [9].



21. ábra: Az anyagmozgatási tevékenységek műveletelemekre bontása [10]

AIM	SEGÉDESZKÖZ NÉLKÜLI KEZI ANYAGMOZGATÁS						ANYAGMOZGATÁS KEZI SZÁLLÍTÓESZKÖZZEL					
	H HELYVÁLTOZTATÁS			HKK HELYVÁLTOZTATÁS			F RAKOMÁNYFELVÉTEL			I INDÍTÁS ÉS MEGÁLLÍTÁS		
	Jel	A rakomány tömege (kg)	Időigény (10 ⁻² min/m)	Jel	A rakomány tömege (kg)	Időigény (10 ⁻² min/m)	Jel	A rakomány tömege (kg)	Időigény (10 ⁻² min/m)	Jel	A rakomány tömege (kg)	Időigény (10 ⁻² min/m)
	H-...I	<2	1,1	HKK-...I	<100	1,2	I	<100	1,2	I	<100	1,2
	H-...II	2...5	1,4	HKK-...II	100...250	1,4	I	100...250	1,4	I	100...250	1,4
	H-...III	5...20	1,7	HKK-...III	250...500	1,7	I	250...500	1,7	I	250...500	1,7
	H-...IV	20...50	2,0	HKK-...IV	500...1000	2,4	I	500...1000	2,4	I	500...1000	2,4
	Időigény (10 ⁻² min/m)			Időigény (10 ⁻² min/m)			Időigény (10 ⁻² min/m)			Időigény (10 ⁻² min/m)		
	A rakomány tömege (kg)	0..10	10..20	20..30	30..40	40..50	A rakomány tömege (kg)	0..10	10..20	20..30	30..40	40..50
	Szint (m)	I	II	III	IV	V	A rakomány tömege (kg)	0..10	10..20	20..30	30..40	40..50

ANYAGMOZGATÁS VILLAMOS GEPI EMELOTARGONCAVAL						ANYAGMOZGATÁS BELSŐGÉSI MOTOROS GEPI EMELOTARGONCAVAL						ANYAGMOZGATÁS KISEMELÉSŰ GEPI EMELOTARGONCAVAL					
HV HELYVÁLTOZTATÁS			HB HELYVÁLTOZTATÁS			HKT HELYVÁLTOZTATÁS			F RAKOMÁNYFELVÉTEL			L RAKOMÁNYHELYEZÉS			HVT HELYVÁLTOZTATÁS		
Mód	Jel	Időigény (10 ⁻² min/m)	Mód	Jel	Időigény (10 ⁻² min/m)	Mód	Jel	Időigény (10 ⁻² min/m)	Mód	Jel	Időigény (10 ⁻² min/m)	Mód	Jel	Időigény (10 ⁻² min/m)	Mód	Jel	Időigény (10 ⁻² min/m)
Üresen	HV - ...Ü	0,75	Üresen	HB - ...Ü	0,45	Üresen	HKT - ...Ü	1,0	Üresen	F - ...Ü	0,75	Üresen	L - ...Ü	0,75	Üresen	HVT - ...Ü	0,6
Rakottan	HV - ...R	0,90	Rakottan	HB - ...R	0,50	Rakottan	HKT - ...R	1,5	Rakottan	F - ...R	1,00	Rakottan	L - ...R	1,00	Rakottan	HVT - ...R	0,8
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	35	60	85	Üresen	Ü	25	40	60	95	Üresen	Ü	25	40	60	95
Rakottan	R	30	40	65	90	Rakottan	R	30	45	65	100	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					
Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0	Szintmagasság (m)	0..0,7	0,7..1,2	1,2..2,4	2,4..3,6	3,6..5,0
Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5	Rakodólap* Jel	I	2	3	4	5
Üresen	Ü	25	40	60	95	Üresen	Ü	20	40	65	85	Üresen	Ü	25	40	60	95
Rakottan	R	30	45	65	100	Rakottan	R	25	45	70	90	Rakottan	R	30	45	65	100
Időigény (10 ⁻² min)						Időigény (10 ⁻² min)						Időigény (10 ⁻² min)					

Táblázat a környezeti tényező (k) értékének meghatározásához			
A minősítés szempontjai			Minősítő pontszám
	(B-1)* értéke egyirányú	(B-2)* értéke kétirányú	
	forgalom esetén		
A közlekedő utak szélessége 10 m-nél hosszabb utak esetén (KSZ)	> 0,6 m	> 0,9 m	1**
	0,4...0,6 m	0,4...0,9 m	3
	0,2...0,4 m	0,3...0,6 m	5
	< 0,2 m	< 0,3 m	8
A közlekedő utak lejtése (UL)	A teljes úthosszon	0 %	1
	A teljes úthossz 50 %-ánál kisebb hossz	< 1 %	1
	A teljes úthossz 50 %-ánál nagyobb hossz	1...5 %	2
	A teljes úthossz 50 %-ánál nagyobb hossz	5...10 %	3
	A teljes úthossz 50 %-ánál nagyobb hossz	> 10 %	4
	A teljes úthossz 50 %-ánál nagyobb hossz	< 1 %	2
	A teljes úthossz 50 %-ánál nagyobb hossz	1...5 %	4
	A teljes úthossz 50 %-ánál nagyobb hossz	5...10 %	6
A megvilágítás erőssége	> 150 lx	nappali fény	1
	100...150 lx	jó	2
	75... 100 lx	közepes	3
	< 75 lx	gyenge	4
A minősítő pontszámok összege (MPÖ)	A környezeti tényező értéke (k)		
3	0,05		
4...6	0,1		
7...9	0,15		
10...12	0,2		
13...15	0,25		
16...17	0,3		
18	0,4		

* B - a közlekedő út szélességi mérete (m)
l - a rakománynak, illetve a szállítóeszköznek a haladás irányára merőleges maximális mérete (m)
** KSZ = 1 vendő akkor is, ha az útszélesség nem korlátozott

23. ábra: A környezeti tényezők [9]

A pihenési tényező (p) értékei

A tevékenység megnevezése	A rakomány tömege (kg)	A pihenési tényező
Segédeszköz nélküli kézi anyagmozgatás	< 2	0,05
	2...5	0,1
	5...20	0,2
	20...50	0,3
Kézi szállítóeszközzel végzett anyagmozgatás	< 100	0,1
	100...250	0,2
	250...500	0,25
	500...1000	0,3
Gépi anyagmozgatás		0,05
Szellemi tevékenység		0,1

24. ábra: A pihenési tényezők [9]

5.2. Az AIMaster program célja

Az előbbiekben látható volt, hogy az AIM módszertan használata a táblázatos megoldás (a táblázatban való keresgélés) miatt nehézkes és lassú, egy-egy művelet korrekt besorolása, majd időalapjának, alapidejének és tervezhető idejének meghatározása körülményes. Az **AIMaster** nevű program fejlesztésével a célom az volt, hogy az AIM módszer használatát gyorsítsam, a tervező szakemberek számára kényelmesebbé, a kevésbé gyakorlott felhasználók esetén pedig könnyebbé tegyem.

A program Borland Delphi fejlesztői környezetben készült el, kiterjesztése *.exe formátumú, rendszerigénye minimális, így bármilyen manapság használatos

számítógépen könnyedén használható. Az *.exe fájl futtatásával a program kezelőfelülete töltődik be (25. ábra).

AIMMaster by Fesus Norbert

Anyagmozgatási mód kiválasztása:
 Segédeszköz nélküli kézi anyagmozgatás

A kiválasztott anyagmozgatási mód esetén elérhető műveletelemek:

A kiválasztott művelettel paramétereinek megadása:
 Rakomány tömege [kg]:
 Mozgatás távolsága [m]:
 Mozgatás szintmagassága [m]:
 Műveletlem gyakorisága [db]:

☐ Adott pótlékoló tényező használata:

Műveletlem felvétele

Pihenési tényező meghatározása
 A tevékenység:
 Mozgatott tömeg:
☐ A művelethez kiegészítő szellemi tevékenység tartozik.
 Összesített pihenési tényező:

Környezeti tényező meghatározása
 Közlekedő út szélessége [m]:
 Mozgatóeszköz és rakomány együttes max. szélessége [m]:
 A forgalom kétirányú:
☐ A közlekedő út lejtése:
☐ A teljes úthossz lejtése: 0%
☐ A teljes úthossz legfeljebb 50%-án lejt
☐ A teljes úthossz legalább 50%-án lejt
 A lejtés mértéke:
☐ < 1%
☐ 1 ... 5%
☐ 5 ... 10%
☐ > 10%
 A megvilágítás erőssége: nappali fény (> 150 lx)
 Összesített környezeti tényező: Számítás

Anyagmozgatási mód szerinti jellemzők:
 Anyagmozgatási mód:
 Összes műveletlem gyakorisága:
 Összes alapidő:
 Összes tervezhető idő:

A teljes folyamat jellemzői:
 Összes műveletlem gyakorisága:
 Összes alapidő:
 Összes tervezhető idő:

Az eredmények mértékegysége:
 Perc (min)

Figyelem! A táblázatban szereplő értékek századpercben (0,01 min) értendők!

Sz.	Anyagmozgatási módja	Megnevezés	Jel	Időállandó	Gyakoriság	Alapidő	Pihen. t.	Körny. t.	Tervezhető idő

Mozgatás fel Mozgatás le Kijelölt műveletlem törlése Műveletlista törlése Exportálás Excel-be AIM értékek módosítása Kilépés

25. ábra: Az AIMMaster nevű program főablaka

5.3. A program funkciói és használata

Az alkalmazás fejlesztése során cél volt, hogy a lehető legegyszerűbben lehessen bevinni a paramétereket és a felhasználónak csak a minimálisan szükséges értékeket kelljen megadnia (természetesen a bevitt paraméterek formailag és technikailag is ellenőrzésre kerülnek, hiba esetén a program értesítést küld a felhasználónak). Az alábbiakban összefoglalom a program funkcióival és használatával kapcsolatos jellemzőket:

1. Új műveletlem felvétele (26. ábra):

Első lépésként ki kell választani az „Anyagmozgatási mód kiválasztása” nevű lenyíló listából azt az elemet, amellyel a felhasználó dolgozni szeretne. A kiválasztás után a lenyíló lista alatt megjelennek az elérhető műveletelemek, amelyek az AIM

szabályrendszere szerint az adott anyagmozgatási mód esetén számba vehető. A kívánt művelet megjelölése után kizárólag a művelet pontos meghatározásához szükséges kitöltendő mezők aktiválódnak, azonban amelyek nem szükségesek, azok nem. Ezzel a felhasználó azonnal tájékoztatást kap arról, hogy mely elemek megadása szükséges az adott művelet esetén. A szükséges paraméterek megadása után a felhasználó eldöntheti, hogy egy általa megadott

26. ábra: Új művelet felvétele

összesített pótlékoló tényezőt akar-e használni, vagy ez esetben is szeretne pontos (p,k) értékekkel számolni. (A gyakorlati életben sokszor nincs lehetőség (a kevés rendelkezésre álló információ miatt) a pihenési (p) és a környezeti (k) tényezők számítására, így sok esetben átlagos értékekkel számolt, összevont pótlékolási tényezővel célszerű számolni.)

Amennyiben a felhasználó egyéb szükséges paramétereket is megadott (pl. mozgatandó rakomány tömege, átlagos mozgatási távolság, le- és felhelyezési átlagos szintmagasságok, gyakoriság és pótlékolási értékek) a „Művelet felvétele” gombbal rögzítheti a kívánt elemet és a következőre léphet.

2. Pótlékoló tényezők meghatározása (27. ábra):

A pótlékoló tényezők megadásánál elsődleges szempont volt, hogy minél kevesebb beavatkozással, azok automatikusan kerüljenek meghatározásra.

A pihenési tényező a korábban kiválasztott anyagmozgatási mód és a megadott tömegű rakomány alapján automatikusan generálódik, a felhasználónak csupán azt kell jelölnie, hogy az adott művelet tartozik-e kiegészítő szellemi tevékenység (ekkor a pihenési tényező

27. ábra: Pótlékoló tényezők meghatározása

értéke az AIM szabályai szerint módosul).

A környezeti tényező számításakor (amennyiben a felhasználó ezt a megoldást választja annak meghatározására) már több paraméter megadása szükséges. A közlekedő útszélességek, az anyagmozgatásra használt eszköz szélessége, a forgalom irányítottsága (egy- vagy kétirányú), a közlekedő út lejtése, valamint a megvilágítás erősségének megadása után a „Számítás” gombbal a környezeti tényező azonnal meghatározásra kerül, így nem kell az egyes súlyozó pontszámokat a táblázatból keresgélni, majd átkonvertálni [9].

3. A tevékenység rögzített műveletelemei (28. ábra):

A tevékenység egyes műveletelemeinek rögzítése után, a műveletelem-lista kibővül az új elemmel. A táblázat a megszokott AIM-táblázat egy kibővített változatát alkalmazza.

A rögzített műveletelemek alábbi paraméterei jelennek meg:

- a művelet sorszáma a tevékenység műveletelem-listájában (Ssz.),
- az anyagmozgatási feladat (akkor van jelentősége, ha több anyagmozgatási feladattípus is részt vesz a teljes folyamatban),
- a műveletelem AIM szerinti megnevezése,
- a műveletelem AIM szerinti jele,
- a műveletelem adott paraméterek szerint meghatározott időállandója,
- a műveletelem gyakorisága,
- a műveletelem alapideje (az időállandó és gyakoriság szorzataként kapjuk),
- a műveletelemhez rendelt pihenési tényező értéke,
- a műveletelemhez rendelt környezeti tényező értéke,
- az előbbiek alapján előálló tervezhető idő értéke.

Megjegyzés: a táblázatban szereplő időértékek minden esetben 10^{-2} percben értendők!

A tevékenység műveletelemeinek felvétele során előfordulhat, hogy a felhasználó meggondolja magát és módosítani szeretné a korábbi tervezését. Az ilyen esetekre az alábbi műveletelem-lista módosítást segítő funkciók kerültek beépítésre:

- mozgatás fel / le: a kijelölt műveletelem listabeli helyét megváltoztathatjuk az adott irányokban,
- a kijelölt műveletelem törlése, valamint
- a teljes műveletelem-lista törlése.

Amennyiben a tevékenység AIM szerinti definiálása teljesen elkészült, a program lehetőséget biztosít arra, hogy a táblázatot a további felhasználás illetve mentés céljából Excel munkafüzetbe exportálhassuk.

Az anyagmozgatási folyamat műveletelemeinek listája: **Figyelem! A táblázatban szereplő értékek századpercben (0,01 min) értendők!** Tölts!

Sz.	Anyagmozgatás módja	Megnevezés	Jel	Időállandó	Gyakoriság	Alapidő	Pihen. t.	Körny. t.	Tervezhető idő
1	Anyagmozgatás villamos gépi emelőtargonccal	Helyváltoztatás	HV - 32 - R	28,8	20	576	0,05	0,25	748,8
2	Anyagmozgatás villamos gépi emelőtargonccal	Rakományelhelyezés	L - 1 - R	30	20	600	0,05	0,25	780
3	Anyagmozgatás villamos gépi emelőtargonccal	Egyéb - Indítás és megállás	IM	16	40	640	0,05	0,25	832
4	Anyagmozgatás villamos gépi emelőtargonccal	Egyéb - Irányváltás	IV	18	40	720	0,05	0,25	936
5	Anyagmozgatás villamos gépi emelőtargonccal	Helyváltoztatás	HV - 32 - Ü	24	20	480	0,05	0,25	624
6	Anyagmozgatás villamos gépi emelőtargonccal	Rakományfelvétel	F - 2 - R	40	20	800	0,05	0,25	1040

Mozgatás fel
Mozgatás le
Kijelölt műveletelem törlése
Műveletlista törlése
Exportálás Excel-be
AIM értékek módosítása
Kilépés

28. ábra: A tevékenység rögzített műveletelemei

4. Az elkészült folyamat összesített eredményeinek megjelenése (29. ábra):

A felhasználó számára várhatóan a leghasznosabb információ az a teljes folyamatot jellemző értékek lesznek.

A végeredményben megjelenik az összes műveletelem száma (a gyakoriságot is figyelembe véve), az összesített alap-, valamint tervezhető idő is. Az időértékek ezen felül tetszés szerint megjeleníthetők másodperc, századperc vagy perc egységben is, amely a könnyebb értelmezhetőséget segíti.

Amennyiben a folyamat több különböző anyagmozgatási feladattípusból is áll (kombinált feladat), akkor az egyes anyagmozgatási feladatokra külön-külön is megjeleníthetők az előbbieken említett értékek (30. ábra).

A teljes folyamat jellemzői

Összes műveletelem gyakorisága: 160
 Összes alapidő: 38,16
 Összes tervezhető idő: 49,608

Az eredmények mértékegysége: Perc (min)

A teljes folyamat jellemzői

Összes műveletelem gyakorisága: 160
 Összes alapidő: 2289,6
 Összes tervezhető idő: 2976,48

Az eredmények mértékegysége: Másodperc (s)

A teljes folyamat jellemzői

Összes műveletelem gyakorisága: 160
 Összes alapidő: 3816
 Összes tervezhető idő: 4960,8

Az eredmények mértékegysége: Századperc (0,01 min)

29. ábra: A tevékenység összesített információi különböző időegységben

Anyagmozgatási mód szerinti jellemzők

Anyagmozgatási mód:

Anyagmozgatás villamos gépi emelőtargonccal

Összes műveletelem gyakorisága: 160

Összes alapidő: 3816

Összes tervezhető idő: 4960,8

30. ábra: Anyagmozgatási mód szerinti jellemzők

5. Az AIM paraméterek módosítása (31. ábra):

A program elkészítése során törekedtem arra, hogy az alkalmazás a továbbiakban a lehetőségekhez mérten naprakészen tartható legyen. Az „AIM értékek módosítása” gomb segítségével előhívható az adatbázis, amelyből a program a számításokhoz szükséges értékeket veszi. Ezen a felületen az összes paraméter módosítható, azonban ez csak kellő körültekintés, illetve indokolt esetben javasolt (például határozott értékekkel rendelkezünk egy adott anyagmozgatási feladattal kapcsolatban).

Amennyiben később úgy dönt a felhasználó, hogy szeretne visszatérni a standard AIM táblázat

AIM AIM paraméterek módosítása

Segésszköz nélküli kézi anyagmozgatás | Anyagmozgatási kézi szállítóeszközzel

Rakodás kisemelésű kézi emelőtargonccal

Anyagmozgatás villamos gépi emelőtargonccal

Anyagmozgatás belsőgésű motoros gépi emelőtargonccal

Anyagmozgatás kisemelésű gépi emelőtargonccal

Anyagmozgatás villamos szállító- vagy vontatótargonccal

Pihenési tényező (p) értékei | Környezeti tényező (k) értékei

Alapértékek visszaállítása

Vissza

HV		Helyváltoztatás		
Mód	Jel			(0,01 min/m)
Üresen	HV - ... - Ű			0,75
Rakottan	HV - ... - R			0,90

F		Rakományfelvétel			
Szintmagasság (m)		0 ... 0,7	0,7 ... 1,2	1,2 ... 2,4	2,4 ... 3,6
Rakodólap	Jel	1	2	3	4
Üres	Ű	25	35	60	85
Rakott	R	30	40	65	90

L		Rakományhelyezés			
Szintmagasság (m)		0 ... 0,7	0,7 ... 1,2	1,2 ... 2,4	2,4 ... 3,6
Rakodólap	Jel	1	2	3	4
Üres	Ű	25	40	60	95
Rakott	R	30	45	65	100

Egyéb tevékenységek		
Megnevezés	Jel	(0,01 min)
Indítás és megállás	IM	16
Irányváltás	IV	18
Szintváltás	SZV	15

31. ábra: Az AIM értékeinek módosítása

értékeihez, akkor az „Alapértékek visszaállítása” gombbal ezt bármikor megteheti.

5.4. A program további fejlesztési lehetőségei

Látható, hogy az *AIMaster* nevű alkalmazás számos igényt elégít ki, azonban további fejlesztési lehetőségeket rejt magában. Tapasztalatok azt mutatják, hogy az AIM módszer által definiálható folyamatok és műveletelemek a hagyományos keretek között nagyon hasonlóak. Bizonyos folyamatokat, függetlenül a környezeti adottságoktól és jellemzőktől ugyanolyan módon írhatunk le: pl. egy betárolási művelet során, amikor nagyemelésű gépi targoncával a rakodófrontról indulva a tárolótérre anyagmozgatást végzünk, akkor a legtöbb esetben ugyanazokból a műveletelemből építhetjük fel a folyamatot, függetlenül attól, hogy melyik vállalat, melyik raktárában történik a tevékenység.

Az előbbieket ismeretében tehát úgynevezett „*AIM-pattern*”-eket, vagyis folyamatmintákat lehet létrehozni, amelyek még inkább megkönnyítenék egy-egy folyamat időszükséglet-számítását, hiszen nem egyenként kellene felvenni a műveletelemeket, hanem egy sablon alapján elegendő lenne csupán a legszükségesebb paramétereket megadni, így nem állna fenn a veszélye annak, hogy műveletelemeket hagyunk ki a folyamatunkból.

Az alkalmazás következő verziójában már szeretném szerepeltetni ezeket a folyamatmintákat is.

Megjegyzés. A raktári tárolótéren történő optimális áruelrendezés keresésekor az inicializációs lépés során meghatározott felkeresési (elérési) idők is egy ilyen folyamatminta alapján készültek el, amely a tárolótéri be- és kitárolás anyagmozgatási feladatát írja le.

ÖSSZEFOGLALÁS

Az optimális áruelrendezés genetikus algoritmus segítségével történő meghatározása egy újszerű megközelítése a felmerült problémának, amely újszerűsége miatt még rengeteg kiaknázatlan lehetőséget rejt magában.

A raktári tárolótérek technológiai tervezése során a tervezők nagy hangsúlyt fektetnek a geometriai paraméterek szakszerű meghatározására, azonban a tárolótér tervezése során nem kap kellő hangsúlyt az optimális áruelrendezés kialakítása, márpedig egy nem optimális layout jelentősen befolyásolja a raktári rendszer működési hatékonyságát, illetve működési költségeit. Az egyes árucikkek tárolótéren történő – előzetes – elrendezése sok esetben csupán intuitív módszerrel történik, amelyek a legtöbb esetben nem vezetnek helyes megoldásra.

A dolgozatomban kifejlesztettem egy alkalmazást, amelybe implementáltam egy korszerű optimum-kereső eljárást (genetikus algoritmus). Az eljárás segítségével különböző szempontok figyelembevételével a tárolótéri optimális áruelrendezés gyorsan, nagy pontossággal, intuitív módszerek használata nélkül meghatározható. A tervezés során figyelembeveendő főbb szempontok közül egyik a tárolótér geometriájából fakadó tárolási lokációk felkeresési ideje és az elhelyezendő árucikkek forgási sebessége alapján történő árukiosztás, a másik tervezési szempont az, hogy az azonos cikkszámúval rendelkező áruk állványrendszeren történő helyfoglalása úgy valósuljon meg, hogy azok egymás melletti tároló lokációkon foglaljanak helyet, vagyis az azonos cikkszámú áruk ne legyenek szétszórva a tárolótér különböző pontjain.

A kifejlesztett alkalmazást MS Excel VBA felületen hoztam létre, amely így széleskörűen felhasználható. A program nagy előnye, hogy használatával az optimális áruelrendezés kérdésköre a korábbiaktól eltérően egészen a raktár technológiai előtervezési fázisáig hozható előre (tehát az áruelrendezés problematikájával már az előtervezés fázisában foglalkozunk) így a következő tervezési fázisban (részletes tervezés) a tervező már egy „pontosabb” előtervvel dolgozhat tovább.

A TDK munkám során egy anyagmozgatási idősükséglet meghatározására alkalmas programot is kifejlesztettem. Az AIMaster alkalmazás létrehozásával számítógépes

környezetben sikerült megvalósítani az anyagmozgatási feladatok idősükségleteinek egy szabályozott keretek között történő meghatározását, amellyel könnyebb, gyorsabb és pontosabb anyagmozgatási idősükséglet számítások végezhetők el, ezáltal a program segítségével lehet a logisztikus tervezőknek a raktári folyamatok tervezése során, továbbá demonstrációs eszközként oktatási célokra is kiválóan alkalmas.

Ábrajegyzék

1. ábra: Egy GA működése	13
2. ábra: Az egyponos keresztezési eljárás alapelve [7]	15
3. ábra: Az invertáláson alapuló mutáció elve [7]	15
4. ábra: A tárolótéri elrendezés adatainak exportálása a rAcKer-ből [2]	19
5. ábra: A tárolótéri lokációk adatai Excelben	19
6. ábra: A készletezési adatok megjelenése Excelben	20
7. ábra: A bázispontok illusztrációja (pirossal jelölve)	22
8. ábra: A legjobb összes felkeresési idő szerinti változat	25
9. ábra: Véletlen elrendezés kiinduló állapota (részlet)	29
10. ábra: Az új hozzárendelés kijelölése (részlet)	30
11. ábra: Az új hozzárendelés érvényesítése (részlet)	30
12. ábra: A "befolyásolt" rulett-kerék illusztrációja, a kromoszómák eltérő fittségének arányában felosztva	31
13. ábra: A Partially Mapped Crossover keresztezési eljárás [7]	33
14. ábra: Az Inverted Displacement mutációs eljárás	34
15. ábra: Az első generáció legjobb elrendezése (véletlenszerű elrendezés)	36
16. ábra: A 100. generáció legjobb elrendezése	36
17. ábra: Az 500. generáció legjobb elrendezése	37
18. ábra: Az 1000. generáció legjobb elrendezése	37
19. ábra: A 7500. generáció legjobb elrendezése	38
20. ábra: A genetikusan tesztelt algoritmus tesztfuttatásának eredménye	39
21. ábra: Az anyagmozgatási tevékenységek műveletelemekre bontása [10]	43
22. ábra: Az AIM módszer időállandói táblázatba foglalva [9]	43
23. ábra: A környezeti tényezők [9]	44
24. ábra: A pihenési tényezők [9]	44
25. ábra: Az AIMaster nevű program főablaka	45
26. ábra: Új művelet felvétele	46
27. ábra: Pótlékoló tényezők meghatározása	46
28. ábra: A tevékenység rögzített műveletelei	48
29. ábra: A tevékenység összesített információi különböző időegységben	48
30. ábra: Anyagmozgatási mód szerinti jellemzők	49
31. ábra: Az AIM értékeinek módosítása	49

Irodalomjegyzék

- [1] K. Bóna (2008): Targoncás kiszolgálású, darabárus italipari nagykereskedelmi elosztóraktár tervezése, elektronikus egyetemi jegyzet
- [2] Á. Kardos (2011): Soros állványos tárolótér tervező program fejlesztése, TDK dolgozat
- [3] Á. Kardos (megjelenés alatt): Soros állványos tárolótérek számítógéppel támogatott tervezése, Logisztikai Híradó
- [4] K. Bóna (2011): Ellátási, elosztási logisztika, előadás jegyzet
- [5] Gy. Retter (2007): Kombinált fuzzy, neurális, genetikus rendszerek (kombinált lágy számítások), ISBN 978-963-87401-0-6
- [6] Deep, Mebrahtu (2012): Variant of partially mapped crossover for the Travelling Salesman Problems. IJCOPI Vol. 3, No. 1, EDITADA. ISSN: 2007-1558.
- [7] M. Mitchell (1998): An Introduction to Genetic Algorithms, A Bradford Book The MIT Press, Cambridge, Massachusetts - London, England, ISBN 0-262-13316-4 (HB), 0-262-63185-7 (PB)
- [8] A.E. Eiben, J.E. Smith (2003): Introduction to Evolutionary Computing, Springer, Natural Computing Series 1st edition, ISBN: 3-540-40184-9 Corr. 2nd printing, 2007, ISBN: 978-3-540-40184-1
- [9] AIM gyakorlat, gyakorlati segédlet, BME Közlekedésüzemi Tanszék
- [10] K. Bóna (2011): Anyagmozgatási és raktározási folyamatok, előadás jegyzet