MISKOLCI EGYETEM
GÉPÉSZMÉRNÖKI ÉS INFORMATIKAI KAR
HATVANY JÓZSEF INFORMATIKAI TUDOMÁNYOK DOKTORI ISKOLA

HOZZÁRENDELÉSI FELADATOK LOGISZTIKAI RÁFORDÍTÁS ALAPJÁN TÖRTÉNŐ OPTIMALIZÁLÁSA HÁLÓZATSZERŰEN MŰKÖDŐ, MŰSZAKI FELÜGYELETET ÉS KARBANTARTÁST ELLÁTÓ RENDSZEREKBEN

Ph.D. értekezés

Készítette:

Kota László

okleveles mérnök informatikus

A doktori iskola vezetője:

Prof. Dr. Tóth Tibor

Tudományos vezetők:

Prof. Dr. Jármai Károly

Dr. Bányai Tamás

Miskolc, 2012

Tartalomjegyzék

Cselényi József professzor úr emlékének

A disszertációban előforduló jelölések jegyzéke

$\boldsymbol{L}$: a rendszer útmátrixa,

$\boldsymbol{Y}$: a rendszer hozzárendelési mátrixa,

p: az objektumok száma,

s: szakértők száma,

ka: karbantartó üzemek száma,

ra: raktára száma,

be: beszállítók száma,

lo: logisztikai szolgáltatók száma,

q: a rendszerelemek száma,

i, j: szakértő futó indexe,

k, l: objektumok futó indexe,

t: ciklusszám futó indexe,

m: karbantartó üzemek futó indexe,

f: raktárak futó indexe,

b: beszállítók futó indexe,

κ_k: a kötelezően előírt vizsgálatok száma objektumonként,

$MTBF_k$: a meghibásodások között eltelt átlagos idő objektumonként,

ϑ: a vizsgált időszak,

ε_k: eseti karbantartási feladatok száma objektumonként ϑ időszak alatt,

τ_k^K : egy-egy karbantartás, műszaki felülvizsgálat átlagos ideje objektumonként,

τ_k^m: a vizsgálatok időközének minimális értéke objektumonként,

$\bar{v}$: a szakértő átlagos sebessége,

τ_k^f: objektum felkeresési ideje,

$\tau_{k,l}$: *k*. és *l*. objektum között úthossz megtételéhez szükséges idő,

P_i: az *i*. szakértők hány karbantartási feladatot végezhet el,

$P_{i\ min}$: minimálisan előírt vizsgálatszám,

$P_{i\ max}$: maximálisan előírt vizsgálatszám,

T: ciklusok száma a ϑ időintervallumban,

τ_i^t: a szakértő által egy ciklusban felhasznált idő,

τ^T : egy ciklus ideje,

c^t : a t-edik ciklusban felkeresendő objektumok száma,

τ_k^k: a k-adik objektum felülvizsgálatának, karbantartásának átlagos ideje,

O : az objektumok halmaza,

O_i : a felkeresendő objektumok halmaza az *i*-edik szakértőnél,

O_i^S: a felkeresendő objektumok rendezett halmaza az *i*-edik szakértőnél,

C^S: a szakértők által felkeresett objektumokhoz kapcsolódó logisztikai ráfordítások,

C^K: karbantartó üzemekhez kapcsolódó logisztikai ráfordítások,

C^B: a beszállítókhoz kapcsolódó logisztikai ráfordítások,

C^R: a raktárakhoz kapcsolódó logisztikai ráfordítások,

C_{BK}: a beszállítás költsége a beszállítóktól a karbantartó üzemekbe,

c_{bk}: a beszállítás fajlagos költsége a beszállítóktól a karbantartó üzemekbe,

B_{BK}: beszállítások számának mátrixa a beszállítóktól a karbantartó üzemekbe,

C_{KO}: a karbantartó üzemek és az objektumok közötti szállítási költség,

c_{ko}: a szállítás fajlagos költsége a karbantartó üzemek és az objektumok között,

B_{KO}a beszállítások számának mátrixa az objektumokból a karbantartó üzemekbe,

C_{KR}: a karbantartó üzemek valamint a raktárak közötti anyagmozgatás költsége,

c_{kr}: a szállítás fajlagos költsége a karbantartó üzemek és a raktárak között,

B_{KR}a beszállítások számának mátrixa az karbantartó üzemekből a raktárakba,

C_{BR}: a beszállítóktól a raktárakba való anyagmozgatás költsége,

c_{br}: a szállítás fajlagos költsége a beszállítóktól a raktárakba,

B_{BR}a beszállítások számának mátrixa a beszállítóktól a raktárakba,

C_{RO}: a raktárak és az objektumok közötti szállítások költsége,

c_{ro}: a szállítás fajlagos költsége a karbantartó üzemektől a raktárakba,

B_{RO}: a beszállítások számának mátrixa a raktárakból az objektumokhoz,

C_R: anyagok, alkatrészek raktározási költsége,

P_m^{KO}: az *m*-edik karbantartó üzemhez rendelt objektumok száma,

P_f^{RO}: az *f*-edik raktárhoz rendelt objektumok száma,

P_b^{BK}: a *b*-edik beszállítóhoz rendelt karbantartó üzemek száma,

P_b^{BR}: a b-edik beszállítóhoz rendelt raktárak száma,

F: az egyed fitnesz értéke,

C_s : a szakértők által megtett út költsége,

B_s : szakértők alkalmazásának költsége,

B_{CC}: szakértők ciklusidő túllépésének büntetőfüggvénye,

B_{SC}: szétszórt objektumok büntetőfüggvénye,

B_F : szakértő minimális terhelés alatt büntetőfüggvénye,

B_M: szakértő maximális terhelés fölött büntetőfüggvénye,

B_N : felügyeleti, karbantartási feladatok közeliségének büntetőfüggvénye,

P_{CC}: a ciklustúllépés büntetőértéke,

T_i : a szakértő által igényelt ciklusok száma,

T_{max}: maximális ciklusszám,

P_F: a megengedettnél kevesebb vizsgálat büntetőértéke,

P_M: a megengedettnél több vizsgálat büntetőértéke,

P_N: a megengedettnél közelebb lévő vizsgálat büntetőértéke,

τ_i^m: a vizsgálatok minimális távolsága,

P_{SC} : a szétszórt objektumok büntetőértéke

s_P: azon szakértők száma, ahol van hozzárendelt objektum,

P_s : szakértő alkalmazásának költsége.

1 Bevezetés

Napjainkban a globalizált termelés és szolgáltatás területén egyre nagyobb mértékben terjednek a hálózatszerűen működő, logisztikával integrált rendszerek. Kezdetben a termelési ágazat globalizálódott, mára már a szolgáltató iparban is jelen vannak az akár több földrészen is jelen lévő multinacionális vállalatok.

A szolgáltatások területén kiemelt jelentőségűek a műszaki felügyeleti és karbantartási rendszerek, mert ezek a termelésnek, szolgáltatásnak - ezek közül kiemelt területek a lakosságot közvetlen érintőek – a biztonságát, megbízhatóságát biztosítják. Ilyen területek például a kommunális szolgáltatások, a víz, a szennyvíz, a gáz, a villamos energia, a távfűtés, az üzemanyag ellátás, a telekommunikációs szolgáltatások, vagy akár a felvonók és a kötélpályák karbantartásához kapcsolódó szolgáltatások [21][22].

Ezek megbízható, balesetmentes és gazdaságos üzemeltetése megköveteli az időszakos műszaki ellenőrzéseket, karbantartásokat. Felülvizsgálatuk, karbantartásuk pedig az esetek túlnyomó többségében speciális, vizsgához kötött szaktudást igényel. Ilyen például az emelőgépek egy sajátos változata a felvonók, amelyek vizsgálata, karbantartása életvédelmi szempontból is igen fontos, így ezt a területet kormányrendelet szabályozza. Hasonlóan kezelhetőek a különböző szolgáltató hálózatok, például villamos energia-, gáz-, hő-, vízellátás biztosítására szolgáló olyan objektumok, biztonsági berendezések, irányító alközpontok, ellenőrző egységek, kritikus hálózati elemek, amelyek időszakos felülvizsgálata, helyszíni ellenőrzése, karbantartása szükséges [23] [24].

Az ilyen tevékenységek végrehajtásánál a következő feladatok jelentkeznek:

- egy-egy személynek kell évente egy vagy több alkalommal a helyszínre kiszállni,
- az alkatrészek, szerszámok, gépek egyéb eszközök helyszínre való ki- és visszaszállítására is szükség lehet,
- az is belátható, hogy a tevékenységet ellátó személyeknek vagy szakértőknek a kiszolgált terület objektumaihoz közel kell lakniuk, mert így képesek kis időráfordítással, költséghatékony tevékenységet végezni, mivel így a célterületre keveset kell utazniuk,
- a szükséges anyagok, alkatrészek, eszközök, gépek a rendszer különböző pontjaiban telepített raktárakban találhatóak, ezekből történik ki- ill. visszaszállításuk,
- illetve előfordulhatnak a helyszínen nem javítható szerkezeti elemek, amelyeket kiszerelés után a karbantartó üzemekben újítanak fel.

2 A kutatás célkitűzései és módszertana

A felvonók vizsgálatát és kötelező karbantartását országunkban - hasonlóan a többi országhoz - törvény írja elő. A felvonóvizsgáló-karbantartó rendszer egész Magyarországot felöleli, hazánkat tekintve ez az egyik legnagyobb nagykiterjedésű karbantartó, felügyelő hálózat. Hazánk világviszonylatban kicsinek mondható, a kifejlesztett rendszer általános, alkalmazható bármely országra, vagy más egyéb kis, vagy nagyléptékű hálózatos rendszerekre. Ilyen nagyméretű, sokszereplős rendszerekben gyakori probléma a diszponálások optimalizálása, a hozzárendelések optimális meghatározása, egyedi igények kielégítése, és a váratlanul fellépő helyzetek gyors, minden igényt kielégítő megoldása.

Célom a rendszer kiterjesztésével és általánosításával az ilyen és az ehhez hasonló, a logisztikában gyakran felmerülő nagykiterjedésű és elemeiben nagy számosságú, műszaki felügyeleti és karbantartási rendszerek matematikai leírása, matematikai módszerek és modellek definiálása, valamint optimalizálási kérdéseinek megoldása.

A szakirodalom kutatása folyamán megállapítható, hogy míg a „tiszta” többszörös utazó ügynök problémának igen kiterjedt irodalma van, az ilyen nagykiterjedésű sok bemeneti paraméterű és sok peremfeltétellel ellátott rendszerek - amelyek a logisztika tudományterületén gyakran előfordulnak - optimalizálása nincs kidolgozva. Bár vannak kutatások, amelyek a probléma alappillérével a több végpontú utazó ügynök problémával foglalkoznak [25], de ezek sem alkalmaznak kiterjedt feltételrendszert, bonyolult többfázisú módszereket használnak, amelyek összehangolása nehézkes.

Kutatásom során a problématerületet többféle módon megvizsgáltam. Megalkottam a műszaki felügyeleti és karbantartási rendszerek általános modelljét, amelyet a kutatómunka folyamán tovább finomítottam a modellek általánosításával és újabb feltételek bevonásával. Az optimalizálási vizsgálatot szűkítettem a szakértő objektum hozzárendelésre, valamint a szakértők bejárási ciklusainak meghatározására, amelyet először egyszerűbb heurisztikákkal vizsgáltam, többfázisú modellek alkalmazásával [9].

Vizsgáltam az optimalizálás területén alkalmazott biológiai működés modellezésén alapuló algoritmusokat is [5][16]. Számos algoritmus tanulmányozása után döntöttem az evolúciós programozás alkalmazása mellett. A választott módszer az evolúciós módszertan szerinti mikroevolúciót valósítja meg. Az evolúciós programozás egy igen általános algoritmus, így konvergenciája lassabb lehet, mint a speciális, egyetlen problémára alkalmazható algoritmusnak [26][27][28], viszont megfelelően kidolgozva az adatstruktúrákat, valamint megfogalmazva a peremfeltételeket, alkalmassá tehető egy ilyen bonyolult probléma egyfázisú optimalizálására, amelyet kutatási eredményeim is igazolnak.

3 A szakirodalom áttekintése

Disszertációm a hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek leírásával, valamint hozzárendelési kérdéseivel foglalkozik. Ezen belül is az objektumok hálózatszerű műszaki felügyeleti és karbantartási rendszerének átfogó modelljének egy alrendszerével: a hálózatszerűen működő, műszaki felügyeletet és karbantartást ellátó szakértő objektum hozzárendelési feladatok optimalizálásával.

A szakirodalom áttekintése után megállapítottam, hogy az eddig megjelent publikációk nem foglalkoznak a disszertációm első részében kidolgozott műszaki felügyeleti és karbantartó rendszerek logisztikai szempontrendszerű leírásával. A szakirodalom egyrészt műszaki oldalról dolgozza fel a karbantartási szakterületet [21], másrészt az életciklus menedzsment [29], valamint a minőségbiztosítási [22] megközelítés az elterjedt. A szakirodalomban ilyen logisztikai szempontú modell eddig nem ismert. Az általam kidolgozott rendszer egyes részei (4. fejezet, 5. fejezet) bekerültek a tanszék oktatási anyagaiba is.

A disszertációm 7. fejezetében kidolgozott hálózatszerűen működő műszaki felügyeleti és karbantartó rendszerek szakértő objektum hozzárendelési problémájának megoldása visszavezethető az úgynevezett többszörös utazó ügynök problémára (MTSP - Multiple Traveling Salesman Problem), bár a megoldás részleteiben eltér a szakirodalomban eddig tárgyalt megoldásoktól.

A fellelhető szakirodalomban az utazó ügynök probléma (TSP - Traveling Salesman Problem) igen széleskörűen tárgyalt, sőt gyakran új algoritmusokat az utazó ügynök feladat megoldására fejlesztenek ki, mint például az Ant Colony algoritmus [30], amelyeket később általánosítanak más problématerületen való alkalmazásra. A többszörös utazó ügynök probléma azonban kevésbé kutatott terület így viszont jelenleg még nem rendelkezik hasonlóan átfogó szakirodalommal.

Az MTSP rövid definíciója a következő: Adott egy csomópont halmaz, amely a felkeresendő helyeket reprezentálja és *m* utazó ügynök. Az *m* utazó ügynöknek fel kell keresnie minden csomópontot úgy, hogy minden csomópontot egyszer és csak egyszer kereshetnek fel, majd az utazó ügynökök visszatérnek a kiinduló csomópontba úgy, hogy a csomópontok felkeresési költsége, ideje minimális [31] [32].

Az MTSP többféle változatban fordul elő az alkalmazási területtől és problémától függően, például:

- Egy vagy több végpontos MTSP:
    - Egy végpontos esetben az utazó ügynökök ugyanabból a pontból indulnak és ide is térnek vissza [33][34].

    - Több végpontos utazó ügynök feladat, ahol az utazó ügynökök több csomópontban helyezkedhetnek el. Ilyenkor megkülönböztethetünk:
        - fix végpontú MTSP: amikor az utazó ügynökök a kiinduló pontba térnek vissza [35].
        - Nem fix végpontú utazó ügynök feladat: amikor az utazó ügynökök a körút végén bármely csomópontba visszatérhetnek, [36] [37].
- Az utazó ügynökök száma is lehet kötött vagy változó.
- Az egyes utazó ügynökökhöz fix költség is rendelhető. Ilyenkor az utazó ügynökök száma nem fix, mivel az utazó ügynökök számának, vagy a teljes rendszer költségének minimalizálására törekszünk.
- Lehetséges időablakhoz kötni a csomópont felkeresésének időpontját [38]. Ilyenek például az iskolabusz körútszervezési problémák (MTSPTW).
- Egyéb speciális problémák, például:
    - az utazó ügynök által megtehető maximális út,
    - a felkeresendő csomópontok alsó és vagy felső korlátja,
    - a felkeresendő csomópontok maximális távolsága, stb.

A MTSP problémakörben alkalmazott közelítő eljárások használata többféleképpen indokolható. A TSP és így az MTSP probléma is NP-nehéz (non-deterministic polynomial-time hard - nem determinisztikusan polinomiális időben eldönthető) [39], ebből következik, hogy az MTSP probléma is az NP-nehéz feladatosztályba tartozik. Az NP-nehéz problémák esetén nem ismeretesek polinom korlátos műveletigényű megoldó eljárások. A rendelkezésre álló algoritmusok műveletigénye általában exponenciális függvénye a probléma méretének. Ennek következtében a nagyobb méretű feladatok megoldásának időigénye annyira nagy, hogy esetenként a megoldás nem realizálható. Így az NP-nehéz problémák esetében létjogosultsága van a közelítő eljárásoknak. Ezek nem a tényleges optimális megoldást szolgáltatják, hanem egy lehetséges megoldást eredményeznek csak. Az így előállított lehetséges megoldással szemben alapvető elvárás, hogy „jó” legyen abban az értelemben, miszerint a hozzátartozó célfüggvény érték közel legyen az optimum értékhez. [40]

Az utazóügynök- és a többszörös utazó ügynök probléma az operációkutatás, valamint a mesterséges intelligencia kutatási területén is megjelenik a szakirodalomban. A megoldási módszerekben megfigyelhető, hogy míg az operációkutatás főleg az egzakt matematikai módszerek felől közelít a megoldáshoz (integer programozás,

korlátozások és szétválasztás, korlátozás és vágás) [41][42], addig a mesterséges intelligencia, az informatikai tudományok modelljeivel és módszereivel támogatja a feladatmegoldást. Így ilyen irányultságú kutatásoknál inkább a biológiai modellezésen alapuló algoritmusok (neurális hálók [43], genetikus algoritmus, hangyakolónia algoritmus [44][45], evolúciós algoritmusok [46], részecskeraj algoritmus [47][48], tabukeresés [49], DNS algoritmus [50] alkalmazása és ezek kifejlesztése az elterjedtebb.

A TSP-hez képest az MTSP a valós szituációkhoz jobban alkalmazkodik, így a logisztikai területén e kutatási terület a feladatok centralizációja, a globalizáció és a nagyméretű hálózatok alkalmazása miatt került előtérbe.

A kutatás folyamán megvizsgáltam a szakirodalomban a témához kapcsolódó eddigi kutatásokat, publikációkat. Mivel a kutatási terület igen szorosan a gyakorlathoz kapcsolódik, így a publikációk is általában egy-egy alkalmazás köré szerveződnek, továbbá látható az is, hogy a kutatások zöme a logisztika területéhez köthető.

A problématerületen több egzakt módszer is ismert [51][52], különféle eljárásokat felhasználva, ilyenek például a következők:

- egészértékű lineáris programozás [53],
- vágósíkok módszere,
- branch and bound módszerek,
- Lagrange relaxációs módszer.

Egészértékű lineáris programozás módszerét használja Bektas et al.[54], ahol a kutatók fix és nemfix végpontú MTSP modelljét építették fel, majd megvizsgálták a nemfix végpontú MTSP modell implementációját. A futási idők már igen kevés bejárandó csomópont esetén is túllépték a kísérletben definiált három órás határt.

Laporte és Nobert [55] szimmetrikus és aszimmetrikus struktúrákra két MTSP modellt mutat be. Egy olyan modellt vizsgálva, ahol az utazó ügynökök mind egy városban helyezkednek el. Publikációjukban először a problémát 1-TSP-vé konvertálják – a konverziós eljárás a peremfeltételek nélküli MTSP problémánál jól használható [56]- majd ennek megoldására egy egyenes és egy fordított algoritmust írnak le:

- egyenes algoritmus: nincsenek alkörút eliminációs és integralitási feltételek, amíg az egészértékű megoldást elérik, ha nem legális alkörutak vannak a megoldásban (amelyek a depót nem érintik) a feltétel bekerül a problémába és az optimalizálás újra megtörténik,

- fordított algoritmus: amikor először csak a fokszám feltételt veszik figyelembe, majd az illegális alkörutakat vizsgálják.

Az integralitást mindkét esetben korlátozás és szétválasztás vagy vágósík (Gomory vágósíkok) módszerrel érik el. A megoldást néhány száz (180-240) városra implementálják, ahol az egyenes algoritmus „hamar elvérzik" memóriaproblémák miatt (már mintegy 70 városnál), de e felett a problémaméret felett a fordított algoritmus megoldási sebessége gyorsan csökken.

Ali és Kennington [57] egy korlátozás és szétválasztás alapú algoritmust dolgozott ki aszimmetrikus MTSP megoldására. Az algoritmusban Lagrange relaxációt és szubgradiens módszert, valamint mohó algoritmust használ fel. Ebben a modellben az utazó ügynökök szintén egyazon városból indulnak. A módszer mintegy hatvan városig és hét utazóügynökig ad elfogadható időn belül megoldást.

Gavish és Srikanth fix számú utazó ügynökre kínál megoldást [58] Lagrange relaxációt valamint a korlátozás és szétválasztás módszerét felhasználva, ahol az utazó ügynökök az előző megoldásokhoz hasonlóan szintén egyazon városból indulnak, valamint ugyan oda érkeznek vissza. A megoldás folyamata a következőképpen foglalható össze:

- Előállítja a probléma felső korlátját heurisztikus csomópont beillesztéses módszerrel, amely a Lagrange multiplikátorok előállítására szolgáló szubgradiens módszerhez szükséges.
- Megoldja a Lagrange problémát.
- Ha a csomópontok fokszáma kisebb, mint az előírt, heurisztikus módszerrel javítja az eredményt a Lagrange probléma megoldását használva kiindulópontként.
- A Lagrange vektor értékeit a szubgradiens módszerrel aktualizálja.
- Ha a megoldás nem elfogadható a korlátozás és szétválasztás módszerrel redukálja a problémaméretet.

A futtatások eredményeiből kiderül, hogy a módszerrel mintegy 500 csomópontig illetve 10 utazó ügynökig szolgáltat megoldást elfogadható időn belül

A publikációk másik nagy csoportja heurisztikus módszerekre fókuszál, e publikációk általában valamilyen gyakorlatorientált problémára adnak megoldást. A logisztika tudományterületén a problémák jellegénél fogva gyakori a heurisztikus módszerek használata [59], hiszen a logisztikai modellek, feltételrendszerek igen bonyolultak is lehetnek, amelyek egzakt módszerekkel nem vagy csak nehezen kezelhetők.

A következő felsorolásban szakirodalomból vett MTSP-hez kapcsolódó néhány példát mutatok be heurisztikus módszerek alkalmazására:

- Újság nyomtatás ütemezése: az egyik első MTSP alkalmazás a nyomdaiparban született, ahol 5 pár henger között fut a papír és mindkét oldalára egyidejűleg nyomtatják egy újság különféle változatait. Egy változat különféle hirdetési oldalakat tartalmazhat, amelyeket 4, 6 és 8 oldalas formában nyomtatnak. Az ütemezés feladata eldönteni, hogy melyik forma melyik nyomtatási menetben kerüljön nyomtatásra. Az MTSP terminológia szerint például a nyomólemez cseréjének költsége a csomópontok közötti közlekedés költsége [60]. Hasonló problémakör az újságba beillesztett előnyomtatott hirdetésinzertek beillesztésének problémája, itt az újságba beillesztett hirdetések az újság terjesztési területének megfelelő földrajzi régiónként változnak. Azaz minden régiónak más hirdetésekkel kerül az újság kinyomtatásra, az MTSP terminológiájában az egyes régiók megfelelnek az egyes csomópontoknak, az egyes gyártósorok pedig az egyes utazóügynököknek. [61]. A probléma megoldására a szerző egy ma már szélesebb körben tárgyalt osztott kromoszómás genetikus algoritmust használ fel.
- Diszponálás és ütemezés: Egy klasszikus logisztikai feladat, amelyben a pénzszállítók a pénzt különböző bankoktól begyűjtik és egy központi lerakóhelyre szállítják. A feladat lényege meghatározni az egyes pénzszállítók begyűjtési útvonalát a minimális költségre törekedve [62]. Lenstra et al. [63] két hasonló alkalmazást ír le. Az első alkalmazás észak Hollandia területén a telefonfülkék javításához kiszálló telefonszerelők hozzárendelése és ütemezése. A második alkalmazás az Utrechti postaautók levélbegyűjtési körútjainak megtervezése, valamint a postaautók számának minimalizálása. Hasonló problémát említ Zhang et al. [64], amelynél a probléma fotóscsoportok hozzárendelése és körutazásaik megtervezése nagyszámú általános és középiskolába.
- Iskolabusz körútjának megtervezése: A problémát Angel et al. [65] vizsgálta az MTSP variációjaként több különféle feltétel bevezetésével. Ebben az esetben a célfüggvény igen komplex, a cél a buszok kihasználtságának maximalizálása, a

buszok által megtett út minimalizálása, viszont a buszok nem szállíthatnak a névleges kapacitásukon felül utasokat és az iskolákba kell érniük a megadott időn belül.

- Interjúk ütemezése: Az MTSP egy variációja, amelyben az utazásszervezők beutazzák a városokat és azokon belül több helyen bemutatókat tartanak, utazásokat értékesítenek. [66]
- Mobil robotok útvonaltervezése: A probléma autonóm mobil robotoknál jelentkezik. Széleskörű elterjedésük folyományaképp ilyen robotok nemcsak a logisztika területén vannak jelen, mint például raktár automatizálás, hanem megjelennek a katonai felderítés területén, UAV-k (Unmanned Aerial Vehicle) útvonaltervezésénél, vagy a postai feladatok diszponálásánál. Különleges felhasználási területük például a bolygók felderítésére alkotott mobil robotok. Sőt már a háztartásokban is megjelennek az autonóm mobilrobotok robotporszívók formájában és az alkalmazások köre a jövőben szélesedhet. Ebben az esetben a cél az optimális út keresése valamilyen speciális feltételrendszer mentén. Például a terület bejárása a legrövidebb út megtétele mellett. Általánosan *n* számú robotnak *m* számú célpontot kell felkeresnie majd visszatérni a kiindulási pontra. Az autonóm robotok útvonaltervezésének problémáival foglalkozik Brummit et al. [67], akik speciális feltételeket igénylő nem strukturált környezetben [68] is vizsgálták a problémát. Hasonló problémakör a pilótanélküli légi járművek útvonalának megtervezése, viszont itt szigorú időtényező, az üzemanyag fogyása is, így a Ryan et al.[69] által kidolgozott alkalmazás az időablakos MTSP tárgykörét ismerteti.
- Meleghengerlés ütemezése: Egy fontos acélipari problémát kutat Tang et al. [70], ahol a hengerek cseréjéhez kapcsolódó átállási költséget minimalizálja MTSP modell segítségével. Az MTSP modellt TSP-vé konvertálja, amely ez esetben mivel a probléma egyvégpontos - lehetséges [71], majd egy módosított genetikus algoritmus segítségével ad optimális megoldást a problémára. Figyelembe véve a speciális jellegzetességeket, így például a megrendeléseket a csomópontok reprezentálják. Az alkalmazott genetikus algoritmus variánsban összehasonlítja az új populációt az előzővel és mindig a legjobb megoldást

használja fel a következő populáció generálásához. A Volgenant and Jonker [72] egzakt algoritmussal összehasonlítva látható hogy az egzakt algoritmus futási ideje már a problémaméret kismértékű emelkedésekor is exponenciálisan növekszik, míg a genetikus algoritmusé sokkal kisebb mértékben emelkedik [73] [74].

- Globális navigációs rendszerek (GNSS Global Navigation Satellite System) alkalmazása a geodéziában: A legújabb kutatások közé tartozik a Saleh és Chelouah [75] által kidolgozott MTSP modell, amely a GPS rendszerrel kijelölt geodéziai hálózat bázispontjainak optimális meghatározására szolgál. A kidolgozott modell alkalmas lineáris vagy trianguláris GPS hálózatoknál az optimális bejárás meghatározására. A kifejlesztett GPS-GA algoritmus képes az optimális bejárás meghatározására, figyelembe véve többek között a vevőegységek számát, valamint az egyes referenciapontok időbeli felkeresési paramétereit, mint bejárási időkorlátokat. A kutatók a kapott eredményeket összevetették egy tabukeresés (TS) és egy szimulált hűtés (SA) algoritmus által szolgáltatott eredményekkel. A kutatók által vizsgált példákban a genetikus algoritmus minden esetben jobb eredményt szolgáltatott a másik két algoritmusnál

A disszertációmban vizsgált műszaki felügyeleti és karbantartási rendszerek az MSTP probléma egy speciális, kevéssé kutatott kiterjesztését képviselik, több végpontos fix végpontú utazó ügynök problémával írhatók körül (MmTSP vagy MDMTSP-vel jelöli a szakirodalom; Multi Depot Multiple Traveling Salesman Problem). A problématerület ezen ága azonban még a generalizált vagy speciális MTSP megoldásoknál is kevésbé kutatott terület. Csak a legújabb kutatások foglalkoznak a problémával, olyan heurisztikus algoritmusok alkalmazása jelenik meg, mint például a következők:

- Ágensalapú modellezés valószínűségi kollektívek módszerével, Anand et al. [37]: Egy többágensű modellt dolgozott ki az MTSP probléma megoldására, ahol a kollektív memória és a valószínűségi kollektívek módszerét használta fel. Az ismertetett algoritmus egyszerű beszúrásos, csere és eliminációs heurisztikát is felhasznál. Két egyszerű esetben vizsgálja az algoritmus működését tizenöt csomóponttal és három utazó ügynökkel.

- Genetikus algoritmus, Suk-Tae Bae et al. [25]: Két fázisra bontva oldják meg a problémát. Első fázisban egy klasztering eljárással a több középpontú problémát lebontják több egy középpontú problémává, majd ezeket genetikus algoritmus segítségével oldják meg. Eljárásukat kilencvenkilenc csomópontig, négy ügynökig vizsgálták. A klasztering eljárás gyorsasága miatt széleskörűen használt módszer [76][77][78] de az esetek döntő többségében lokális optimumot ad eredményül. További gyakran használt a megoldott TSP probléma partícionálása is, amely szintén a gyors heurisztikák közé sorolható [79].
- Hangyakolónia algoritmus, Ghafurian et al. [80]: A Marco Dorigo által kidolgozott [81] hangyakolónia algoritmust felhasználva oldja meg az általános MmTSP modellt. A problémafán mesterséges hangyák - a biológiai hangyák élelemkeresési metódusát modellezve - keresik az optimális megoldást. A szerzők az algoritmust összehasonlították a Lingo szoftver megoldásával és kimutatták, hogy a hangyakolónia algoritmus hasonlóan jó, vagy jobb megoldást ad, viszont az algoritmust csak mintegy negyven csomópontig és öt ügynökig vizsgálták.

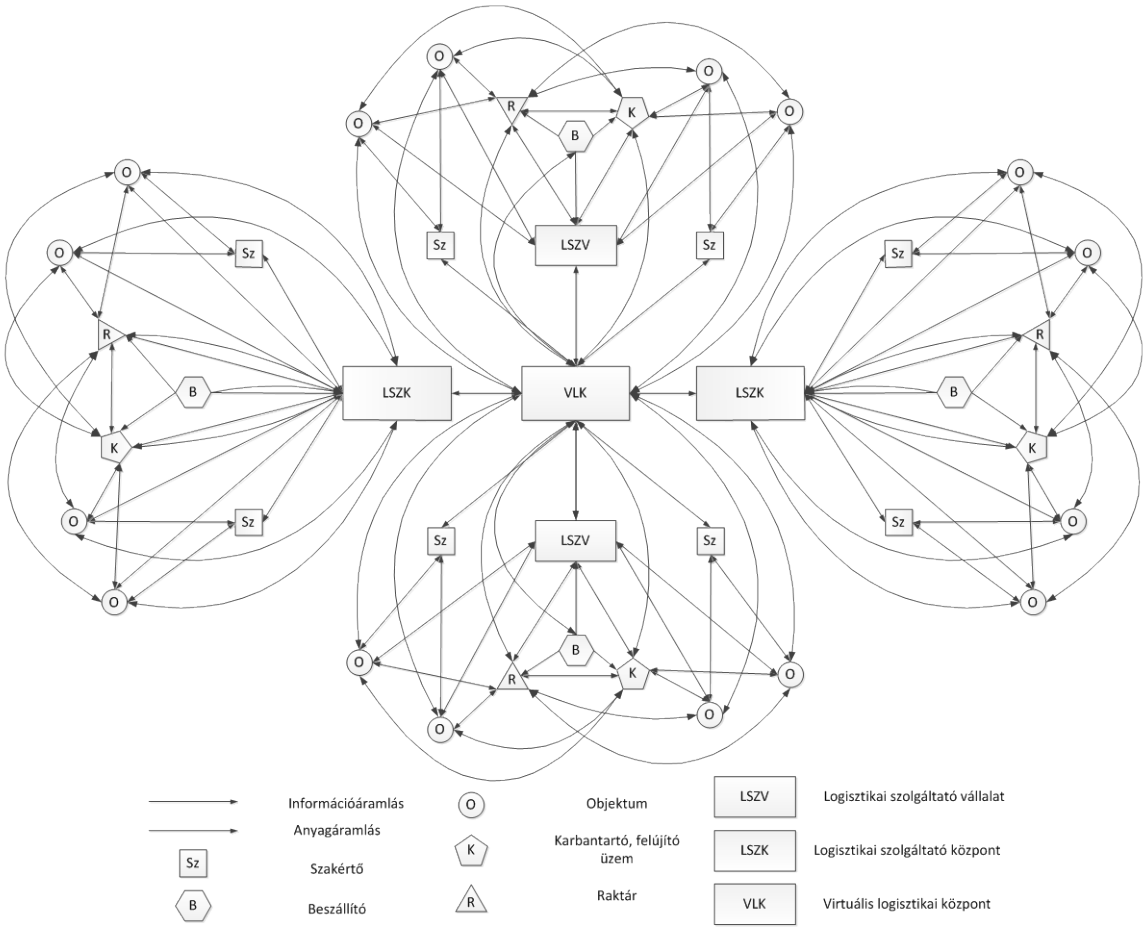
Ezekkel a módszerekkel csak maximum száz csomópontig vizsgálták a problémát, valamint a modellek nem terjednek ki speciális, a műszaki felügyeleti és karbantartási rendszerekben felmerülő jellemzőkre.

A szakirodalom alapján megállapítható, hogy az alkalmazott egzakt módszerek csak igen kis problémaméreteknél (néhány 10 utazóügynök, valamint néhány száz bejárt pont) adnak kivárható időn belül megoldást [82].

A logisztika területén előforduló nagyméretű problémák – mint például a műszaki felügyeleti és karbantartó rendszerekkel kapcsolatos optimalizálási feladatok – megoldása során előfordulhat akár több tízezer felkeresendő csomópont (objektum) is, amelyet tízes nagyságrendű vagy akár a százat is meghaladó számú ügynök (szakértő) felügyel. Ilyen esetekben - a legújabb kutatási irányzatokat figyelembe véve - a heurisztikus és a korszerű mesterséges intelligencia módszerek használata komoly sikerekkel kecsegtet.

4 Hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek struktúrája

A hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek kiterjedhetnek egy városra, egy régióra, egy országra, lehetnek kontinensen belüli vagy akár földrészeken átívelő rendszerek. Feladatuk egyrészt a hálózat megfelelő pontjaiban az előírásoknak megfelelő időpontokban rendszeres felülvizsgálatok és vizsgálatok elvégzése szakértők által, másrészt karbantartások és felújítások megvalósítása. A karbantartási feladat hatékony végrehajtását egy vagy több térben szétszórt anyag- és eszközraktár, karbantartó üzem segíti.



1. ábra Hálózatszerűen működő, regionális decentrumokkal rendelkező műszaki felügyeleti és karbantartási rendszer általános struktúrája

A logisztikai rendszer feladata, hogy biztosítsa a felülvizsgálathoz és a karbantartáshoz szükséges erőforrások és szakértők egymáshoz rendelését, valamint a karbantartáshoz szükséges anyagok rendelkezésre állását. Tekintettel arra, hogy a logisztikai erőforrások és igények, szakértők, anyagok, eszközök, objektumok térben szétszórtan

helyezkednek el, a hálózatszerűen működő műszaki felügyeleti és karbantartási integrált rendszer akkor működtethető optimálisan, ha virtuális logisztikai központ, vagy az irányítási és anyagmozgatási feladatok egy szervezetbe tömörítése esetén logisztikai szolgáltató központ [83] látja el az ilyen típusú feladatokat.

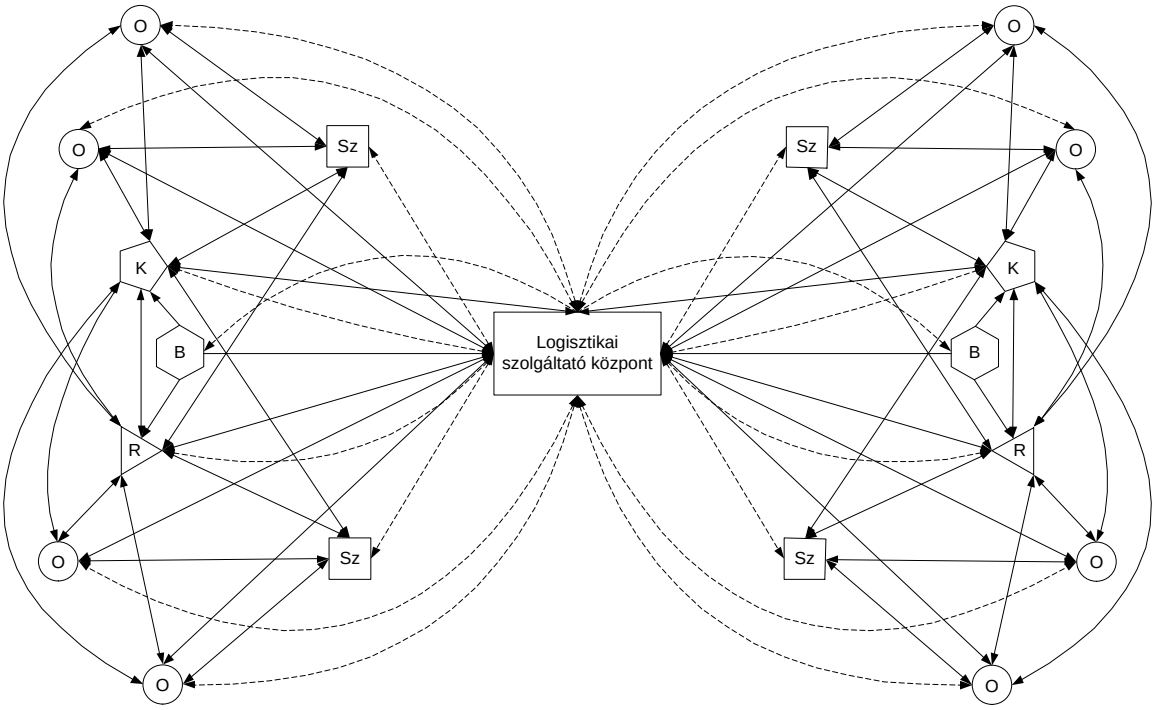
A rendszert egy virtuális logisztikai központ irányítja (1. ábra). Kisebb kiterjedésű rendszerek esetén, amelyek lehetnek regionális vagy akár országos rendszerek is, olyan rendszerváltozat is lehetséges, ahol az irányítási, információáramoltatási és feldolgozási, valamint az anyagmozgatási feladatokat egyszerre ellátó logisztikai szolgáltató központ képezi a rendszer magját. A rendszert irányító virtuális logisztikai központ információs kapcsolatban áll a rendszert alkotó:

- szakértőkkel,
- raktárakkal,
- karbantartó egységekkel,
- alkatrészek és eszközök beszállítóival,
- szállítmányozó vállalatokkal, fuvarozókkal, logisztikai szolgáltatókkal, amelyek a szükség szerint jelentkező szállítási feladatokat ellátják [84].

A virtuális logisztikai központ (VLK) nem vesz részt a fizikai folyamatokban, a személy, az anyag és az eszközáramlásban, hanem csak diszponálja azokat.

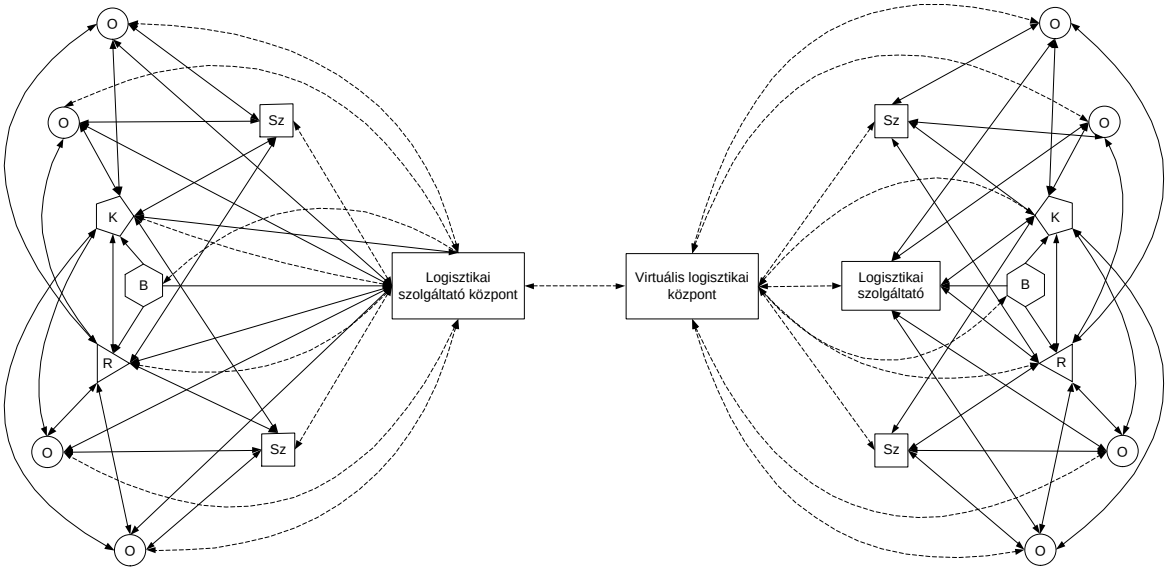
A VLK feladatai:

- kiválasztja a szakértőket a rendelkezésre álló szakértők közül, optimális esetben – amikor megadható, hogy a szakértő lakhelye hol legyen - meghatározza a szakértők helyét, azonban ez az esetek döntő többségében kötött,
- hozzárendeli az objektumokat (4.1.1) a szakértőkhöz,
- ütemezi, nyilvántartja a szükséges műszaki ellenőrzéseket, vizsgálatokat,
- kezeli az eseti meghibásodásokat,
- kiválasztja a raktárakat, karbantartó egységeket a rendelkezésre állók közül, optimális esetben - új raktár telepítése - meghatározza helyüket,
- amennyiben lehetséges megválasztja a beszállítókat, hiszen sok esetben nem áll rendelkezésre több beszállító,
- diszponálja, ütemezi a raktárakba, karbantartó egységekbe való beszállításokat, onnan történő kiszállításokat,
- diszponálja a szállítóeszközöket.



2. ábra Regionális hálózatszerűen működő műszaki felügyeleti és karbantartási rendszer

A rendszert irányító logisztikai központ a döntéseihez matematikai modellekre épülő optimalizálási eljárásokat használ, amelyekben a működtetési követelményként előírt feltételeket, mint korlátozó feltételeket veszi figyelembe.



3. ábra Decentralizált hálózatszerűen működő műszaki felügyeleti és karbantartási rendszer

Kisméretű rendszerek esetén eredményes lehet logisztikai szolgáltató központ alkalmazása a rendszerben. Ilyenkor az 1. ábrából levezethető, egyszerűbb struktúrát

építhetünk ki (2. ábra), amelynek irányítását már nem virtuális, hanem valós logisztikai szolgáltató központ végzi. Amennyiben ilyen erőforrások nem állnak rendelkezésre, a befektetés költsége, a megtérülési mutatók, illetve a fő kompetenciák figyelembevétele alapján gazdaságosabb lehet a logisztikai folyamatok kihelyezése (outsourcing) és az 1. ábrán is látható virtuális logisztikai központ alkalmazása.

A rendszer méretének növekedése estén, a szolgáltatás minőségének fenntartása valamint a versenyképesség növelése érdekében szükség lehet regionális decentrumok kiépítésére (3. ábra) [85][86].

A decentrumok kiépítésének előnyei:

- csökkennek a logisztikai költségek,
- a decentrum irányítási feladatokat is átvállalhat,
- a szolgáltatási területhez közelebb történő tervezéssel és irányítással csökkenthető a bizonytalanság,
- a szolgáltatás minősége javul.

4.1 A rendszer elemeinek és azok funkcióinak definiálása

A következő fejezetben bemutatom a műszaki felügyeleti és karbantartási rendszerek tipikus rendszerelemeit, valamint meghatározom ezek jellegzetes funkcióit.

4.1.1 Objektumok

- Olyan műszaki felügyeletet, ellenőrzést és karbantartást igénylő műszaki berendezések, amelyeken típusuktól vagy funkciójuktól függően adott ϑ időintervallum alatt $\kappa_{k\ (k=1..p)}$ számú felülvizsgálat elvégzése előírt, ahol p a vizsgálat, illetve az ellenőrzés alá vont műszaki berendezések száma.
- A vizsgálatok tetszőleges (nem rövid) időközönként elvégezhetők a megadott ϑ időintervallumon belül.
- Az előírt objektumonkénti κ_k számú felülvizsgálatot kötelezően el kell végezni. Ez egyes rendszerekben törvényi előírás is lehet.
- Hiba felmerülése, vagy vélhető felmerülése esetén, az objektumokon eseti karbantartási feladatokat is el kell végezni.

4.1.2 Szakértők

A műszaki szakértők a rendszer objektumain ϑ időintervallum alatt előírt, objektumonként κ_k számú felülvizsgálatot, illetve az objektumok meghibásodása miatti ε_k számú karbantartási feladatot végzik.

Feladatuk a rendszer típusától függően lehet:

- csak műszaki felügyeleti, ellenőrző vizsgálat,
- kisebb karbantartási feladatok elvégzése,
- nagyobb karbantartási feladatok felügyelete, végrehajtásuk jóváhagyása.

4.1.3 Karbantartó üzemek

A minden objektumnál jelentkező karbantartási, illetve felügyeleti, vagy vizsgálati igények nagy részét a szakértők a helyszínen végzik, esetlegesen a karbantartási üzem szakembereivel együtt. A karbantartási üzemekben a helyszínen el nem végezhető műveleteket hajtják végre. Karbantartó üzemek igénye elsődlegesen nagyméretű alkatrészek javításánál, bonyolult vagy hosszúidejű műveletek esetén merül fel.

A karbantartó üzemek feladatai:

- a beszállított berendezések, vagy alkatrészek javítása,
- a beszállított berendezések, vagy alkatrészek felújítása,
- a beszállított alkatrészek újrahasznosítása,
- a felújított alkatrészek esetleges értékesítése.

A karbantartó üzem a rendszerben anyagmozgatási tevékenységet nem végez. Az anyagmozgatási tevékenységet a külső logisztikai szolgáltató vállalat, vagy a logisztikai szolgáltató központ esetenként akár maguk a szakértők végzik.

4.1.4 Raktárak

A raktárak feladatai:

- pótalkatrészek raktározása,
- felújított alkatrészek raktározása,
- saját tulajdonú mobil anyagmozgatási eszközök raktározása,
- a karbantartáshoz szükséges kis- és nagyméretű szerszámok, kellékanyagok raktározása.

A raktárakban a költséghatékony működést szem előtt tartva az anyagokból, illetve építőelemekből olyan készletszint legyen, amely optimális mértékű, figyelembe véve a

karbantartási idő alatti szolgáltatási- illetve termelési kieséséből származó veszteségeket, valamint a lekötött tőke mértékét.

4.1.5 Beszállítók

A beszállítók feladatai:

- alkatrészek előállítása,
- eszközök, kellékanyagok előállítása,

A beszállításhoz kapcsolódó anyagmozgatási tevékenységet végezheti:

- a beszállító,
- a külső logisztikai vállalat.

4.1.6 Logisztikai szolgáltatók

A külső logisztikai szolgáltatók feladata a rendszer szállítási, raktározási, vagy egyéb komplex logisztikai feladatainak ellátása. Ide tartozik például az alkatrészek JIT beszállítása. Tevékenységük szerint megkülönböztethetünk:

- logisztikai szolgáltató vállalatot, amennyiben csak szállítási, vagy csak raktározási feladatot végez,
- logisztikai szolgáltató központot, amennyiben szállítási és raktározási feladatot is ellát, valamint komplex szolgáltatásokat nyújt.

A logisztikai szolgáltatók szállítási feladatai:

- alkatrészek, szerszámok kiszállítása a raktárakból az objektumokhoz,
- a beszerzett anyagok beszállítása a raktárakba,
- a beszerzett anyagok beszállítása a karbantartó üzemekbe,
- a karbantartó üzemekből a felújított alkatrészek beszállítása a raktárakba.

Raktározási feladatok ellátása esetén feladataik bővülnek a 4.1.4 pontban leírtakkal, mivel ilyenkor a hálózatban raktárként funkcionálnak. Alkalmazásuk azonban nem zárja ki a helyszínhez közel telepített saját raktárak üzemeltetését.

A logisztikai szolgáltató központ az anyagmozgatási és raktározási vagy akár a lokális diszponálási szolgáltatásokat együttesen végezheti komplex szolgáltatást nyújtva. Logisztikai szolgáltató központ kialakítása vagy bevonása a hálózatba általában valamilyen regionális szinten történik. Mivel szolgáltatásaik széleskörűek akár teljesen kiválthatják az adott regionális szinten a raktárak és a logisztikai szolgáltató vállalatok

szerepét. Regionális szinten akár a virtuális logisztikai központ egyes irányítási, diszponálási feladatait is elláthatja.

4.1.7 Virtuális logisztikai központ

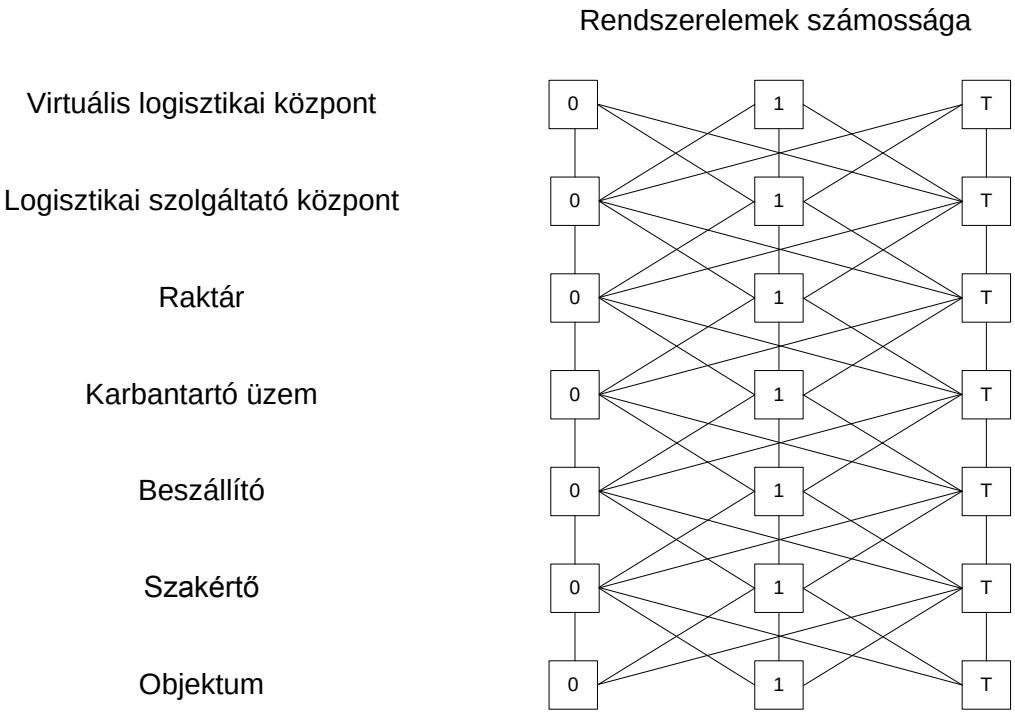
A virtuális logisztikai központ (VLK) a rendszer magja, központi eleme.

A virtuális logisztikai központ a rendszer működését koordinálja, elvégzi:

- megválasztja, meghatározza, és egymáshoz rendeli a szükséges logisztikai erőforrásokat,
- meghatározza, összegyűjti és feldolgozza a logisztikai hálózat elemeinek megválasztásához, egymáshoz rendeléséhez szükséges adatokat,
- működteti a rendszer elemeit összekötő információs hálózatot,
- kialakítja a megválasztott elemekhez illeszkedően az irányítási struktúrát, működteti a logisztikai hálózatot,
- meghatározza, illetve folyamatosan gyűjti, feldolgozza és tárolja a logisztikai hálózat belső és külső körében lévő azon adatokat, információkat, amelyek a működés ellenőrzéséhez szükségesek,
- a hálózat működésének hatékonysága érdekében vizsgálja, mely tevékenységek ellátása nem gazdaságos, így átszervezés, vagy a tevékenység kiszervezése (outsourcing) válhat szükségessé, valamint vizsgálja a decentrumok kialakításának lehetőségét.

5 Műszaki felügyeleti és karbantartó rendszerek jellegzetes rendszerváltozatai

Az 1. ábrán látható műszaki felügyeleti és karbantartó rendszer általános struktúrája alapján különféle rendszerváltozatok képezhetők, amelyek blokksémáit az 4-9. ábrák mutatják be.



4. ábra Rendszerváltozatok képzése

Ahol a rendszerelemek számossága lehet:

- 0: ekkor ez az elem nem található meg a rendszerben,
- 1: az elemből egy található a rendszerben,
- T: azaz több, ezen belül egyes rendszerekben a szakértők és az objektumok számossága lehet:
    - kevés: számosságuk a többi rendszerelem számosságához mérhető, de egynél nagyobb,
    - sok: a rendszer teljes számosságának a legnagyobb hányadát ezen elemek adják.

A 4. ábra alapján levezethetők jellegzetes változatok, mint az

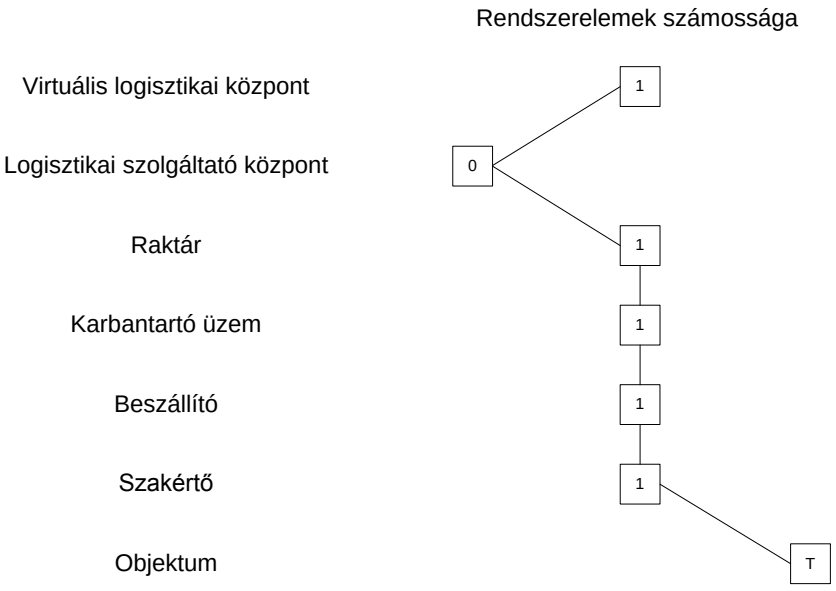
- ***országos felvonó felügyeleti és karbantartó rendszer,***

5. ábra Országos felvonó karbantartó és felügyeleti rendszer

jellemzője:

- kiterjedése országos méretű,
- több területi decentrum alkalmazása,
- országosan több raktár, beszállító, karbantartó és felújító üzem,
- több száz szakértő, valamint
- több ezer objektum van jelen a rendszerben.

- ***országos ipari-robot karbantartó rendszer***,

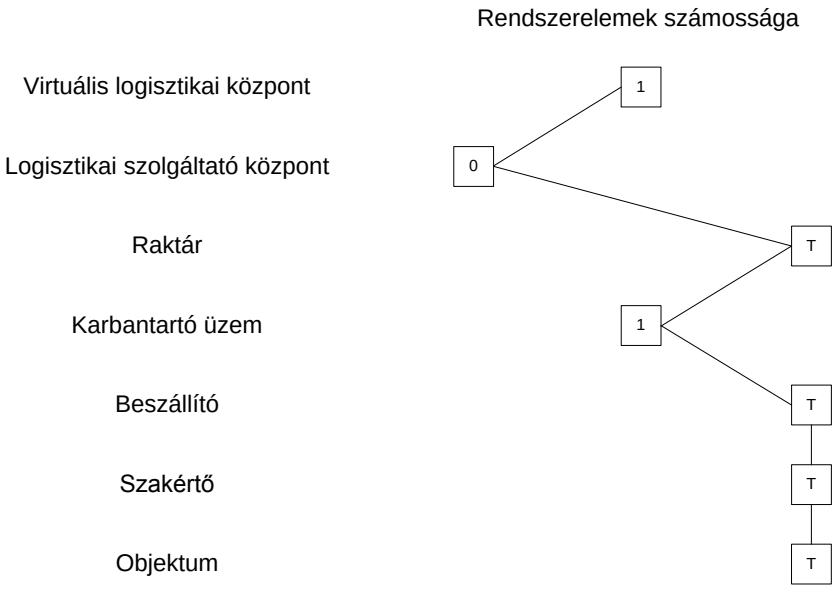


6. ábra Országos kiterjedésű ipari robot karbantartó rendszer

jellemzője:

- országos kiterjedés,
- az országban egyetlen centrummal rendelkezik, amely egyben raktárként is funkcionál,
- az alkatrészeket külföldről, az anyacégtől szerzik be,
- egyetlen szakértő, valamint
- néhány száz objektum.

- ***országos gázátadó állomás felügyeleti és karbantartó rendszere***,

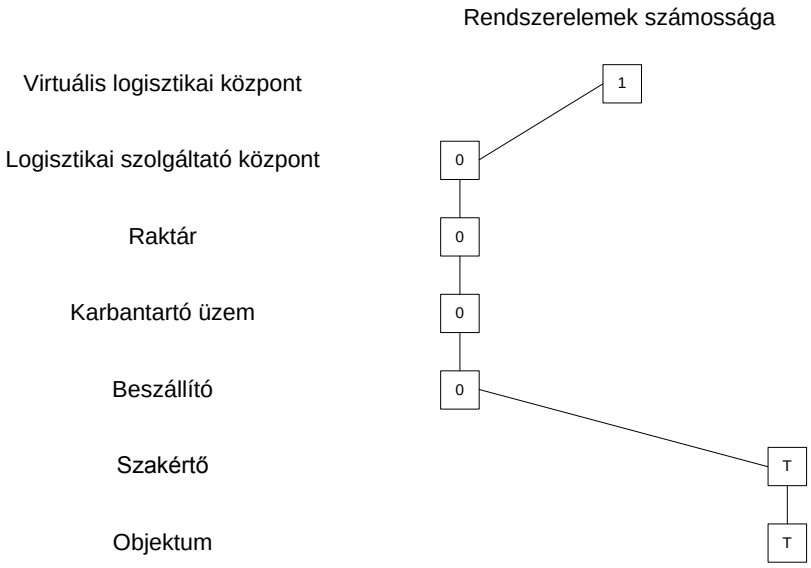


7. ábra Országos gázátadó állomás felügyeleti és karbantartó rendszere

jellemzője:

- országos kiterjedés,
- több országosan elosztott raktár,
- egy központi karbantartó üzem,
- több beszállító,
- néhány tíz szakértő alkalmazása, valamint
- néhány száz objektum.

- ***nemzetközi kötélpálya specialisták szakértői szervezete*** (O.I.T.A.F),



8. ábra Európai kötélpálya specialisták szervezete

jellemzője:

- egy virtuális centrummal rendelkezik,
- a szervezet nem rendelkezik raktárakkal, karbantartó üzemekkel, beszállítókkal,
- néhány tíz szakértő alkalmazása, akiknek feladatuk:
- néhány száz objektum ellenőrzése, valamint felülvizsgálata az üzemeltetőkkel együttműködve,

- ***világméretű tengeri fúrótorony felügyeleti és karbantartó rendszer***.

9. ábra Tengeri fúrótornyok felügyeleti és karbantartási rendszere

jellemzője:

- világméretű rendszer,
- a világon szétszórva több virtuális centrummal működik, amelyeket összefog egy virtuális centrum,
- a rendszer óriási méretéből adódóan több raktárt, karbantartó üzemet, valamint beszállítót tartalmaz,
- néhány tíz szakértő, valamint néhány száz objektum.

A továbbiakban a felvonó karbantartó és felügyeleti rendszert vizsgálom meg a dolgozatomban. Mivel ez a rendszer rendelkezik a műszaki felügyeleti és karbantartási rendszerek minden jellegzetességével, így a feltárt összefüggések általánosíthatók, más rendszerekre változatlan formában vagy egyszerűsítésekkel alkalmazhatók.

6 A műszaki felügyeleti és karbantartó rendszer matematikai leírása

6.1 Rendszerparaméterek, célfüggvények és korlátozások meghatározása

6.1.1 Rendszerszintű paraméterek

A rendszerszintű paraméterek közé tartozik a rendszer útmátrixa, amely megmutatja az egyes rendszerelemek távolságát egy másik rendszerelemtől. A rendszer útmátrixa egy több partícióból felépülő négyzetes mátrix, mindig annyi partícióval amennyi rendszerelemtípus szerepel az adott rendszerben.

A rendszer útmátrixa általánosított alakban:

$$L = \begin{array}{c} \\ 1 \\ i \\ q \end{array} \begin{array}{c} \begin{array}{ccc} 1 & j & q \end{array} \\ \begin{bmatrix} & & \\ & l_{ij} & \\ & & \end{bmatrix} \end{array} \tag{1}$$

Ahol:

- a mátrix mérete: $q = p + s + ka + ra + be + lo$,
- p: az objektumok száma,
- s: a szakértők száma,
- ka: a karbantartó üzemek száma,
- ra: a raktárak száma,
- be: a beszállítók száma,
- lo: a logisztikai szolgáltatók száma.

Az útmátrix a rendszerelemek szerint particionálható. Az egyes partíciók általában szimmetrikusak, de előfordulhat, hogy két rendszerelem között nem ugyanolyan hosszú utat kell bejárnunk mindkét irányban, például nagyvárosra kiterjedő rendszerekben az egyirányú utak megtörhetik a szimmetriát.

Az útmátrix partícionálható:

$$L = \left[L_{ij}(l_{mn})\right] \tag{2}$$

Az útmátrix partíciókra bontása az egyes rendszerelemek kapcsolatai szerint:

$$L = \begin{array}{c|c|c|c|c|c|c|} & \text{objektumok} & \text{szakértők} & \text{karbantartó üzemek} & \text{raktárak} & \text{beszállítók} & \text{logisztikai szolgáltatók} \\ \hline \text{objektumok} & L_{11} & L_{21} & L_{13} & L_{14} & L_{15} & L_{16} \\ \hline \text{szakértők} & L_{21} & L_{22} & L_{23} & L_{24} & L_{25} & L_{26} \\ \hline \text{karbantartó üzemek} & L_{31} & L_{32} & L_{33} & L_{34} & L_{35} & L_{36} \\ \hline \text{raktárak} & L_{41} & L_{42} & L_{43} & L_{44} & L_{45} & L_{46} \\ \hline \text{beszállítók} & L_{51} & L_{52} & L_{53} & L_{54} & L_{55} & L_{56} \\ \hline \text{logisztikai szolgáltatók} & L_{61} & L_{62} & L_{63} & L_{64} & L_{65} & L_{66} \\ \hline \end{array} \tag{3}$$

A rendszer kimenő paramétere a hozzárendelési mátrix, az útmátrixhoz hasonló felépítésű, megmutatja az egyes rendszerelemek kapcsolatát.

$$Y = \begin{array}{c} \\ 1 \\ i \\ q \end{array} \begin{array}{c} \begin{array}{ccc} 1 & j & q \end{array} \\ \left[\begin{array}{ccc} & & \\ & y_{ij} & \\ & & \end{array}\right] \end{array} \tag{4}$$

Ahol:

- $y_{ij} = \begin{cases} 1 \\ 0 \end{cases}$ értéket vehet fel aszerint, hogy az egyes rendszerelemek egymáshoz vannak e rendelve (1) vagy sem (0),
- a mátrix mérete q.

A hozzárendelési mátrix az útmátrixhoz hasonlóan particionálható:

$$Y = \left[Y_{ij}(y_{mn})\right] \tag{5}$$

A hozzárendelési mátrix partíciós bontása:

$$Y = \begin{array}{c|c|c|c|c|c|c} & \text{objektumok} & \text{szakértők} & \text{karbantartó üzemek} & \text{raktárak} & \text{beszállítók} & \text{logisztikai szolgáltatók} \\ \hline \text{objektumok} & Y_{11} & Y_{12} & Y_{13} & Y_{14} & Y_{15} & Y_{16} \\ \hline \text{szakértők} & Y_{21} & Y_{22} & Y_{23} & Y_{24} & Y_{25} & Y_{26} \\ \hline \text{karbantartó üzemek} & Y_{31} & Y_{32} & Y_{33} & Y_{34} & Y_{35} & Y_{36} \\ \hline \text{raktárak} & Y_{41} & Y_{42} & Y_{43} & Y_{44} & Y_{45} & Y_{46} \\ \hline \text{beszállítók} & Y_{51} & Y_{52} & Y_{53} & Y_{54} & Y_{55} & Y_{56} \\ \hline \text{logisztikai szolgáltatók} & Y_{61} & Y_{62} & Y_{63} & Y_{64} & Y_{65} & Y_{66} \end{array} \tag{6}$$

Az y_{ij} meghatározása képezi a rendszerben azt a hozzárendelési feladatot, amely az előírt célfüggvények adott feltételek melletti optimalizálásával oldható meg.

A hozzárendelési mátrix partíciói és funkcióik:

- Y_{11}: felhasználható az objektumok klaszter képzésére,
- $Y_{12} = Y_{21}^T$: objektumok és szakértők egymáshoz rendelése,
- $Y_{22} = Y_{33} = Y_{44} = Y_{55} = Y_{66}$: a legtöbb esetben nem értelmezhető, kivéve a raktár-raktár kapcsolat, a modellben nem értelmezett.
- $Y_{31} = Y_{13}^T$: karbantartó üzemek és objektumok egymáshoz rendelése,
- $Y_{41} = Y_{14}^T$: raktárak és objektumok egymáshoz rendelése,

- $Y_{51} = Y_{15}^T = 0$: beszállítók és objektumok egymáshoz rendelése, a modellben nem értelmezett,
- $Y_{61} = Y_{16}^T = 0$: logisztikai szolgáltatók és objektumok egymáshoz rendelése, a modellben nem értelmezett,
- $Y_{32} = Y_{23}^T = 0$: karbantartó üzemek és szakértők egymáshoz rendelése, a modellben nem értelmezett,
- $Y_{42} = Y_{24}^T = 0$: raktárak és szakértők egymáshoz rendelése, a modellben nem értelmezett,
- $Y_{52} = Y_{25}^T = 0$: beszállítók és szakértők egymáshoz rendelése, a modellben nem értelmezett,
- $Y_{62} = Y_{26}^T = 0$: logisztikai szolgáltatók és szakértők egymáshoz hozzárendelése, a modellben nem értelmezett,
- $Y_{43} = Y_{34}^T$: raktárak és karbantartó üzemek egymáshoz rendelése,
- $Y_{53} = Y_{35}^T$: beszállítók és karbantartó üzemek egymáshoz rendelése,
- $Y_{63} = Y_{36}^T$: logisztikai szolgáltatók és karbantartó üzemek egymáshoz rendelése,
- $Y_{54} = Y_{45}^T$: beszállítók és raktárak egymáshoz rendelése,
- $Y_{64} = Y_{46}^T$: logisztikai szolgáltatók és raktárak egymáshoz rendelése,
- $Y_{65} = Y_{56}^T$: logisztikai szolgáltatók, beszállítók egymáshoz rendelése.

6.1.2 Objektumok

Az objektumok fő paraméterei:

- p: az objektumok száma, lehet statikus vagy dinamikusan változó,
- ***L*** mátrix: az objektumok lokációja, kapcsolata más rendszerelemekkel, amely az útmátrixszal és a hozzárendelési mátrixszal jellemezhető (6) felhasználásával kiszámítható az egyes objektumok felkeresésére, illetve a szakértő sebességének ismeretében az egyes objektumok közötti távolság megtételére fordított idő,
- $\kappa_{k\ (k=1..p)}$ a kötelezően előírt vizsgálatok száma objektumonként,
- $MTBF_k$: a *k*-adik objektumnál a meghibásodások között eltelt átlagos idő (Mean Time Between Failures)

- $\varepsilon_{k\ (k=1..p)}$: eseti karbantartási feladatok száma ϑ időszak alatt, amely leszármaztatható az intervallumból és az objektum MTBF értékéből,

  ahol:

$$\left.\begin{aligned}\varepsilon_k &= \frac{\vartheta}{MTBF_k}\\ \varepsilon &= [\varepsilon_k]_{k=1..p}\end{aligned}\right\}, \tag{7}$$

  ahol:

  - $\tau_k^{K}{}_{\ (K=1..p)}$ a *k*-adik objektumon elvégzett eseti karbantartás, műszaki felülvizsgálat átlagos ideje a k-adik objektumon.

A felülvizsgálatok, karbantartások számát egyes berendezéseknél biztonsági megfontolások valamint az emberi élet védelme miatt akár törvény is előírhatja.

A vizsgálatok viszont nem történhetnek egymás után tetszőlegesen rövid időközönként, minden objektumra definiálni kell egy minimális, amelynél rövidebb időn belül a következő felülvizsgálat nem végezhető el:

$$\tau^m = [\tau_k^m]_{k=1..p}. \tag{8}$$

A felülvizsgálatok időközére megadható a

$$\tau_k^m \cdot (\varepsilon_k - 1) \leq \vartheta, \tag{9}$$

korlátozó feltétel.

Modellünkben - mint általában a valóságban is - az egyes objektumoknál történő felügyeleti, karbantartási feladatokat mindig ugyanaz a szakértő végzi, így a karbantartási feladatok során szerzett az adott berendezésre jellemző speciális tapasztalatok hasznosíthatók, ami a karbantartási idők rövidülését eredményezheti.

6.1.3 Szakértők

A szakértők matematikai leírásához szükséges paraméterek:

- s: a szakértők száma, amely a modellben konstans, nem változik. Dinamikus modell esetén a szakértőszám változhat:
  - szakértő kilépése,
  - rendszer bővítés esetén új szakértő bevonása,
  - objektumok számának csökkenése esetén szakértő elbocsátása.

- $\boldsymbol{L}$ mátrix: definiálja a szakértők állomáshelyét, illetve megadja a szakértők távolságát a rendszer többi elemétől,
- $\bar{v}$: a szakértő átlagos sebessége, állandónak tekintjük minden szakértőre.

A szakértő teljesítményének meghatározásakor figyelembe vehetjük a szakértő sebességét (6.1.3 fejezet). A modell dinamizálása esetén a szakértők teljesítményébe beépíthető a szakértők eltérő sebessége. Nagyobb átlagsebesség esetén a szakértő több objektumot tud megvizsgálni, így teljesítménye is nagyobb lehet, viszont figyelembe kell venni, hogy lehetnek akár hatósági szabályozások is, amely korlátok nem léphetők át.

Ezekből a paraméterekből származtatható az adott *k*-adik objektum felkeresési idejének összefüggése az *i*-edik szakértőnél:

$$\tau_k^f = \frac{l_{p+i,k}}{\bar{v}}, \tag{10}$$

ahol:

- $l_{p+i,k}$: az *i*-edik szakértő és a *k*-adik objektum közötti úthossz.

Adott *k*-adik és *l*-edik objektum közötti út megtételéhez szükséges idő:

$$\tau_{i,j} = \frac{l_{k,l}}{\bar{v}}, \tag{11}$$

ahol:

- $l_{k,l}$: a *k*-adik és a *l*-edik objektum közötti úthossz,

valamint:

- P_i: az *i*-edik szakértő teljesítménye, az értéke megmutatja hány felügyeleti, karbantartási feladatot végez a szakértő.

Korlátozó feltételek:

A szakértő teljesítménye (P_i : az *i*-edik szakértő teljesítménye) az:

- előírt minimum ($P_{i\ min}$) és
- előírt maximum ($P_{i\ max}$)

érték között változhat, ez lehet globálisan meghatározott korlát vagy egyedileg megállapított korlát is.

$$P_{i\,min} < P_i \; < P_{i\,max}, \tag{12}$$

ahol:

$$P_i \; = \sum_{j=1}^{p} \left(Y_{12\,i,j} \cdot \varepsilon_j \; \right). \tag{13}$$

Korlátot szab az egy ciklus (t) alatt felkeresendő objektumok vizsgálatára és a felkeresésre fordított idő összege is. Bármely adott szakértőnél az adott ciklusban az objektumok bejárásának és felülvizsgálatának, valamint a telephelyre való visszatérés idejének kisebbnek kell lennie, mint a ciklusidő, bármely tetszőleges bejárás esetén, például:

$$\tau^t \; = \tau_{0,1}^f + \tau_1^k + \sum_{i=2}^{c^t} \left(\tau_i^k + \tau_{i-1,i} \right) + \tau_{z,0}^f < \tau^T, \tag{14}$$

ahol:

- c^t : a t-edik ciklusban felkeresendő objektumok száma,
- $\tau_{0,1}^f$: az első objektum felkeresési ideje,
- $\tau_{z,0}^f$: visszatérés az utolsó objektumtól a szakértő bázisállomására,
- τ_i^k: az *i*-edik objektum felülvizsgálatának, karbantartásának átlagos ideje,
- τ^t : az az időintervallum, amelyben az adott szakértő állomáshelyéről elindul, vizsgálatokat végez, majd oda visszatér,
- τ^T : egy ciklus ideje, ez az országos vagy regionális rendszereknél lehet egy műszak, 8 vagy 16 óra, vagy 1 nap, nagyobb kiterjedésű rendszereknél az időfelbontás azonban ettől eltérő is lehet:

$$\sum_{t=1}^{T} \tau_t^T = \vartheta, \tag{15}$$

ahol:

- T: a ciklusok száma a ϑ időintervallumban,
- τ^T egy ciklus ideje.

Definiálható a felkeresendő objektumok halmaza az *i*-edik szakértőnél:

$$\left.\begin{matrix} O_i := \left\{ o_k \mid Y_{12\,s,k} = 1; k = 1..p \right\}, \\ \left| O_i \right| = P_i \ , \end{matrix}\right\}, \tag{16}$$

valamint részhalmazai, az egy ciklus (t) alatt felkeresendő objektumok halmaza.

$$O_i^t \subseteq O_i \ , \tag{17}$$

Definiálható O_i^S: az *i*-edik szakértőhöz rendelt objektumok rendezett halmaza, amelyben a rendezési reláció:

$$o_k \in O_i; o_l \in O_i; o_k < o_l \; ahol \; t_{o_k} < t_{o_k}, \tag{18}$$

ahol:

- t_{o_k} az o_k vizsgálat ciklusának időpontja,

  t_{o_l} az o_l vizsgálat ciklusának időpontja,

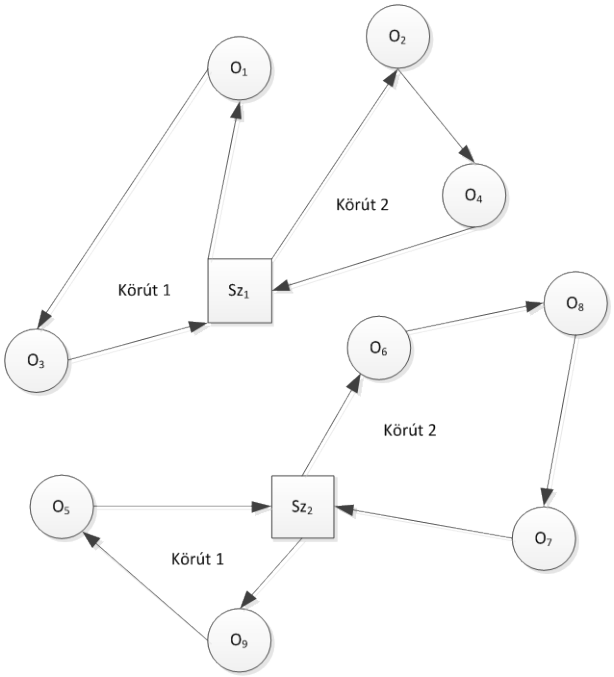
vagyis a halmaz a felkeresés ideje alapján rendezett.

$$\left.\begin{matrix} |O_i^t| = c_i^t \leq P_i \\ \bigcup_{t=1}^{T} O_i^t = O_i^S \end{matrix}\right\}, \tag{19}$$

illetve:

$$\bigcup_{i=1}^{s} O_i^S = O \ . \tag{20}$$

Mivel a szakértő egy objektumnál több vizsgálatot is végez (10-11. ábra) így egy objektum annyiszor kell, hogy szerepeljen a (17) összefüggésnél definiált halmazokban, ahány vizsgálatot kell végezni rajta.



10. ábra Több körutas rendszer objektumonként egy vizsgálattal

A vizsgálatok időközének meghatározásához a következő távolságfüggvény alkalmazható:

$$d\left(o_k; o_l \middle| o_k \in O_i^t; o_l \in O_j^t\right) = i - j, \tag{21}$$

tehát a (9) feltétel alapján:

$$min\left\{d\left(o_k; o_l \middle| o_k \in O_i^t; o_l \in O_j^t\right)\right\} \geq \tau_k^m. \tag{22}$$

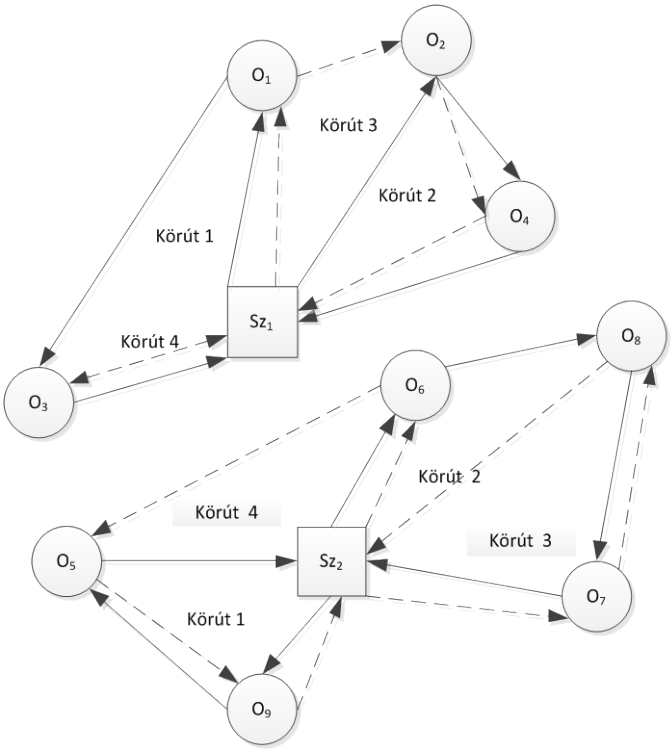
Így az *i*-edik szakértő által az egy ciklus (t) alatt az objektumok felkeresése miatt megtett út leírható az:

$$l_i^t = l_{0,o_i^t(1)} + \sum_{k=1}^{|o_i^t|-1} \left(l_{o_i^t(k),o_i^t(k+1)}\right) + l_{o_i^t(|o_i^t|),0} \tag{23}$$

összefüggéssel, az *i*-edik szakértő által a teljes időintervallumban megtett út pedig az:

$$l_i^T = \sum_{t=1}^{T} \left[l_{0,o_i^t(1)} + \sum_{k=1}^{|o_i^t|-1} \left(l_{o_i^t(k),o_i^t(k+1)}\right) + l_{o_i^t(|o_i^t|),0} \right] = \sum_{t=1}^{T} l_i^t \tag{24}$$

összefüggéssel.



11. ábra Több körutas rendszer objektumonként több vizsgálattal

A szakértők (s) által a megadott időintervallum (T) alatt felkeresett objektumokhoz kapcsolódó ráfordítások:

$$C^S = \left[\sum_{i=1}^{s}\left(\sum_{t=1}^{T} l_i^t\right)\right] \cdot c_u + \left[\sum_{i=1}^{s} P_i\right] \cdot c_v, \tag{25}$$

ahol a:

- c_u: az 1 kilométerre jutó fajlagos költség,
- c_v: az egy objektumra jutó fajlagos vizsgálati költség.

$$C^S \rightarrow min, \tag{26}$$

vagyis a ráfordítások legyenek minimálisak a korlátozó feltételek figyelembevételével.

6.1.4 Karbantartó üzemek

A karbantartó üzemek az alábbi paraméterekkel definiálhatók:

- ka: a karbantartó üzemek száma, amelyet statikus értéknek tekinthetünk,
- $\boldsymbol{L}$ mátrix: a karbantartó üzemek lokációja, távolsága a rendszer többi elemétől.

A karbantartó üzemek legtöbbször adottak, mint például:

- saját már meglévő karbantartó üzem esetén,
- olyan speciális kihelyezett karbantartó üzem, amelyből nem áll rendelkezésre akár az adott régióban akár globálisan másik üzem.

Ilyenkor lokációjának megválasztásakor nincs optimalizálásra lehetőség.

Amennyiben a karbantartó üzem lokációja nem adott, vagy választási lehetőségek állnak rendelkezésre, törekedni kell a logisztikai ráfordítások minimalizálására. A karbantartó üzemek kiválasztásánál figyelembe kell venni:

- logisztikai ráfordítások,
- minőségi kérdések,
- rendelkezésre álló erőforrások,
- időbeli átfutási, rugalmassági kérdéseket.

A karbantartó üzemekhez logisztikai ráfordítások megadhatók az:

$$C^K = C_{BK} + C_{KO} + C_{KR} \tag{27}$$

összefüggéssel, ahol:

- C_{BK}: beszálltás költsége a beszállítóktól a karbantartó üzemekbe:

$$C_{BK} = B_{BK} \cdot Y_{53} \cdot L_{53} \cdot c_{bk}, \tag{28}$$

ahol:

- c_{bk}: a beszállítás fajlagos költsége a beszállítóktól a karbantartó üzemekbe, beszállítónként eltérő lehet, jelen modellben konstansnak tekintjük,
- B_{BK} a beszállítások számának mátrixa

$$B_{BK} = \begin{matrix} \\ 1 \\ \\ b \end{matrix} \begin{matrix} 1 & & k \\ \multicolumn{3}{c}{\left[\begin{matrix} & & \\ & b_{bk} & \\ & & \end{matrix}\right]} \end{matrix} \tag{29}$$

ahol:

- b_{bk}: beszállítások száma a b-edik beszállítótól a k-adik karbantartó üzembe a ϑ intervallumban,

- C_{KO}: a karbantartó üzemek és az objektumok közötti szállítási költség

$$C_{BK} = B_{KO} \cdot Y_{31} \cdot L_{31} \cdot c_{ko}, \tag{30}$$

ahol:

- c_{ko}: a szállítás fajlagos költsége a karbantartó üzemek és az objektumok között,
- B_{KO} a beszállítások számának mátrixa

$$B_{KO} = \begin{matrix} \\ 1 \\ \\ k \end{matrix} \begin{matrix} \begin{matrix} 1 & & o \end{matrix} \\ \begin{bmatrix} & b_{ko} & \end{bmatrix} \end{matrix} \tag{31}$$

ahol:

- b_{bo}: beszállítások száma adott karbantartó üzemtől megadott objektumhoz a ϑ intervallumban,

- C_{KR}: a karbantartó üzemek valamint a raktárak közötti anyagmozgatás költsége.

$$C_{BK} = B_{KR} \cdot Y_{43} \cdot L_{43} \cdot c_{kr}, \tag{32}$$

- c_{kr}: a szállítás fajlagos költsége a karbantartó üzemek és a raktárak között

$$B_{KR} = \begin{matrix} \\ 1 \\ \\ k \end{matrix} \begin{matrix} \begin{matrix} 1 & & r \end{matrix} \\ \begin{bmatrix} & b_{kr} & \end{bmatrix} \end{matrix}, \tag{33}$$

ahol:

- b_{kr}: beszállítások száma adott karbantartó üzemtől adott raktárhoz a ϑ intervallumban,

$$C^K \rightarrow min, \tag{34}$$

korlátozó feltétel: a karbantartó üzemhez hozzárendelhető objektumok száma (P_m^{KO}) az

- előírt minimum ($P_{m\,min}^{KO}$) és
- előírt maximum ($P_{m\,max}^{KO}$)

érték között változhat.

6.1.5 Raktárak

A raktárakat definiálja:

- ra: a raktárak száma, amelyet statikus értéknek tekinthetünk,
- $\boldsymbol{L}$ mátrix: a raktárak lokációja, távolságok a rendszer többi elemétől.

A rendszerben lévő raktáraknál - hasonlóan a karbantartó üzemeknél leírtakhoz – ha már rendelkezésre áll saját raktár, akkor a raktár lokációjának megválasztásakor nincs optimalizálásra lehetőség, viszont meg lehet vizsgálni a raktározás gazdaságosságát, kihelyezésének hatásait.

Amennyiben a raktárak lokációja nem adott, saját raktárt vagy raktárakat építünk. Amennyiben választási lehetőségek állnak rendelkezésre, például kihelyezett raktárak választása vagy raktár építése esetén, törekedni kell a logisztikai ráfordítások minimalizálására. A raktárak kiválasztásánál figyelembe kell venni:

- logisztikai ráfordításokat,
- minőségi szempontokat,
- a raktár rendelkezésre álló erőforrásait,
- időbeli átfutási, rugalmassági feltételeket.

A raktárakhoz kapcsolódó logisztikai ráfordítások megadhatók az:

$$C^R = C_{BR} + C_{KR} + C_{RO} + C_R \tag{35}$$

összefüggéssel, ahol:

- C_{BR}: beszállítás költsége a beszállítóktól a raktárakba,

$$C_{BR} = B_{BR} \cdot Y_{54} \cdot L_{54} \cdot c_{br} \tag{36}$$

ahol:

- c_{br}: a szállítás fajlagos költsége a beszállítóktól a raktárakba,
- B_{BR}a beszállítások számának mátrixa,

$$B_{BR} = \begin{matrix} \\ 1 \\ b \end{matrix} \begin{matrix} \begin{matrix} 1 & & r \end{matrix} \\ \begin{bmatrix} & b_{br} & \end{bmatrix} \end{matrix}, \quad (37)$$

ahol:

- b_{br}: beszállítások száma adott beszállítótól adott raktárba a ϑ intervallumban,

- C_{KR}: beszállítás költsége a karbantartó üzemektől a raktárakba,
- C_{RO}: a raktárak és az objektumok közötti szállítások költsége,

$$C_{RO} = B_{RO} \cdot Y_{41} \cdot L_{41} \cdot c_{ro}, \quad (38)$$

ahol:

- c_{ro}: a szállítás fajlagos költsége a karbantartó üzemektől a raktárakba,
- B_{RO}a beszállítások számának mátrixa,

$$B_{RO} = \begin{matrix} \\ 1 \\ r \end{matrix} \begin{matrix} \begin{matrix} 1 & & o \end{matrix} \\ \begin{bmatrix} & b_{ro} & \end{bmatrix} \end{matrix}, \quad (39)$$

ahol:

- b_{ro}: beszállítások száma adott raktárból az adott objektumhoz a ϑ intervallumban,

- C_R: anyagok, alkatrészek raktározási költsége.

A célfüggvény a raktárakhoz kapcsolódó logisztikai ráfordítások minimalizálása:

$$C^R \rightarrow min, \quad (40)$$

korlátozó feltétel: a raktárakhoz hozzárendelhető objektumok száma (P_f^{RO}) az

- előírt minimum ($P_{f\ min}^{RO}$) és
- előírt maximum ($P_{f\ max}^{RO}$)

között változhat.

6.1.6 Beszállítók

A beszállítókat definiálja:

- be: a beszállítók száma, amely a modellben statikus érték,
- $\boldsymbol{L}$ mátrix: a beszállítók lokációja, távolságok a rendszer többi elemétől.

A beszállítók megfelelő megválasztása igen bonyolult sokparaméteres optimalizálási folyamat, amelyben a logisztikai ráfordítások mellett igen nagy szerepet játszik a

- beszállított termékek, alkatrészek ára,
- a beszállított termékek, alkatrészek minősége,
- mennyiségi kedvezmények,
- rugalmasság, határidők.

A logisztikai ráfordítások megadhatók:

$$C^B = C_{BR} + C_{BK} \tag{41}$$

összefüggéssel, ahol:

- C_{BR}: beszálltás költsége a beszállítóktól a raktárakba,
- C_{BK}: beszálltás költsége a beszállítóktól a karbantartó üzemekbe.

A célfüggvény a beszállítókhoz kapcsolódó logisztikai ráfordítások minimalizálása:

$$C^B \rightarrow min. \tag{42}$$

Korlátozó feltétel: a beszállítókhoz hozzárendelhető karbantartó üzemek száma (P_b^{BK}) az

- előírt minimum ($P_{b\,min}^{BK}$) és
- előírt maximum ($P_{b\,max}^{BK}$),

valamint a beszállítókhoz hozzárendelhető raktárak száma (P_b^{BR}) az

- előírt minimum ($P_{b\,min}^{BR}$) és
- előírt maximum ($P_{b\,max}^{BR}$),

értékek között változhat.

6.1.7 Logisztikai szolgáltatók

A logisztikai szolgáltatókat megadja az:

- lo: a logisztikai szolgáltatók száma, amely általában statikus érték,
- $\boldsymbol{L}$ mátrix: a logisztikai szolgáltatók lokációja, távolságuk a rendszer többi elemétől.

Ha a logisztikai szolgáltató raktározási feladatot végez, akkor a logisztikai ráfordítások azonosak a raktáraknál (6.1.5) leírtakkal.

Amennyiben a logisztikai szolgáltató szállítási feladatot végez, külön logisztikai költségek nem merülnek fel. A szolgáltató költségei a szállítási költségekben jelennek meg.

$$F(C_{BK} + C_{KO} + C_{KR} + C_{BR} + C_{BO} + C_{BS}) \tag{43}$$

6.2 A rendszer célfüggvényeinek összevetése

$$\left.\begin{array}{c} C^S \rightarrow min \\ C^K = C_{BK} + C_{KO} + C_{KR} \\ C^R = C_{BR} + C_{KR} + C_{RO} + C_R \\ C^B = C_{BR} + C_{BK} \end{array}\right\} \tag{44}$$

A rendszer általános célfüggvénye:

$$C^S + C^K + C^R + C^B \rightarrow \text{min}, \tag{45}$$

a redundánsan előforduló költségkomponensek redukálásával pedig:

$$C = C_{BK} + C_{KO} + C_{KR} + C_{BR} + C_{RO} + C_R \rightarrow min \tag{46}$$

célfüggvény adódik.

7 Multi-kromoszómás evolúciós programozási algoritmus a vizsgált rendszerben jelentkező szakértő - objektum relációk optimalizálására

7.1 Evolúciós programozás

Az evolúciós programozás az evolúciós algoritmusok családjába tartozó módszer, amelyeknek alapvetően négy fajtáját különböztethetjük meg:

- genetikus algoritmus,
- genetikus programozás,
- evolúciós programozás,
- evolúciós stratégia [87].

A felsorolt módszerek közös ismérve, hogy a biológiai evolúció alapján működnek, általában egyszerűsített modelljei annak.
Az evolúciós programozás során is egy adott probléma lehetséges megoldásaiból álló populációt kezelünk. Az evolúciós programozás során nincs megkötés a megoldások ábrázolási módjára, az eredeti feladat által meghatározott formában tároljuk a megoldásokat. Az evolúciós programozás során nincs szükség arra, hogy az egyedeket bitvektorrá vagy numerikus vektorrá kódoljuk [88], mint a genetikus algoritmusnál. Itt is egy véletlenül választott populációval indul az algoritmus, egy lépés során először az összes egyedről másolatot készítünk, majd a lemásolt egyedek mutáción esnek át. A mutáció különböző mértékű lehet, de bevett módszer, hogy a kisebb mutációknak nagyobb a valószínűsége, mint a nagy változást okozóknak. A módosult egyedek fitneszértékének kiszámítása után már csak annak az eldöntése van hátra, hogy az eredeti populáció és a megváltozott populáció mely egyedei kerülnek az új populációba.
Ez a kiválasztás többnyire egy bajnokság segítségével történik. A legegyszerűbb változata a bajnokságnak, ha véletlenül kiválasztunk 2 egyedet, és a nagyobb fitneszértékkel rendelkező kerül az új populációba. Ezt addig ismételjük, amíg fel nem töltődik az új populáció. Az evolúciós programozás során az esetek döntő részében nem alkalmaznak keresztezést, így biológiai szemszögből nézve ez olyan, mint amikor több faj fejlődését vizsgáljuk, hiszen a fajok között sincs kereszteződés. Tehát az egyedek itt egy-egy fajt képviselnek viszont a szakirodalomban az egységesség megőrzése érdekében a populáció elnevezés használatos [87].

Az evolúciós programozás technikáját főleg bonyolult sok-feltételes problémák megoldására használják, problématerület specifikus adatstruktúrákkal valamint funkciókkal. Az evolúciós programozás folyamata a következőképpen definiálható pszeudó kódban:

- kezdeti populáció inicializálása
- kezdeti populáció fitneszértékeinek meghatározása
- while kilépési feltétel nincs kielégítve
    - új populáció előállítása mutációs operátorokkal
    - fitneszérték kiszámítása
    - egyedek kiválasztása a következő populációba
- end while

7.2 A kidolgozott algoritmus feltételrendszerének, döntési változóinak rövid összefoglalása

A disszertációmban kidolgozott és alább ismertetésre kerülő algoritmus egy fázisban megoldja a körutakra bontott fix végpontú, több bázisállomású többszörös utazóügynök problémát a műszaki felügyeleti és karbantartó rendszereknél felmerülő speciális feltételrendszerek figyelembevételével. Ezek a következők:

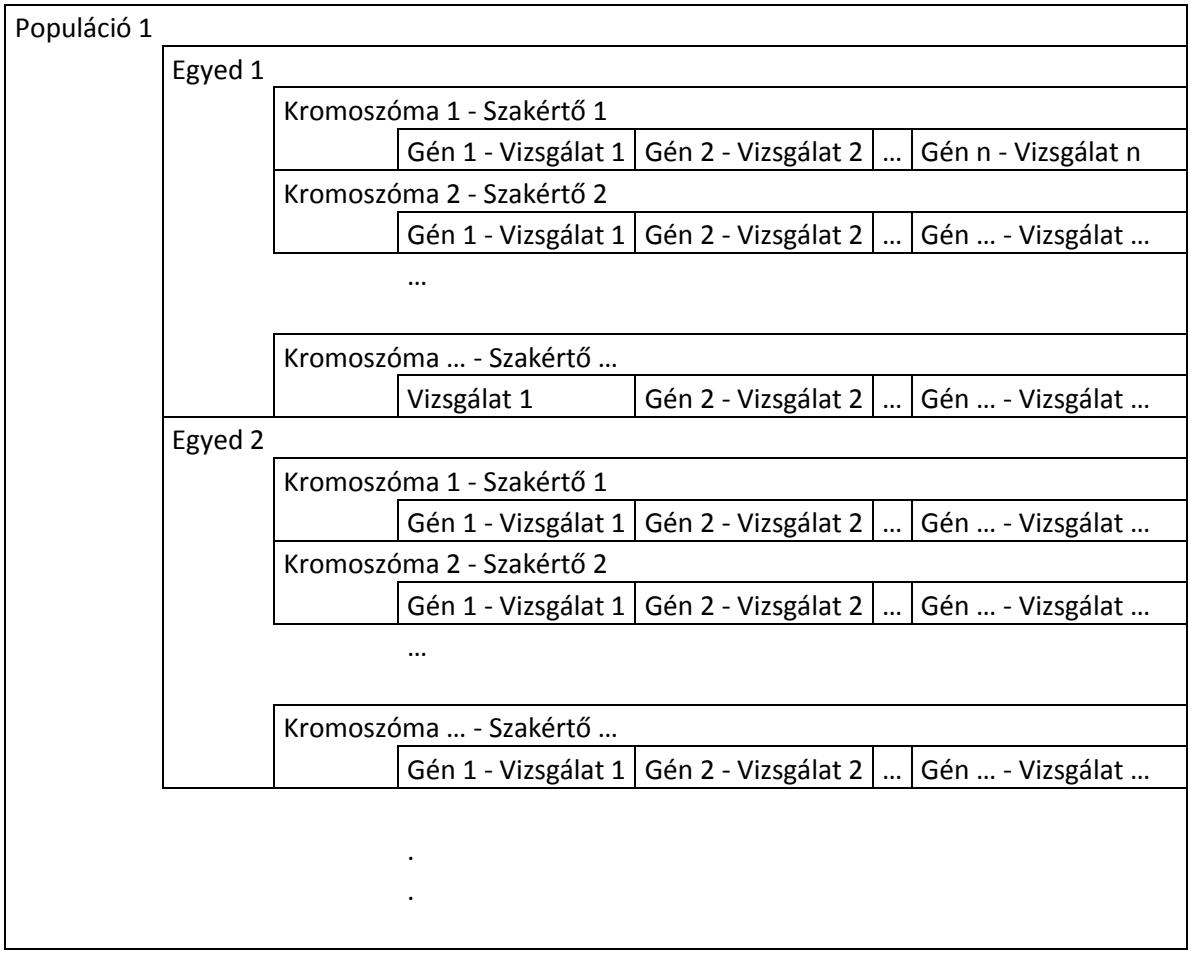
- az objektumok helye fix,
- az objektumokon több műveletet kell elvégezni, így egyes objektumokat többször kell felkeresni,
- egyes objektumokat mindig ugyanaz a szakértő keres fel, de a felügyeleti stratégia szerint ez a feltétel figyelmen kívül hagyható,
- a szakértők helye fix, de számuk meghatározása az optimalizálás tárgykörébe tartozik, mivel:
    - az egyes szakértők foglalkoztatása extra költséggel jár,
- az egyes szakértők:
    - minimális és maximális elvégezhető vizsgálatszáma definiált,
    - adott az egy ciklus alatt bejárható út hossza, valamint
    - az adott intervallumban lévő ciklusok száma.

A kidolgozott algoritmus a szakértők által megtett utat optimalizálja, amely konkrét esetekben költségkomponensekkel súlyozható.

7.3 Az algoritmus multi-kromoszómás struktúrája

A megadott feltételrendszer kielégítése érdekében az algoritmus adatstruktúrája a genetikus algoritmusok körében kevésbé alkalmazott [89] – elsősorban párhuzamos evolúciós, illetve neuro-evolúciós algoritmusoknál használt - multi-kromoszómás

struktúrán alapul. Ezek a technikák csak nemrégiben terjedtek el a genetikus módszerek körében [90]. Az adatstruktúra egy kaszkád rendszerrel írható le:



12. ábra Az optimalizálás kaszkád adatstruktúrája

A legnagyobb egység a populáció. Mindent iterációban (7.1) egy új populáció jön létre az előző populációra alapozva:

- mutációs operátorok,
- versengéses kiválasztás, valamint
- véletlenszerűen generált egyedek segítségével.

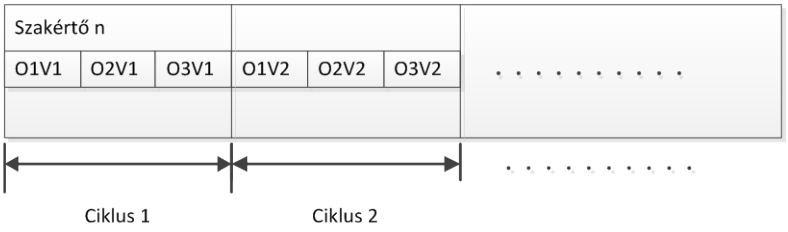
Egy populáció egyedeket tartalmaz. Az egyedek száma fontos paramétere az optimalizálásnak, az egyedek számától függ:

- az optimalizálás sebessége, mivel nagy egyedszámnál az algoritmus lassabb,
- az algoritmus konvergenciájának sebessége, nagyobb egyedszámnál több lehetséges megoldás kerül kiértékelésre,

- a lokális optimumok elkerülésének valószínűsége, nagyobb egyedszámnál nagyobb a mutációs valószínűség, valamint a véletlenszerűen előállított egyedek száma is nagyobb.

Nagyobb problémaméret esetén célszerű az egyedszámot is növelni, különben az algoritmus lokális optimumhoz konvergálhat.

A kromoszómákat az egyes szakértők jelképezik, mindig annyi kromoszóma van ahány szakértő. A vizsgálatok pedig az egyes kromoszómák génjei.



13. ábra Kromoszómastruktúra

A kromoszóma egyes génjei, ahogy az ábrán is látható az egyes objektumok vizsgálatait tartalmazzák: (O1 – 1. objektum, V1 – 1. vizsgálat). A génszekvenciák tovább csoportosíthatók a szakértő által egy ciklusban bejárható út alapján. Ahogy az ábra is mutatja a kromoszómák ábrázolásakor az egyes napi ciklusok csak logikailag különülnek el, mivel így a mutációs operátorok jelentősen egyszerűsödnek, ezáltal az algoritmus sebessége nő és nagyobb problémaméret kezelhető.

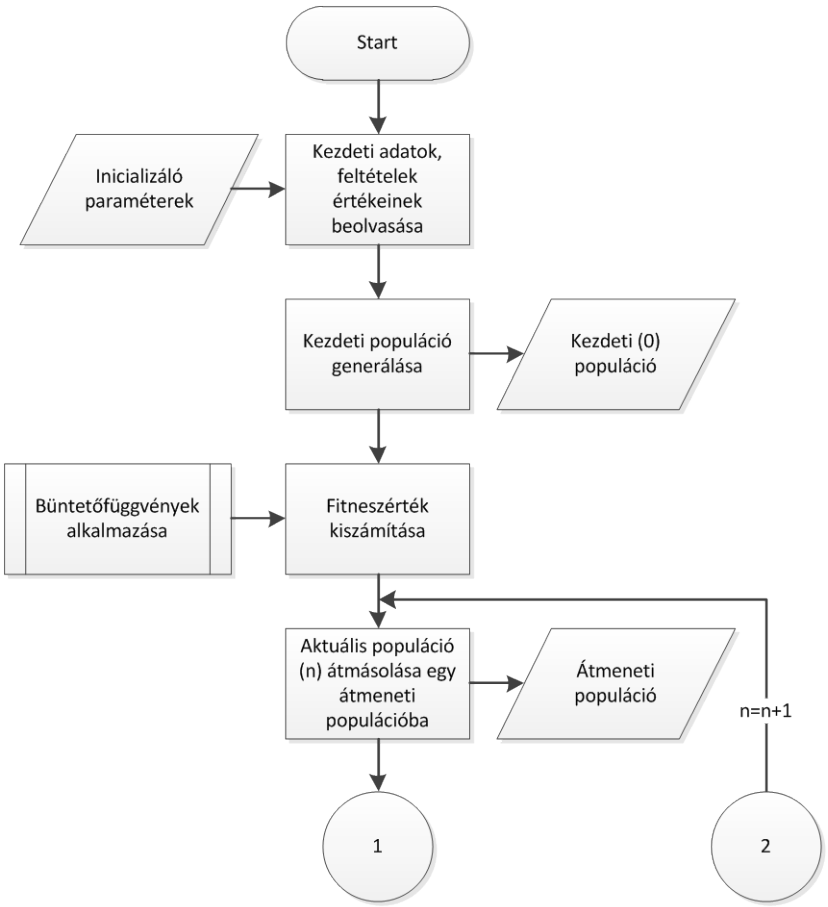
7.4 Az algoritmus működése

A kidolgozott algoritmus az általános evolúciós programozási algoritmus struktúráját követi [87]. Első lépésként inicializálja az adatstruktúrákat a bemenő paraméterek alapján.

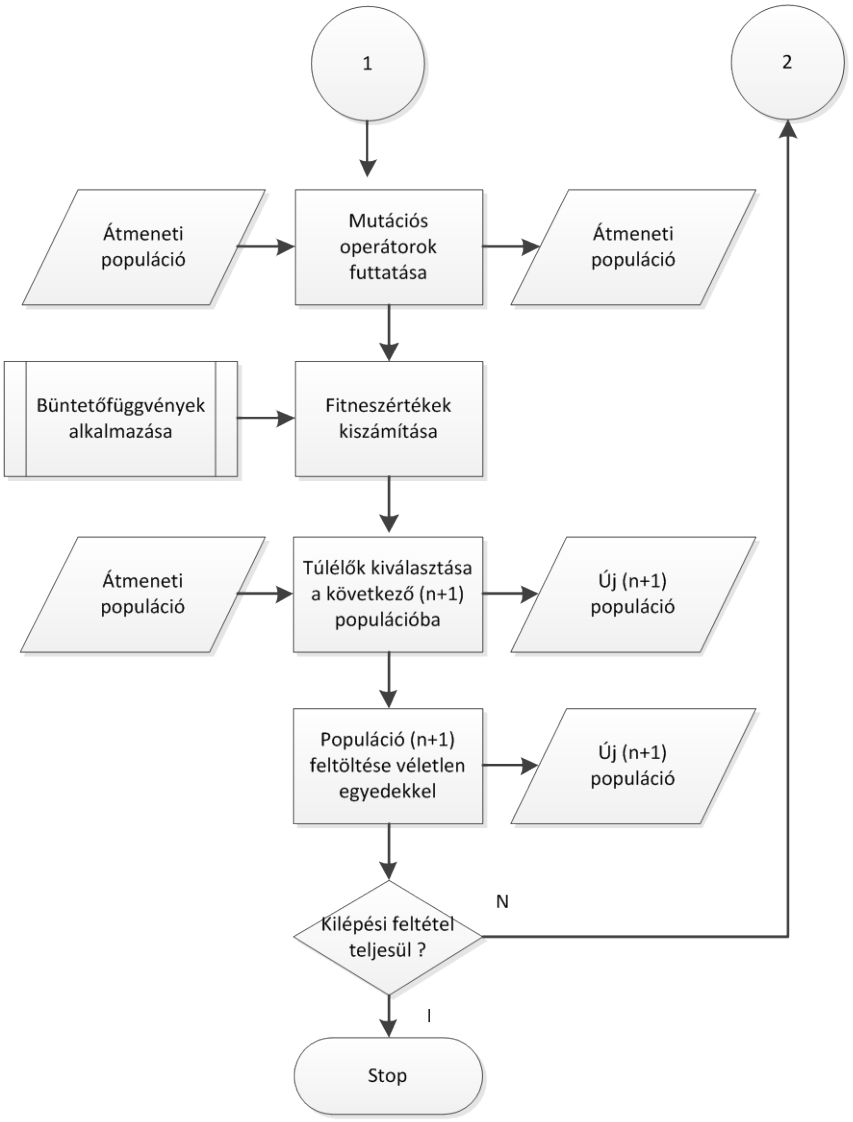
Az algoritmus paraméterei a következők:

- objektumok:
  - száma,
  - lokációja,
  - műszaki karbantartási, felügyeleti eseményeinek száma,
- a karbantartási, felügyeleti események minimális távolságának ciklusszáma
- szakértők:
  - száma,
  - lokációja,

  - minimális műveletszám,
  - maximális műveletszám,
  - egy ciklus alatt bejárható maximális út,
  - maximális ciklusok száma,
  - alkalmazásának költsége,
- minimális műveletszám alatti műveletek büntetőértéke,
- maximális műveletszám feletti műveletek büntetőértéke,
- maximális ciklusszám feletti ciklus büntetőértéke,
- szétszórtság (megadott objektum vizsgálatai nem ugyanazon szakértőhöz vannak hozzárendelve),
- vizsgálatok közelségének büntetőértéke,
- populáció méret,
- iteráció szám.

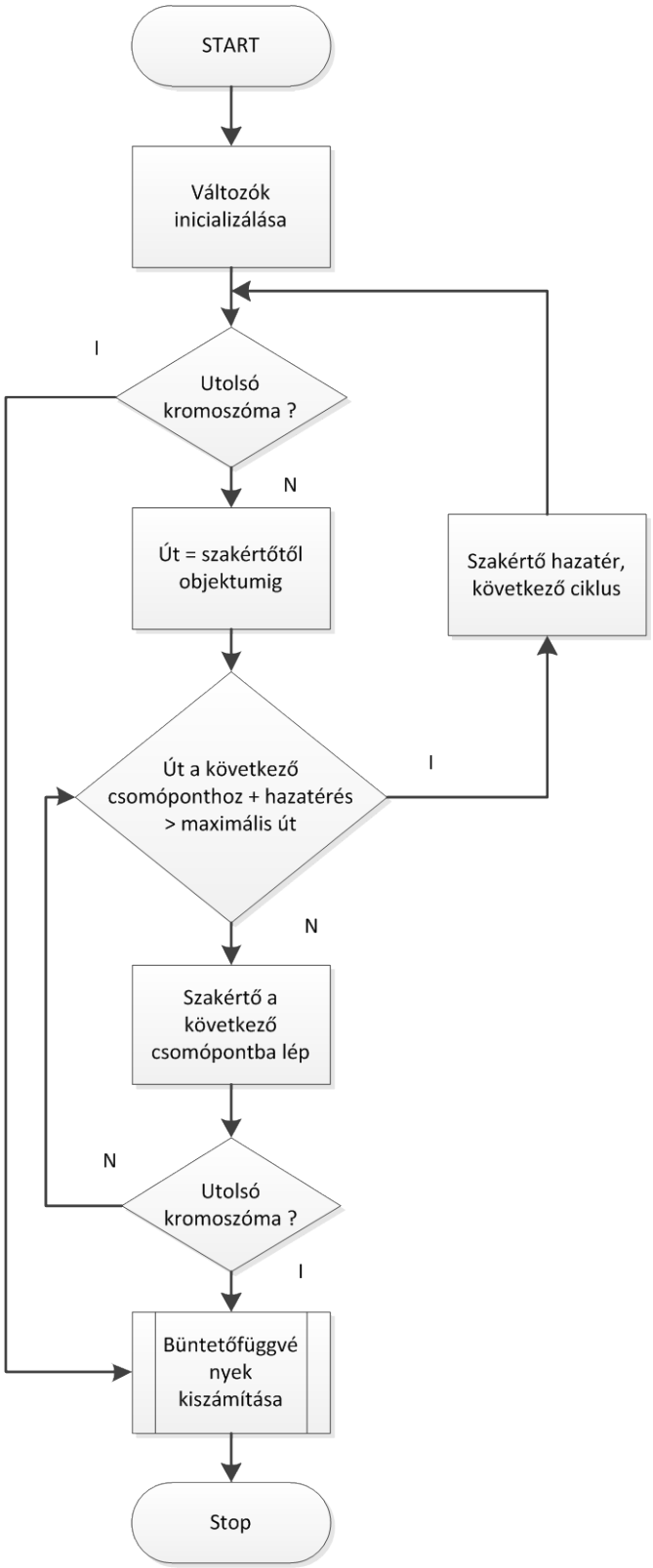


14. ábra Az algoritmus folyamatábrája (1)



15. ábra Az algoritmus folyamatábrája (2)

Az inicializálási fázis után feltölti a kezdeti populációt véletlenszerűen generált egyedekkel a megadott paraméterek alapján és kiszámítja az egyedek fitneszértékét, amelyet a büntetőfüggvények által megadott értékkel növel.



16. ábra Megtett út kiszámítása, ciklusonkénti bontás

7.5 Fitnesz kiszámítása

Az egyed fitneszének kiszámítása a következő formula alapján történik:

$$F = C_s + B_s + B_{CC} + B_{SC} + B_F + B_M + B_N , \quad (47)$$

ahol:

- F: az egyed fitneszértéke,
- C_s : a szakértők által megtett út költsége (25),
- B_s : szakértők alkalmazásának költsége (53),
- B_{CC}: szakértők ciklusidő túllépésének büntetőfüggvénye (48),
- B_{SC}: szétszórt objektumok büntetőfüggvénye (52),
- B_F : szakértő minimális terhelés alatt büntetőfüggvénye (49),
- B_M : szakértő maximális terhelés fölött büntetőfüggvénye (50),
- B_N : felügyeleti, karbantartási feladatok közeliségének büntetőfüggvénye (51).

A szakértő által megtett út költségének kiszámítási algoritmusa egyben megadja a szakértő által egy ciklusban megtehető utat is, biztosítja, hogy a szakértő egy ciklusban ne léphesse túl a (14) feltételt (16. ábra).

7.6 Büntetőfüggvények

A büntetőfüggvények alkalmazása a legegyszerűbb módja a megoldások jóságának értékelésére. Alkalmazásuk az erősen feltételes problémáknál [91] - mint esetünkben is – előnyös, mivel így az algoritmus megfelelően gyors lesz a nagyméretű problémák megoldásához. Nem szükséges az egyes elemek ellenőrzése, miszerint az egyedek megfelelnek-e a feltételeknek, hanem az egyedek jóságát a büntetőfüggvények határozzák meg. Az algoritmus kétféle büntetőfüggvényt használ:

- lokális büntetőfüggvények,
- globális büntetőfüggvények.

A büntetőfüggvények lokalitása vagy globalitása az egyes egyedekre és a szakértőkre, vagyis kromoszómákra vonatkozik. A lokális büntetőfüggvények egy kromoszóma adatain kerülnek végrehajtásra, míg a globális büntetőfüggvények több kromoszóma – akár az összes – adatait használják bemenő paraméterként, vagyis globálisan a teljes egyedre vonatkoznak.

A következő fejezetekben bemutatott büntetőfüggvények lineáris büntetést alkalmaznak, mivel az általánosan elterjedt négyzetes büntetőfüggvények nagyobb problémaméretek esetén átléphetik a változók ábrázolási tartományát. Ilyenkor a túlcsordulás miatt a megoldó algoritmus hamis eredményt ad. Mivel a probléma

nehezen észlelhető négyzetes büntetés használata adott problémaméreten csak tesztfuttatások adatainak elemzése után javasolt.

7.6.1 Lokális büntetőfüggvények

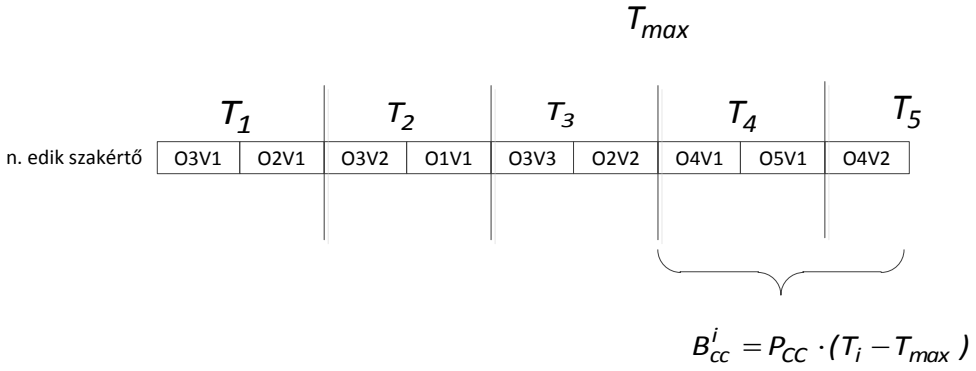
Ahogy a (47) egyenlet is mutatja, összesen négyféle lokális büntetőfüggvény kerül alkalmazásra:

- Ciklusidő túllépés: mikor a szakértő a megadott ciklusszámnál (például 365 nap) több ciklust fut be, egyes rendszerekben ez akár fizikai, technikai korlátokba ütközhet, amely akár a „halálbüntetés” módszerét is indokolhatja. Ilyenkor a büntetőfüggvény értéke végtelen.

$$B_{CC} = P_{CC} \cdot (T_i - T_{max}), \tag{48}$$

ahol:

- P_{CC}: a ciklustúllépés büntetőértéke, konstans,
- T_i : a szakértő által igényelt ciklusok száma,
- T_{max}: maximális ciklusszám (15).



17. ábra Ciklusidő túllépés büntetőfüggvényének működése

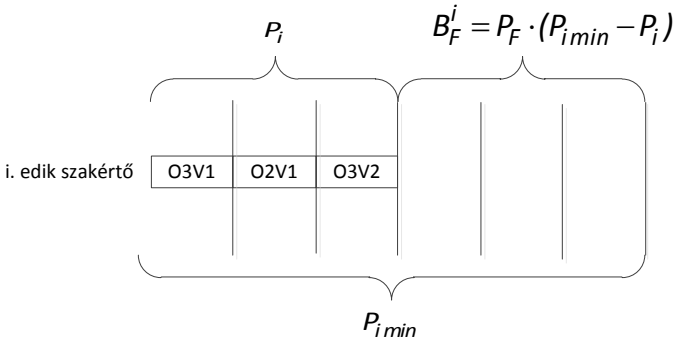
- Kevesebb vizsgálat: a szakértőknek teljesíteni kell egy minimális számú vizsgálatot, hogy alkalmazásuk gazdaságos legyen.

$$B_F = P_F \cdot (P_{i\,min} - P_i), \tag{49}$$

ahol:

- P_F: a megengedettnél kevesebb vizsgálat büntetőértéke, konstans,

- $P_{i\,min}$: szakértő minimális vizsgálatainak száma (12),
- P_i : szakértő vizsgálatainak száma.



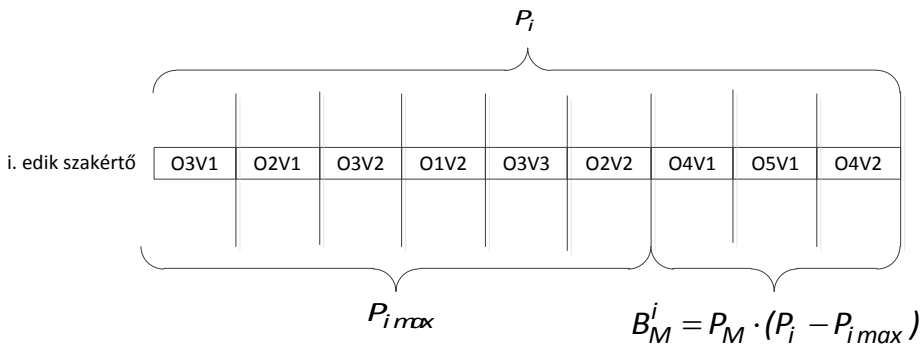
18. ábra A megengedettnél kevesebb vizsgálat büntetőfüggvényének működése

- Megengedettnél több vizsgálat: a szakértő a megengedettnél több vizsgálatot teljesít, ez egyes esetekben akár törvényileg is szabályozott lehet, amely már nem gazdaságossági kérdés, mint a minimális vizsgálat átlépése, így itt is indokolt lehet a halálbüntetés értéke.

$$B_M = P_M \cdot (P_i - P_{i\,max}), \tag{50}$$

ahol:

- P_M: a megengedettnél több vizsgálat büntetőértéke, konstans,
- $P_{i\,max}$: szakértő maximális vizsgálatainak száma (12),
- P_i : szakértő vizsgálatainak száma.



19. ábra A megengedettnél több vizsgálat büntetőfüggvényének működése

7.6.2 Globális büntetőfüggvények

A globális büntetőfüggvények a lokális büntetőfüggvények után kerülnek kiértékelésre. Ezek a függvények a teljes egyedre vonatkoznak. Kétféle globális büntetőfüggvény kerül bevezetésre:

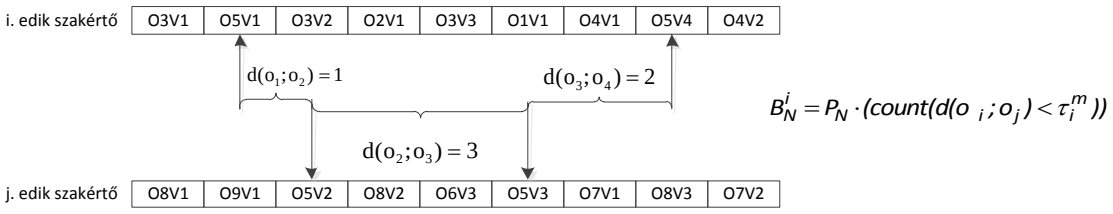
- A „közel” lévő vizsgálatok: a műszaki felügyeleti rendszerek bizonyos típusainál amennyiben egy objektumot többször kell vizsgálni, mint például felvonóvizsgáló rendszer esetén, előírható egy minimális időbeli távolság a két vizsgálat között (22).

$$B_N = P_N \cdot \left[count\left(d\left(o_i; o_j\right) < \tau_i^m\right)\right], \tag{51}$$

ahol:

- P_N: a megengedettnél közelebb lévő vizsgálat büntetőértéke, konstans,
- $d\left(o_i; o_j\right)$: egy objektum *i*-edik és *j*-edik vizsgálatának távolsága (21),
- τ_i^m: a vizsgálatok minimális távolsága,
- $count()$: függvény a feltétel teljesülésének számlálása.

A büntetőfüggvény tervezésekor figyelembe kellett venni, hogy az objektumok egyes vizsgálatai – amennyiben nem alkalmazzuk a szétszórtsági büntetőfüggvényt - szétszórtan helyezkedhetnek el.



20. ábra A közel lévő vizsgálatok büntetőfüggvényének működése

- szétszórtság: A szétszórtsági büntetőfüggvény alkalmazása két alapvető filozófia elkülönítését teszi lehetővé.
  - Egyes rendszerekben ugyanazon objektumot minden esetben ugyanazon szakértő lát el, így az idővel felhalmozott speciális tudás és tapasztalat jobban hasznosulhat,

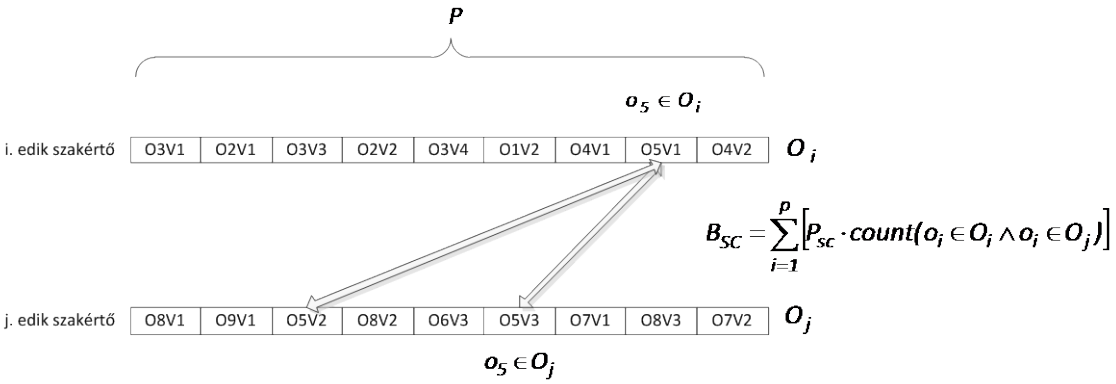
- míg más rendszerekben minden vizsgálatot más szakértő láthat el, amely új szemlélet, más megközelítések alkalmazásával alaposabb vizsgálatot eredményezhet.

Az alapmodell az első esetet feltételezi (6.1.2), de e büntetőfüggvény értékének megadásával az algoritmus képes második eset optimalizálására is. A (12), (13), (14) alapján

$$B_{SC} = \sum_{k=1}^{p} [P_{SC} \cdot count(o_k \in O_x \wedge o_i \in O_y)], \tag{52}$$

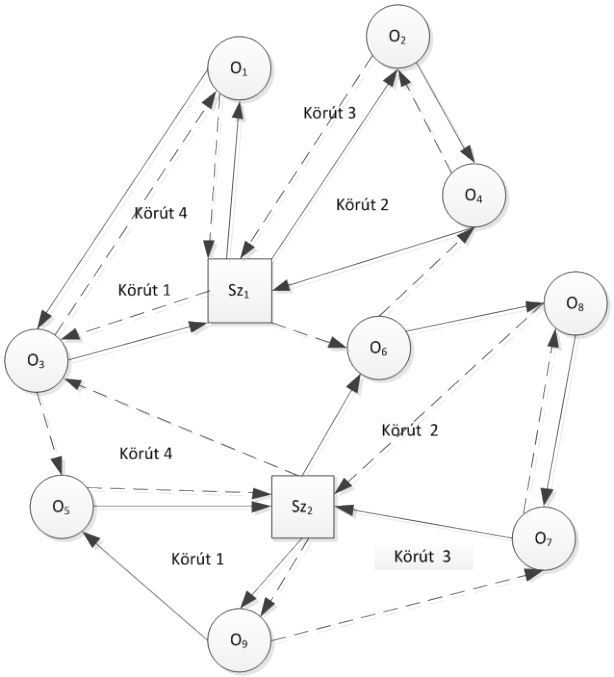
ahol:

- P_{SC} : a szétszórt objektumok büntetőértéke, konstans,
- p: az objektumok száma.



21. ábra A szétszórt vizsgálatok büntetőfüggvényének működése

Ha egy objektum nem csak egy szakértőhöz van hozzárendelve, a büntetőérték hozzáadódik a fitneszfüggvényhez. A büntetőfüggvény alkalmazásának célja, hogy egy objektum egy szakértőhöz lesz rendelve minden vizsgálatával (11. ábra). Ha a büntetőfüggvényt nem alkalmazzuk, akkor az algoritmus egy adott objektum vizsgálatait szétoszthatja a szakértők között, így egy objektum több szakértőhöz lesz hozzárendelve.



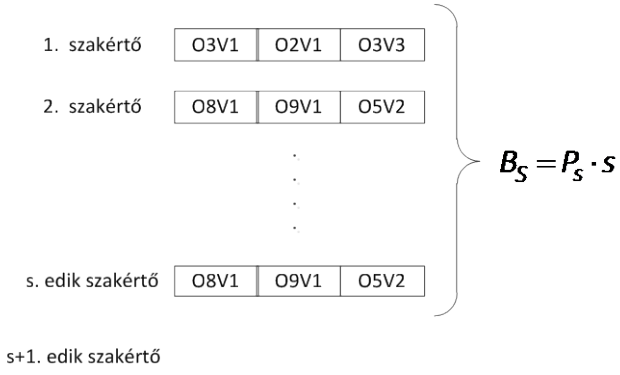
22. ábra Több körutas rendszer objektumonként több vizsgálattal, szétszórt vizsgálatokkal

- szakértők száma: a szakértők számának büntetőértéke az alkalmazott szakértők számával arányos, ahol a vizsgálatok száma nem nulla. Értéke azt mutatja, hogy egy új szakértő alkalmazása mekkora költséggel jár. A disszertációban kidolgozott modellben ez konstans érték, de az algoritmus képes kezelni az egyedi értékeket is. A büntetőérték:

$$B_S = count(P_i \neq 0|_{i=1..s}) * s_P, \tag{53}$$

ahol:

- s_P: azon szakértők száma, ahol van hozzárendelt objektum,
- P_s : szakértő alkalmazásának költsége.



23. ábra A szakértők száma büntetőfüggvényének működése

7.7 Mutációs operátorok

Az algoritmus a büntetőfüggvény értékeket is magába foglaló megnövelt fitneszértékek kiszámítása után belép a fő ciklusba. A populáció egyedeit lemásolja egy átmeneti populációba fitneszértékeik sorrendjében, tehát a legjobb fitneszértékű egyed kerül az első pozícióba. Majd a mutációs operátorok [92] a meghatározott paraméterek alapján végrehajtódnak az egyedeken.

Az algoritmus kétféle mutációs operátor csoportot alkalmaz, a multi-kromoszómás struktúrának megfelelően lehetnek:

- lokális operátorok: ahol az operátor egy kromoszómán belül végez műveletet,
- globális operátorok: ahol az operátor kromoszómák között végez műveletet.

A mutációs operátorok végrehajtási sebessége nagymértékben befolyásolja az algoritmus sebességét, ezért a nagyméretű problémák hatékony kezelése érdekében az egyszerű, gyorsan végrehajtható operátorok kidolgozására helyeztem a hangsúlyt.

7.7.1 Lokális operátorok

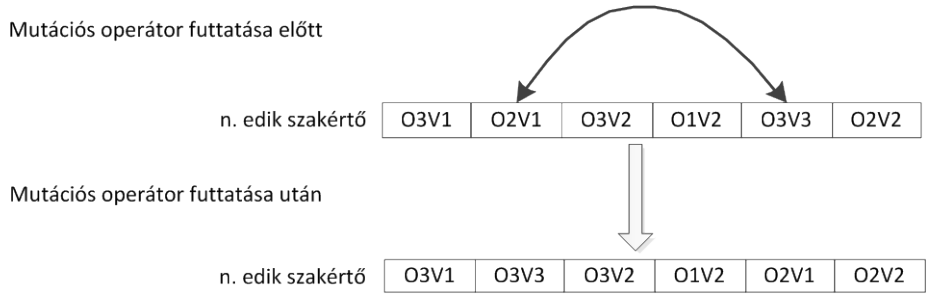
A lokális operátorok a lokális büntetőfüggvényekhez hasonlóan egy kromoszómán belül hajtódnak végre, vagyis egy kromoszóma adatait módosítják.

Az algoritmus három lokális operátor típust használ, ezek a következők:

- gén áthelyezés,
- géncsere,
- gén szekvencia fordítás.

7.7.1.1 Gén áthelyezés operátor

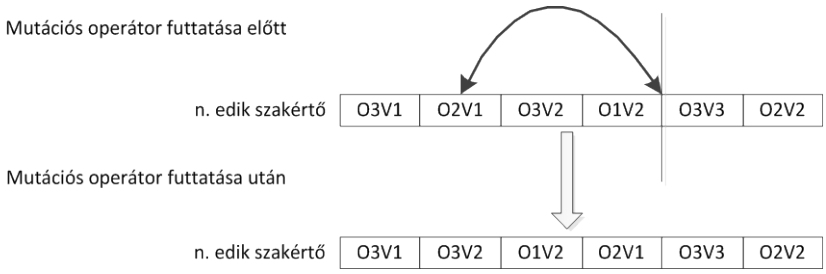
A gén áthelyezés operátora egy gént kiemel az eredeti pozíciójából és egy új pozícióba illeszti be az adott kromoszómán belül. Az áthelyezendő gén kiválasztása véletlenszerűen történik csakúgy, mint az új pozíció kiválasztása.



24. ábra Gén áthelyezés

7.7.1.2 Géncsere

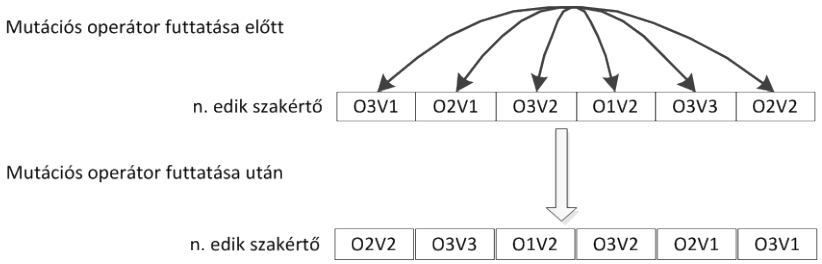
A géncsere operátora egy kromoszómán belül cserél fel két gént. A gének kiválasztása véletlenszerűen történik. Amennyiben a két véletlen érték megegyezik, nem történik művelet.



25. ábra Géncsere

7.7.1.3 Gén szekvencia fordítás

A génszekvencia fordító operátor egy véletlenszerű pozíciótól kezdődően egy véletlen hosszúságú génszekvenciában cseréli fel a géneket. Amennyiben a génszekvencia hossza egy, az operátor nem végez műveletet.



26. ábra Gén szekvencia fordítás

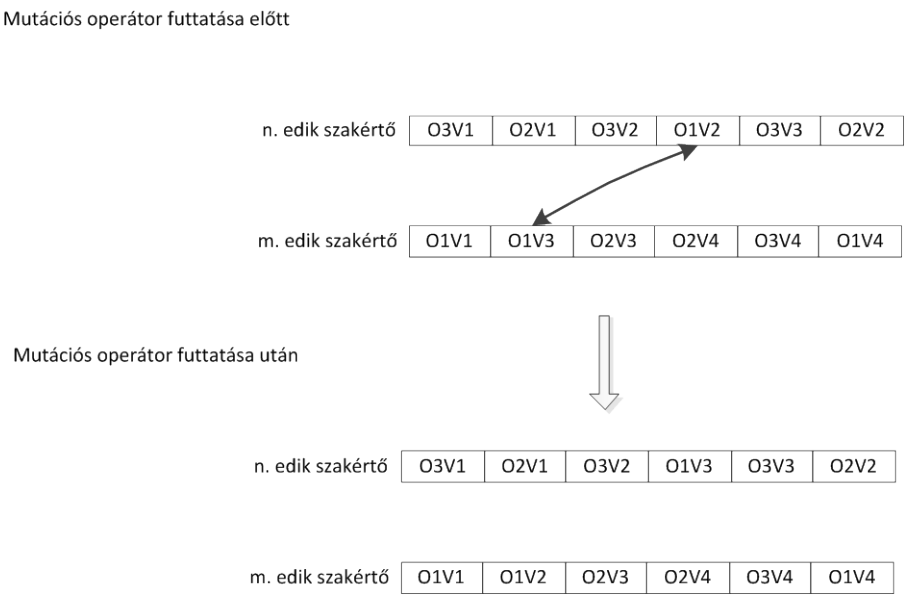
7.7.2 Globális operátorok

A globális operátorok jellemzője az egyedre vonatkoztatott globalitás, vagyis működésük több kromoszómára terjed ki. Az algoritmusban háromféle globális operátor típus létezik:

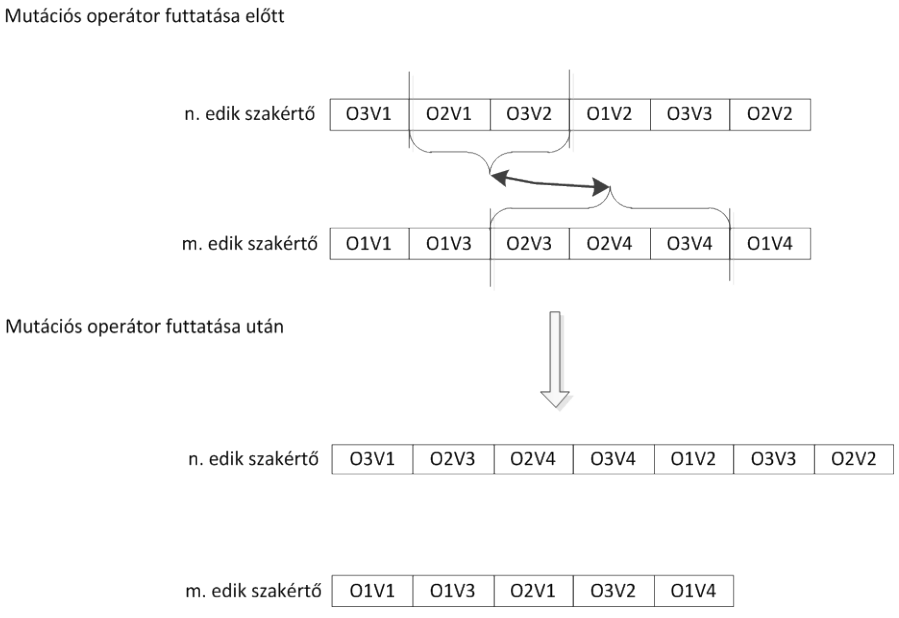
- kromoszómák közötti géncsere,
- kromoszómák között gén szekvencia csere,
- kromoszóma kontrakció.

7.7.2.1 Kromoszómák közötti géncsere

A kromoszómák közötti géncsere operátora egy véletlenszerűen kiválasztott szakértő (*n*-edik szakértő) és egy másik véletlenszerűen kiválasztott szakértő (*m*-edik szakértő) egy-egy véletlenszerűen kiválasztott génjét kicseréli. Amennyiben a két véletlenszerűen kiválasztott szakértő megegyezik, az operátor nem végez műveletet.



27. ábra Kromoszómák közötti géncsere



28. ábra Kromoszómák közötti génszekvencia csere

7.7.2.2 Kromoszómák közötti génszekvencia csere

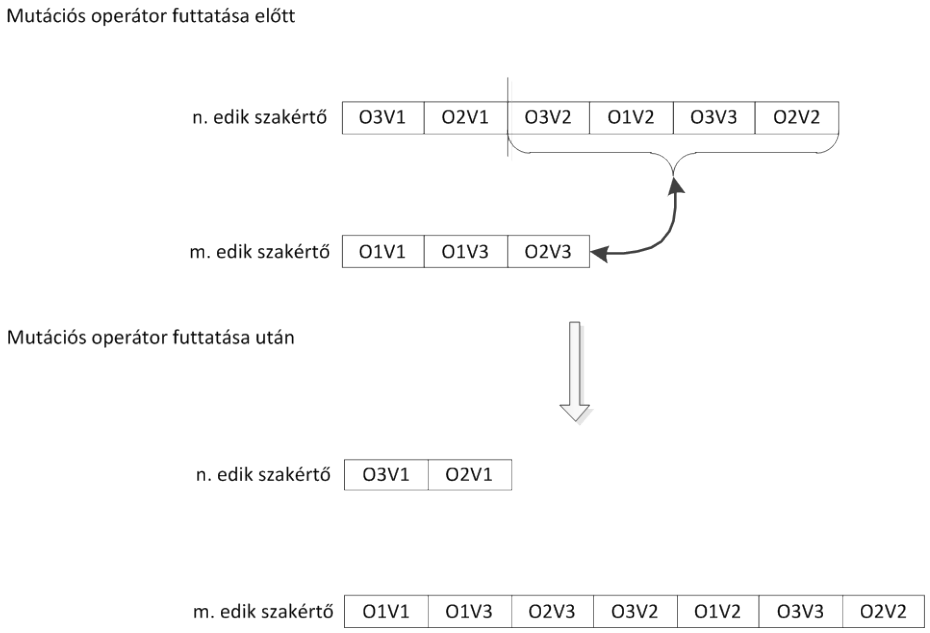
A kromoszómák közötti génszekvencia csere operátora egy véletlenszerűen kiválasztott szakértő (*n*. szakértő) és egy másik - *n*-től különböző - véletlenszerűen kiválasztott szakértő (*m*. szakértő) egy-egy véletlenszerűen meghatározott hosszúságú génszakaszát cseréli ki (28. ábra).

7.7.2.3 Kromoszóma kontrakció

A megoldó algoritmus elméleti kidolgozása során a kontrakciós operátorok két típusát definiáltam:

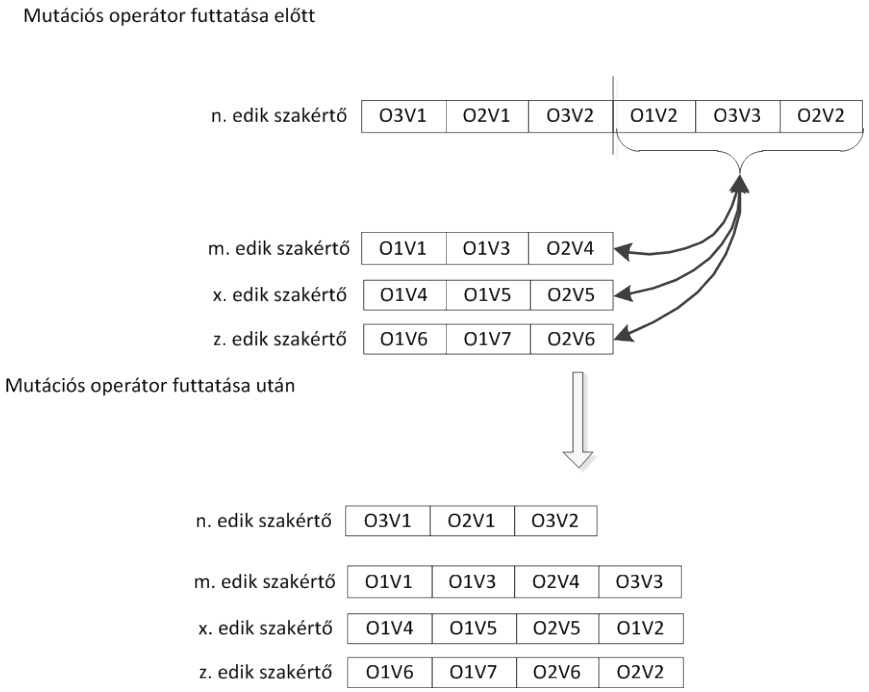
- 1. típus: a leválasztott géneket egyazon szakértőhöz helyezi át (29. ábra),
- 2. típus: a leválasztott géneket véletlenszerűen helyezi el minden kromoszómánál egyenként meghatározva más-más szakértőhöz (30. ábra).

Az első típusú kontrakciós operátor egy véletlenszerűen kiválasztott kromoszóma (*n.* szakértőhöz rendelt vizsgálatok listájának) hosszát rövidíti a gének áthelyezésével úgy, hogy a kromoszóma végéről egy véletlenszerű hosszúságú génszakaszt egy másik véletlenszerű kromoszómához (*m*. szakértő vizsgálati listájához) illeszt.



29. ábra Kromoszómák közötti kontrakciós operátor

A második típusnál a kromoszóma kontrakciós operátor egy véletlenszerűen kiválasztott kromoszóma (*n*. szakértő) hosszát rövidíti, úgy hogy a kromoszóma végéről egy véletlenszerű hosszúságú génszakaszt génenként véletlenszerűen meghatározva egy másik kromoszómához (*m*. szakértő) illeszt. Működését a 30. ábra mutatja.



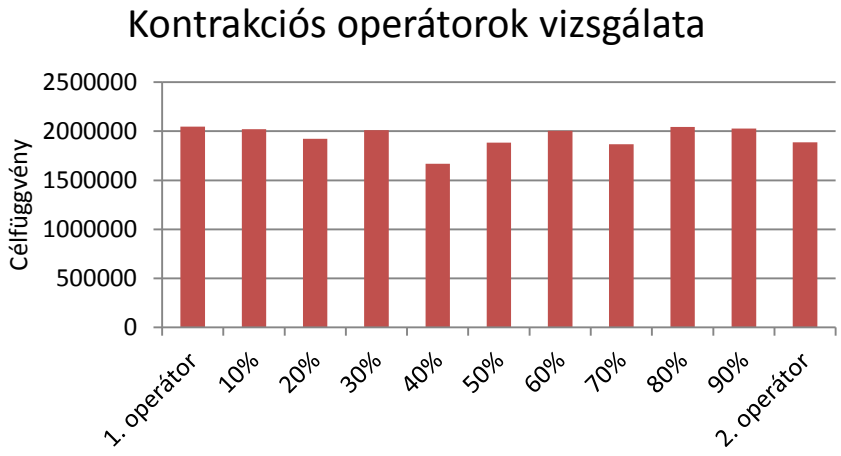
30. ábra Kromoszómák közötti kontrakciós operátor (2. típus)

7.7.2.4 Kontrakciós operátorok vizsgálata

A két kontrakciós operátor használata ellentmondó eredményeket szolgáltatott, így az operátorok hatásának kiterjedt vizsgálatára volt szükség. Több vizsgálatot végeztem, ahol a futtatásoknál azonos paraméterek mellett emeltem a 2. operátor futásának valószínűségét.

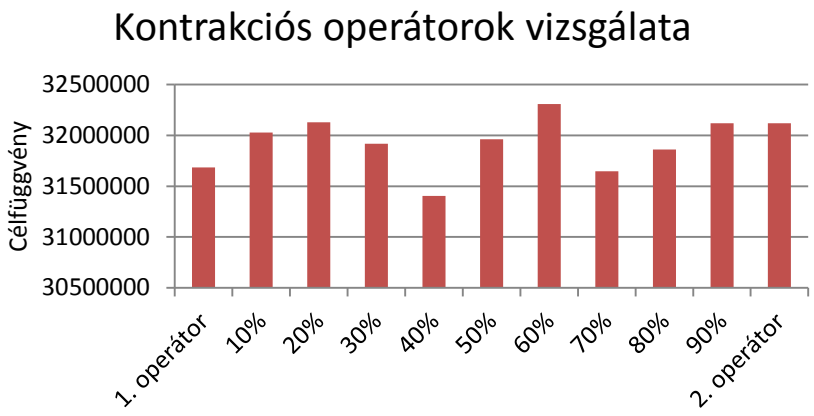
1. vizsgálat 3 szakértővel, 48 csomóponttal, vizsgálatok száma 2-4 között.

A diagramok (31. ábra, 32. ábra, 33. ábra) X tengelyén a 2. operátor kiválasztási valószínűsége szerepel (0 – 100 %).



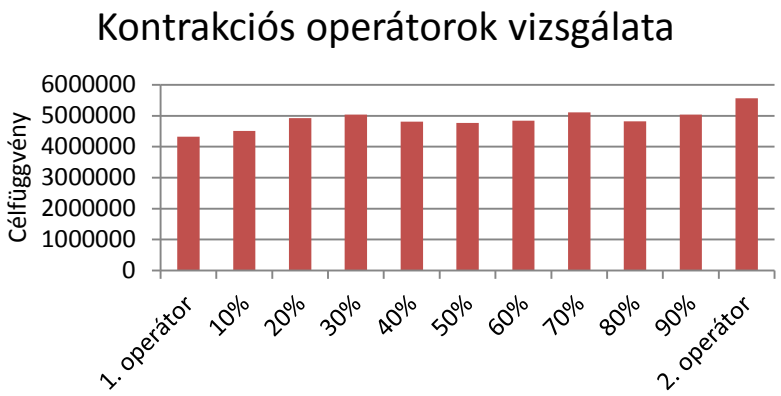
31. ábra Kontrakciós operátorok vizsgálata (1. vizsgálat)

2. vizsgálat 3 szakértővel, 48 csomóponttal, vizsgálatok száma 5-10 között.



32. ábra Kontrakciós operátorok vizsgálata (2. vizsgálat)

3. vizsgálat 3 szakértővel, 48 csomóponttal, vizsgálatok száma 10-15 között.



33. ábra Kontrakciós operátorok vizsgálata (3. vizsgálat)

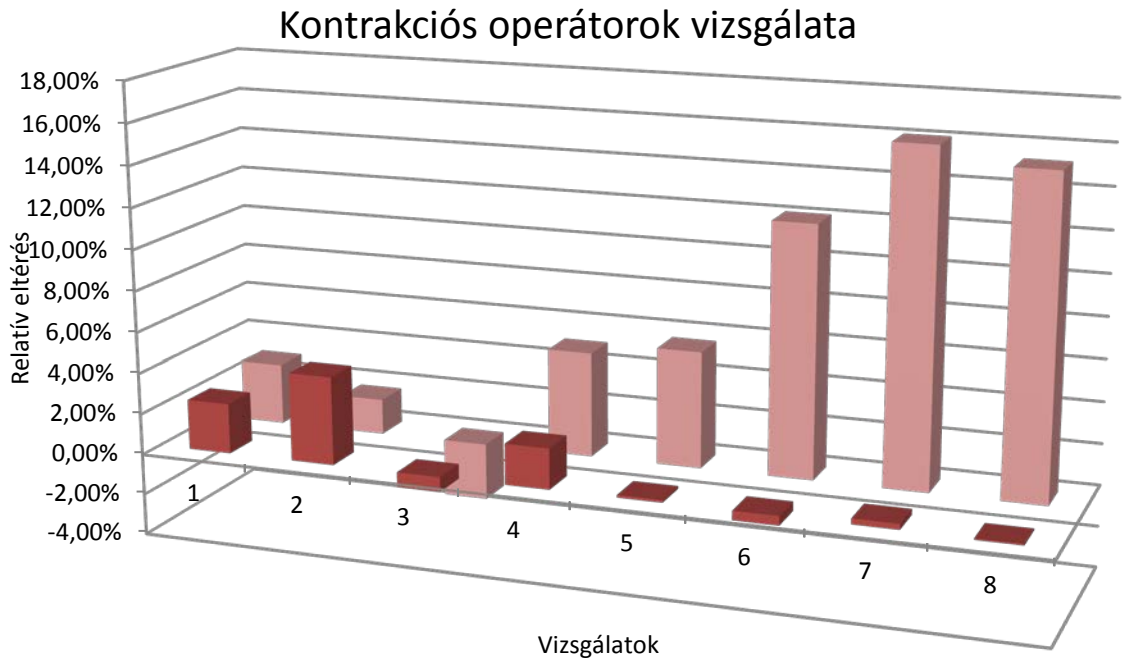
A vizsgálatokból kitűnik, hogy az operátorok alkalmazása a 2. operátor 40 százalékos valószínűséggel való alkalmazása esetén jobb megoldást ad azonos iteráció szám mellett a vizsgált esetek kétharmadánál. Így szükségesnek láttam a vizsgálatok kiterjesztését a 40 százalékos valószínűség vizsgálatára egyre növekvő problémaméret esetén.

	Vizsgálat sorszáma							
	1	2	3	4	5	6	7	8
	Célfüggvény							
1. operátor	8002,5	5246833,3	17397317,3	33781333,6	46382896,0	115069309,1	344822051,5	456533669,5
1. op. + 40 % 2. op.	8203,1	5482336,3	17271951,2	34462749,7	46438808,3	114553051,1	345782665,9	456247933
2. operátor	8246,0	5338075,4	16930432,8	35578076,0	49140600,5	130939609,0	410956545,9	539052208,7

	Relatív eltérés							
1. operátor	0	0	0	0	0	0	0	0
1. op. + 40% 2. op.	2,45%	4,30%	-0,73%	1,98%	0,12%	-0,45%	0,28%	-0,06%
2. operátor	2,95%	1,71%	-2,76%	5,05%	5,61%	12,12%	16,09%	15,31%

	Paraméterek							
Ciklusszám	5	3	3	5	5	5	5	5
Min. vizsgálat	1	2	5	10	10	100	400	500
Max vizsgálat	1	4	10	16	25	180	500	650
Vizsgáltat tól.	15	15	35	35	50	30	80	100
Vizsgáltat ig.	17	22	55	50	100	50	150	200

1. táblázat Kontrakciós operátorok vizsgálata, a második operátor alkalmazásának 40 százalékos valószínűsége mellett

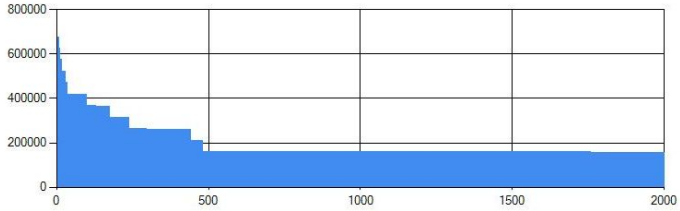


34. ábra Kontrakciós operátorok vizsgálata, a második operátor alkalmazásának 40 százalékos valószínűsége mellett

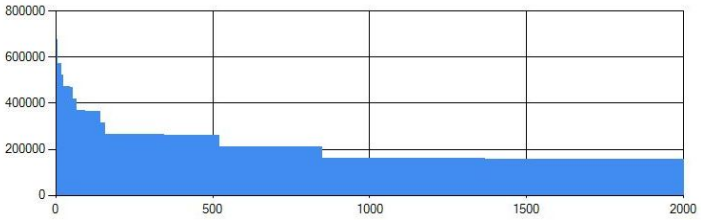
A kontrakciós operátorok vizsgálata azt mutatta, hogy a 40 százalékos valószínűségi aránynál és adott iterációnál csak egyes egyedi esetekben állt elő jobb célfüggvényérték, és a javulás nem éri el az 1 százalékot, így ezen tesztek alapján alkalmazása nem, vagy csak részleges tesztfuttatás után indokolt.

Azonban konvergencia tesztek kimutatták, hogy a 2. kontrakciós operátor alkalmazása gyorsíthatja az algoritmus konvergenciáját az optimálás kezdeti szakaszán.

A 35. ábra, valamint a 36. ábra mutatják az algoritmus konvergenciát az 1.-es, valamint a 2.-es kontrakciós operátor használatával. Az ábrákon jól látszik, hogy a 2. operátorral a konvergencia sebesség nagyobb az optimalizálási folyamat kezdeti szakaszában.

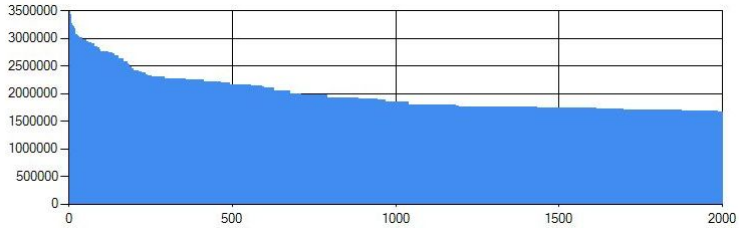


35. ábra A célfüggvény konvergenciája az 1. kontrakciós operátorral

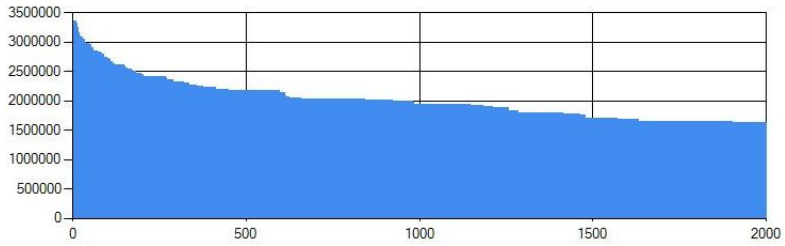


36. ábra A célfüggvény konvergenciája a 2. kontrakciós operátorral

A 36. ábra, 40. ábra egy másik teszt problémán futtatva mutatják a konvergenciát az 1.-es valamint a 2.-es kontrakciós operátor használatával. Az ábrákon látható, ebben az esetben is a 2. operátor alkalmazásával a konvergencia sebesség az optimalizálás kezdeti szakaszában nagyobb.



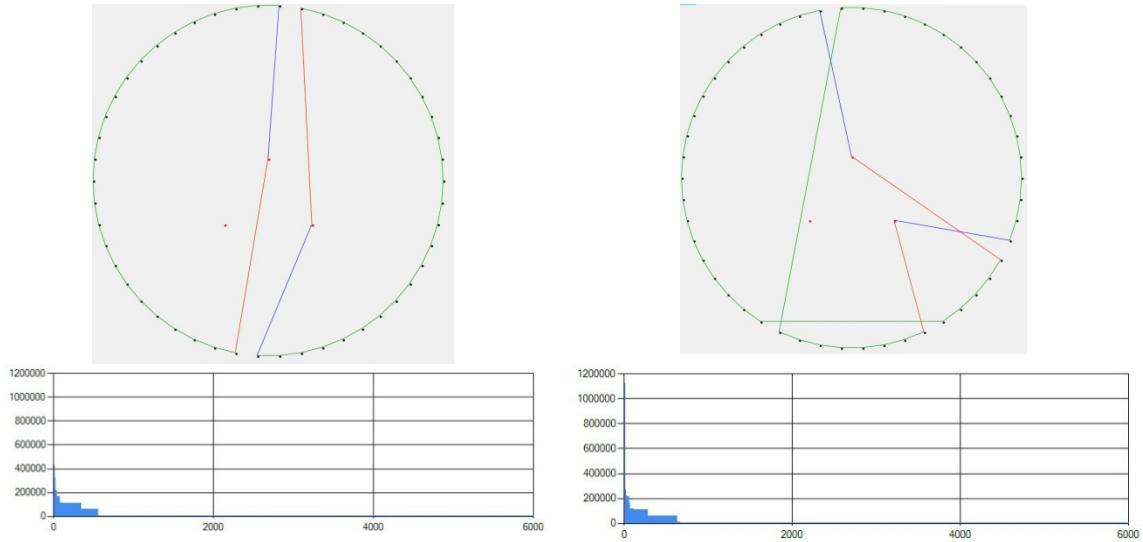
37. ábra A célfüggvény konvergenciája az 1. kontrakciós operátorral



38. ábra A célfüggvény konvergenciája a 2. kontrakciós operátorral

Az ábrák azt mutatják, hogy ebben az esetben a 2. operátor a tesztfuttatásnál megadott intervallumban az iterációs ciklusok végén is jobb eredményt szolgáltat, azonban ez a tesztfuttatás a teljes optimálási folyamat mintegy tizede. A teljes futtatások nem igazolták az operátor jóságát a teljes folyamaton, így további vizsgálatokra volt szükség.

A további tesztek azonban azt mutatták, hogy főleg olyan futtatásoknál, ahol a szakértők száma is optimalizálandó, tehát egyes szakértők elhagyhatók a rendszerből, az operátor az optimalizálás végső szakaszában lassan konvergál a megoldáshoz. A 39. ábra látható egy olyan vizsgálat, amely 6000 iterációs ciklus után mutatja az optimálás eredményét, valamint a célfüggvény konvergenciáját a két operátor használata esetén.



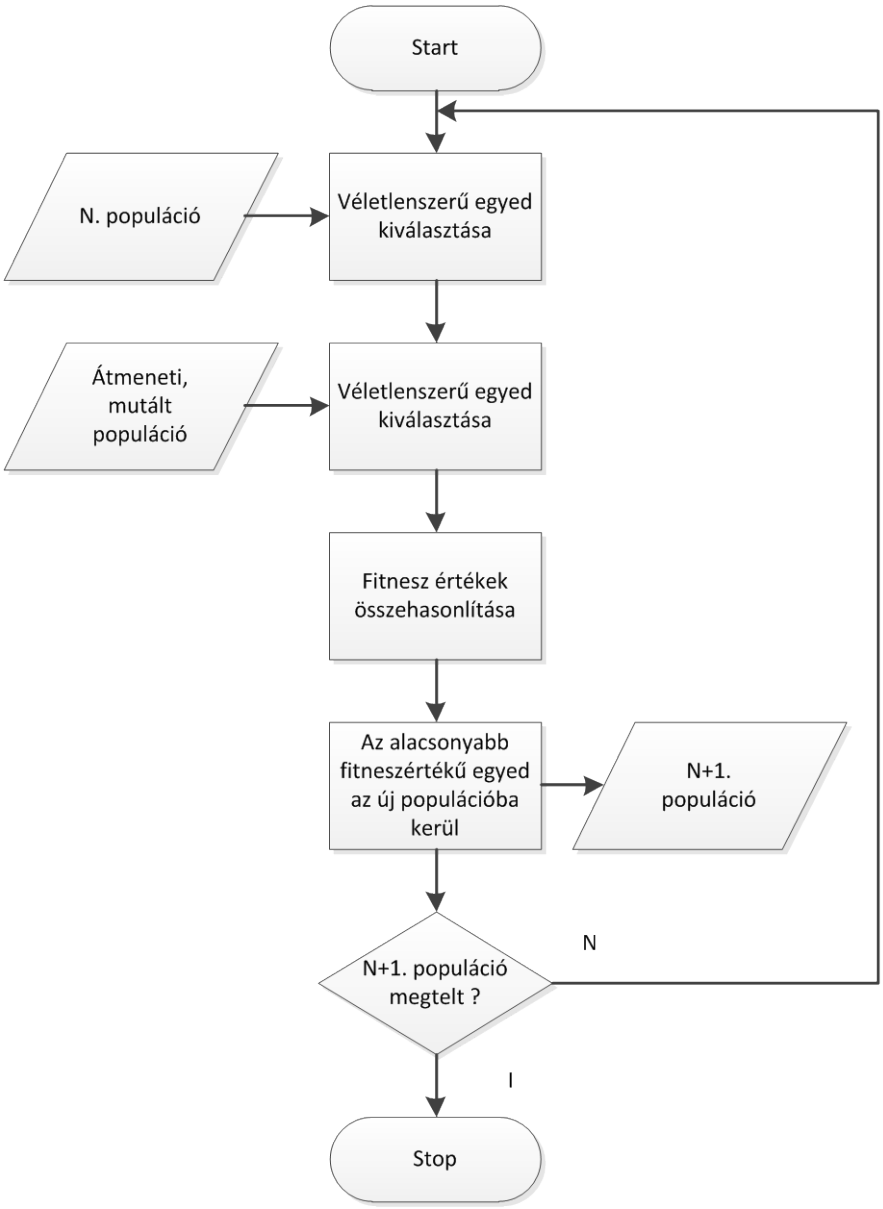
1. kontrakciós operátorral 2. kontrakciós operátorral

39. ábra Optimalizálás alkalmazott szakértők számának redukálásával

A jelenség magyarázata az, hogy a 2. operátor a kontrakcióra kiválasztott génszekvenciát szétosztja a kromoszómák között, tehát egy véletlenszerűen kiválasztott szakértő vizsgálatait szétosztja több szakértő között, így növeli az egyedek sokféleségét. Viszont kisebb valószínűséggel jönnek létre olyan egyedek, amelyeknél egy-egy szakértőhöz nem tartozik objektum, azonban mivel ez sok feladatban az optimalizálás célja, így az operátor használatát meg kell fontolni, alkalmazása előtt tesztfuttatásokat kell végezni. Általánosan kijelenthető azonban, hogy alkalmazása az optimalizálás első szakaszában – az iterációszám mintegy 10 százalékában - az algoritmus konvergenciájának sebességét jelentősen javíthatja.

7.8 Szelekció

Az algoritmus a szelekció menetét, a legjobb egyedek kiválasztását bajnoksággal valósítja meg. Az evolúción alapuló algoritmusok szelekciós módszerekből igen széles skálát használnak. Ezek közül az evolúciós programozásban széles körűen használt bajnoksággal történő kiválasztás az egyik leggyorsabb, valamint a bajnokságban résztvevő egyedek véletlenszerű kiválasztásával biztosítja a lokális optimumok elkerülésének lehetőségét is [93]. A bajnokság végrehajtása előtt a legjobb, úgynevezett elitista egyed változatlanul átkerül a leszármazott populációba [94]. A gyors végrehajtási sebesség érdekében az algoritmus kétszereplős bajnokságot használ - (1+1) stratégia [95].



40. ábra Szelekciós bajnokság menete

A bajnokság első lépéseként véletlenszerűen kiválasztunk egy-egy egyedet az eredeti, valamint a mutáción átesett populációból. A jobb fitneszértékű egyedet, ami ebben az esetben a kisebb célfüggvényértékű egyedet jelenti, a leszármazott populációba helyezzük. A folyamatot addig ismételjük, amíg a leszármazott populáció meg nem telik.

A szelekció egy a bementi paraméterek között meghatározott számig tölti fel a leszármaztatott populációt. A teljes populáció méret eléréséig a populáció véletlenszerűen generált egyedekkel töltődik fel.

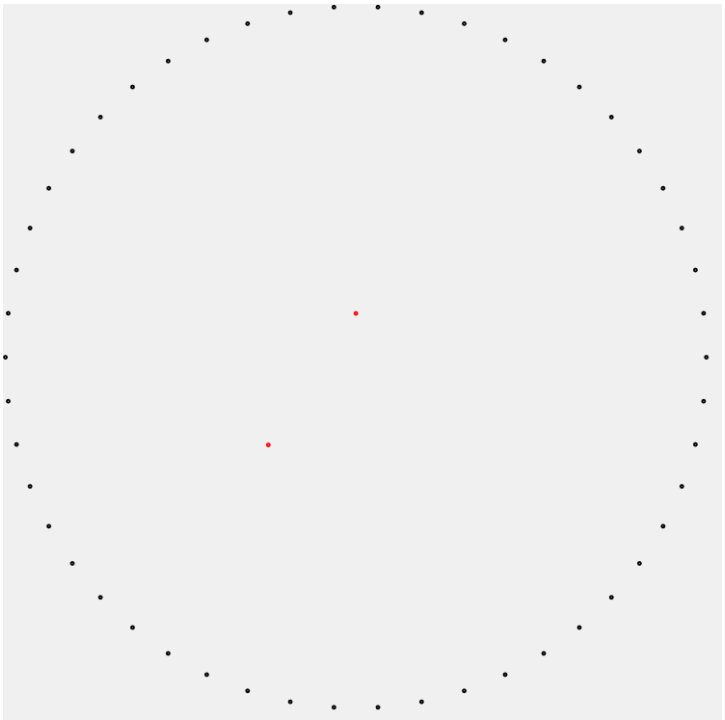
8 Eredmények

8.1 Az algoritmus tesztelése jellegzetes struktúrákkal

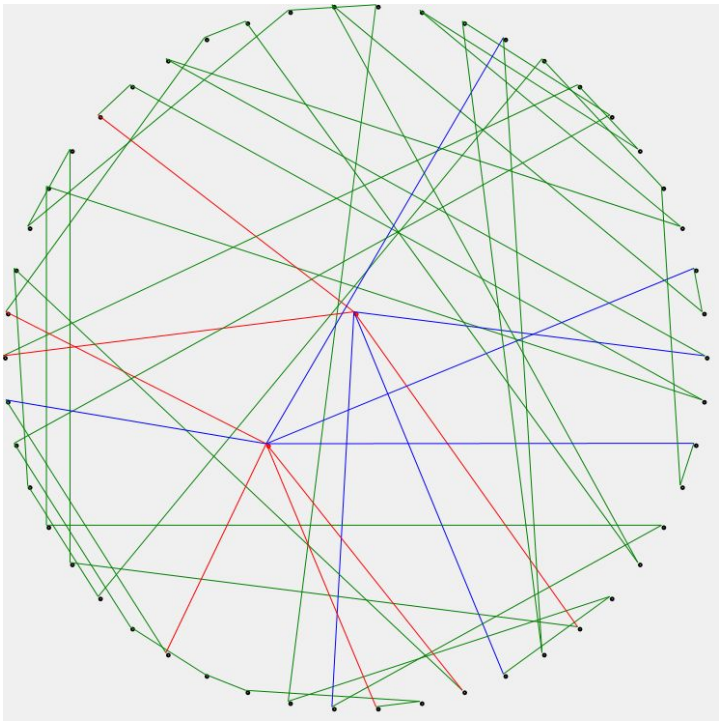
Az algoritmus kidolgozásakor a nagyméretű logisztikai problémákra való hatékony alkalmazás volt a cél. Az algoritmus tesztelésére készült egy számítógépes program C# nyelven, amely megvalósítja az algoritmust minden paraméterével, így módot adva az alapos és tervszerű tesztelésre, paraméter érzékenység vizsgálatokra. A tesztelés folyamán a globális és lokális mutációs operátorok aránya, valamint a büntetőértékek ezek alapján, tapasztalati úton lettek meghatározva.

8.1.1 Tesztfeladat két szakértővel

Az első tesztfuttatás egy egyszerű tesztrendszer két szakértővel és körben elhelyezkedő objektumokkal, ahol minden objektumot csak egyszer kell vizsgálni. A feladat egyszerűségéből kifolyólag az algoritmus megoldásának jósága jól látható. A 41. ábrán látható a szakértők és objektumok elhelyezkedése a tesztfeladatban, a 42. ábra mutatja az algoritmus által generált véletlenszerű kiinduló állapotot. A 43. ábrán látható az evolúciós programozási algoritmus által szolgáltatott megoldás. A 44. ábrán látható a célfüggvény konvergálása az optimumhoz.



41. ábra Szakértők és objektumok elhelyezkedése



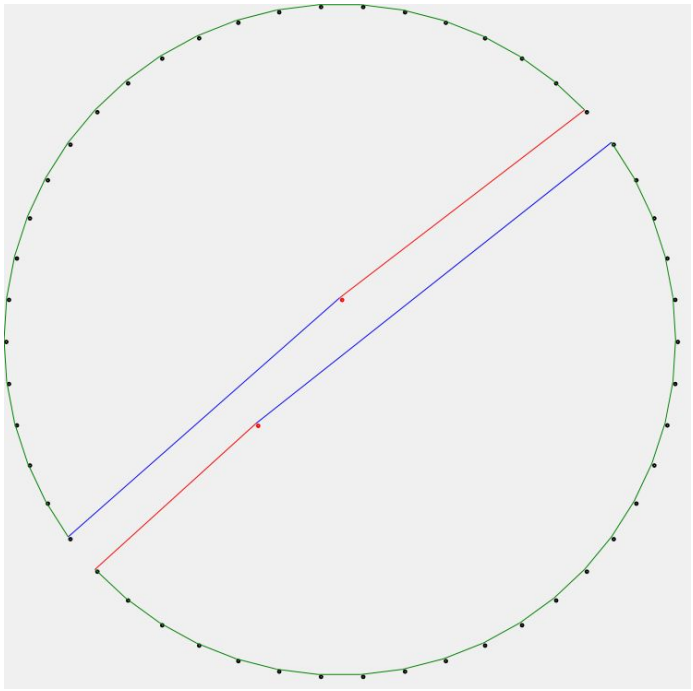
42. ábra Kiinduló állapot

Bemenő paraméterek listája:

Iterációszám	10000	Mutációs operátorok valószínűsége	
Populáció méret	500	Nincs mutáció	0
Véletlen egyedek száma	50	Lokális mutáció	50
Szakértők száma	2	Globális mutáció	50
Csomópontok száma	48		
Minimális vizsgálatszám	23	Lokális operátorok	
Maximális vizsgálatszám	25	Géncsere	40
Maximális körúthossz	4000	Génbeszúrás	40
Maximális ciklusszám	1	Génszekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	1	Géncsere	45
Objektum vizsgálatok ig	1	Génszekvencia csere	45
Büntetőértékek		Génkontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	0		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	50000		
Szakértő költsége	0		

2. táblázat A tesztfeladat paraméterei

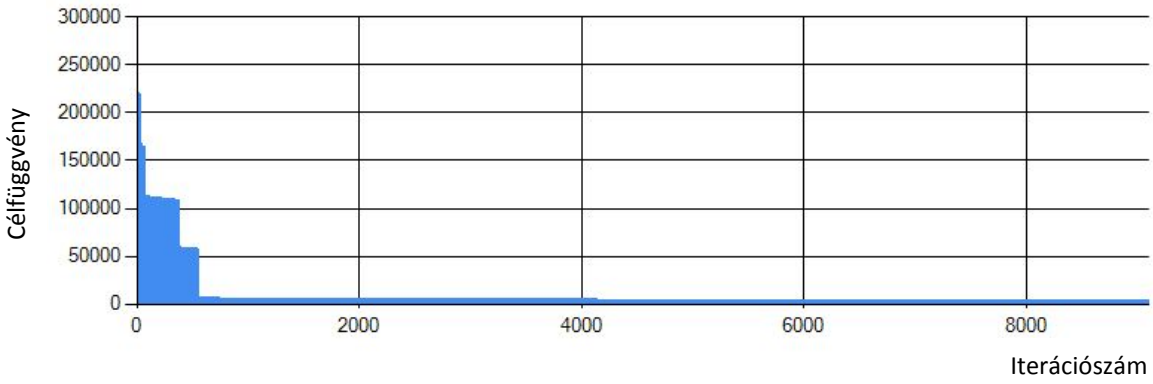
A futtatás eredménye:



43. ábra A tesztfeladat megoldása

Az ábra jelmagyarázata:

- a piros pontok a szakértők,
- a fekete pontok az objektumok,
- a zöld vonal a szakértő körútja közben bejárt útvonal,
- a kék vonal a szakértő körútjának kezdete,
- a vörös vonal a szakértő a körútjának befejezése.



44. ábra Célfüggvény konvergenciája

Iterációszám	9100
Megoldási idő	13 min 06 sec
Büntetés	0
Költség	4008,85

3. táblázat A tesztfeladat futási eredményei

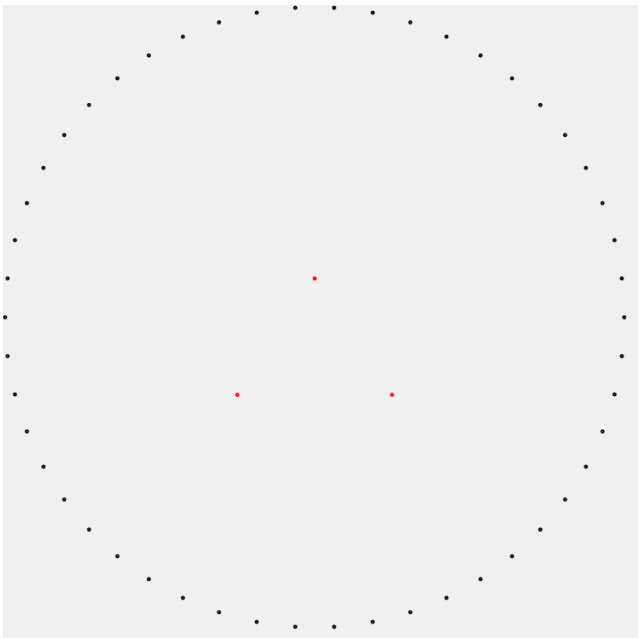
Vizsgáljuk meg, hogy az optimalizálás a keresési tér hány százalékát járja be:

- Összes egyed: $\frac{48!}{(48-24)!} \cdot 2 =$ 4,00159E+37 egyed.
- Az optimalizálás során kiszámított egyedek száma: $9100 \cdot 500 =$ 4550000.

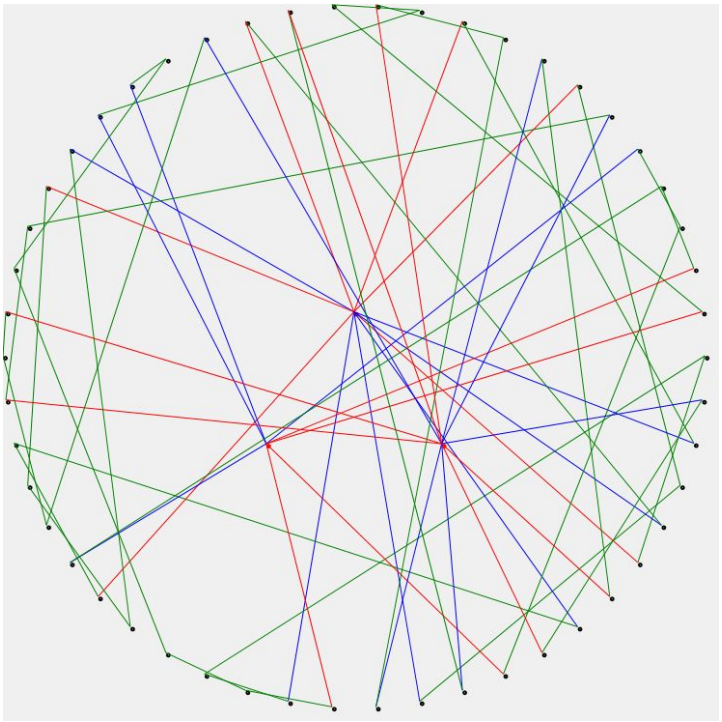
Az algoritmus a keresési tér 1,13705E-29 százalékát járta be.

8.1.2 Tesztfeladat három szakértővel

A második tesztfeladat az előzőhöz hasonló rendszer viszont nem két, hanem három szakértővel (45. ábra). Az előző példához hasonlóan a szakértő minden objektumot csak egyszer keres fel, így a megoldás jósága (47. ábra) vizuálisan ennél a problémánál is könnyen ellenőrizhető.



45. ábra Szakértők és objektumok elhelyezkedése



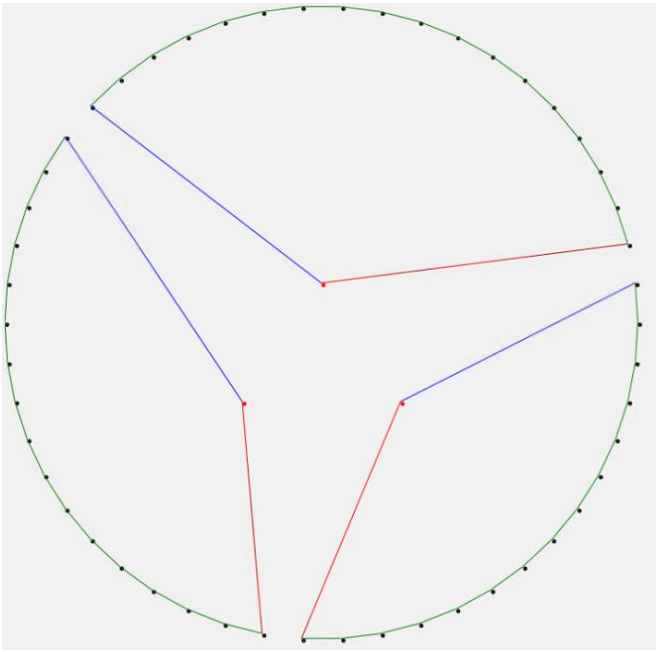
46. ábra Kiinduló állapot

Bemenő paraméterek listája:

Iterációszám	20000	Mutációs operátorok valószínűsége	
Populáció méret	500	Nincs mutáció	0
Véletlen egyedek száma	50	Lokális mutáció	50
Szakértők száma	3	Globális mutáció	50
Csomópontok száma	48		
Minimális vizsgálatszám	16	Lokális operátorok	
Maximális vizsgálatszám	18	Géncsere	40
Maximális körúthossz	2000	Génbeszúrás	40
Maximális ciklusszám	1	Génszekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	1	Géncsere	45
Objektum vizsgálatok ig	1	Génszekvencia csere	45
Büntetőértékek		Génkontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	0		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	50000		
Szakértő költsége	0		

4. táblázat A tesztfeladat paraméterei

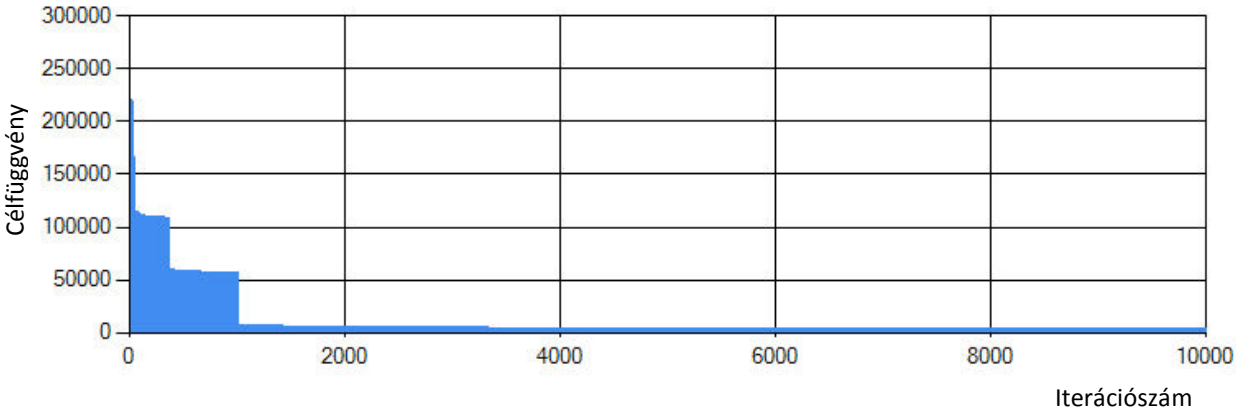
A futtatás eredménye:



47. ábra A tesztfeladat megoldása

Iterációszám	16434
Megoldási idő	24 min 35 sec
Büntetés	0
Költség	4483,96

5. táblázat A tesztfeladat futási eredményei



48. ábra Célfüggvény konvergenciája

A 47. ábrán látható hogy a megoldó algoritmus megtalálja a megoldásként „elvárt” 120 fokos elrendezést.

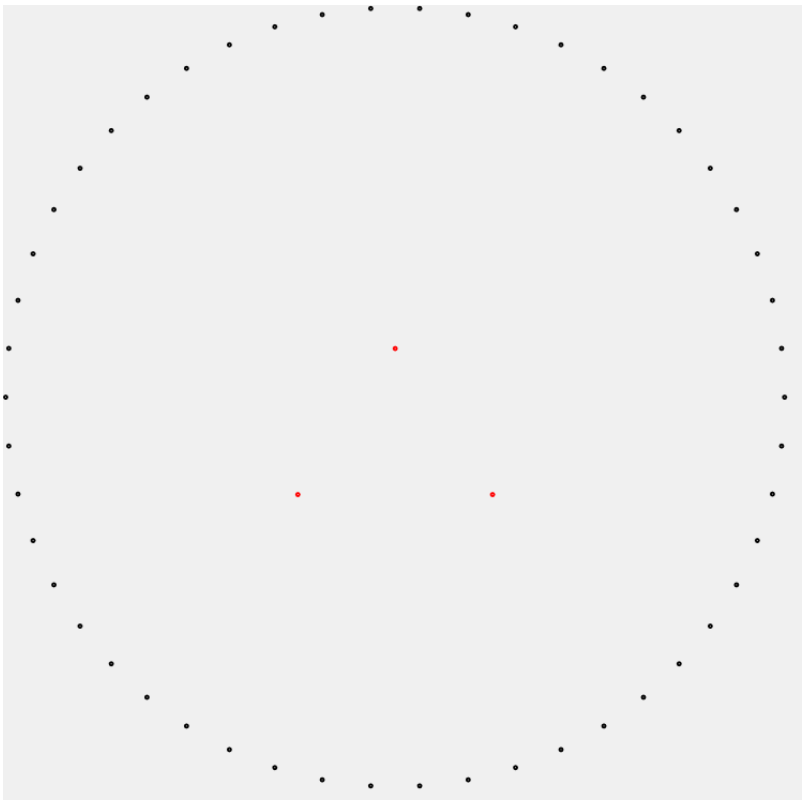
Vizsgáljuk meg, hogy az optimalizálás a keresési tér hány százalékát járja be:

- Összes egyed: $\frac{48!}{(48-16)!} \cdot 3 =$1,41533E+26 egyed.
- Az optimalizálás során kiszámított egyedek száma: $16434 \cdot 500 =$ 8217000.

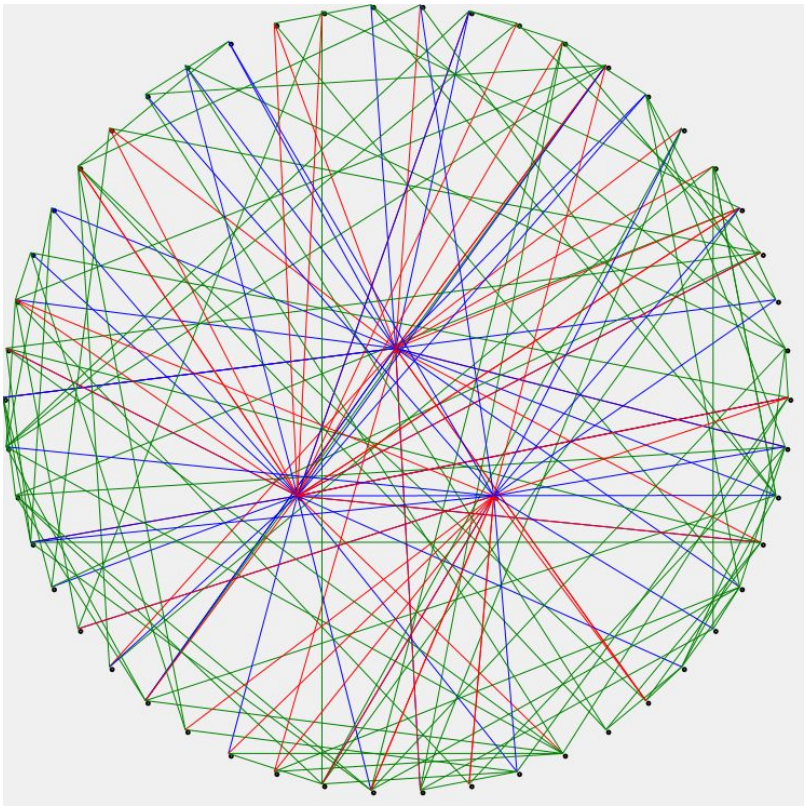
Az algoritmus a keresési tér 5,80571E-18 százalékát járta be.

8.1.3 Összetett tesztfeladat három szakértővel

A harmadik az előzőhöz hasonló rendszer, három szakértővel (49. ábra). Ebben az esetben az objektumokat, ellentétben az előző feladattal, már többször kell felkeresni. A véletlenszerűen felvett kiinduló állapot ábrája (50. ábra) jól szemlélteti a feladat bonyolultságát. A megoldás (51. ábra) itt már vizuálisan nem ellenőrizhető.



49. ábra Szakértők és objektumok elhelyezkedése

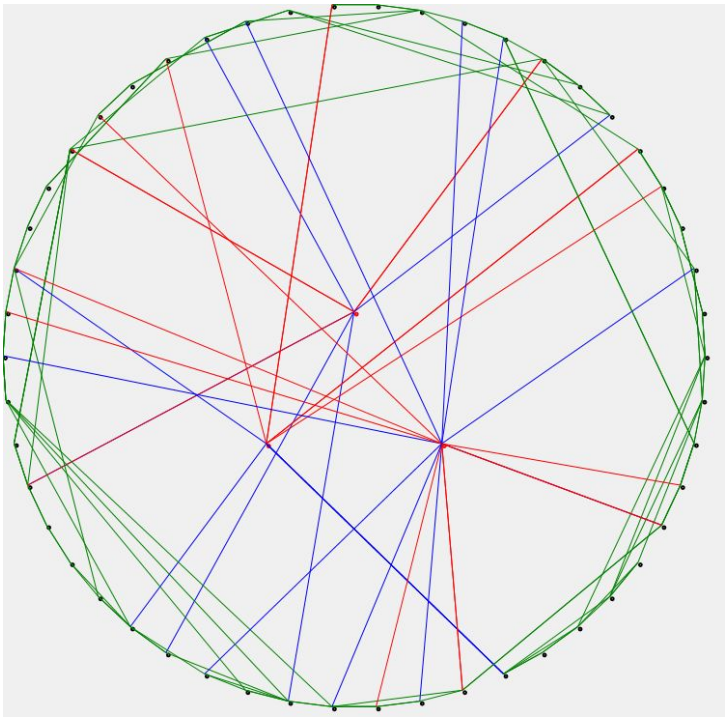


50. ábra Kiinduló állapot

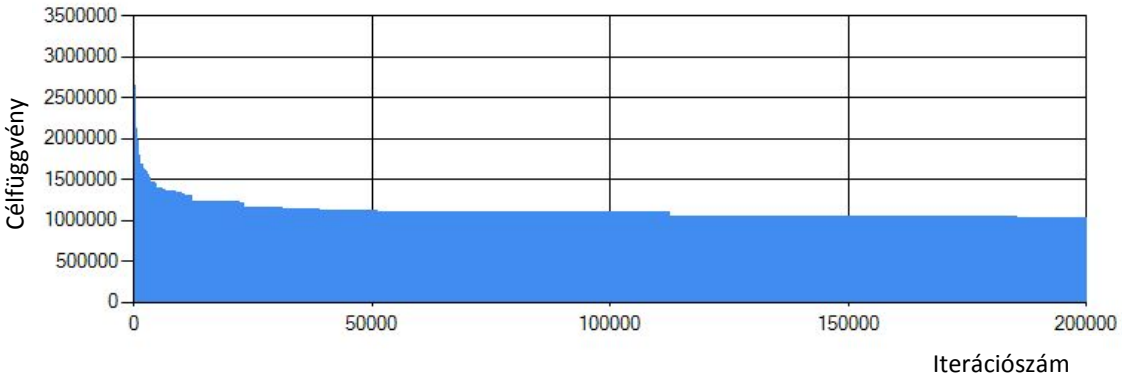
Paraméterek listája:

Iterációszám	50000	Mutációs operátorok valószínűsége	
Populáció méret	500	Nincs mutáció	0
Véletlen egyedek száma	50	Lokális mutáció	50
Szakértők száma	3	Globális mutáció	50
Csomópontok száma	48		
Minimális vizsgálatszám	32	Lokális operátorok	
Maximális vizsgálatszám	64	Géncsere	40
Maximális körúthossz	2000	Génbeszúrás	40
Maximális ciklusszám	5	Gén szekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	2	Géncsere	45
Objektum vizsgálatok ig	5	Gén szekvencia csere	45
Büntetőértékek		Gén kontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	50000		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	50000		
Szakértő költsége	0		

6. táblázat A tesztfeladat paraméterei



51. ábra A tesztfeladat megoldása



52. ábra Célfüggvény konvergenciája

Iterációszám	200000
Megoldási idő	9 h 01 min 12 sec
Büntetés	11
Költség	217534,75

7. táblázat A tesztfeladat futási eredményei

Vizsgáljuk meg, hogy az optimalizálás a keresési tér hány százalékát járja be:

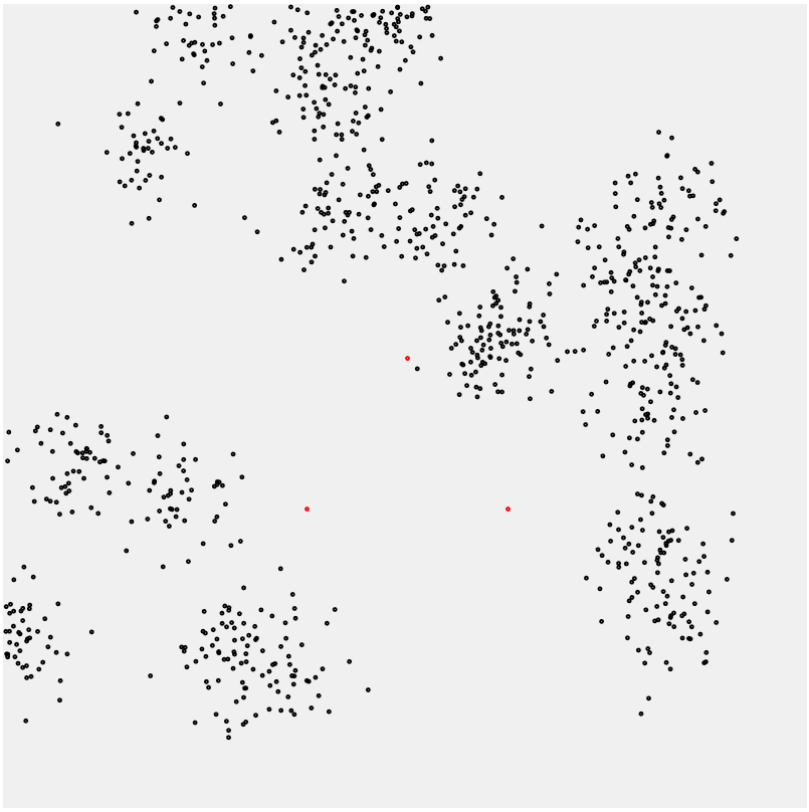
- Összes egyed: $\frac{144!}{(144-48)!} \cdot 3 =$ 7,47836E+62 egyed.
- Az optimalizálás során kiszámított egyedek száma: $200000 \cdot 500 = 100000000$.

Az algoritmus a keresési tér 1,33719E-53 százalékát járta be.

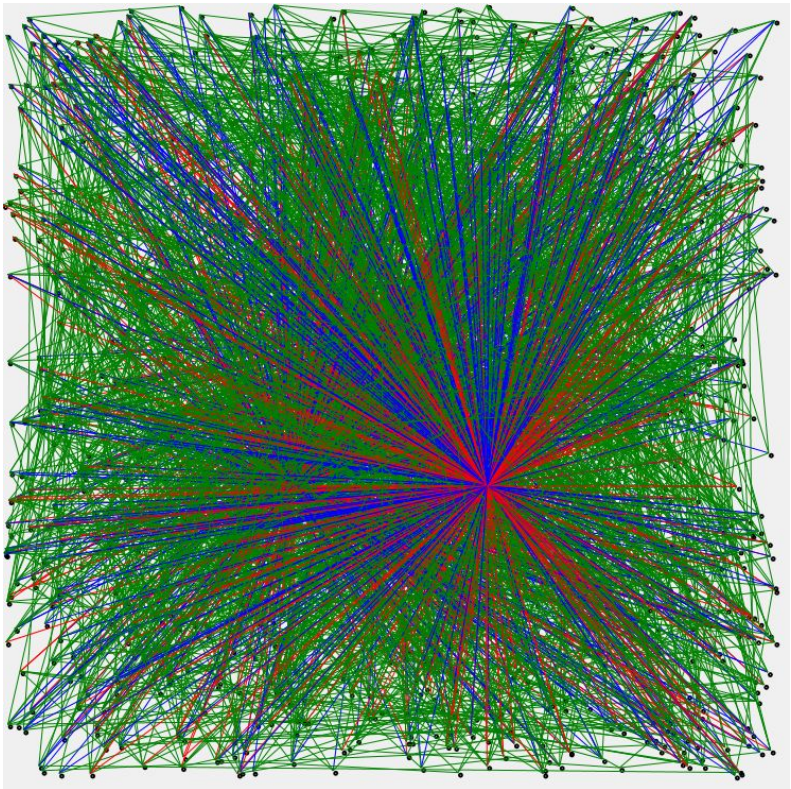
A program futása 200000 iteráció után került leállításra, a kapott megoldás mutatja, hogy az algoritmus 11 büntetést nem tudott eliminálni adott iteráció számon belül (7. táblázat), ezek főleg szétszórt objektumokból adódnak. A feladat megoldásának grafikus megjelenítéséből (51. ábra) látható hogy ebben az esetben a megoldás ábrázolása már nem átlátható, jósága - mint általában a heurisztikus megoldások esetében általában - nem bizonyítható.

8.1.4 Nagyméretű rendszer optimalizálása

A negyedik tesztfeladat egy nagyméretű rendszer optimalizálása. Az előzőekhez hasonlóan ebben a feladatban is csak három szakértő szerepel. Az objektumok száma azonban az előzőektől eltérően 1000 darab (53. ábra). Az objektumokat többször kell felkeresni (8. táblázat). A véletlenszerűen felvett kiinduló állapot ábrán (54. ábra) látható hogy a vizuális ellenőrzés ebben az esetben egyáltalán nem lehetséges a kis méretek és az egymást fedő utak miatt.



53. ábra Szakértők és objektumok elhelyezkedése

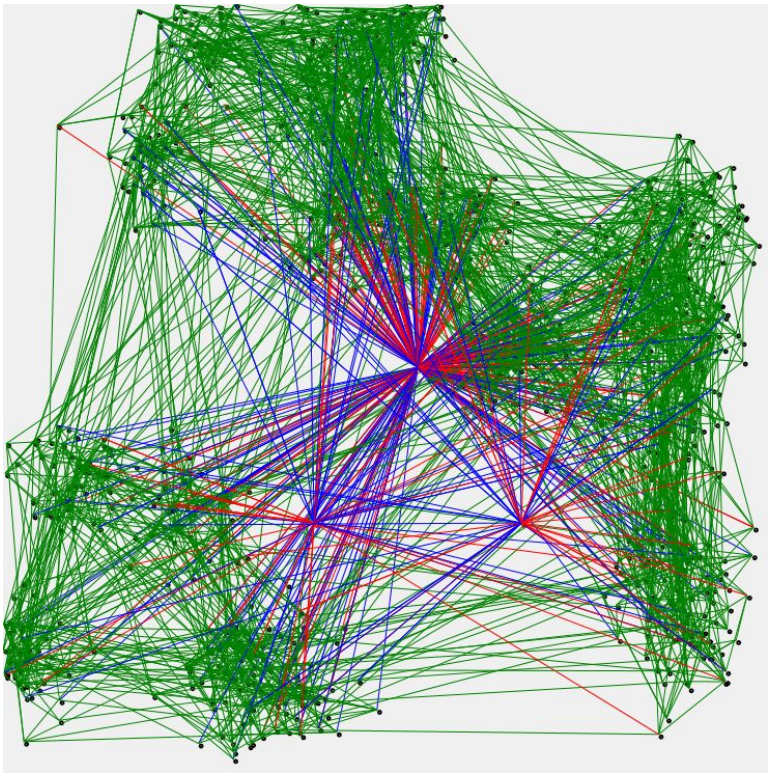


54. ábra Kiinduló állapot

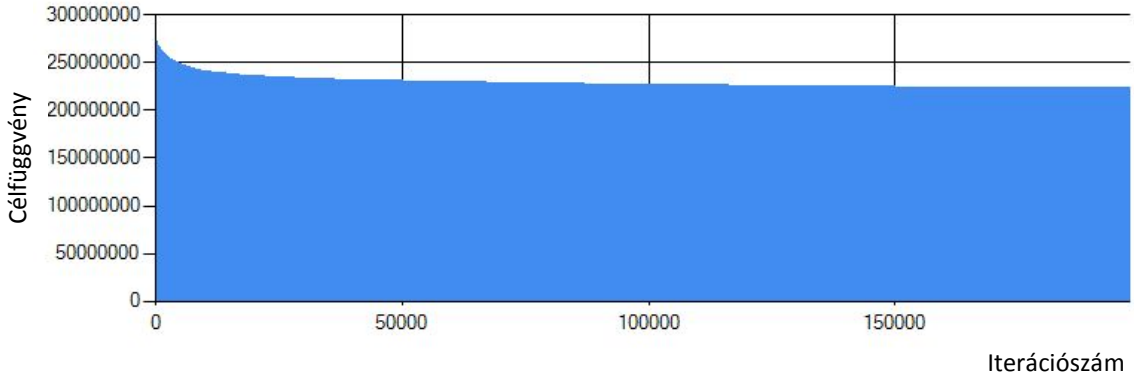
Paraméterek listája:

Iterációszám	200000	Mutációs operátorok valószínűsége	
Populáció méret	500	Nincs mutáció	0
Véletlen egyedek száma	5	Lokális mutáció	50
Szakértők száma	3	Globális mutáció	50
Csomópontok száma	1000		
Minimális vizsgálatszám	200	Lokális operátorok	
Maximális vizsgálatszám	280	Géncsere	40
Maximális körúthossz	2000	Génbeszúrás	40
Maximális ciklusszám	50	Génszekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	2	Géncsere	45
Objektum vizsgálatok ig	5	Génszekvencia csere	45
Büntetőértékek		Génkontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	50000		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	100000		
Szakértő költsége	100000		

8. táblázat A tesztfeladat paraméterei



55. ábra A tesztfeladat megoldása



56. ábra Célfüggvény konvergenciája

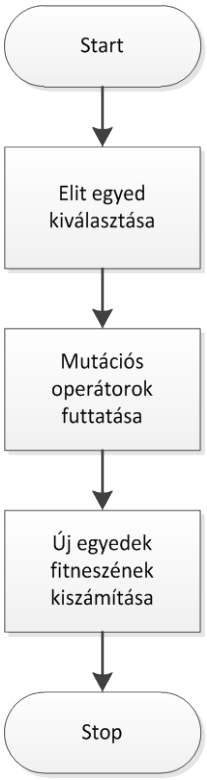
Iterációszám	200000
Megoldási idő	4d 11 h 26 min
Büntetések:	szétszórt objektum 16 eset ciklusszám túllépés 1 esetben szakértő kapacitás túllépés 3 esetben
Költség	223373911,13

9. táblázat A tesztfeladat futási eredményei

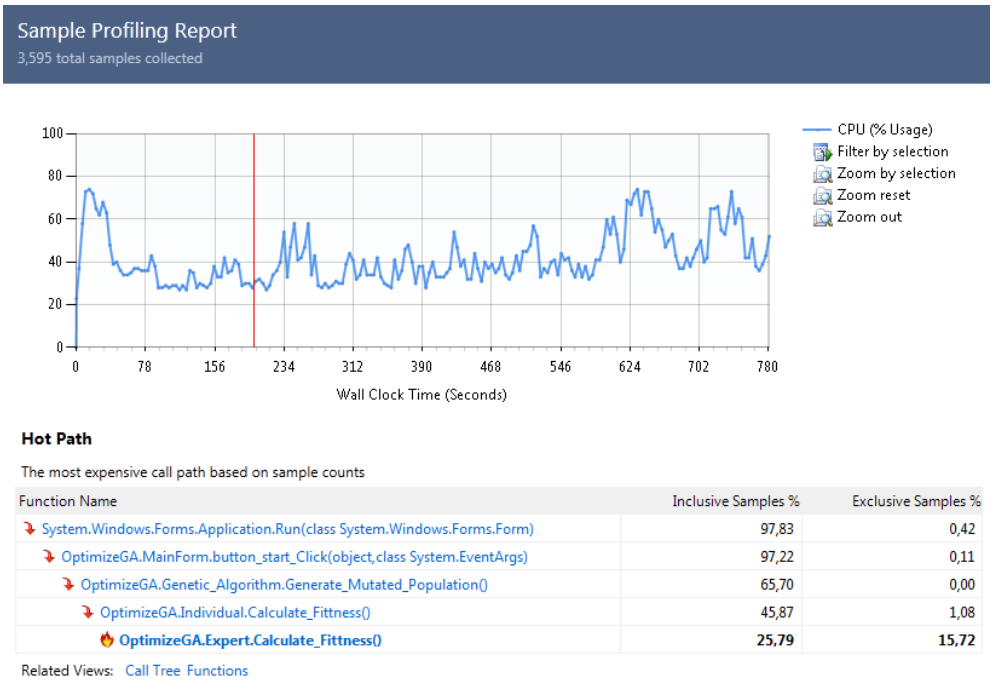
Ebben az esetben a program futása 200000 iteráció után került leállításra. A kapott megoldás azt mutatja, hogy az algoritmus 16 büntetést nem tudott eliminálni adott iteráció számon belül, ezek főleg szétszórt objektumokból adódnak. Azonban látható, hogy ebben az esetben a futási idő már jelentős, több mint 4 nap. Ilyen nagyméretű rendszerek optimalizálása esetén kiemelt figyelmet kell fordítani a büntetőértékek meghatározására, valamint arra, hogy az adott paraméterekkel kapható e megoldás vagy büntetőértékek maradnak a rendszerben a paraméterek rossz megválasztása miatt. A bejárt keresési tér megállapításához a teljes egyedszám meghaladja a számítógépes szoftverek ábrázolási tartományát, így összevetése a teljes leszámítolással nem lehetséges.

8.1.5 Az algoritmus párhuzamosítása

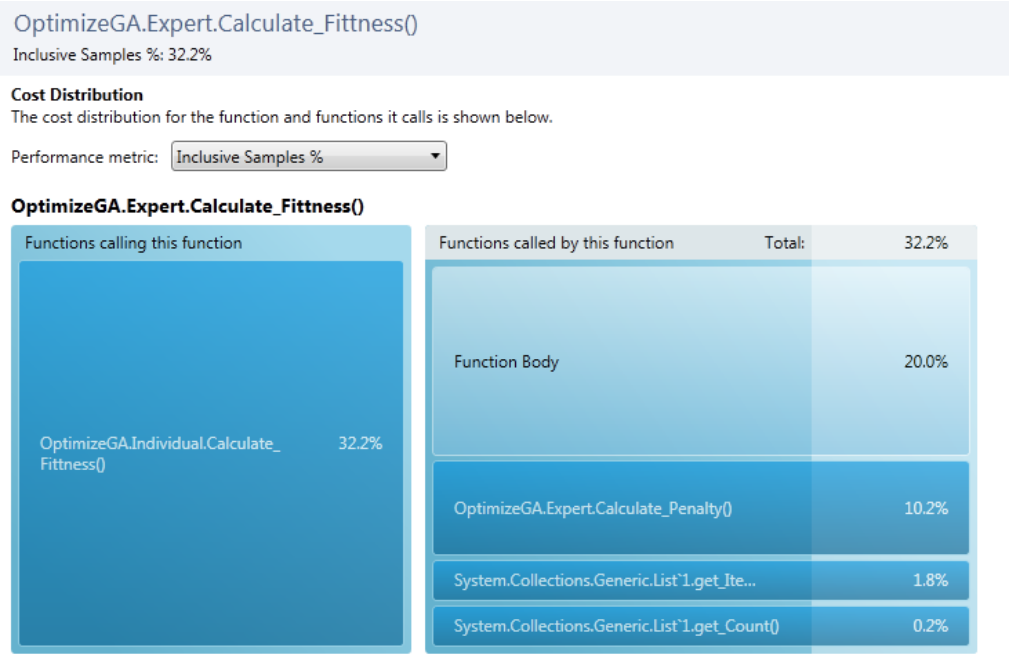
Az algoritmust nagyméretű rendszereken tesztelve látható, hogy a futási idők a rendszer méretével együtt nem lineárisan emelkednek. Felmerül az algoritmus párhuzamosításának igénye [96], amely alkalmassá teszi többmagos, többprocesszoros számítógépeken való futtatásra, grafikus kártyán való futtatásra, vagy akár felhő alapú számítógép rendszereken történő futtatásra.



57. ábra A mutáció lineáris folyamata



58. ábra A mutációs függvény költsége a VS Profiler szoftverben



59. ábra A fitnesz kalkulációs eljárás hívásai

Az algoritmusban a mutáció és a fitnesz kiszámítás a két leggyakrabban használt és a legtöbb futásidőt igénylő eljárás, ahogy ezt a Microsoft Visual Studio Profiler szoftvere is mutatja (58. ábra, 59. ábra).

A mutációs eljárás a multi-kromoszómás megoldás miatt nehezen párhuzamosítható, mivel a folyamatok az összes kromoszómán műveleteket végeznek, ennélfogva egy

közös memóriás rendszerben a zárolások és a várakozások lelassítanák a rendszert, így ennek hasznossága nehezen felbecsülhető. Viszont a fitneszfüggvény kiszámítása - mivel adatmódosítást nem végez – könnyen párhuzamosítható (60. ábra).

A párhuzamosítás sebességnövekménye viszont nem triviális, hiszen a sebességnövekedés nem lineáris, korlátos függvény. Az operációs rendszer a szálakat automatikusan ütemezi és elosztja a rendelkezésre álló erőforrások (processzormagok, processzorok) között. Viszont a szálak adminisztrációja többletköltséggel, – memóriafoglalás, közös memória védelme, kölcsönös kizárások figyelése, stb. - időveszteséggel jár, így kisméretű feladatok esetén a párhuzamosítás vesztesége meghaladhatja a nyereséget.

60. ábra A fitnesz kiszámításának párhuzamosítása

A 59. ábrán látható, hogy a fitnesz kiszámítása két fő részre oszlik, a nagyobb részt a körutak, illetve a körutak megtételéhez szükséges ciklusok kiszámítása teszi ki, a másik a büntetőfüggvények kiszámítása. A körutak kiszámítása a büntetőfüggvényekhez hasonlóan csak ugyanazon a kromoszómán végez műveleteket, valamint bemenő paraméterként az útmátrixot használja fel a számításhoz. Olyan esetekben, amikor az algoritmus nem egy számítógépen fut, hanem egy klaszteren vagy számítási felhőben a sebesség vizsgálata különösen fontos, hiszen az útmátrixot le kell másolni minden egyes feldolgozó csomópontba, amely jelentős adatmozgatással járhat a probléma méretétől függően, míg adott architektúrán belül közös memóriás megvalósítással az adatmozgatásra nincs szükség.

8.2 Az algoritmus párhuzamosításának vizsgálata

A párhuzamosítás vizsgálatához nagyméretű problémákra van szükség. Ezeket előállíthatjuk véletlenszerűen, vagy valamely ezzel a problémakörrel foglalkozó referencia adatbázisból. A vizsgálathoz végül az optimalizálásban de facto szabványnak tekinthető a Heidelbergi Egyetem Alkalmazott Matematika Tanszéke által karbantartott TSPLIB [97] egyedeket használtam fel. A vizsgálati módszer minden esetben

- ugyanazon értékkel inicializált véletlenszám generátor,
- ugyanazon tsplib adatok,
- megegyező számú iteráció.

A megvizsgált futtatások egy 2.8 GHz-es Intel i7-920 négy processzormagos hyperthreading technológiával ellátott számítógépen készültek.

8.2.1 Az algoritmus párhuzamosításának vizsgálata kis egyedszám mellett

Első vizsgálatot kis egyedszám mellett végeztem el. A vizsgálat bemenő paraméterei a következőképpen alakultak:

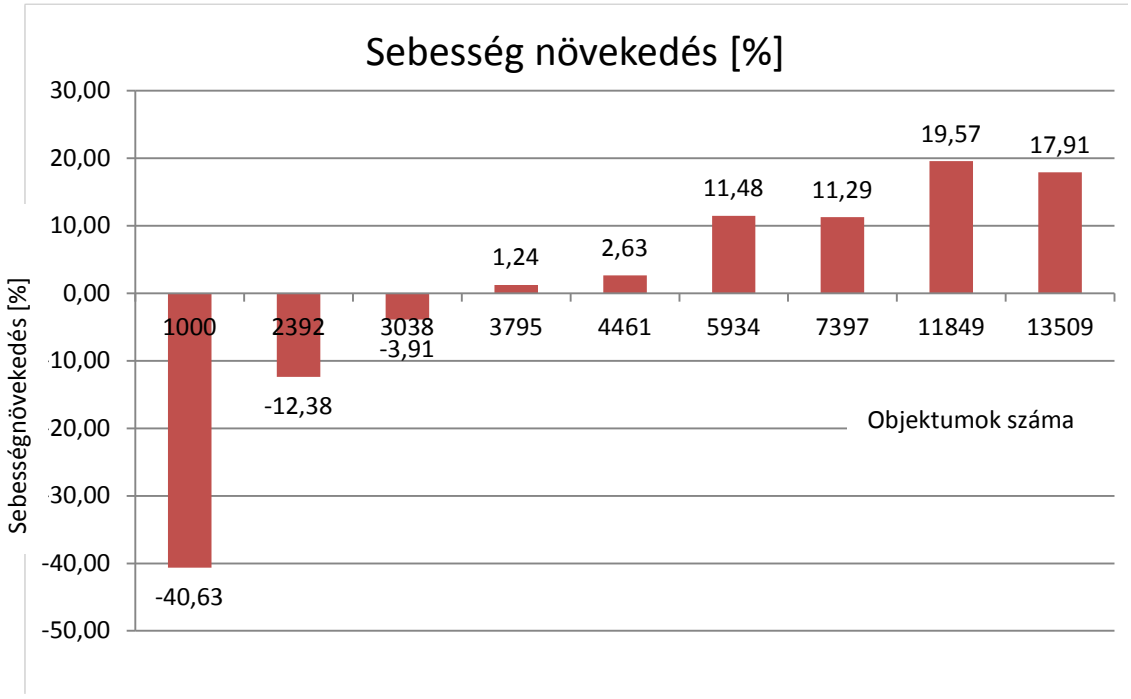
Iterációszám	100	Mutációs operátorok valószínűsége	
Populáció méret	100	Nincs mutáció	0
Véletlen egyedek száma	20	Lokális mutáció	50
Szakértők száma	30	Globális mutáció	50
Csomópontok száma	változó		
Minimális vizsgálatszám	50	Lokális operátorok	
Maximális vizsgálatszám	150	Géncsere	40
Maximális körúthossz	2000	Génbeszúrás	40
Maximális ciklusszám	10	Génszekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	2	Géncsere	45
Objektum vizsgálatok ig	5	Génszekvencia csere	45
Büntetőértékek		Génkontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	50000		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	50000		
Szakértő költsége	100000		

10. táblázat Bemenő paraméterek

A vizsgálat eredményei:

Iterációk száma	Egyed (tsplib)	Csomópontok száma	Párhuzamosított algoritmus	Futási idő [mm:ss]	Sebesség növekedés [%]
100	dsj1000	1000	nem	00:38	
100	dsj1000	1000	igen	01:04	-40,63
100	pr2392	2392	nem	01:32	
100	pr2392	2392	igen	01:45	-12,38
100	pcb3038	3038	nem	02:03	
100	pcb3038	3038	igen	02:08	-3,91
100	fl3795	3795	nem	02:43	
100	fl3795	3795	igen	02:41	1,24
100	fnl4461	4461	nem	03:15	
100	fnl4461	4461	igen	03:05	2,63
100	rl5934	5934	nem	04:32	
100	rl5934	5934	igen	04:04	11,48
100	pla7397	7397	nem	05:55	
100	pla7397	7397	igen	05:19	11,29
100	rl11849	11849	nem	12:01	
100	rl11849	11849	igen	10:03	19,57
100	usa13509	13509	nem	13:56	
100	usa13509	13509	igen	11:49	17,91

11. táblázat A párhuzamosítás vizsgálata kis egyedszámmal



61. ábra Sebességnövekedés a csomópontok számának függvényében

A 11. táblázat és az 61. ábra alapján jól látható hogy kis problémaméretek esetén a fitnesz-számoló folyamatok feldolgozásának és ütemezésének költsége meghaladja az

elérhető nyereség szintjét, így a korábban vázolt párhuzamosítás ezen a számítógépen mintegy körülbelül 3500 csomóponttól gazdaságos ilyen bemenő paraméterek mellett. Az eredményekből az is látható, hogy az algoritmus sebességének növekedése, amikor az algoritmus mind a négy processzormagot használja, esetünkben 20 százalék körüli értékre áll be. Ez az érték függ a processzor pillanatnyi terhelésétől és a háttérben futó folyamatoktól, így az érték egy bizonyos határon belül ingadozik. A vizsgált tesztfeladatok megválasztásánál figyelembe vettem, hogy valós körülmények között ennél nagyobb problémák nem, vagy csak nagyon ritkán adódnak.

A folyamatok, szálak operációs rendszer által történő adminisztrációjának emelkedő költségét megfigyelhetjük a populáció méret emelkedésével, ekkor a korlátozott erőforrásra - a processzorra - váró folyamatok ütemezése nagyobb késleltetéssel jár, így a következő vizsgálatot ötszörös populáció mérettel végeztem el.

8.2.2 Az algoritmus párhuzamosításának vizsgálata megnövelt egyedszám mellett

Ha az egyedszámot az ötszörösére emeljük, de az algoritmust futtató rendszer paraméterei nem változnak. A bemenő paraméterek:

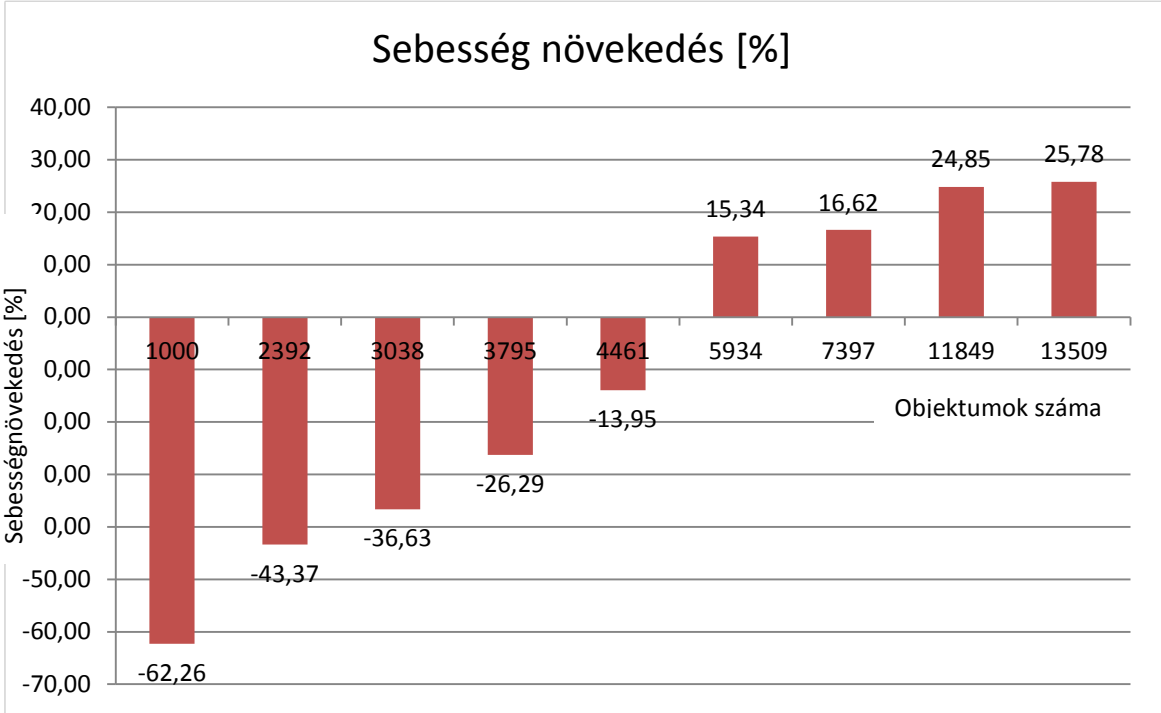
Iterációszám	50	Mutációs operátorok valószínűsége	
Populáció méret	500	Nincs mutáció	0
Véletlen egyedek száma	50	Lokális mutáció	50
Szakértők száma	30	Globális mutáció	50
Csomópontok száma	változó		
Minimális vizsgálatszám	50	Lokális operátorok	
Maximális vizsgálatszám	150	Géncsere	40
Maximális körúthossz	2000	Génbeszúrás	40
Maximális ciklusszám	10	Génszekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	2	Géncsere	45
Objektum vizsgálatok ig	5	Gén szekvencia csere	45
Büntetőértékek		Gén kontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	50000		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	50000		
Szakértő költsége	100000		

12. táblázat Bemenő paraméterek

A vizsgálat eredményei, a kiválasztott tsplib egyedeken:

Iterációk száma	Egyed (tsplib)	Csomópontok száma	Párhuzamosított algoritmus	Futási idő [mm:ss]	Sebesség növekedés [%]
50	dsj1000	1000	nem	01:20	
50	dsj1000	1000	igen	03:32	-62,26
50	pr2392	2392	nem	03:25	
50	pr2392	2392	igen	06:02	-43,37
50	pcb3038	3038	nem	04:23	
50	pcb3038	3038	igen	06:55	-36,63
50	fl3795	3795	nem	06:10	
50	fl3795	3795	igen	08:22	-26,29
50	fnl4461	4461	nem	07:18	
50	fnl4461	4461	igen	08:29	-13,95
50	rl5934	5934	nem	09:54	
50	rl5934	5934	igen	08:35	15,34
50	pla7397	7397	nem	12:52	
50	pla7397	7397	igen	11:02	16,62
50	rl11849	11849	nem	24:02	
50	rl11849	11849	igen	19:15	24,85
50	usa13509	13509	nem	28:18	
50	usa13509	13509	igen	22:30	25,78

13. táblázat A párhuzamosítás vizsgálata megnövelt egyedszámmal



62. ábra Sebesség növekedés a csomópontok számának függvényében, növelt csomópontszám esetén

A megnövelt egyedszámú vizsgálat eredményeiből látható (13. táblázat, 62. ábra) a szűk keresztmetszetet képező processzor miatt az ütemezési várakozások tovább nőnek, így a diagram értékei jobbra tolódnak. Az előző vizsgálathoz képest majdnem kétszer nagyobb a párhuzamosítás megtérülési csomópontszáma. A kapott eredményeket interpolálva, mintegy 5200 csomópontra tehető. Nyilvánvaló tehát, hogy ilyen nagyméretű problémák megoldására gazdaságos lehet sokprocesszoros számítógépek vagy számítási felhők alkalmazása, valamint speciális GPU farmok használata.

8.3 Érzékenységvizsgálatok

8.3.1 A populáció-méret vizsgálata

Felmerül a kérdés, hogy hogyan változik a végeredmény, ha nem az iterációk számát növeljük meg, hanem a populációméretet változtatjuk, illetve megváltoztatjuk a véletlen generált egyedek arányát a populáción belül. Fontos kérdés az is, hogy hogyan képes az algoritmus kilépni a lokális optimumokból, ha csökkentjük a véletlen generált egyedek arányát. Ennek megválaszolására elvégeztem az algoritmus érzékenységvizsgálatát a populáció méretre vetítve. A kiindulási alap 8.1.2 fejezet három szakértős tesztfeladata volt.

Iterációszám	1000	Mutációs operátorok valószínűsége	
Populáció méret	változó	Nincs mutáció	0
Véletlen egyedek száma	változó	Lokális mutáció	50
Szakértők száma	3	Globális mutáció	50
Csomópontok száma	48		
Minimális vizsgálatszám	16	Lokális operátorok	
Maximális vizsgálatszám	18	Géncsere	40
Maximális körúthossz	2000	Génbeszúrás	40
Maximális ciklusszám	1	Génszekvencia csere	20
Minimális ciklusszám	1		
Közelségi határérték	1	Globális operátorok	
Objektum vizsgálatok tól	1	Géncsere	45
Objektum vizsgálatok ig	1	Génszekvencia csere	45
Büntetőértékek		Génkontrakció	5
Szétszórt objektum	10000		
Közeli vizsgálatok	0		
Kevesebb vizsgálat	50000		
Több vizsgálat	50000		
Több ciklus	50000		
Szakértő költsége	0		

14. táblázat Az érzékenységvizsgálat bemenő paraméterei.

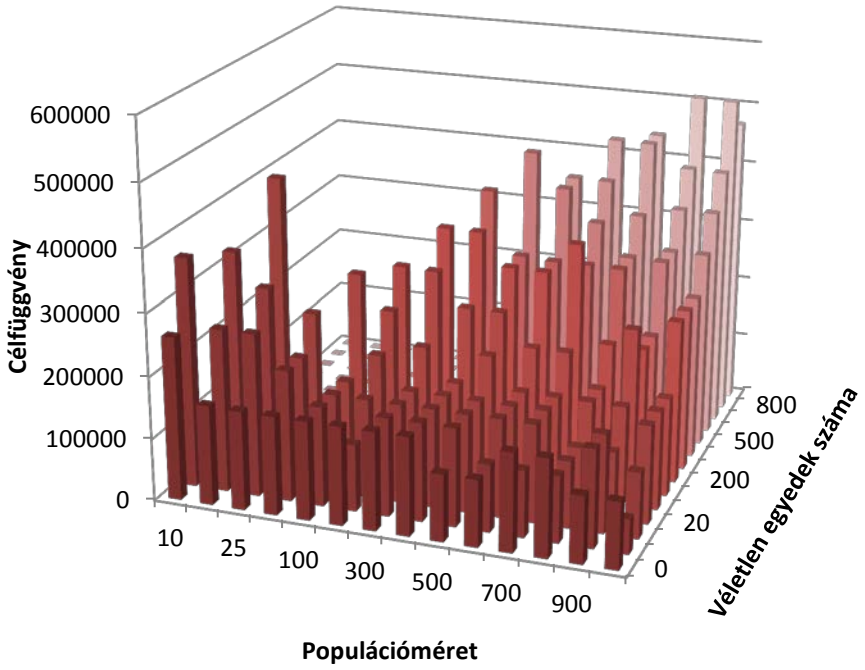
Az érzékenységvizsgálat eredményei: (a legjobb célfüggvény érték zöld színnel jelölt)

		Populáció méret						
		10	20	25	50	100	200	300
Véletlen egyedek száma	0	262147,41	160300,48	158774,1	158330,53	158421,46	158033	158219,08
	5	367630,06	262188,85	261761,12	210446,86	160049,98	107682,99	158103,69
	10	-	365443,7	314261,95	211049,09	158930,41	158805,26	158249,05
	20	-	-	470334,75	263240,72	159747,81	209857,65	158833,65
	50	-	-	-	-	314076,62	262342,62	210368,71
	100	-	-	-	-	-	315404,74	313768,66
	200	-	-	-	-	-	-	366780,29
	300	-	-	-	-	-	-	-
	400	-	-	-	-	-	-	-
	500	-	-	-	-	-	-	-
	600	-	-	-	-	-	-	-
	700	-	-	-	-	-	-	-
	800	-	-	-	-	-	-	-
	900	-	-	-	-	-	-	-

15. táblázat Az célfüggvény változása a populáció méret és a véletlen egyedszám függvényében (1)

		Populáció méret						
		400	500	600	700	800	900	1000
Véletlen egyedek száma	0	157955,21	107545,32	107215,26	158309,14	157414,73	107832,6	106779,7
	5	158538,91	157945,82	108574,83	107437,69	107556,71	157603,09	56174,49
	10	158907,74	158227,07	158565,84	158039,39	107418,38	157906,86	106265,12
	20	159237,94	158423,14	158033,83	158684,29	106752,63	107745,62	157781,22
	50	159222,57	210896,47	159411,16	158886,62	158843,58	158450,84	158429,52
	100	261921,75	261951,37	211042,73	211105,01	158283,73	261510,72	158332,04
	200	366377,61	315208,69	314279,49	364707,29	211106,92	210566,73	261773,05
	300	416787,63	316873,58	313776,11	314413,83	313991,51	212120,57	261151,58
	400	-	468406,88	415795,6	365993,76	315628,49	314367,26	261779,57
	500	-	-	417931,19	417772,18	367165,7	313930,85	315149,76
	600	-	-	-	470131,99	469421,26	366624,13	366926,91
	700	-	-	-	-	469115,21	418974,4	417495,43
	800	-	-	-	-	-	521295,51	520073,15
	900	-	-	-	-	-	-	469285,56

16. táblázat Az célfüggvény változása a populáció méret és a véletlen egyedszám függvényében (2)



63. ábra A célfüggvény változása a populáció-méret és a véletlen egyedszám függvényében

Az eredményekből látható, hogy ennél a problémánál alacsony populáció méret esetén a véletlen egyedek hatása nem érvényesül, inkább ront a célfüggvény értékén, hiszen a bejárt keresési tér kicsi. Nagyobb populációméretnél már észrevehető a véletlenszerűen generált egyedek hatása. A vizsgálat alapján javasolt arányuk a populáció méretének 2-3 százaléka.

Ha megvizsgáljuk a problémát azonos méretű keresési térben:

Iterációszám	Populáció méret	Vizsgált egyedszám
10000	10	100000
5000	20	100000
4000	25	100000
2000	50	100000
1000	100	100000
500	200	100000
333	300	99900
250	400	100000
200	500	100000
166	600	99600
142	700	99400
125	800	100000
111	900	99900
100	1000	100000

17. táblázat Iterációszám meghatározása

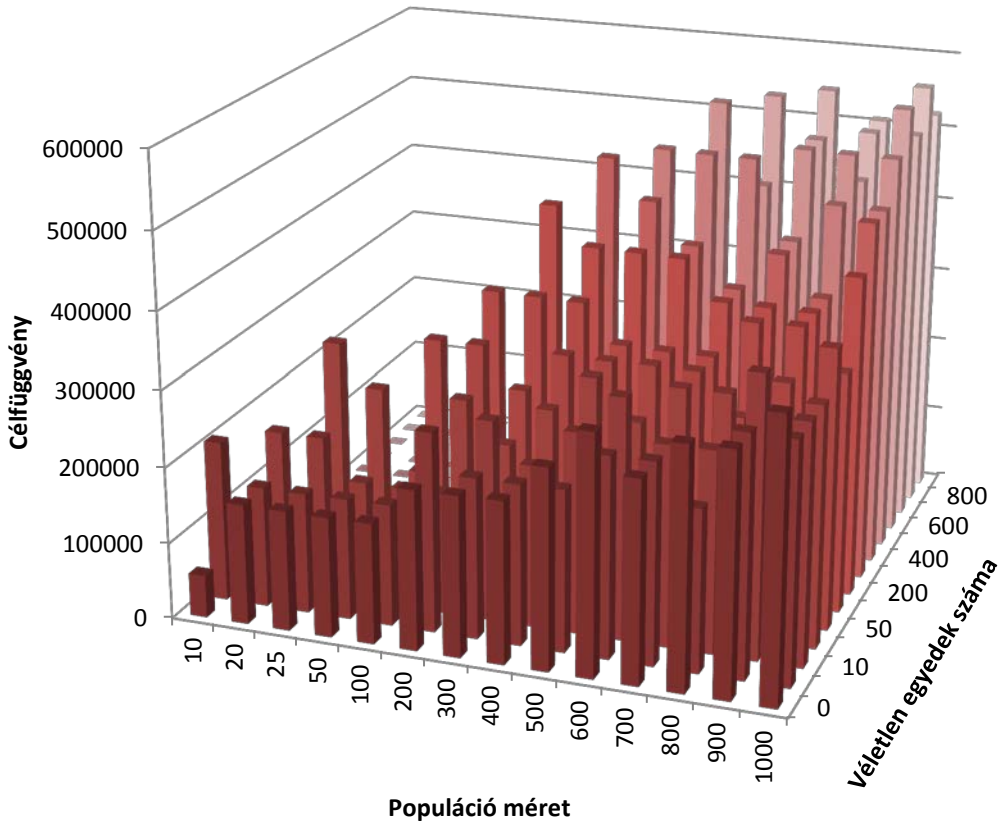
A következő eredményeket kapjuk:

		Populáció méret						
		10	20	25	50	100	200	300
Véletlen egyedek száma	0	55951,46	158334,55	158008,94	157770,09	158421,46	209782,38	210728,87
	5	210556,4	158660,78	158357,34	159260,35	160049,98	262039,25	211126,09
	10	-	210818,29	211100,73	159313,42	158930,41	160241,57	263370,5
	20	-	-	315210,41	262172,15	159747,81	262550,83	211108,47
	50	-	-	-	-	314076,62	314856,48	262600,5
	100	-	-	-	-	-	366792,2	366268,57
	200	-	-	-	-	-	-	468790,58
	300	-	-	-	-	-	-	-
	400	-	-	-	-	-	-	-
	500	-	-	-	-	-	-	-
	600	-	-	-	-	-	-	-
	700	-	-	-	-	-	-	-
	800	-	-	-	-	-	-	-
	900	-	-	-	-	-	-	-
		10000	5000	4000	2000	1000	500	333
		Iterációszám						

18. táblázat Az célfüggvény változása a populáció-méret és a véletlen egyedszám függvényében (1)

		Populáció méret						
		400	500	600	700	800	900	1000
Véletlen egyedek száma	0	212248,01	262190,95	313917,48	264179,11	314288,39	315548,66	366054,69
	5	211217,26	212174,58	263090,52	263758,78	212794,85	314842,36	314462,27
	10	212798,04	263654,66	315512,02	263160,72	264119,96	366713,93	315107,07
	20	264379,08	313330,9	263381,38	313940,42	314754,14	265100,54	315326,35
	50	314916,29	314182,9	315912,94	315163,88	263890,49	315607,6	365578,87
	100	365307,11	315883,01	314519,21	314939,88	365646,09	367097,56	314419,16
	200	419323,32	418746,37	417607,54	366973,62	366634,96	365900,83	418172,79
	300	520380,75	469494,91	417193,98	366382,58	418090,33	366859,98	470574,46
	400	-	521841,86	520747,43	520786,31	418626,89	471016,59	469912,09
	500	-	-	573818,67	470332,43	522793,76	521398,17	521406,94
	600	-	-	-	573804,97	521275,66	471914,58	571301,22
	700	-	-	-	-	572568,34	521201,2	522919,64
	800	-	-	-	-	-	522945,92	572292,19
	900	-	-	-	-	-	-	522050,56
		250	200	166	142	125	111	100
		Iterációszám						

19. táblázat Az célfüggvény változása a populáció méret és a véletlen egyedszám függvényében (2)



64. ábra A célfüggvény változása a populáció méret és a véletlen egyedszám függvényében

Az eredményekből kitűnik, hogy ebben az esetben már megfigyelhető a lokális optimumok elkerülésére generált véletlen egyedek hatása. Az arányuk optimális értéke a 0-3 százalékos sávban ingadozik, néhány kiugró érték kivételével, amelyek elérhetik a 10 százalékot is. A véletlen egyedek számának populáció méretre vetített aránya egyértelműen nem meghatározható, így értékét az adott feladathoz kísérletileg kell beállítani. Az elvégzett vizsgálatok szerint azonban a 3 százalékos értéket nem ajánlott túllépni. Viszont az eredmények rámutatnak arra a tényre, hogy az elemszámot nem érdemes a csomópontok számának sokszorosára beállítani, mert az algoritmus futási ideje indokolatlanul megnövekszik adott iteráció szám mellett. Ha viszont az iteráció számot csökkentjük, nem érvényesül a szelekció hatása, amely a nem megfelelő egyedeket eltávolítja a populációból.

8.3.2 A lokális és globális operátorok arányának vizsgálata

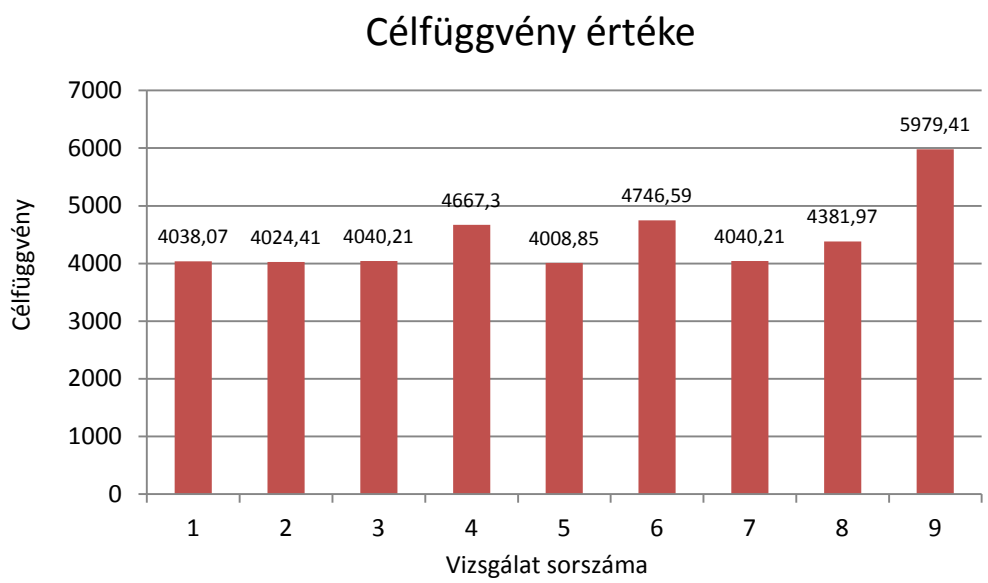
A következő vizsgálatok mutatják az algoritmus érzékenységét a lokális és globális operátorok arányának változtatására. Minden esetben a lokális és globális operátorok aránya 10-90 százalék között változik 10 százalékos lépcsőkben.

8.3.2.1 Két szakértős egyszerű tesztfeladat

Az első vizsgált példa a 8.1.1 fejezetben ismertetett két szakértős tesztfeladat.

Futtatás sorszáma	Lokális operátorok aránya [%]	Globális operátorok aránya [%]	Célfüggvény értéke	Iterációk száma
1	10	90	4038,07	7400
2	20	80	4024,41	8000
3	30	70	4040,21	1850
4	40	60	4667,3	9100
5	50	50	4008,85	9100
6	60	40	4746,59	9100
7	70	30	4040,21	8100
8	80	20	4381,97	9100
9	90	10	5979,41	9100

20. táblázat A célfüggvény értékének változása az operátorok arányának függvényében



65. ábra A célfüggvény értékek változása az operátorok arányának függvényében.

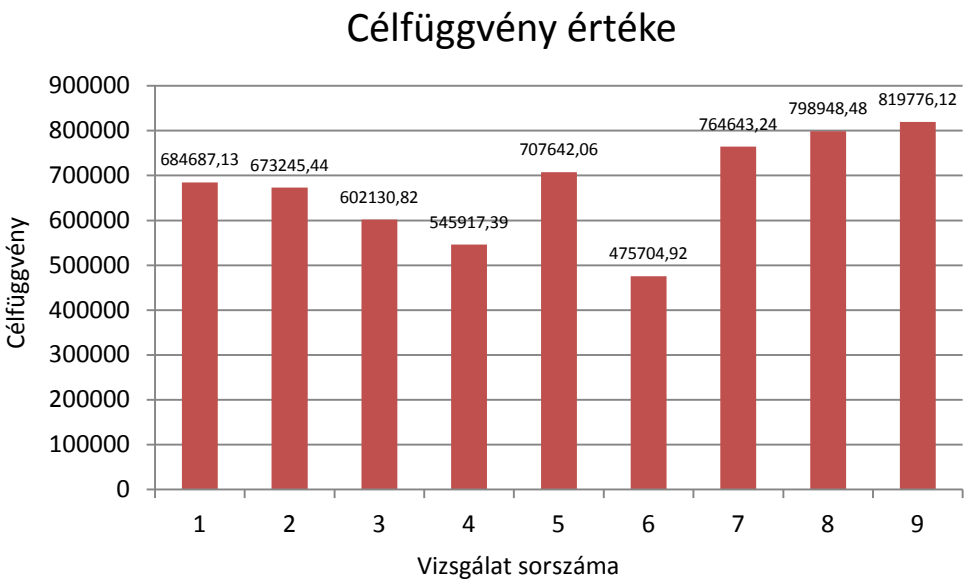
A futtatásoknál 50 százalékos aránynál elért legjobb megoldás 9100 iterációnál adódott, így a vizsgálatok is eddig az iterációig történtek. A 20. táblázat iterációk sorszáma oszlopa mutatja, hogy az adott aránynál elért érték mely iterációnál állt be. A 65. ábrán láthatók az eredmények diagram formában innen könnyen leolvasható, hogy a legjobb eredményt az 50-50 százalékos megoszlásnál kapjuk. Figyelemreméltó azonban, hogy a 30-70 százalékos aránynál már 1850 iteráció után beáll egy az optimumtól csak 0,08 százalékban eltérő megoldás.

8.3.2.2 Három szakértős összetett tesztfeladat

A következő vizsgált példa a 8.1.3 fejezetben ismertetett három szakértős összetett tesztfeladat. Ennek vizsgálata két iterációs lépcső elérése esetén is megtörtént.

Futtatás sorszáma	Lokális operátorok aránya [%]	Globális operátorok aránya [%]	Célfüggvény értéke	Iterációk száma
1	10	90	684687,13	10000
2	20	80	673245,44	10000
3	30	70	602130,82	10000
4	40	60	545917,39	10000
5	50	50	707642,06	10000
6	60	40	475704,92	10000
7	70	30	764643,24	10000
8	80	20	798948,48	10000
9	90	10	819776,12	10000

21. táblázat A célfüggvény értékének változása az operátorok arányának függvényében

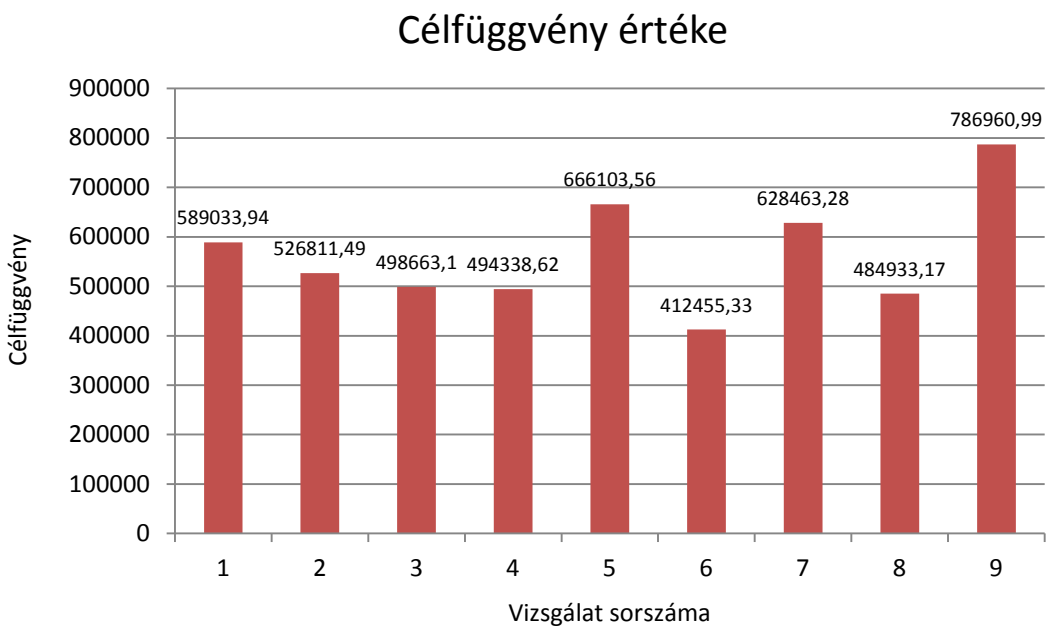


66. ábra A célfüggvény értékek változása az operátorok arányának függvényében.

Futtatás sorszáma	Lokális operátorok aránya [%]	Globális operátorok aránya [%]	Célfüggvény értéke	Iterációk száma
1	10	90	589033,94	20000
2	20	80	526811,49	20000
3	30	70	498663,1	20000
4	40	60	494338,62	20000
5	50	50	666103,56	20000
6	60	40	412455,33	20000
7	70	30	628463,28	20000
8	80	20	484933,17	20000
9	90	10	786960,99	20000

22. táblázat A célfüggvény értékének változása az operátorok arányának függvényében

Az első vizsgálat - 10000 iteráció esetén (21. táblázat, 66. ábra) – adatai viszont azt mutatják, hogy ebben az esetben az 50-50 százalékos megoszlás nem ideális, a 60-40 százalékos megoldás adja a legjobb értéket. A vizsgálatot tovább futtatva 20000 iterációig a 22. táblázatban lévő eredményeket kapjuk.



67. ábra A célfüggvény értékek változása az operátorok arányának függvényében.

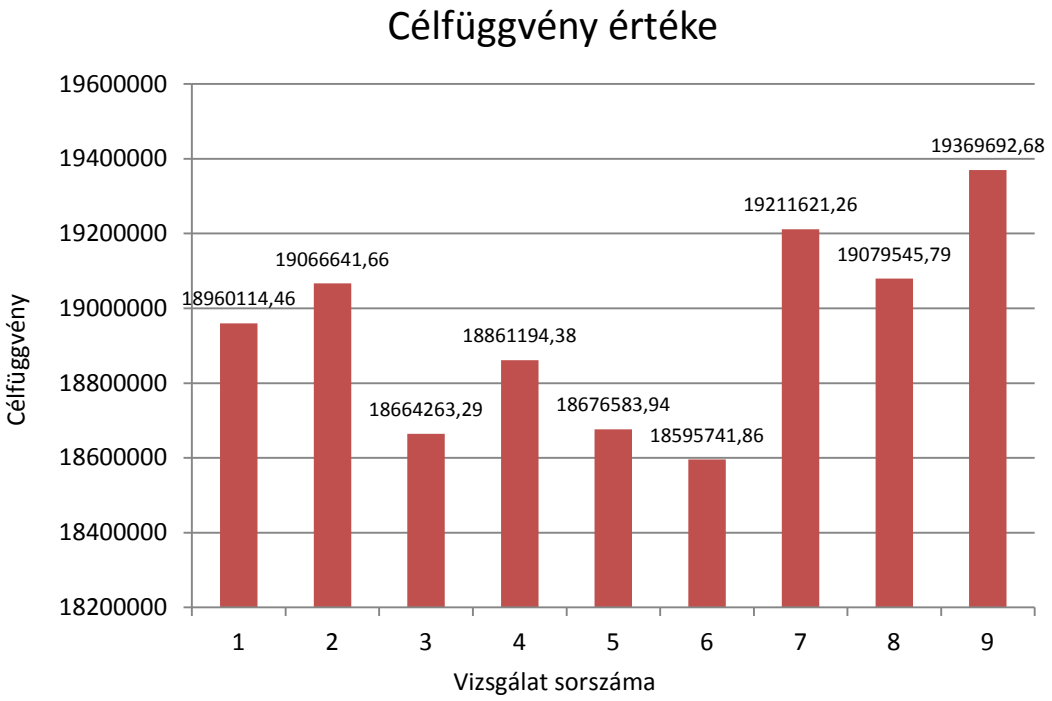
A kapott adatokból (22. táblázat, 67. ábra) látható, hogy a 60-40 százalékos lokális-globális operátorok aránya magasan a legjobb megoldást szolgáltatja, azonban a 2. helyre a 20-80 százalékos arány került, viszont az 50-50 százalékos arány megoldása nem javult.

8.3.2.3 Három szakértős nagyméretű tesztfeladat

A vizsgált példa a 8.1.4 fejezetben ismertetett három szakértős nagyméretű tesztfeladat, amely 1000 vizsgálandó objektumot tartalmaz.

Futtatás sorszáma	Lokális operátorok aránya [%]	Globális operátorok aránya [%]	Célfüggvény értéke	Iterációk száma
1	10	90	18960114,5	2000
2	20	80	19066641,7	2000
3	30	70	18664263,3	2000
4	40	60	18861194,4	2000
5	50	50	18676583,9	2000
6	60	40	18595741,9	2000
7	70	30	19211621,3	2000
8	80	20	19079545,8	2000
9	90	10	19369692,7	2000

23. táblázat A célfüggvény értékének változása az operátorok arányának függvényében



68. ábra A célfüggvény értékek változása az operátorok arányának függvényében.

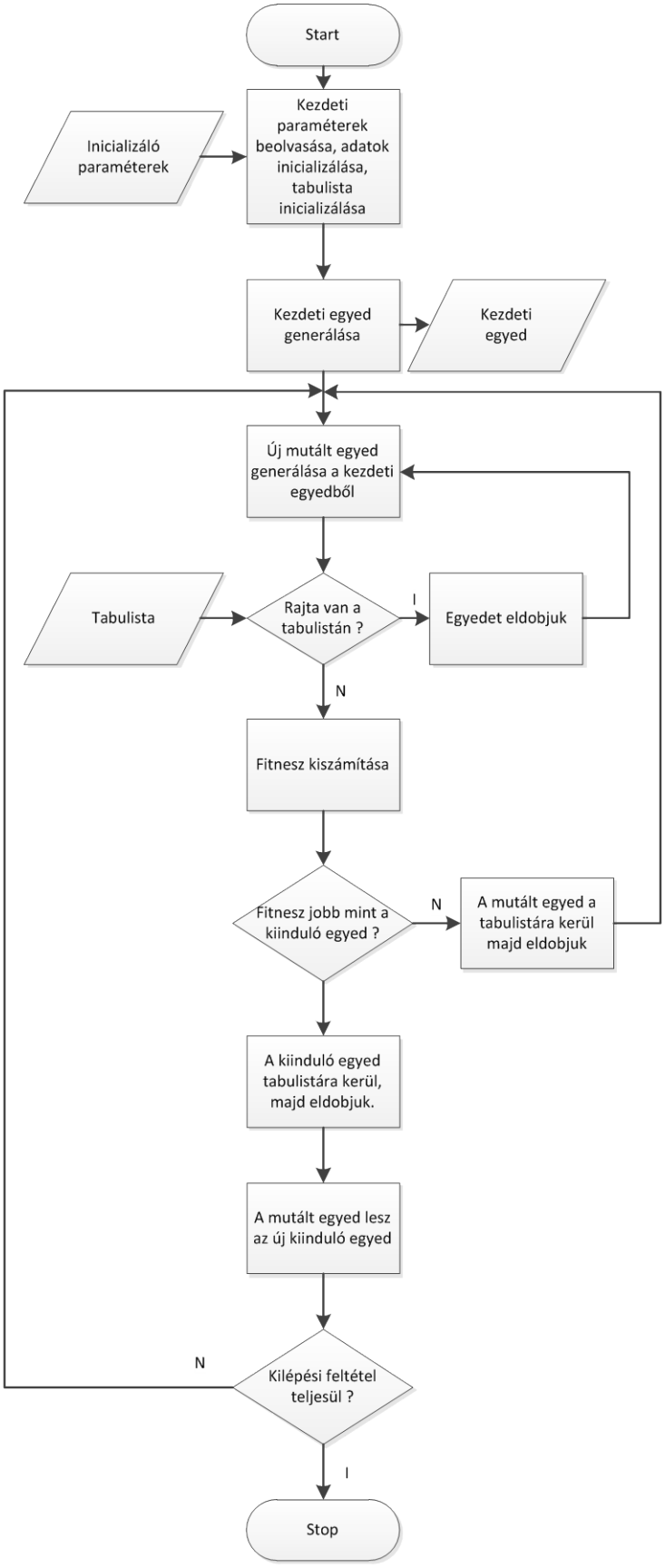
A kapott adatokból (23. táblázat, 68. ábra) látható, hogy ebben az esetben is a 60-40 százalékos arány adja a legjobb célfüggvény-értéket, adott iteráció szám mellett viszont itt az 50-50 százalékos aránynál a célfüggvény értéke a második legjobb.

A vizsgálatok eredményeként levonható a következtetés: az operátorok arányának ideális megválasztása feladatfüggő. Ajánlott a teljes futtatás előtt – amennyiben ez hosszú idejű – tesztfuttatásokkal megállapítani a lokális és globális operátorok ideális arányát.

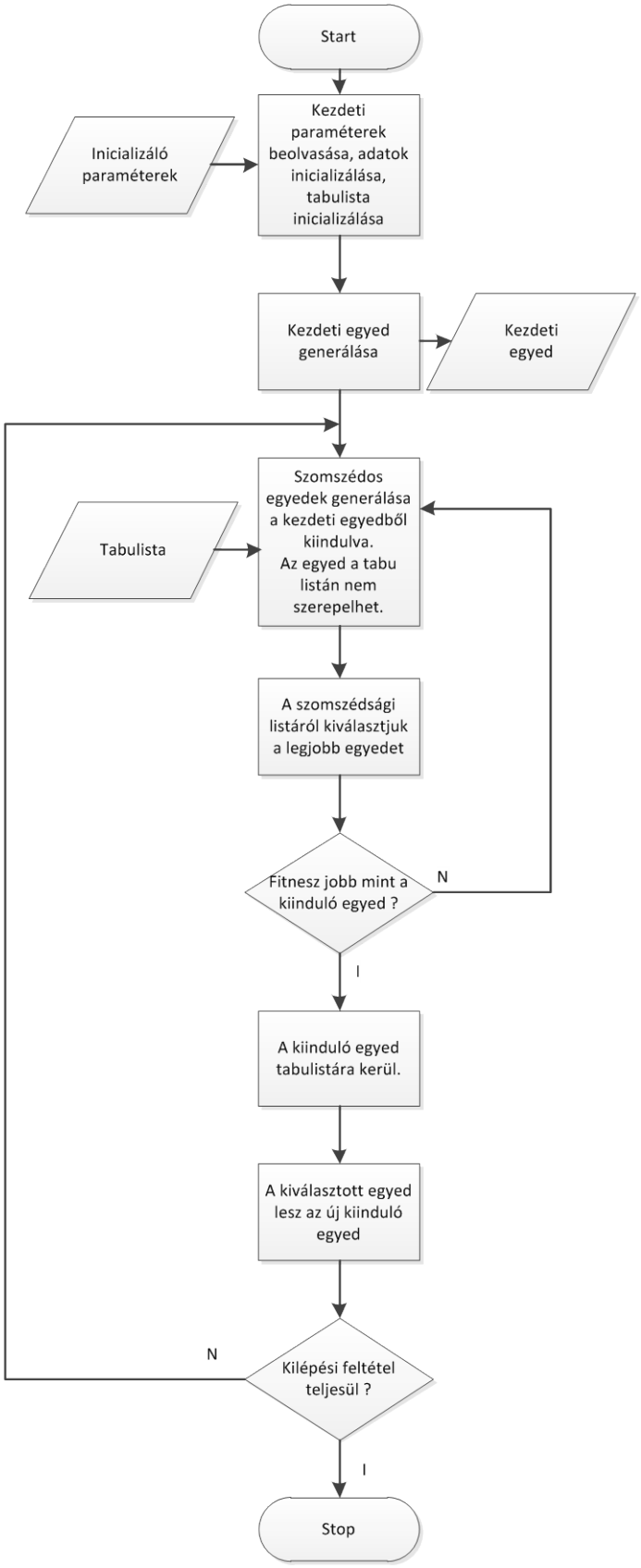
8.4 A kidolgozott algoritmus összehasonlítása a tabu keresés algoritmussal

A tabu keresés egy széles körben használt problémafüggetlen metaheurisztikus kereső eljárás. Többféle változata, illetve kiegészítése ismert.

Az egyszerűsített tabu keresés (69. ábra), amelyet főleg a mesterséges intelligencia tudományterületén alkalmaznak, nem használ szomszédsági listát. A kereső operátor az evolúciós algoritmushoz hasonlóan a mutáció. A keresés során egy tabu listát tartunk fenn, amely a legutóbb megvizsgált néhány megoldásból áll. A tabu lista mérete az algoritmus egyik paramétere. Az új populáció, azaz az új aktuális megoldás kiválasztásához először megnézzük, hogy a mutációval létrehozott új elem szerepel-e a tabu listában. Ha igen, akkor nem fogadjuk el, egyébként pedig ha nem rosszabb, mint a régi megoldás, akkor elfogadjuk. A régi megoldás a tabu listára kerül, és a tabu lista legrégebbi eleme törlődik [98][99].



69. ábra Az egyszerűsített tabu keresés algoritmusa



70. ábra A tabu keresés algoritmusa szomszédsági listával

A tabu keresés széleskörűen használt változata az egyszerűsített tabukereséssel szemben egy szomszédsági listát használó kereső eljárás (70. ábra) [100]. A kezdeti egyed generálása után előállítjuk annak szomszéd egyedeit. Megvizsgáljuk, hogy a szomszédos egyedek rajta vannak e a tabulistán, ha igen eldobjuk, ha nem, akkor ezt az egyedet a tabu listára helyezzük. Majd a szomszédsági listáról kiválasztjuk a legjobb egyedet ez lesz az új „kezdeti egyed”. Az iterációs ciklus a kilépési feltétel eléréséig folytatjuk.

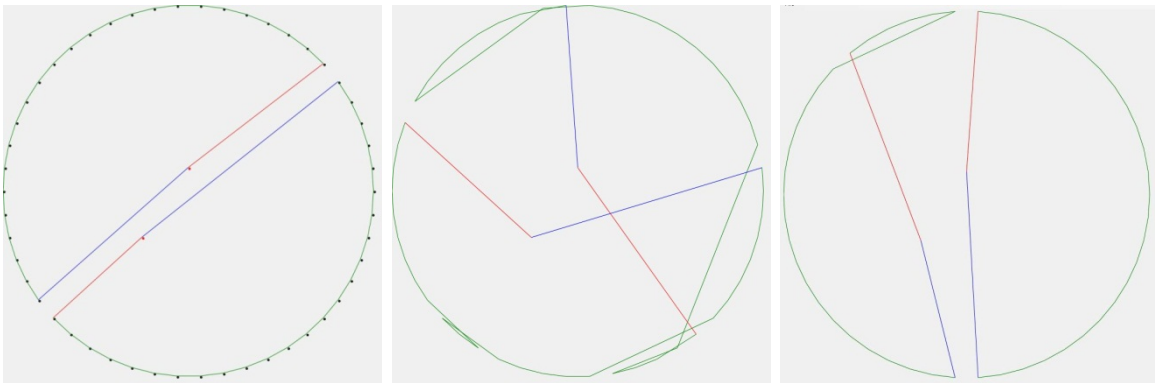
A tabu keresés egy általános kereső módszer, a módszer futtatásakor ugyanazokat az operátorokat és büntetőfüggvényeket használtam, mint az evolúciós programozási algoritmusnál (későbbiekben EP). A tabu keresés algoritmusát teszteltem a disszertációban korábban szereplő négy feladattal. A tabulista hosszát mind a négy esetben 500-ra választottam az egyszerű tabu keresés algoritmusánál (későbbiekben TE), mivel néhány kísérletből ez adta a legjobb eredményt futási sebesség és célfüggvény érték szempontjából. A tabu keresés szomszédsági listás algoritmusánál (későbbiekben TS) a szakirodalom alapján [101] a tabulista méretét 5-re, míg a vizsgált szomszédsági lista elemszámát 40-re választottam meg. Míg az evolúciós programozási algoritmusnál az első három problémánál a populációméret 500, 50 véletlen generált egyed mellett, míg az utolsó problémánál, mivel ez nagyméretű probléma, a populációméret 10, 1 véletlenszerűen generált egyeddel.

8.4.1 Egyszerű két szakértős tesztfeladat vizsgálata

Az első tesztfeladat kisméretű probléma két szakértővel, 50 objektummal, objektumonként 1 vizsgálattal (8.1.1. fejezet).

Tesztfeladat	Algoritmus	Célfüggvény	Relatív eltérés	Megoldási idő	Büntetések	Iterációszám
1	EP	4008,85	100,00%	0:13:06	0	9100
	TE	5491,27	73,00%	0:13:22	0	15000
	TS	4200,52	95,43%	0:13:27	0	15500

24. táblázat Tabu keresés és az evolúciós programozási algoritmus összehasonlítása az első tesztfeladat esetén



a.) Evolúciós programozási algoritmus, EP

b.) Tabu keresési algoritmus, TE

c.) Tabu keresési algoritmus, TS

71. ábra Futási eredmények

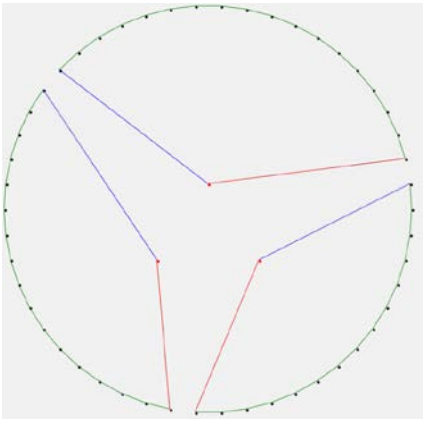
A kapott eredményekből (24. táblázat, 71. ábra) látható, hogy az evolúciós programozási algoritmushoz képest a tabu keresés algoritmusa rosszabb eredményt ad ugyanannyi futásidő (mintegy 13 perc) alatt, a TE a célfüggvény értékét 73 százalékra, a TS algoritmus azonban 95,43 százalékra közelíti meg.

8.4.2 Második tesztfeladat vizsgálata

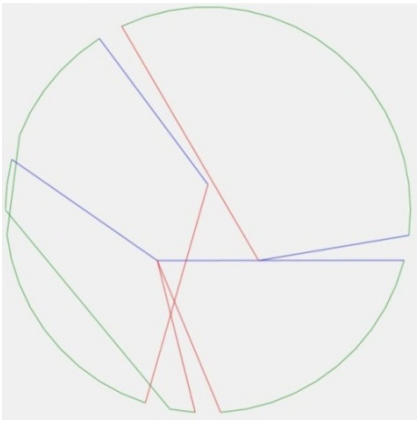
A második tesztfeladat kisméretű probléma három szakértővel, 50 objektummal, objektumonként 1 vizsgálattal (8.1.2. fejezet)

Tesztfeladat	Algoritmus	Célfüggvény	Relatív eltérés	Megoldási idő	Büntetések	Iterációszám
2	EP	4483,96	100,00%	0:24:35	0	16434
	TE	159581,79	2,81%	0:28:57	3	30000
	TS	55365,99	8,09%	0:26:40	1	30000
	TS*	5232,89	85,68%	0:27:59	0	30000

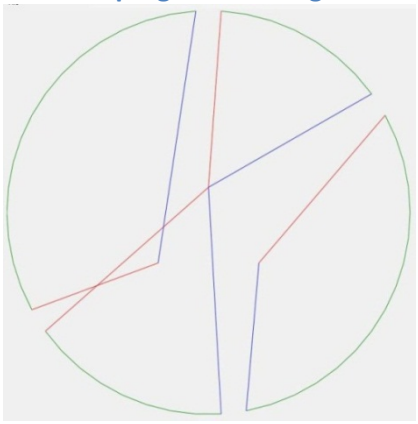
25. táblázat Tabu keresés és evolúciós programozási algoritmus összehasonlítása a második tesztfeladat esetén



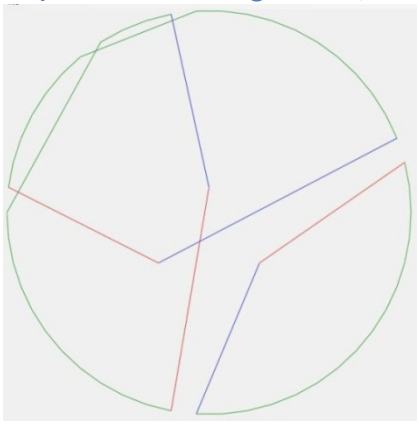
a.) Evolúciós programozási algoritmus, EP



b.) Tabu keresési algoritmus, TE



c.) Tabu keresési algoritmus, TS



d.) Tabu keresési algoritmus, TS* (legjobb)

72. ábra Futási eredmények

A kapott eredményekből (25. táblázat, 72. ábra) látható, hogy az evolúciós programozási algoritmushoz képest a tabu keresés algoritmusa rosszabb eredményt ad. Ennél a problémánál a tabu keresés megoldása jelentősen rosszabb, mint az evolúciós programozási feladat megoldása, a célfüggvény értékét a TE csak mintegy

2,81 százalékra a TS 8,09 százalékra közelíti meg. Az itt tapasztalt nagy relatív eltérés miatt a tabukeresési algoritmust többször is lefuttattam, minden alkalommal más értékkel inicializálva a véletlenszám generátort (26. táblázat). Az eredmények azt mutatták, hogy míg a TE algoritmusnak nem, a TS algoritmusnak azonban sikerült elkerülni a lokális optimumot (csillaggal jelölt), ekkor 85,68 százalékra megközelítette az evolúciós programozási algoritmust. A TS algoritmus a lokális optimumokat a szomszédsági lista segítségével általában elkerüli.

Futtatás	Célfüggvény TE	Célfüggvény TS
1*	159532,87	5232,89
2	158366,67	55418,98
3	159189,03	56014,61
4	158633,57	55633,31
5	157738,56	106964,54
6	160244,28	55791,98

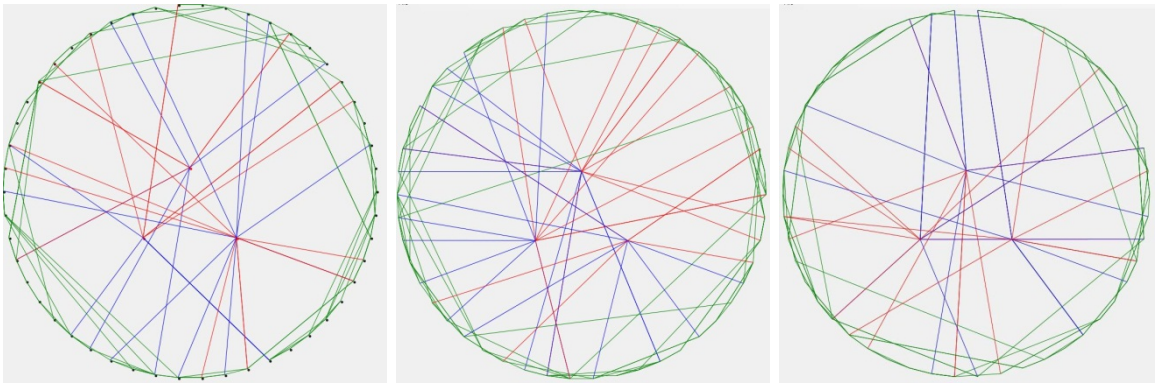
26. táblázat Tabu keresési algoritmus futtatásai

8.4.3 Harmadik tesztfeladat vizsgálata

A harmadik tesztfeladat közepes méretű probléma három szakértővel, objektumonként 2-4 vizsgálattal. A evolúciós programozási algoritmus 200000 iterációig futott a tabukeresési algoritmusok 650000 iterációig, a TE algoritmus evolúciós programozási algoritmussal azonos ideig, míg a TS algoritmus futása azonos iterációszám mellett több időt vett igénybe (8.1.3. fejezet).

Tesztfeladat	Algoritmus	Célfüggvény	Relatív eltérés	Megoldási idő	Büntetések	Iterációszám
3	EP	217534,75	100,00%	9:01:12	11	200000
	TE	495470,75	43,90%	9:02:10	14	650000
	TS	443791,30	49,01%	9:59:00	14	650000

27. táblázat Tabu keresés és evolúciós programozási algoritmus összehasonlítása a harmadik tesztfeladat esetén



a.) Evolúciós programozási algoritmus | b.) Tabu keresési algoritmus TE | c.) Tabu keresési algoritmus TS

73. ábra Futási eredmények

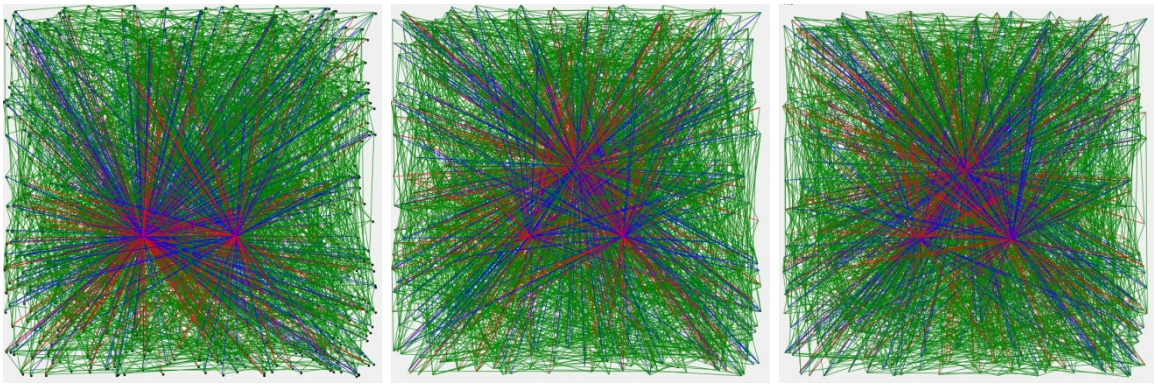
A kapott eredményekből (27. táblázat, 73. ábra) látható, hogy az evolúciós programozási algoritmushoz képest a tabu keresési algoritmusok rosszabb eredményt adnak. A TE algoritmus ugyanannyi futásidő alatt, a tabu keresés a célfüggvény értékét 43,9 százalékra közelíti meg. A TS algoritmus a TE algoritmussal azonos iterációszámig futtatva annál jobb eredményt ad, azonban az evolúciós programozási algoritmus eredményét csak 49,01 százalékra közelíti meg.

8.4.4 Negyedik tesztfeladat vizsgálata

A negyedik tesztfeladat nagyméretű méretű probléma három szakértővel, 1000 objektummal objektumonként 2-4 vizsgálattal. A evolúciós programozási algoritmus 5500 iterációig, a tabukeresési algoritmus TE 29000, a TS 5180 iterációig futott, az evolúciós programozási algoritmussal azonos ideig (8.1.4. fejezet).

Tesztfeladat	Algoritmus	Célfüggvény	Relatív eltérés	Megoldási idő	Büntetések	Iterációszám
4	EP	138245712,9	100,00%	0:57:05	631	5500
	TE	138781405,3	99,61%	0:57:46	645	29000
	TS	140099069,8	98,68%	0:58:01	595	5180

28. táblázat Tabu keresés és evolúciós programozási algoritmus összehasonlítása a negyedik tesztfeladat esetén



a.) Evolúciós programozási algoritmus **b.) Tabu keresési algoritmus TE** **b.) Tabu keresési algoritmus TS**

74. ábra Futási eredmények

A kapott eredményekből (28. táblázat, 74. ábra) látható, hogy az evolúciós programozási algoritmushoz képest a tabu keresés mindkét algoritmusa kismértékben rosszabb eredményt ad. Ennél a problémánál viszont a tabu keresés megoldása nem tér el jelentősen az evolúciós programozási feladat megoldásától. A célfüggvény értékét a TE algoritmus 99,61 százalékra a TS algoritmus 98,68 százalékra megközelíti.

8.4.5 Összegzés

Az egyszerűsített tabu keresés, valamint a szomszédsági listás tabu keresés sem adott jobb eredményt a kidolgozott evolúciós programozási algoritmusnál egyik esetben

sem, habár az utolsó esetben nagymértékben megközelítette az evolúciós programozási megoldás eredményét. A 2. példában pedig csak a megoldás töredékét szolgáltatta azonos időn belül, a lefuttatott tesztek azt mutatták, hogy ez nemcsak a véletlen szórás eredménye, hiszen több futtatás is hasonló eredményeket produkált. Itt azonban a szomszédsági listás algoritmusnak sikerült elkerülni a lokális optimumba „ragadást". A tabu keresés egyik előnye, hogy nem számítja ki fölöslegesen egyes, már megvizsgált elemek eredményét, ez ebben az esetben sokszor hátrányává válik, hiszen az operátorok által előállított elemek szétszórtsága igen magas.

Míg általános függvényszerű, folytonos, vagy néhány dimenziós diszkrét megoldás keresésekor könnyen megvalósítható a szomszédság fogalma a keresési térben - ami az egy lépéssel elérhető újabb megoldás - itt ezen elemek halmaza nagyobb problémáknál, gyakorlatilag olyan magas hogy teljes meghatározásuk nem megvalósítható. Így a szomszédsági listás kereséshez a teljes szomszédság nem állítható elő, hanem csak egy részhalmaza vizsgálható. Ebben az esetben a szomszédos elemek az összes, jelen esetben hat operátor által egy paraméterváltozattal előállítható elemek, amik a probléma méretével együtt növekednek, hiszen az operátorokat különféle szakértőkön különféle paraméterezéssel lehet végrehajtani. Így a tabulista elemein kevés találat születik.

A gyakorlatban a tesztfuttatások során egy közepes méretű problémát megvizsgálva a tabulistán nem volt egyetlen találat sem. Viszont a tabulistán való keresés, az egyedek összehasonlítása, főleg nagyméretű egyedeknél, valamint az egyedek elhelyezése a tabulistára (memóriafoglalás, egyed átmásolása) olyan sok időt vehet igénybe, amely összemérhető az elem fitneszfüggvényének kiszámításával.

9 Összefoglalás

A kutatómunkám kitűzött célja a hozzárendelési feladatok logisztikai ráfordítás alapján történő optimalizálása hálózatszerűen működő, műszaki felügyeletet és karbantartást ellátó rendszerekben. Ezen cél elérése érdekében:

- feltártam a hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek struktúráját,
- megalkottam a különböző jellegzetes rendszermodelleket, rendszerváltozatokat,
- elkészítettem a rendszerben előforduló sok feltételes objektum-szakértő hozzárendelési probléma matematikai modelljét,
- olyan optimalizáló eljárást dolgoztam ki, amely képes a kidolgozott modell esetében a logisztika területén előforduló nagyméretű problémákat, nagy elemszámú rendszereket is kezelni.

A kutatómunka eredményei az alábbi módon foglalhatók össze tézisszerűen:

1. *Hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek általános struktúrája*
   Feltártam a hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek általános struktúráját, kitérve a kis- és nagykiterjedésű rendszerek centralizált/decentralizált működésére.
   A rendszerstruktúrában szereplő modellek alkalmasak a kis kiterjedésű regionális hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek, valamint a közepes, továbbá a nagy kiterjedésű decentralizáltan működő műszaki felügyeleti és karbantartási rendszerek leírására.
   Definiáltam a műszaki felügyeleti és karbantartó rendszerek rendszerelemeit, azok jellegzetes funkcióit, feladatköreit, logisztikai szempontból.
   Megalkottam a műszaki felügyeleti és karbantartó rendszerek jellegzetes rendszerváltozatait leíró struktúrát.

2. *Hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek matematikai modellje*
   Kidolgoztam a hálózatszerűen működő műszaki felügyeleti és karbantartási rendszerek matematikai leírását.
   Meghatároztam a rendszert leíró rendszerszintű paramétereket, valamint az egyes rendszerkomponensek paramétereit.
   A szakértők, valamint az objektumok paramétereinél ismertettem azokat a feltételeket, korlátozásokat, amelyeket az optimalizáló eljárás kidolgozásakor figyelembe kell venni.

Definiáltam a több szakértős, több körutas rendszerek problémáját, valamint a kapcsolódó logisztikai ráfordításokat.

3. *Multikromoszómás evolúciós programozáson alapuló algoritmus több körutas több szakértős hozzárendelési feladat megoldására*
   Kidolgoztam egy multikromoszómás evolúciós programozáson alapuló algoritmust, amely egyrészt alkalmas a műszaki felügyeleti és karbantartási rendszerek több szakértős, több körutas hozzárendelési problémájának megoldására, másrészt a szakértők által bejárt körutak meghatározására. Figyelembe véve a matematikai modellben megadott feltételeket és korlátozásokat, az algoritmus multikromoszómás ábrázolásmódot használ, ahol egy gén egy adott objektum adott vizsgálatát jelenti. A kidolgozott heurisztikus algoritmus lokális, valamint a kromoszómák között globális operátorokat használ a körutak meghatározására. Az egyedek jóságát az algoritmusban büntetőfüggvények határozzák meg, amelyek az operátorokhoz hasonlóan szintén globálisak vagy lokálisak lehetnek.
   Az operátorok tervezésekor megvizsgáltam az egyes kontrakciós operátorok működését, hatásukat a célfüggvény konvergenciájára.

4. *A kifejlesztett algoritmus vizsgálata, jóságának igazolása, összehasonlítása a tabu keresés algoritmusának két változatával*
   Az algoritmus vizsgálatára, jóságának igazolására implementáltam az algoritmust, amely lehetővé tette az algoritmus tesztelését, működésének elemzését kis és nagyméretű problémákon.
   Kisméretű problémákon az algoritmus a program által igazoltan helyes eredményt adott, nagyméretű problémákon a megoldás helyessége már nem igazolható, azonban a tesztelés folyamán nyilvánvalóvá vált, hogy egy processzormagon szekvenciális végrehajtással a nagyméretű problémák futásideje igen magas.
   Szükségessé vált az algoritmus párhuzamosítása, amelynek hatását a megoldás sebességére nagyméretű példákon keresztül vizsgáltam.
   A kifejlesztett alkalmazás felhasználásával az algoritmust több példán keresztül összehasonlítottam a tabu keresés egy egyszerűsített, valamint szomszédsági listás általános algoritmusával. A vizsgálatok és összehasonlítások megerősítették a kidolgozott algoritmus jóságát.

10 Az értekezés tudományos eredményeinek gyakorlati hasznosíthatósága, továbbfejlesztés lehetőségei

A kifejlesztett modell és algoritmus széleskörűen felhasználható valós hálózatszerű műszaki felügyeleti és karbantartó rendszerek optimalizálására, mint például a bevezetőben említett felvonó karbantartó rendszerek optimalizálására. Az algoritmus a büntetőfüggvények módszerével képes igen sok egyéb feltétel bevezetésével is megoldást szolgáltatni, ezáltal igen rugalmas eszköznek tekinthető hasonló rendszerek tervezésekor és optimalizálásakor. Használatával akár az is kimutatható, hogy az adott feltételrendszerhez nem létezik megvalósítható megoldás, de az algoritmus képes megadni a feltételeket legkevésbé megsértő kvázi-megoldást.

Az értekezés témaválasztásában meghatározó szerepet játszott az ÉMI-TÜV Bayern Kft. számára végzett „Felvonóvizsgáló szakértők optimális hozzárendelési kérdései” című kutatás-fejlesztési munka. Felismertem, hogy a felvonóvizsgáló szakértők feladatokhoz és felvonókhoz történő hozzárendelése új - a szakirodalomban eddig nem tárgyalt - modellek és módszerek kifejlesztését és alkalmazását teszi szükségessé. A kidolgozott új modellek és módszerek alkalmasak a gyakorlatban is jól hasznosítható megoldások elérésére, előállítására.

A kidolgozott algoritmus evolúciós jellegéből adódóan is sok továbbfejlesztési lehetőséget kínál, ilyenek például a következők:

- új mutációs operátorok bevezetése,
- új szelekciós módszerek alkalmazása.

Igen nagy potenciál rejlik az algoritmus párhuzamosíthatóságának kihasználásában, hiszen a párhuzamosítás különböző technikáit felhasználva nagyméretű rendszerek optimalizálása rövidebb (futási) időn belül lehetővé válik.

Az algoritmus megvalósítható nagyteljesítményű számítógép klasztereken vagy számítási felhőkön, bár külön vizsgálat tárgyává kell tenni a kommunikáció hatását az algoritmus sebességére, hiszen az adatokra való várakozás és a kommunikáció miatt a sebességnövekedés a számításba bevont számítógépek számával nem egyenesen arányos.

Érdemes megvizsgálni még az algoritmus párhuzamos szálainak futtatási lehetőségét általános célú grafikus processzorokon (GPGPU General Purpose Graphics Processing Unit). Mivel ezek a speciális processzorok egyszerre több számítási szál futtatásával nagy teljesítményre képesek, megfelelően megírt szoftverek esetén párhuzamos teljesítményük sokszorosan meghaladja az általános célú processzorokét, így használatukkal a futásidő jelentősen csökkenthető.

11 Köszönetnyilvánítás

A TÁMOP-4.2.1.B-10/2/KONV-2010-0001 „A felsőoktatás minőségének javítása kiválósági központok fejlesztésére alapozva a Miskolci Egyetem stratégiai kutatási területein” című projekt támogatásával. „A projekt az Új Magyarország Fejlesztési Terv Keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.” 2011. december 31.-ig

A TÁMOP-4.2.2.B-10/B-10/1-2010-0008 „A Miskolci Egyetem tudományos képzés műhelyeinek támogatása” című projekt támogatásával. „A projekt az Új Magyarország Fejlesztési Terv Keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.” 2012.01.01. – 2012.06.01.

Ezúton szeretnék köszönetet mondani mindazoknak, akik munkájukkal és segítségükkel hozzájárultak az értekezés elkészültéhez.
Különösen tudományos vezetőimnek, Dr. Jármai Károly professzor úrnak és Dr. Bányai Tamás egyetemi docens úrnak, akik szakmai irányításukkal, segítőkész munkájukkal és támogatásukkal nélkülözhetetlen segítséget nyújtottak kutatómunkámhoz.
Szintén köszönetet mondok Dr. Tóth Tibor professzor úrnak, a Doktori Iskola vezetőjének hasznos tanácsaiért és támogatásáért. Köszönöm a Miskolc Egyetem Anyagmozgatási és Logisztikai Tanszéke kollektívájának a szakmai és erkölcsi támogatást.
Végül köszönetet szeretnék mondani családomnak és menyasszonyomnak, akik mindvégig mellettem álltak és bíztattak munkám során.

12 Summary

The purpose of my research was the optimization of the logistics expenditures in a network like technical inspection and maintenance systems. To achieve this aim:

- I have described the structure of the network like technical inspection and maintenance systems.
- I have created the standard systems models and system variants.
- I have made the mathematical model of the multi constrained object-expert assignment in the system.
- I have created an optimization algorithm which able to handle large scales systems with a large number of elements appearing in the field of logistics.

The results of the research work can be summarized as follows:

1. *General structure of the of the network like technical inspection and maintenance systems*
   I have described the general structure of the network like technical inspection and maintenance systems, including the centralized/decentralized operation of the small and large scale systems.
   The models in the system structure are appropriate to describe the small scale network like technical inspection and maintenance systems operating regionally and the medium and large scale network like technical inspection and maintenance systems operating decentralized.
   I had defined the system elements of the technical inspection and maintenance systems with their characteristic functions and roles in a logistics point of view.
   I had created the generator structure which describes the standard system variants of the technical inspection and maintenance systems.

2. *Mathematical model of the network like technical inspection and maintenance systems*
   I had developed the mathematical description of the network like technical inspection and maintenance systems.
   I had determined the system wide parameters which describe the system and the parameters of the individual system components.
   I had presented the constraints and conditions which have to be taking into consideration at the development of the optimization process.
   I had defined the typical problems of the multi expert, multi route systems and the associated logistics expenditures.

3. *Development of an evolutionary programming algorithm*

   I had developed an evolutionary programming algorithm based on multi chromosome technique which in one hand is suitable to solve the multi expert, multi object assignment problem of the technical inspection and maintenance systems and on the other hand, it is appropriate to determine the routes travelled by the experts. Considering the constraints and conditions described in the mathematical model the algorithm using a multi chromosome model where one gene describes an examination of an object.

   The developed heuristic algorithm uses local and global operators to determine the routes of the experts.

   The goodness of individuals is determined by penalty functions, which can be also local and global like the operators.

   At the design process of the operators, I have examined several contraction operators and their impact on the convergence of the target function.

4. *Examination of the developed algorithm, comparison with two variants of the tabu search algorithm*

   I had implemented the algorithm to confirm and justify the goodness of the algorithm which allowed to evaluate and analyze the algorithm on small and large scale problems.

   On small scale problems, the algorithm had given accurate results certified by the naked eye, but on large scale problems, the goodness of the algorithm cannot be certified, but during the tests, it was obvious that the run time of the algorithm on a single processor core with sequential processing at large scale problem could be extremely high.

   It has become necessary the parallelization of the algorithm whose impact on the speed was examined on a large scale problems.

   Using the developed application, the algorithm was compared to the simple and to the generic neighborhood list tabu search algorithm. The examinations and comparison tests had confirmed the goodness of the developed algorithm.

13 Felhasznált irodalmak jegyzéke

13.1 Az értekezés témakörében megjelent saját publikációk

[1] Kota László, Dr. Cselényi József: Felügyeletet ellátó szakértők optimális hozzárendelése különböző helyeken üzemelő felvonókhoz, Miskolc, PhD fórum 2000, pp. 39-46.

[2] Kota László: Air cargo raktár logisztikai rendszerének vizsgálata, Ferihegy: kelet-európai hub, *Logisztikai Híradó*, 2001 február, XI. Évf 1. szám, pp. 3-6.

[3] Prof. Dr. Cselényi József, Dr. Illés Béla, Kota László: Szétszórt objektumok karbantartásának, időszakos műszaki ellenőrzésének optimális működtetésére szolgáló virtuális logisztikai központ, Országos Emelőgép- Üzemeltetői Biztonságtechnikai Konferencia. 2001. Nyíregyháza. 2001. június 20-21. Gépgyártás XLI. évfolyam 5. szám, pp. 15-18.

[4] Kota László, Cselényi József: Felvonó szakértők felvonókhoz való hozzárendelésének metamatikai modellei és módszerei dinamikusan változó felvonószám esetén, Ph.D. fórum 2001 november 6., pp. 103-109.

[5] Kota László: Termelési mélység optimalizálása Ant Colony algoritmus alkalmazásával, Repüléstudományi konferencia 2009, Szolnok, *Repüléstudományi közlemények* különszám, 2009. április 24., www.szrfk.hu/rtk/, Zrínyi Miklós Nemzetvédelmi Egyetem, HU-ISSN 1789-770X

[6] Dr. József Cselényi, László Kota, Dr. Ágota Bányai, Dr. Tamás Bányai: Dynamic Assignment of Experts Doing Analyses to Elevators in Different Places, *GÉP*, 2001, vol. LII. pp. 28-33.

[7] Cselényi J., Kota L., Bányainé Tóth Á., Bányai T.: Dynamic assignment of experts doing analyses to elevators in different places. Proceedings of MicroCAD 2001, University of Miskolc, pp. 23-32.

[8] László Kota, Dr. József Cselényi, Dr. Ágota Bányai, Dr. Tamás Bányai: Mathematical models and methods of the assignment of elevator experts to elevators in case of different constrains, Proceedings of 3rd International Conference of Ph.D. Students, University of Miskolc, Miskolc, ISBN 963 661 480 6 (2001 aug.), Engineering Sciences, Vol. I, pp. 245-252.

[9] László Kota, Dr. József Cselényi: Mathematical models and methods of the assignment of elevator experts to elevators, in case of dynamically changed elevator number, Miskolcer Gespräche 2001 „Die neusten ergebnisse auf dem Gebiet Fördertechnik und Logistik" Universtität Miskolc. ISBN 9636614938, Universität Miskolc, 13-14 September 2001, Miskolc pp. 118-122.

[10] József Cselényi, Béla Illés, László Kota: Die Virtuelle Zentrale zum optimalen Betreiben bei der Wartung, bei der periodisch technischen Kontrolle von zerstreuten Objekten, *GÉP*, 2002/2-3 vol LII. pp: 43-46

[11] L. Kota, J. Cselényi: Defining the locations of new experts in an elevator maintenance-examination system, *Modelling and Optimisation of Logistic Systems*, 2001, University of Miskolc, ISBN 963 661 510 1, pp. 91-96.

[12] Cselényi J., Kota L.: Determination of storage capacity requirement of EEEs before packaging. MicroCAD 2002, International Scientific conference 7-8 Marc. 2002. Selection I, Material Flow Systems, Logistical Informatics Miskolc, pp. 69-75. ISBN 963 661 515 2

[13] L. Kota, J. Cselényi: Defining the new experts location in an elevator maintenance examination system, Вестник Национального технического Университета «ХПИ» 9'2002 т. 10, Харьков, Сборник научных трудов Тематический выпуск "Технологии в машиностроении" Удк 621.9, страницы:96-102.

[14] Prof. Dr. hc. Mult. József Cselényi, Dr. Béla Illés, László Kota: Virtuelle Zentrale zur Disposition der Inspektion räumlich verteilter Objekte, *Magdeburger Schriften zur Logistik*, Otto von Guericke Universität Magdeburg 2003, ISSN 1436-9109

[15] László Kota, József Cselényi: Optimal Layout Planning in Case of Combined Assignment Model, microCAD 2004, International Scientific Conference 18-19 March 2004, Supplementary Volume ISBN 963 661 608 6 ö, ISBN 963 661 624 8

[16] László Kota: Ant Colony Based Optimization of Production Depth, XXIII microCAD 2009 International Scientific Conference 19-20 March 2009, ISBN 978-963-661-866-7 ö, ISBN 978-963-661-880-3

[17] T. Bányai, L.Kota: BOM optimisation to advance assembly processes, Lamdamap 2009, 9th International Conference and Exhibition on Laser Metrology, Machine Tool, CMM & Robotic Performance, 30 june- 2 july, Brunel University, West London, ISBN 978-0-9553082-7-7, pp. 156-164

[18] Kota László, Jármai Károly: Műszaki felügyeleti és karbantartó rendszerek optimálása, GÉP LXII: évfolyam, 2011/7-8, I. Kötet, pp.: 75-78, ISSN: 0016-8572

[19] Kota László: Genetikus programozás és tabu keresés összehasonlítása műszaki felügyeleti és karbantartó rendszerek optimálási feladatainál, GÉP LXIII: évfolyam, 2012/2, pp.: 33-36, ISSN: 0016-8572

[20] Kota L.: Optimisation of Large Scale Maintenance Networks with Evolutionary Programming, DAAAM International Scientific Book 2011 ISSN 1726-9687, ISBN 978-3-901509-84-1, pp.: 495-512, Chapter 40., (reviewed) (könyvfejezet), doi: 10.2507/daaam.scibook.2011.40

13.2 Az értekezésnél felhasznált nem saját irodalom

[21] Joel Levitt: *The handbook of maintenance management*, Industrial Press Inc, 2009 , ISBN 978-0-8311-3389-4, 477 p.

[22] Dr. habil Illés Béla: A karbantartási logisztika, a minőségbiztosítási logisztika alapjai, folyamatainak matematikai modellezése, Miskolci Egyetem habilitációs füzetei, 2005

[23] Illés B., Cselényi J., Németh J.: Hálózatszerűen működő karbantartás, felújítás és hálózatépítés logisztikai rendszereinek tervezési és irányítási módszerei, A Miskolci Egyetem Közleménye, Mechatronika, Anyagtudomány, Vol. 1., No. 2 (2005), pp. 149-164. oldal, ISSN 1589-827X

[24] David Achermann: Modelling, Simulation and Optimization of Maintenance Strategies under Consideration of Logistic Processes, PhD. thesis, Swiss Federal Institue of Technology, Zurich 2008, Diss. ETH No. 18152

[25] Suk-Tae Baea, Heung Suk Hwanga, Gyu-Sung Choa and Meng-Jong Goan: Integrated GA-VRP solver for multi-depot system, *Computers & Industrial Engineering*, Vol. 53, No. 2, (2007), pp. 233-240, ISSN: 0360-8352, doi:10.1016/j.cie.2007.06.014

[26] David H. Wolpert, Wiliam G. Macready: No Free Lunch Theorems for Search, Technical Report SFI-TR-95-02-010 (Santa Fe Institute), 1995, http://econpapers.repec.org/RePEc:wop:safiwp:95-02-010

[27] David H. Wolpert, William G. Macready: No Free Lunch Theorems for Optimization, *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp. 67-82, ISSN: 1089-778X,

[28] Travis C. Service: A No Free Lunch theorem for multi-objective optimization, *Information Processing Letters*, Vol. 110, No. 21, (2010), pp. 917–923, ISSN (Print) 0020-0190, doi:10.1016/j.ipl.2010.07.026

[29] U Dinesh Kumar, John Crocker, J. Knezevic, M El-Haram: Reliability Maintenance and Logistic Support: A Life Cycle Approach, Springer 2009, ISBN: 978-0412842405, p.490

[30] Marco Dorigo, Vittorio Maniezzo, Alberto Colorni: Ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics*--Part B , Vol. 26, No. 1, (1996), pp. 29-41, ISSN: 1083-4419, doi: 10.1109/3477.484436

[31] Arthur E. Carter a, Cliff T. Ragsdale: A new approach to solving the multiple traveling salesperson problem using genetic algorithms*, European Journal of Operational Research*, Vol. 175 (2006), No 1, pp. 246–257

[32] Johannes Wilhelm Joubert: An integrated and intelligent metaheuristic for constrained vehicle routing, Phd dissertation, Department of Industrial and Systems Engineering, University of Pretoria 2007, 167p., doi:10.1016/0377-2217(94)00011-Z

[33] Gilbert Laporte, Yves Nobert, Serge Taillefer: Solving a family of Multi-Depot Vehicle Routing and Location-Routing Problems, *Transportation Science*, Vol.22 1988, No. 3, pp. 161-172, doi: 10.1287/trsc.22.3.161

[34] Sivakumar Rathiam, Ranja Sengupta: Algorithms for Routing Problems Involving UAVs, *Studies in Computational Intelligence*, Vol. 70, No. 1 (2007), Springer-Verlag, ISBN 978-3-540-72695-1 2007, DOI: 10.1007/978-3-540-72696-8_6, pp. 147-172.

[35] Bruce Golden, S Raghavan, Edward Wasil: The vehicle routing problem: latest advances and new challenges, *Operations Research/Computer Science Interfaces Series*, Vol. 43, Springer Science+Business Media 2008, 589 p., ISBN 978-0-387-77777-1

[36] Imdat Kara, Tolga Bektas: Integer linear programming formulations of multiple salesman problems and its variations, *European Journal of Operational Research*, Vol. 174, No. 3 (2006), pp. 1449–1458, doi:10.1016/j.ejor.2005.03.008

[37] Anand J. Kulkarni, K. Tai: Probability Collectives: A multi-agent approach for solving combinatorial optimization problems, *Applied Soft Computing*, Vol. 10, No. 3, (2010), pp. 759–771, doi:10.1016/j.asoc.2009.09.006

[38] Xiong Chen, Weishui Wan, Xinhe Xu: Modeling rolling batch planning as vehicle routing with time windows, *Computers & Operations Research*, Vol. 25, No. 12, pp. 1127-1136, 1998, doi:10.1016/S0305-0548(98)00018-5

[39] David L. Applegate, Robert E. Bixby, Vasek Chvátal, William J. Cook: The Traveling Salesman Problem: A Computational Study, Princeton University Press 2006, ISBN 978-0-691-12993-8

[40] Imreh Balázs,Imreh Csanád: *Kombinatorikus optimalizálás*, Egyetemi tankönyv, Szegedi Tudományegyetem 2005, 339 p., ISBN: 9639056367

[41] Fábos Róbert: Operációkutatás, az elfeledett tudomány a logisztikában*, Katonai Logisztika*, Vol. 14, No. 2 (2006), Honvédelmi Minisztérium, pp. 56-70.

[42] T.K. Ralphs, L. Kopman,·W.R. Pulleyblank, L.E. Trotter: On the capacitated vehicle routing problem, Springer Verlag 2003, *Mathematical Programming*, Vol. 94, No. 2-3, pp. 343-359, doi: 10.1007/s10107-002-0323-0

[43] Hong Qu, Zhang Yi, HuaJin Tang: A columnar competitive model for solving multi-traveling salesman problem, *Chaos, Solitons and Fractals*, Vol. 31, No. 4, (2007), pp. 1009–1019, doi:10.1016/j.chaos.2005.10.059

[44] Michel Gendreau, Gilbert Laporte, Jean-Yves Potvin: Metaheuristics for the Vehicle routing problem, University of Montreal, Séminaire D'Intégration interdisciplinaire sur les transports: Logistique et distributique 200, szemináriumi segédanyag

[45] Balogh Sándor: Többszempontú gazdasági döntéseket segítő genetikus algoritmus kidolgozása, Phd értekezés, Kaposvári Egyetem. Gazdaságtudomány Kar. Informatika Tanszék, 2010, 143 p.

[46] Zhifeng Hao, Han Huang and Ruichu Cai: Bio-inspired Algorithms for TSP and Generalized TSP, *Traveling Salesman Problem*, Federico Greco (Ed.), InTecH 2008, p. 202, ISBN: 978-953-7619-10-7

[47] Donald Sofge, Alan Schultz Kenneth De Jong: Evolutionary Computational Approaches to Solving the Multiple Traveling Salesman Problem Using a Neighborhood Attractor Schema, *Applications of Evolutionary Computing Lecture Notes in Computer Science*, 2002, Vol. 2279/2002, pp. 153-162, ISBN: 3-540-43432-1, doi: 10.1007/3-540-46004-7_16

[48] József Farkas, Károly Jármai: *Design and Optimization of Metal Structures*, Horwood Publishing Limited 2008, ISBN 978-1-904275-29-9, 300 p.

[49] Thomas Weise, Hendrik Skubch, Michael Zapf, Kurt Geihs: Global Optimization Algorithms and their Application to Distributed Systems, Technical Report, Distributed Systems Group, FB 16, University of Kassel, Kasseler Informatikschriften 2008, 3, pp. 1-69.

[50] B.S.E.Zoraida, Michael Arock, R.Ponalagusamy: DNA Algorithm Employing Temperature Gradient for Multiple Traveling Salesperson Problem, *International Journal of Computer Applications*, Vol.1, No. 22, (2010), IJCA Journal, doi: 10.5120/441-674

[51] Tolga Bektas: The multiple traveling salesman problem: an overview of formulations and solution procedures, *The International Journal of Management Science*, Omega Vol. 34 (2006), No. 3, pp. 209-219.

[52] G. Laporte: The traveling salesman problem: An overview of exact and approximate algorithms, *European Journal of Operational Research*, Vol. 59, No. 2, (1992), pp. 231-247, ISSN: 0377-2217, doi:10.1016/0377-2217(92)90138-Y

[53] Roberto Wolfler Calvo, Roberto Cordone: A heuristic approach to the overnight security service problem*, Computers & Operations Research*, Vol. 30, No. 9, (2003), pp. 1269-1287, ISSN: 0305-0548, doi:10.1016/S0305-0548(02)00070-9

[54] Imdat Kara, Tolga Bektas: Integer linear programming formulations of multiple salesman problems and its variations, *European Journal of Operational Research*, Vol. 174, No. 3, (2006), pp. 1449–1458, doi:10.1016/j.ejor.2005.03.008

[55] Gilbert Laporte and Yves Nobert: A Cutting Planes Algorithm for the m-Salesmen Problem, *The Journal of the Operational Research Society*, Vol. 31, No. 11 (1980), pp. 1017-1023, ISSN: 01605682, doi:10.1057/jors.1980.188

[56] Yang GuoXing: Transformation of multidepot multisalesmen problem to the standard travelling salesman problem, *European Journal of Operational Research,* Vol. 81, No. 3 (1995), pp. 557-560.

[57] A. Iqbal Ali, Jell L. Kennington: The asymmetric m-traveling salesman problem: A duality based branch-and-bound algorithm, *Journal of Discrete Applied Mathematics*, Vol. 13, No. 2-3, (1986), pp. 259-276, ISSN: 0166-218X, doi 10.1016/0166-218X(86)90087-9

[58] Bezalel Gavish, Kizhanathan Srikanth: An optimal solution method for large-scale multiple traveling salesman problems, *Journal of Operations Research*, Vol. 34, No. 5, (1986), pp. 698-717, ISSN: 0030364X, doi 10.1287/opre.34.5.698

[59] David Simchi-Levi, with Xin Chen and J. Bramel: *Logic Of Logistics: Theory, Algorithms, And Applications For Logistics and Supply Chain Management*, Springer-Verlag 2004, ISBN: 0387221999, 355 p.

[60] Gorenstein S.: Printing press scheduling for multi-edition periodicals. *Management Science*, Vol. 16, No. 6, (1970), pp: B373–83., doi: 10.1287/mnsc.16.6.B373

[61] Carter AE, Ragsdale CT.: Scheduling pre-printed newspaper advertising inserts using genetic algorithms, *Omega The International Journal of Management Science*, Vol. 30, No. 6, (2002), pp.415–421., ISSN 0305-0483, doi: 10.1016/S0305-0483(02)00059-2

[62] Joseph A. Svestka, Vaughn E. Huckfeldt: Computational experience with an m-salesman traveling salesman algorithm. *Management Science*, Vol. 19, No. 7, (1973), pp. 790–799, doi: 10.1287/mnsc.19.7.790

[63] J.K. Lenstra,A. H. G. Rinnooy Kan: Some simple applications of the traveling salesman problem, *Operational Research Quarterly*, Vol. 26, No. 4, (1975), pp. 717–33., ISSN: 00303623

[64] Tiehua Zhang Gruver, W.A. Smith, M.H.: Team scheduling by genetic search, *Intelligent Processing and Manufacturing of Materials*, Vol. 2,(1999). p. 839–844., ISBN: 0-7803-5489-3, doi: 10.1109/IPMM.1999.791495

[65] R. D. Angel, W. L. Caudle, R. Noonan, A. Whinston : Computer assisted school bus scheduling. *Management Science*, Vol. 18, No. 6, February 1972, pp. B-279-B-288 doi: 10.1287/mnsc.18.6.B279

[66] Kenneth C. Gilbert, Ruth B. Hofstra: A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem, *Decision Sciences*, Vol. 23, No. 1, pp. 250–259, doi: 10.1111/j.1540-5915.1992.tb00387.x

[67] Barry L. Brumitt, Anthony Stentz: Dynamic mission planning for multiple mobile robots. Proceedings of the IEEE international conference on robotics and automation, Vol 3., (1996), pp. 2396-2401, ISBN: 0-7803-2988-0, doi: 10.1109/ROBOT.1996.506522

[68] Barry L. Brumitt, Anthony Stentz: GRAMMPS: A generalized mission planner for multiple mobile robots. Proceedings of the IEEE international conference on robotics and automation, Vol 2., (1998), pp. 1564-1571, ISBN: 0-7803-4300-X, doi: 10.1109/ROBOT.1998.677360

[69] Joel L. Ryan , T. Glenn Bailey, James T. Moore, William B. Carlton: Reactive Tabu search in unmanned aerial reconnaissance simulations. Proceedings of the 1998 winter simulation conference, Vol. 1, 1998. pp. 873–879., ISBN:0-7803-5134-7, doi: 10.1109/WSC.1998.745084

[70] Tang L, Liu J, Rong A, Yang Z.: A multiple traveling salesman problem model for hot rolling scheduling in Shangai Baoshan Iron & Steel Complex, *European Journal of Operational Research*, Vol. 124, No. 2, (2000), pp. 267-282, doi: 10.1016/S0377-2217(99)00380-X

[71] Roy Jonker and Ton Volgenant: An Improved Transformation of the Symmetric Multiple Traveling Salesman Problem, *Operations Research*, Vol. 36, No. 1 (1988), pp. 163-167, ISSN: 0030364X, doi: 10.1287/opre.36.1.163

[72] Ton Volgenant, Roy Jonker: A branch and bound algorithm for the symmetric traveling salesman problem based on the 1-tree relaxation, *European Journal of Operational Research*, Vol. 9, No. 1, (1982) pp. 83-89, doi:10.1016/0377-2217(82)90015-7

[73] Philippe Lacomme, Christian Prins, Wahiba Ramdane-Chérif: Evolutionary algorithms for periodic arc routing problems, *European Journal of Operational Research*, Vol. 165, No. 2, (2005), pp. 535–553, doi:10.1016/j.ejor.2004.04.021

[74] Ismail H. Toroslu Yilmaz Arslanoglu: Genetic algorithm for the personnel assignment problem with multiple objectives, *Information Sciences*, Vol. 177, No. 3 (2007), pp. 787–803, doi:10.1016/j.ins.2006.07.032

[75] Hussain Aziz Saleh, Rachid Chelouah: The design of the global navigation satellite system surveying networks using genetic algorithms, *Engineering Applications of Artificial Intelligence*, Vol. 17, No. 1, (2004), pp. 111-122, ISSN 0952-1976, doi:10.1016/j.engappai.2003.11.001

[76] R.Nallusamy, K.Duraiswamy, R.Dhanalaksmi, P. Parthiban: Optimization of Non-Linear Multiple Traveling Salesman Problem Using K-Means Clustering, Shrink Wrap Algorithm and Meta-Heuristics, *International Journal of Nonlinear Science*, Vol.8 No.4 (2009),pp.480-487, ISSN 1749-3889 (print), 1749-3897 (online)

[77] Ding Chao, Cheng Ye, He Miao: Two-Level Genetic Algorithm for Clustered Traveling Salesman Problem with Application in Large-Scale TSPs, *Tsinghua Science and Technology*, Vol. 12, No. 4, (2007), pp459-465, ISSN 1007-0214 15/20

[78] T.Y.El Mekkawy, S.Liu: A new memetic algorithm for optimizing the partitioning problem of tandem AGV systems, *International Journal of Production Economics*, Vol. 118, No. 2, (2009), pp. 508–520, doi:10.1016/j.ijpe.2009.01.008

[79] Gur Mosheiov: Vehicle routing with pick up and delivery: Tour Partitioning heuristics, *Computers and Indistrial Engineering*, Vol. 34, No. 3, pp. 669-684, 1998, doi:10.1016/S0360-8352(97)00275-1

[80] Soheil Ghafurian, Nikbakhsh Javadian: An ant colony algorithm for solving fixed destination multi-depot multiple traveling salesmen problems*, Applied Soft Computing*, Vol. 11, No. 1, (2011), pp. 1256–1262, doi:10.1016/j.asoc.2010.03.002

[81] M. Dorigo, T. Stützle: *Ant Colony Optimization*, MIT Press 2004.. ISBN 0-262-04219-3, 319 p

[82] San Nah Sze, Wei King Tiong: A Comparison between Heuristic and Meta-Heuristic Methods for Solving the Multiple Traveling Salesman Problem, *World Academy of Science, Engineering and Technology*, Vol. 25, 2007, pp. 300-303.

[83] Kulcsár Béla*: Ipari Logisztika*, LSI Oktatóközpont 1998, ISBN: 963 577 242 4, 385p.

[84] Benkő János: *Logisztika I*, Szent István Egyetemi Kiadó, Gödöllő 2010, p 396

[85] Béla Illés, Elke Glistau, Norge I. Coello Machado: *Logistik und Qualitätsmanagement*, Miskolc, 2007, ISBN 978-963-87738-1-4

[86] Hartványi Tamás, Földesi Péter, Kovács János, Tóth Lajos: Multi-centre logistics systems for improving competitive status of logistics service suppliers in Hungary*, Hungarian Electronic Journal Of Sciences*, 2004., HU ISSN 1418-7108

[87] Salamon András*: Genetikus algoritmusok*, egyetemi jegyzet, ELTE TTK 2003, p. 127

[88] Barrie M. Baker, M.A. Ayechew: A genetic algorithm for the vehicle routing problem*, Computers & Operations Research*, Vol. 30, No. 5, (2003), pp. 787–800, ISSN: 0305-0548, doi:10.1016/S0305-0548(02)00051-5

[89] Hans J. Pierrot , Robert Hinterding: Using Multi-chromosomes to Solve a Simple Mixed Integer Problem, AI '97 Proceedings of the 10th Australian Joint Conference on Artificial Intelligence: Advanced Topics in Artificial Intelligence, pp. 137-146 , ISBN:3-540-63797-4

[90] Ronald, S. Kirkby, S. Eklund, P.: Multi-chromosome mixed encodings for heterogeneous problems, *Evolutionary Computation*, (1997), pp. 37-42, ISBN: 0-7803-3949-5, doi: 10.1109/ICEC.1997.592264

[91] Alice E. Smith and David W. Coit: Penalty Functions, *Handbook of Evolutionary Computation*, Section C 5.2, Institute of Physics Publishing and Oxford University Press, Bristol, U.K., 1997, p. 1130, ISBN: 978-0750303927

[92] András Király, János Abonyi: Optimization of Multiple Traveling Salesmen Problem by a Novel Representation based Genetic Algorithm, *Studies in Computational Intelligence*, Vol. 313, (2010), pp. 141-151, doi: 10.1007/978-3-642-15220-7_12

[93] Byoung-Tak Zhang, Jung-Jib Kim: Comparison of Selection Methods for Evolutionary Optimization, *Evolutionary Optimization*, An International Journal on the Internet ,Vol. 2, No. 1, (2000), pp.55-70

[94] Sayan Maity, Kumar Gunjan, Swagatam Das: An Improved Evolutionary Programming with Voting and Elitist Dispersal Scheme, *Computer Science*, Vol. 6466, (2010), pp. 206-213, doi: 10.1007/978-3-642-17563-3_25

[95] Thomas Weise: *Global Optimization Algorithms – Theory and Application – 2nd edition*, Ebook, (2009), http://www.it-weise.de, p. 820

[96] Myungryun Yoo: Real-time task scheduling by multiobjective genetic algorithm, *The Journal of Systems and Software*, Vol. 82, (2009), pp. 619–628

[97] Ruprecht-Karls-Universität Heidelberg, Institut für Informatik: Library of sample instances for the TSP (and related problems), http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[98] Jelasity Márk: *Mesterséges Intelligencia 1 fejezet*, ed.: Futó Iván, Aula Kiadó 1999, p:1006

[99] Balogh Sándor: Többszempontú gazdasági döntéseket segítő genetikus algoritmus kidolgozása és alkalmazásai, Kaposvári egyetem 2009, p. 143

[100] Fred W. Glover, Manuel Laguna: *Tabu Search*, Springer 1997, ISBN 978-0-7923-9965-0 p.

[101] Siamak Sarmady: An Investigation on Tabu Search Parameters, Universiti Sains Malaysia 2007, p. 11