BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
VILLAMOSMÉRNÖKI ÉS MÉRNÖKINFORMATIKAI KAR
IRÁNYÍTÁSTECHNIKA ÉS INFORMATIKA TANSZÉK

VÁROSI BUSZOK ÚTVONALTERVEZÉSE MESTERSÉGES INTELLIGENCIA MÓDSZEREKKEL

TDK dolgozat

Dimitriu Adonisz

Konzulens:
Dr. habil. Harmati István

2020

Absztrakt

Az intelligens közlekedési rendszerek (ITS: Intelligent Transportation systems) egyik fő feladata a földfelszíni közlekedés fejlesztése. A tömegközlekedés jellemzően kötött vonalú utazást jelent fix útvonallal, megállókkal és menetrenddel, melyben a városi autóbuszok kulcsfontosságú szerepet töltenek be. Ebben a tanulmányban a tömegközlekedés egy új formáját vetjük fel, melyben a buszok szabadon, az utasokhoz alkalmazkodva közlekednek (és nem fordítva) úgy, hogy felveszik őket az aktuális pozíciójukon és elszállítják őket a céljukhoz.

El lehet képzelni egy mobil applikációt, amelynek a felhasználói bejelölhetik aktuális helyzetüket és a céljukat egy úthálózaton. A buszok útvonalának illeszkednie kell erre az utas-elrendeződésre. A probléma közel áll a multi-vehicle multi-depot Dial-a-Ride Problem-hez (MM-DARP) és az On-Demand Transportation task-hoz (ODT), ahol a járműveknek ki kell szolgálni az utasok szállítási igényét. Az MM-DARP-vel és ODT-vel kevés kutatás foglalkozik, a problémák komplex természete miatt. A szakirodalom áttekintésével azt láthatjuk, hogy a legtöbb tanulmány a témában környezet-specifikus megoldásokat biztosít, amelyeket nem, vagy csak nehezen lehet szélesebb körben alkalmazni.

Az úthálózat egyszerűen reprezentálható egy gráffal, amelyben a csomópontok az útkereszteződéseknek, az élek pedig az ezeket összekötő útszakaszoknak felelnek meg. A feladat olyan busz-útvonalak keresése a hálózaton, amelyek mentén a buszok el tudják szállítani az összes utast úgy, hogy az átlagos utazási idő minimális legyen. A költségfüggvény más paramétereket is tartalmazhat, például a buszutak hossza vagy az utasok prioritása (munkába tartó emberek). Ebben a dolgozatban megadtuk a probléma formális leírását és több módosított hangyakolónia algoritmust (ACO: Ant Colony Optimization), melyekkel optimális megoldást kapunk.

Abstract

Intelligent Transportation Systems (ITS) focus on many issues relating to the subject of developing ground transportation. Public transportation consists mostly of fixed transit systems, which have fixed stations, routes, and schedules. Urban buses have a key role in the transit system offering scheduled routes throughout the city. In this paper, a new approach for bus routing in public transportation is proposed, in which buses are traveling unbounded, adapting to passengers (not vice versa) by picking them up at their current location and transferring them to their destinations.

One can picture a mobile application in which users can indicate their positions and destinations on the road-network. Bus routes need to be adjusted to the current passenger-layout, satisfying several constraints. This problem is considered as a variant of the multi-vehicle multi-depot Dial-a-Ride Problem (MM-DARP) and close to the On-Demand Transportation task (ODT), where vehicles need to serve transportation requests of customers. MM-DARP and ODT have rarely been advanced, due to the complex nature of the problems. Surveying the literature one can observe, that most studies on the subject focus on particular problem variants and derive domain-specific solutions, that can not be addressed to a broader set of problems.

Road-networks are represented by graphs, where nodes correspond to junctions, and edges illustrate road segments, connecting junctions. The goal is to find a globally optimal set of paths on the road-network, for a given number of buses, such that all passengers are transferred to their destinations, while the average travel time is minimal. The objective function can include other parameters, such as the length of the bus tours or the prioritization of passengers (for example people going to work). In this paper, a formal description of the problem is presented and several modified Ant Colony Optimization (ACO) algorithms - using different concepts - are utilized to solve it.

Tartalomjegyzék

1 Bevezetés

A tanulmány egy alternatív tömegközlekedési koncepciót mutat be, melyben a buszok a kötött pályás közlekedés helyett szabadon mozoghatnak az úthálózaton, elszállítva az utasokat az aktuális pozíciójuktól a célpozíciójukig.

1.1 Motiváció és problémafelvetés

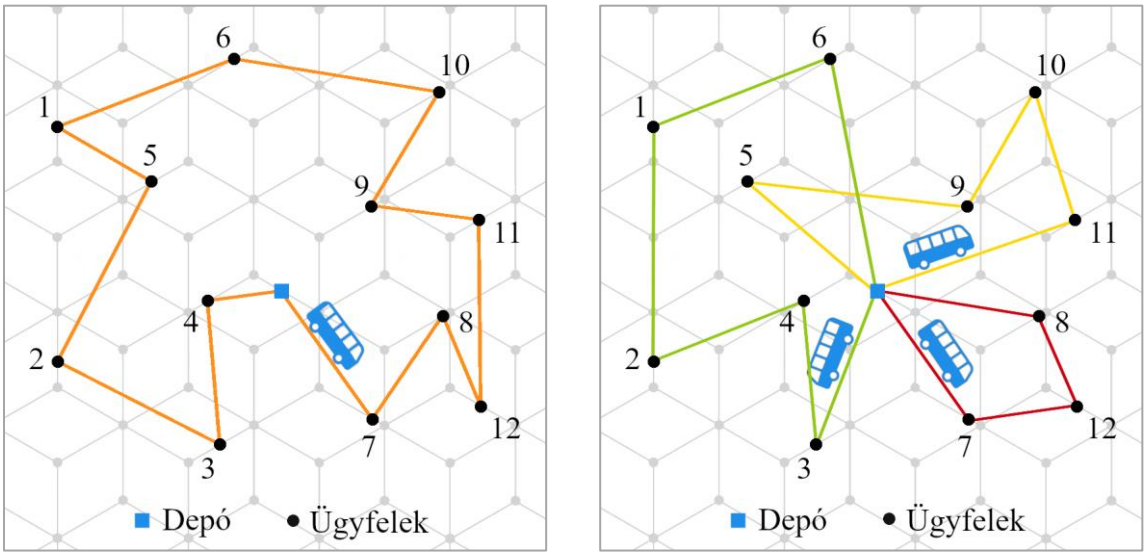
A napjainkban zajló technológiai fejlődés és urbanizáció szükségessé teszi a rohamszerűen megnövekedett városi gépjármű forgalom és a közösségi közlekedés hatékony irányítását. Több európai városban már olyan buszhálózatot terveznek, ahol a buszok nem járatszámtól függő útvonalakon közlekednek, hanem az on-line generált lakossági igények alapján. A hatékony útvonalak szervezése egy komplex feladat, melyben a buszvállalatoknak egyszerre kell figyelembe venniük az utasok igényeit és a gazdasági vonatkozásokat. A klasszikus rendszerben az utasoknak kell alkalmazkodniuk a többnyire fix tömegközlekedési hálózathoz, amely gyakran több eszköz használatát igényli, mely növekedett várakozási és utazási idővel jár. Emellett az idősek és a mozgáskorlátozottak körében az átszállás, valamint a megállók megközelítése – főleg a külváros környezetében, ahol ezek elszórtan helyezkednek el – további nehézséget okoz.

Az utas-központú felvetés egy alternatív megközelítést, az adaptív (az utas-elrendeződésre illeszkedő) útvonalak használatát indítványozza, mely beleillik az okos város koncepcióba is. A probléma komplex: olyan útvonaltervezést igényel, mellyel egy adott mennyiségű busz a véletlenszerűen elrendezett utasokat a céljukhoz szállítja, mindezt úgy, hogy az átlagos utazási idő (várakozási idő + buszon töltött idő) minimális legyen. A probléma értelmezhető egy több-kiindulópontú több-célpontú útvonaltervezésnek (MOMD: Multiple-Origin-Multiple-Destination problem) szigorú megkötésekkel. A következő alfejezetben ismertetésre kerül az ezen felvetéshez kapcsolódó releváns szakirodalom.

1.2 Irodalomkutatás

A jól ismert utazóügynök probléma (TSP: Traveling Salesman Problem) általánosan egy vagy több ország városainak az optimális bejárásáról szól, melyben az utazóügynök egy várost csak egyszer érinthet. A cél a legrövidebb útvonal megtalálása. Ennek egy ekvivalens gráfelméleti megfogalmazása: egy élsúlyozott teljes gráfban keressük a legrövidebb Hamilton-utat. A TSP bizonyítottan az NP-nehéz problémaosztályba tartozik [1], ezért egzakt megoldást kereső algoritmusok helyett heurisztikus módszereket alkalmaznak az optimális megoldás közelítésére. A jármű útvonaltervezési probléma (VRP: Vehicle Routing Problem) a TSP egy általánosítása, melyben egy olyan úthalmazt keresünk járművek számára, mely optimálisan

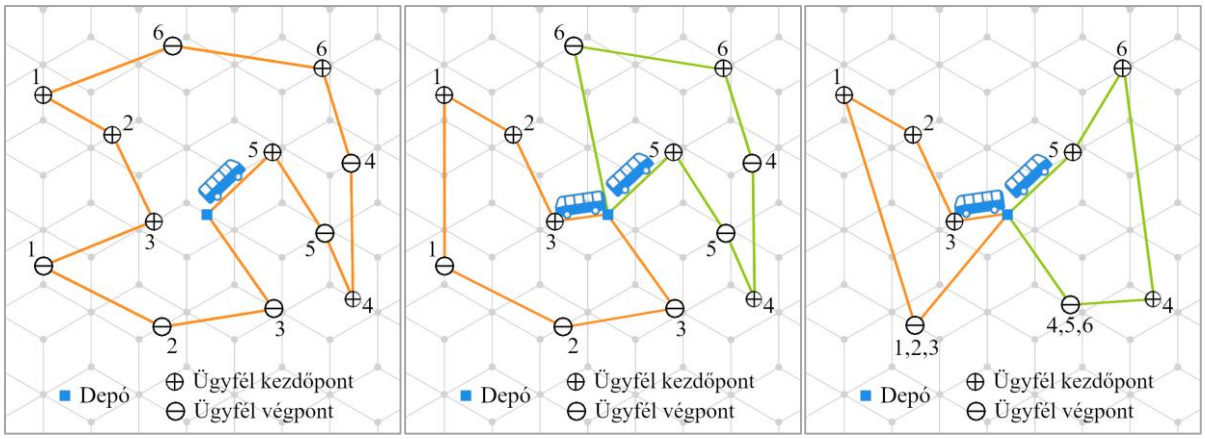
szállítja ki az ügyfelek megrendeléseit egy adott telephelyről (a továbbiakban „depó”). A klasszikus VRP – más néven kapacitált VRP (CVRP: Capacitated Vehicle Routing Problem) – egy depót feltételez, ahol egy bizonyos számú (áru-kapacitással ellátott) jármű tartózkodik, hogy kiszolgálják több ügyfél szállítási igényét.



1.1. ábra: TSP (bal oldal) és VRP (jobb oldal)

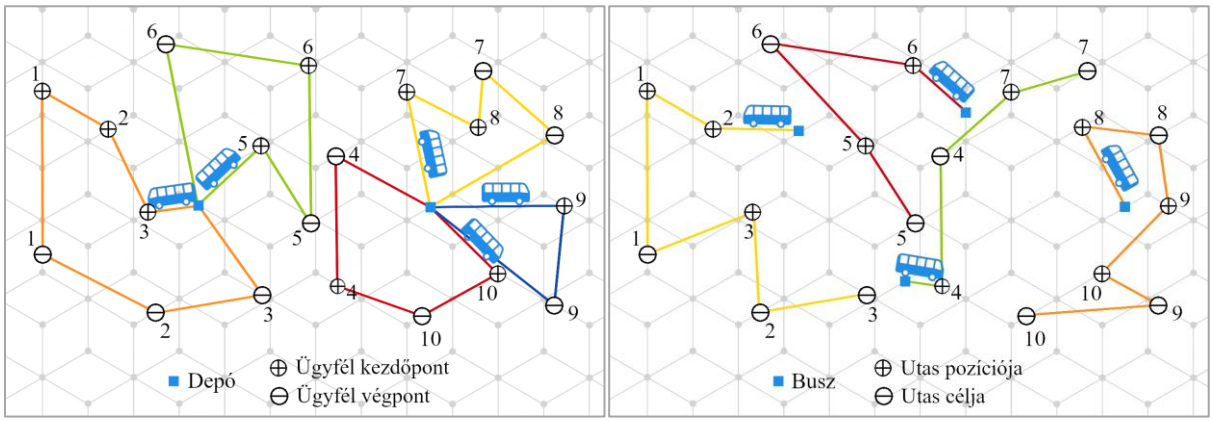
Az 1.1. ábrán a TSP és VRP útvonalainak illusztrációi láthatók. Kényszereket csatolva a VRP-hoz, lehetőségünk van a standard modellt a különböző alkalmazásokhoz illeszteni. P. Toth és D. Vigo egy tekintélyes terjedelmű és kiemelkedő jelentőségű munkája a témában ismerteti az addig megjelent variánsokat és a megoldásukra irányuló különböző megközelítéseket is [2]. Az árufelvevő-átadó VRP (PDVRP: Pickup and Delivery Vehicle Routing Problem) vagy még általánosabban árufelvevő-átadó probléma (PDP: Pickup and Delivery Problem) egy speciális variánsa a VRP-nak, ahol a járművek szállítási kérelmeket teljesítenek. Az 1-1 PDP-ben a járművek árucikket vagy embereket szállítanak egy kezdő és végpozícióból álló lokáció-pár között. Ha a szállítás emberekre vonatkozik, akkor a probléma neve hívj-egy-fuvart (DARP: Dial-a-Ride Problem), melyben az ügyfeleket a céljukhoz kell szállítani. Egy átfogó ismertető a problémáról megtalálható a [4]-es hivatkozásban.

A DARP problémacsaládot az ügyfélközpontú perspektíva lényegesen megkülönbözteti a többi VRP variánstól, lévén az emberi elégedettség – vagy éppen elégedetlenség – nem egy matematikailag egzakt jelenség. Létezik egy-járműves [5] és több-járműves [6] DARP, az előbbiben egyetlen jármű, az utóbbiban több jármű is rendelkezésre áll a kiszolgálásra. Egy gyakorlati alkalmazási területe a több-járműves DARP-nak az úgynevezett iskolabusz probléma (SBRP: School Bus Routing Problem), mely az amerikai iskolabuszok útvonaltervezésével foglalkozik. Az iskolabuszok egyetlen depóban helyezkednek el, innen indulnak a diákokért, akiket a környező iskolákba szállítanak, mintegy háztól-házig való közlekedést megvalósítva.



1.2. ábra: egy-járműves DARP (bal oldal), több járműves DARP (középen) és SBRP (jobb oldal)

Egy figyelemre méltó összefoglaló a témáról megtalálható a [7]-es hivatkozásban. Az 1.2. ábrán ennek a 3 variánsnak a különböző jellegzetességű útvonalai láthatók. A több-depós DARP-ban [18], [19] a járművek különböző helyekről indulhatnak az ügyfelekért. Ez az általánosítás az, amely a legtöbb fejtörést okozza a problémával foglalkozóknak. Az 1.3. ábrán a több-járműves DARP és a dolgozatban szereplő útvonaltervezés összevetése látható ugyanabban a környezetben.



1.3. ábra: több-járműves DARP (bal oldal) és buszok útvonaltervezése (jobb oldal)

A több-járműves több-depós DARP áll legközelebb a városi buszok útvonaltervezéséhez, az összehasonlítás végett a főbb tulajdonságai alább látható:

- *(i)* A problémát egy teljes gráf reprezentálja
- *(ii)* Minden jármű a neki kijelölt depóból indul
- *(iii)* Minden jármű valamelyik depóban fejezi be az útját
- *(iv)* Egy ügyfélt csak egy jármű szolgálhatja ki
- *(v)* Az ügyfelek felvétele időben megelőzi a szállításukat
- *(vi)* A megoldás útvonalak olyan halmaza, mely minden csomópontot (kivéve esetleg a depókat) csakis egyszer tartalmaz

(vii) A cél az úthalmaz minimalizálása az ügyfél-elégedetlenség szempontjából

A *(iv)*-es, *(v)*-ös és *(vii)*-es tulajdonság igaz a jelen problémára is. A *(ii)* tulajdonságban a depók értelmezhetők úgy, mint a buszok kiindulási pozíciója, ahol minden depóban egyetlen busz helyezkedik el. A *(iii)*-as tulajdonság nem áll fenn, mivel a buszok az utolsó letett utasnál fejezik be az útjukat, ezzel a depó fogalom elveszti jelentőségét. Az *(i)* tulajdonságban említett teljes gráf pontjai az ügyfelek kezdő- és végpontjaiból és a depókból áll, minden pontpár között fut egy él, melyen egy költségfüggvény értelmezett, ami általában távolság vagy utazási idő alapú, mint a [6], [8] hivatkozásokban. Fontos megjegyezni, hogy valójában az összes VRP variáns teljes gráfot használ, melyet az ügyfelekből (a továbbiakban „utasok") és a depókból generálnak. Az ilyen típusú gráfok leírására a *ügyfél-alapú gráf* (*customer-based graph*) terminust használjuk, melyet Huang és munkatársaitól adaptáltunk [9], másrészről az *úthálózati gráf* kifejezés alatt olyan gráfokat értünk, melyek a valós úthálózatot reprezentálják (lásd később: 2.1. Úthálózat reprezentációja). Mint említettük, a teljes gráf pontjai a depók és az utasok kezdő- és végpontjai, az ezeket összekötő élek súlya a valós úthálózati elrendezést veszi figyelembe. Ez kétféleképpen történhet:

a) Az egyik eset, hogy az élsúlyok az egyes utasok és depók közötti légvonalbeli távolságot jelölik az úthálózaton. Ilyenkor a DARP megoldása egy olyan pontsorozatot ad, mely csak szűk keretek között tekinthető érvényesnek, például akkor, ha a feladat több város bejárása, mert ekkor a városok meglátogatásának sorrendje fontosabb lehet magánál az útvonaltól. Azonban ez hibára vezet, ha például a valóságban folyó fut a városok között.
b) A másik mód, hogy az élsúlyok a legrövidebb távolságot tartalmazzák két pont között. Ebben az esetben a DARP megoldásaként kapott pontsorozat érvényesnek tekinthető, viszont egy n csomópontú ($n = depók\ száma + 2 \cdot utasok\ száma$) ügyfél-alapú gráfban $\frac{n(n-1)}{2}$ számú útvonalat kell előre meghatározni, valamilyen útkereső algoritmus (például A*) segítségével.

Az a) esetben a légvonalbeli távolság használata legtöbb esetben megbízhatatlan eredményre vezet és további számításokat igényel, hogy tényleges útvonalat kapjuk. A b) esetben előfeldolgozást kell végezni, amely során végeredményben csökken a gráf pontjainak a száma, viszont nő az élek száma.

Ebből a perspektívából nézve vitatható az az állítás, hogy az ügyfél-alapú gráf adaptálása teljesítményben és minőségben felülmúlja az úthálózati gráfok használatát. Ennélfogva a jelen tanulmányban úthálózati gráfot alkalmazunk, ügyfél-alapú gráf helyett. Következésképp a megoldásunk olyan úthalmaz, mely – *(vi)* ellenére – nem feltétlen tartalmazza az összes csomópontot a gráfban.

Egy másik hasonló probléma a kérésre-szállítás (ODT: On Demand Transportation task), ahol háztól-házig való közlekedést használnak a klasszikus tömegközlekedés mellett. A [13]-as cikkben egy branch-and-price módszert alkalmaznak több ODT rendszerre.

A dolgozatnak nem célja kimeríteni a feladat kapcsán felmerülő összes problémát, így például nem foglalkozik a következő tulajdonságokkal:

- Buszok kapacitása
- Idő-ablakok [10]
- Jármű heterogenitás [11]
- A probléma dinamikus változata [12]

Nem ez az első munka, mely valós úthálózati gráfot vezet be a VRP világába, egy viszonylag friss tanulmány a témában megtalálható a [14]-ös hivatkozásban, ahol az ügyfél-alapú gráfok alternatíváit ismertetik.

Mivel a DARP a TSP egy általánosítása, ez is az NP-nehéz problémaosztályba tartozik, így többen metaheurisztikus algoritmusokat fejlesztettek ki ennek a megoldására. J.-F. Cordeau és G. Laporte egy tabu-search algoritmust dolgozott ki az egy-depós DARP-re [6], ezt később egy branch-and-cut algoritmus követte [15], majd mások genetikus algoritmust [16] és variable neighborhood search metaheurisztikát [17] alkalmaztak. A több-depós DARP-vel kevés tanulmány foglalkozott eddig, a probléma komplex természete miatt. A [18] szerzői sikeresen adaptálták a branch-and-cut algoritmust a több-depós szcenárióhoz, majd P. Detti és munkatársai egy tabu-search és egy variable neighborhood search algoritmust [19] publikáltak. A hangyakolónia algoritmus (ACO: Ant Colony Optimization) eddig elkerülte a kutatók figyelmét, összesen egy példát találtunk a szakirodalomban, ahol Tripathy és a munkatársai egy hangyakolónia rendszert (ACS: Ant Colony System) használtak az egy-depós DARP-hoz [20]. A multi-depós variánsra nem találtunk hangyakolóniás megoldást, valamint olyan publikációt sem, amelyben úthálózati gráfot használnának a DARP problémaosztály bármelyik variánsára.

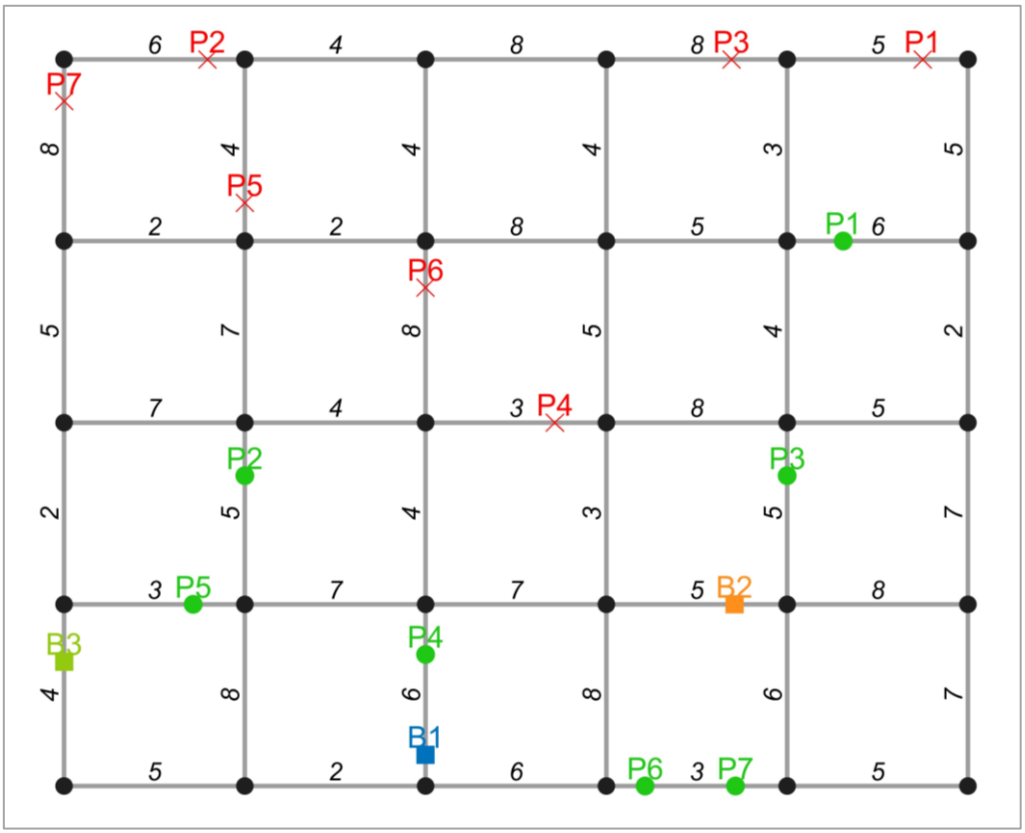
1.3 A dolgozat célja, felépítése

A dolgozat egy alternatív tömegközlekedési rendszert vet fel, melyben a buszok fix pályás közlekedés helyett háztól-házig való szállítást valósítanak meg. Az ehhez szükséges matematikai modellt mi magunk állítottuk fel, ugyanis a szakirodalomban nem találtunk hasonló kezdeményezést. A modell felállításakor úthálózati gráf reprezentációt használtunk az irodalomban gyakori ügyfél-alapú gráf helyett. A feladat megoldására 2 különböző ACO algoritmust dolgoztunk ki, melyek optimális megoldást nyújtanak. Mindkét algoritmus egy eltérő koncepcióval módosított Max-Min Hangyarendszer (MMAS: Max-Min Ant System), melyek az eredeti [24] általánosításai. A dolgozat célja, hogy összevesse a két algoritmus által elért eredményeket, valamint hogy népszerűsítse a valós úthálózat alapú megközelítést.

2 A feladat formalizálása

2.1 Úthálózat reprezentációja

Az úthálózat reprezentálására a 1.2. alfejezetben említett úthálózati gráfot alkalmazunk, mely a dolgozatban többnyire egy négyzetrácsos elrendezésű úthálózatot jelent. Egy ilyen részlet látható az 2.1. ábrán, ahol a csomópontok az úthálózat kereszteződéseit, az élek az ezeket összekötő útszakaszokat reprezentálják. Az élek súlyai jelenthetik az útszakaszok hosszát, forgalmasságát vagy több paraméter együttes költségét. A buszokat színes négyzetek és a B1, B2,..,Bk címkék jelölik, az utasok kezdőpontját zöld pötty és P1, P2, …, Pn címkék, valamint a végpontját piros „x” és P1, P2, …, Pn címkék jelölik. Az utasok összetartozó kezdő- és végpontjának a címkéje eltérő színű (zöld vagy piros), de ugyanazt a sorszámot hordozza.



2.1. ábra: Úthálózati reprezentáció

2.2 Matematikai modell

Legyen $G = (V, E, c)$ egy összefüggő, irányítatlan és élsúlyozott gráf az úthálózat reprezentációja, V csúcshalmazzal, $E = \{(i,j) \mid i,j \in V, i \neq j\}$ élhalmazzal és $c(i,j)$ élköltséggel minden $(i,j) \in E$ élen. Praktikus okokból V felbontható részhalmazokra: $V = I \cup O \cup D \cup V'$, ahol $I = \{1,2,\dots,k\}$ a buszok kiinduló pozícióinak a halmaza, $O = \{k+1, k+2,\dots,k+n\}$ az utasok kezdeti pozícióinak a halmaza, $D = \{k+n+1, k+n+2,\dots,k+2n\}$

az utasok célpozícióinak a halmaza, és $V' = \{k + 2n + 1, k + 2n + 2, \dots\}$ az úthálózat kereszteződéseinek a halmaza. Ettől fogva I, O, D halmazok elemei tekinthetőek az úthálózat virtuális kereszteződéseinek, míg az élek a kereszteződéseket összekötő útszakaszoknak. Az így definiált gráf illusztrációja az 2.1. ábrán látható. Minden $P_h = (p_o, p_d),\ h \in \{1,\ 2, \dots n\}$, utast egy $p_o = orig(P_h) \in O$ kezdeti- és egy $p_d = dest(P_h) \in D$ célpozícióból álló utas-pontpár határoz meg. A busz útvonala $t_z = (v_1, v_2, \dots, v_m),\ (v_i, v_{i+1}) \in E, \forall\ i = 1,2, \dots, m - 1$, amely legalább egy utas-pontpárt tartalmaz $v_l = orig(P_h) \in t_z,\ v_q = dest(P_h) \in t_z, l < q$ sorrendben, vagyis az útvonalnak hamarabb kell tartalmaznia az utas felvételét, mint a lerakását.

Megjegyzendő, hogy a cél az utazási idő (várakozási és buszon töltött idő) minimalizálása az utasok számára és ez különbözik a buszok utazási idejének a minimalizálásától. Emiatt a költségfüggvényt nem a buszokra, hanem az utasokra kell definiálni. Legyen a P_h utashoz rendelt útvonal a következő pontsorozat:

$$r_h = \left(v_1, v_2, \dots, v_q\right),\ (v_i, v_{i+1}) \in E, \forall\ i = 1,2, \dots, q - 1$$

Minden ilyen r_h útra:

$$v_1 \in I, \tag{1}$$

$$v_q = dest(P_h) \in D, \tag{2}$$

$$v_l = orig(P_h) \in r_h,\ l < q. \tag{3}$$

A P_h utashoz rendelt út az őt felvevő busz útvonalának egy részeként (*szakaszaként*): $r_h \subseteq t_z$ van értelmezve, amely a busztól (1) az utas céljáig (2) tart. A (3) -as kifejezés biztosítja, hogy a pontsorozatnak előbb kell tartalmaznia az utas felvételét, mint a lerakását. Továbbá megjegyzendő, hogy r_h tekinthető az alábbi két szakasz kompozíciójának:

- $(v_1, v_2, \dots, v_l)$ az utas eléréséig megtett útvonal
- $(v_l, v_{l+1}, \dots, v_q)$ az utas szállításához tartozó útvonal

Ezzel a felbontással a várakozási idő az előbbi, a buszon töltött idő az utóbbi szakasznak feleltethető meg. Legyen $s_h = \left(e_1, e_2, \dots, e_{q-1}\right),\ e_i = (v_i, v_{i+1}) \in E, \forall\ i = 1,2, \dots, q - 1$ az utashoz rendelt r_h út élsorozata. A P_h utasra vonatkozó út-költség a következőképpen írható fel:

$$c(r_h) = c(s_h) = \sum_{e_i \in s_h} c(e_i) \tag{4}$$

mely azon élek súlyainak az összege, melyeket az utashoz rendelt út érint. A költségfüggvény, az összes utasra vonatkoztatott út-költségek átlaga:

$$\min \frac{1}{n} \sum_{h=1}^{n} \sum_{e_i \in s_h} c(e_i) \tag{5}$$

Ha szükséges, akkor ez a költségfüggvény kibővíthető további szempontokat is figyelembe vevő költségekkel, például a buszok útvonalának hosszával. Emellett több olyan alkalmazás is elképzelhető, ahol az utasokhoz prioritás rendelhető (például munkába induló utasok). Ez egyszerűen szimulálható azzal, ha minden utas út-költségét egy a_h skalárral súlyozzuk. A kibővített költségfüggvény a következő:

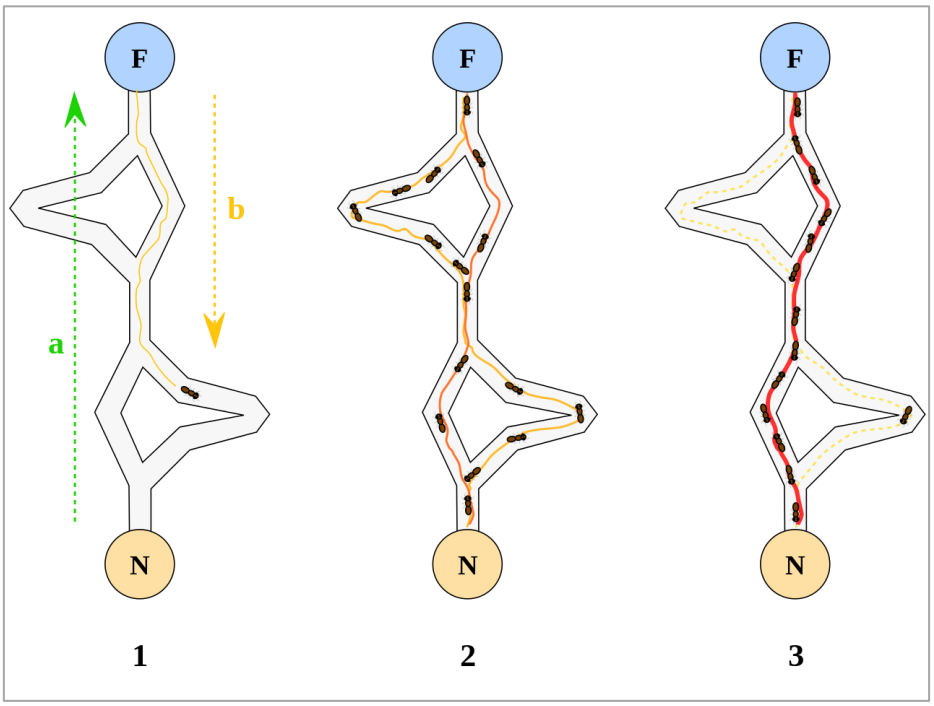
$$\min \left\{ \gamma \frac{1}{n} \sum_{h=1}^{n} a_h \sum_{e_i \in s_h} c(e_i) + \delta \frac{1}{k} \sum_{z=1}^{k} \sum_{e_i \in d_z} c(e_i) \right\} \tag{6}$$

A (6)-os egyenletben a második összegzés a buszok útvonalának a hosszára vonatkozik, ahol $d_z = (e_1, e_2, \ldots, e_{m-1}),\ e_i = (v_i, v_{i+1}) \in E, \forall\, i = 1,2,\ldots,m-1$ a t_z busz útvonalának az élsorozata, γ és δ az utasok út-költsége és a buszok útvonalhosszának a relatív fontossága. A jelen tanulmányban minden utas egyenlő prioritással rendelkezik: $a_h = 1, \forall\, h = 1,2,\ldots,n$, valamint csak az utasok út-költségeit vesszük figyelembe, tehát az (5)-ös számú egyenletet tekintjük a probléma költségfüggvényének, melyre a továbbiakban *útvonal-költség*ként hivatkozunk.

3 Hangyakolónia algoritmus

A rajintelligencia egy népszerű kutatási területe lett az utóbbi éveknek [21]. A kutatók rengeteg energiát fektettek be olyan algoritmusok kifejlesztésébe, melyek bizonyos organizmusok – elsősorban egy fajhoz tartozó állatok – kollektív viselkedésén alapulnak. Ilyen viselkedés például a madarak „V” alakzatban való repülése, a méhek pollen és nektár gyűjtése, a tengeri halrajok örvénylő mozgása és a hangyakolónia táplálékszerzése. A szóban forgó rajok egyedei önmagukban nem rendelkeznek számottevő intelligenciával, viszont egymással és a környezetükkel való interakció során intelligens képességet mutatnak, úgynevezett kollektív intelligenciával rendelkeznek.

Az ACO egy optimalizációs módszer, melyet Marco Dorigo és társai fejlesztettek ki néhány különleges hangyafaj viselkedése alapján [22], [23]. Ezek a hangyák feromonokat bocsátanak ki a táplálékkeresésük során, mely folyamatos párolgásnak van kitéve. Minden hangya egy úgynevezett feromonösvényt hagy hátra a hangyaboly és az élelemforrás között, amely előnyös útvonalat jelölhet ki a többi hangya számára.



3.1. ábra: Hangyák táplálékkeresése (forrás: https://it.wikipedia.org/wiki/Swarm_intelligence)

Az 3.1. ábra ezt a mechanizmust mutatja: kezdetben, amikor egy hangya elágazáshoz ér, véletlenszerűen választja ki az utat magának, feromonokat hagyva hátra a bejárt útja mentén. Az utána következő hangyák útvonalát befolyásolja az előbbi hangya feromonösvénye: minél nagyobb a feromonszint értéke az adott úton, annál nagyobb eséllyel választják ezt az útvonalat. A hosszabb út bejáráshoz több idő szükséges, emiatt a rövidebb utat bejárt hangyák feromonösvénye kevesebb párolgásnak van kitéve, tehát visszaéréskor a rövidebb út nagyobb

feromonszinttel fog rendelkezni. Egy idő után a legrövidebb útnak olyan magas lesz a feromonkoncentrációja, hogy a legtöbb hangya ezt az utat fogja választani. Az ACO olyan mesterséges egységeket használ, melyek az eredeti hangyák viselkedését imitálják.

Az első ACO - Hangya Rendszer (AS: Ant System) 1992-es publikációja [26] meghatározó jelentőségű volt, idővel különböző variánsai jelentek meg.

3.1. táblázat: ACO variánsok (forrás: https://github.com/thiagodnf/jacof)

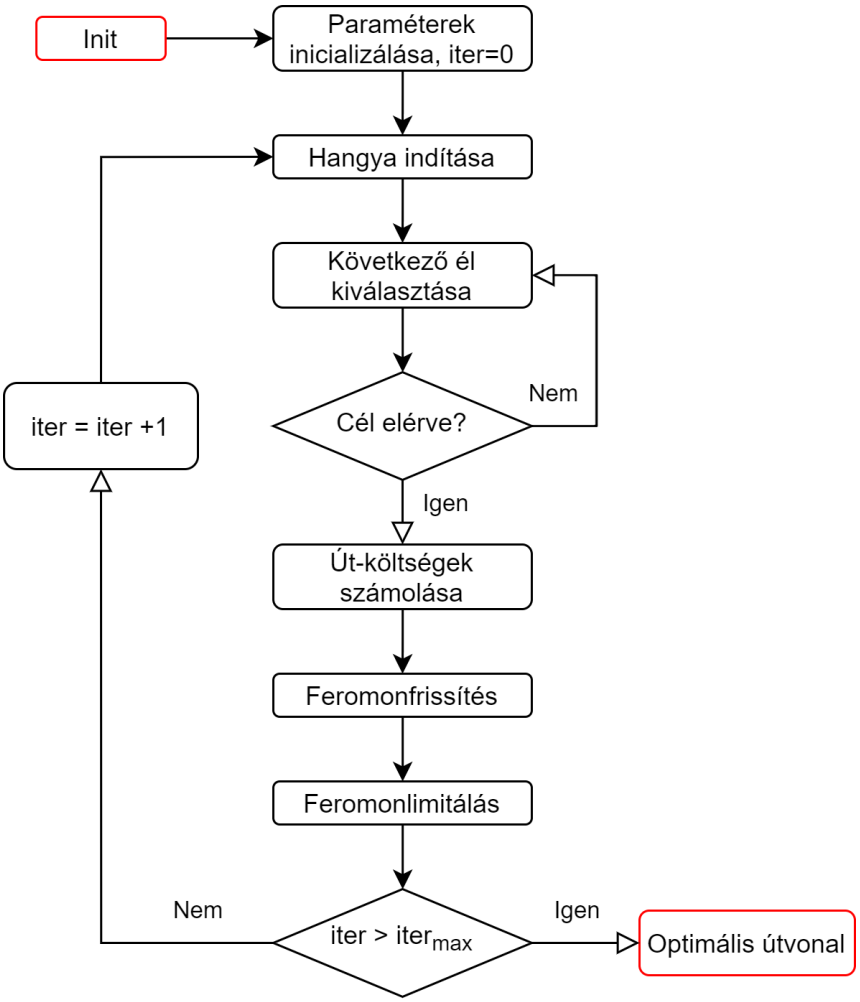
Algoritmus	Rövidítés	Szerzők	Év	Hivatkozás
Hangya Rendszer (Ant System)	AS	Dorigo, Maniezzo és Colomi	1992	[26]
Elitista Hangya Rendszer (Elitist Ant System)	EAS	Dorigo	1992	[26], [27]
Hangyakolónia Rendszer (Ant Colony System)	ACS	Dorigo és Gambardella	1997	[28]
Rangsorolt Hangya rendszer (Rank-based Ant System)	ASRank	Bullnheimer, Hartl és Strauss	1997	[29]
Max-Min Hangya Rendszer (Max-Min Ant System)	MMAS	Stützle és Hoos	2000	[24]

Az 3.1. táblázat a főbb ACO algoritmusokat sorolja fel a megjelenésük sorrendjében. Ezek az algoritmusok effektív és robosztus metaheurisztikának bizonyultak több problémával – különösképpen a VRP variánsaival – szemben [25]. A [20] szerzői például kettő ACS algoritmust alkalmaztak az egy-depós DARP-re: egyet a járműflotta és egy másikat az útköltség minimalizálására.

Ebben a kutatási projektben 2 különböző koncepcióval módosított MMAS algoritmust használtunk a buszok útvonaltervezésére, ezt az alaptípust találtuk legalkalmasabbnak az adott probléma megoldására.

3.1 Max-Min Hangyarendszer

Az eredeti MMAS folyamatábrája az 3.2. ábrán látható. Az algoritmus minden egyes iterációjában egy adott számú hangyaflotta keresi a megoldást. Ezek közül a legjobbat tekintjük az iteráció hangyájának. A feromonfrissítést egyetlen hangya végezheti, amelyik tipikusan vagy az *iteráció-legjobb* vagy az *eddigi-legjobb* hangya lehet.



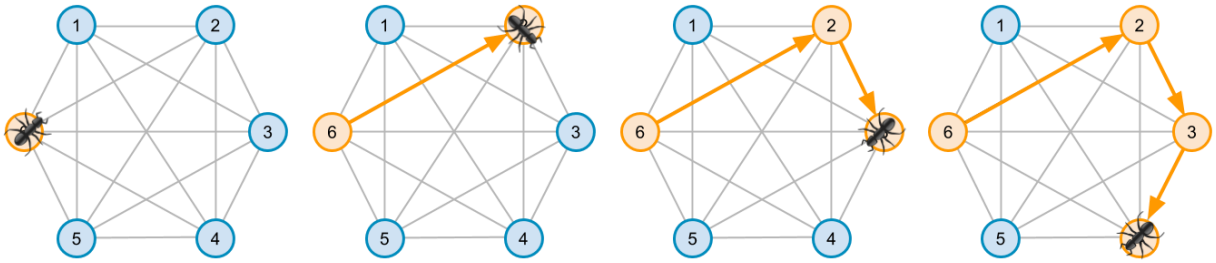
3.2. ábra: MMAS algoritmus folyamatábrája

Ez az algoritmus képes olyan optimális megoldást találni egy hangya számára, mely összefüggő utat eredményez, ilyen például az 1.1. ábra bal oldalán (TSP) és az 1.2. ábra bal oldalán (egy-depós VRP) látható útvonal, valamint ilyen jellegű problémát vázol fel az [5]-ös publikáció. Egy kis módosítással az algoritmus képes lehet egyetlen út helyett egy egész úthalmaz megtalálására, melynek elemei olyan útvonalak, amelyek egy pontból indulnak ki és oda is térnek vissza, lásd: 1.1. ábra jobb oldalán (VRP) és az 1.2. ábra közepén és jobb oldalán (több-járműves DARP) látható útvonalak és a [6]-os hivatkozás problémakörnyezete. A több depós szcenárió már komplexebb természetű, emiatt komolyabb megfontolást igényel. A szakirodalomban eddig nem jelentkezett a probléma ACO algoritmussal történő megoldása, feltételezhető, hogy azért, mert a kutatóknak nem állt rendelkezésre olyan eszköz, mellyel kibővíthették volna a hangyák képességeit. A következő fejezetben ismertetünk 2 módosított MMAS algoritmust, amely alkalmas a probléma megoldására. A matematikai formulák felírásához a [23]-es cikk jelölésrendszerét vettük alapul.

4 A feladat megoldása

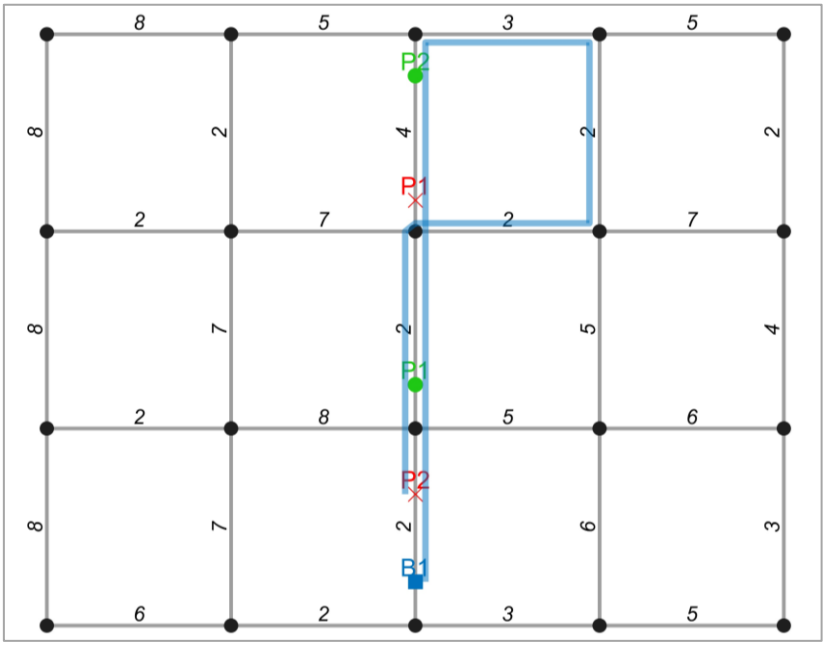
4.1 Élválasztás

A hangyák iteratív módon, véletlenszerűen építik az útjukat, az élek feromonszintje és *kívánatossága* alapján. Az útépítés közben a hangyák nem választhatják az útvonaluk utolsó élét, vagyis közvetlenül nem fordulhatnak vissza. Ez különbözik az – 1.2. fejezetben említett – ügyfél-alapú gráfokon definiált megszorítástól, ahol a hangya az összes eddig meglátogatott pontra nem léphet [2], [20], mely persze előfeltétele a Hamilton-utak keresésének.



4.1. ábra: Éválasztás ügyfélalapú gráfon (forrás: https://github.com/thiagodnf/jacof)

Ezt szemlélteti a 4.1. ábra, ahol a kék csomópontok az engedélyezett, a sárgák pedig a lezárt állomások. A tiltott lépések a $tabu_z$ változóban tárolódnak. Mivel úthálózati reprezentációt használunk, a mi esetünkben az utolsó bejárt él az egyetlen olyan lépés, amely nem engedélyezett. Ez egy fontos különbség, hiszen mi nem Hamilton-utat keresünk. Előfordulhat ugyanis, hogy 2-nél több pontú köröket („visszafordulást") tartalmaz az útvonal. Ilyen körre szükség is lehet, abban az esetben, ha a busz útvonala tartalmazza az utas célpozícióját, de magát az utast még nem vette fel.



4.2. ábra: Élválasztás úthálózati gráfon: a buszok visszafordulhatnak.

Ilyen elrendezést mutat a 4.2. ábra, ahol a *B1* busz a *P1* utas felvevése közben átmegy a *P2* utas célján, majd a *P1* lerakása után felveszi a *P2* utast és visszafordulva elszállítja a céljához (*P2*).

Annak az esélye, hogy a z-edik hangya az (i,j) élt választja az adott iterációban, a következő kifejezés határozza meg:

$$p_{i,j}(z) = \begin{cases} \dfrac{\tau_{i,j}^{\alpha}\eta_{i,j}^{\beta}}{\sum_{x \notin tabu_z} \tau_{i,j}^{\alpha}\eta_{i,j}^{\beta}}, & ha\ j \notin tabu_z \\ 0, & egyébként \end{cases} \tag{7}$$

ahol $\tau_{i,j}$ és $\eta_{i,j}$ az (i,j) él feromonsűrűsége és kívánatossága, α és β a feromonösvény és a kívánatosság relatív fontossága, a summázás pedig az összes engedélyezett élre vonatkozik, hogy normalizáljuk a valószínűséget. A kívánatosság egy heurisztikus információt tartalmazó paraméter, ebben a dolgozatban az élköltség reciprokja: $\eta_{i,j} = \frac{1}{d_{i,j}}$, ahol $d_{i,j}$ az (i,j) él költsége (súlya).

4.2 Feromonfrissítés

MMAS algoritmusban minden iteráció végén egy kiválasztott hangya frissíti a feromonokat. Ebben a dolgozatban csak az adott iterációig futott legjobb hangya hagyhat feromonokat. A feromonfrissítés párolgásból és egy megerősítésből áll:

$$\tau_{i,j} \leftarrow [(1-\rho)\cdot\tau_{i,j} + \triangle\tau_{i,j}^{best}]_{\tau_{min}}^{\tau_{max}} \tag{8}$$

ahol $\rho \in [0,1]$ a párolgás mértéke, τ_{max} a felső, τ_{min} az alsó határa a feromonértékeknek. A megerősítést az eddigi legjobb hangya útvonalán végezzük, $\triangle\tau_{i,j}^{best}$ az eddigi legjobb útvonal feromonértéke, amely a következő formula szerint számítódik:

$$\triangle\tau_{i,j}^{best} = \begin{cases} \dfrac{Q}{L_{best}}, & ha\ a\ legjobb\ útvonal\ tartalmazza\ (i,j)\ élt \\ 0, & egyébként \end{cases}$$

ahol L_{best} az algoritmus kezdete óta talált legjobb útvonal-költség (az utasok út-költségének átlaga, lásd: (5) egyenlet), Q pedig a hangyák kezdeti feromoncsomagjának értéke.

4.3 Adaptív paraméterhangolás

Az ACO algoritmusok paraméterhangolása igen nehéz feladat, mivel sok, egymástól független paraméter befolyásolja a működését:

- α: feromon fontossága
- β: kívánatosság fontássága
- ρ: feromonok párolgása
- τ_{min}: feromonszint alsó határa
- τ_{max}: feromonszint felső határa

A paraméterek értékei tapasztalati úton kerülnek beállításra, melyeket illeszteni kell az adott környezet sajátosságaihoz. Az optimumhoz való konvergencia gyorsasága az α és β paraméterektől nagy mértékben függ, értékük helyes megválasztása kritikus fontosságú.

Eredetileg a fenti paraméterek értékei az inicializálásukat követően konstans értékűek, az algoritmus ezek használja az élválasztáshoz szükséges valószínűség eloszlás kiszámításához (4.1. fejezet) és a feromonfrissítéshez (4.2. fejezet). Ebben a dolgozatban azonban az α és β paramétereket futási időben, az iteráció függvényében változtatjuk, melyet adaptív paraméterhangolásnak neveztünk el.

Nem ez az első munka, amely adaptív paraméterhangolást használ, az evolúciós algoritmusoknál például ez egy visszatérő téma [30] és az ACO algoritmusokban is többször előkerült már [31]. Yao és a munkatársai lineárisan változó α és β paramétereket használtak egy ACO algoritmusban, tengeri herkentyűk szállítási útvonalára [32].

A konstans értékű paraméterekkel a probléma a következő:
Az α paraméter felelős azért, hogy a hangyák az intenzívebb feromonösvény felé orientálódjanak, a β pedig, hogy ezt a rövidebb élek mentén tegyék. Tegyük fel, hogy az α értékét egy nagy számra állítottuk. Ekkor kezdetben a feromonok intenzitása a legjobb útvonal mentén nem sokkal nagyobb, mint a minimális τ_{min} feromonszint értéke a környező éleken, emiatt a hangyák „elcsábulnak” és letérnek a fő feromonösvényről (az α nagy értéke ellenére). Ezzel a hangyáknak lehetőségük van az útvonalat levágásokkal, rövidebb utakkal kiegészíteni, ám ahogy egyre jobb útvonalat találnak, egyre kevésbé fognak letérni a fő feromonösvényről, mert annyira vonzó lesz számukra az eddig talált legjobb útvonal. Következésképp az algoritmus könnyen beragadhat egy lokális minimumba.

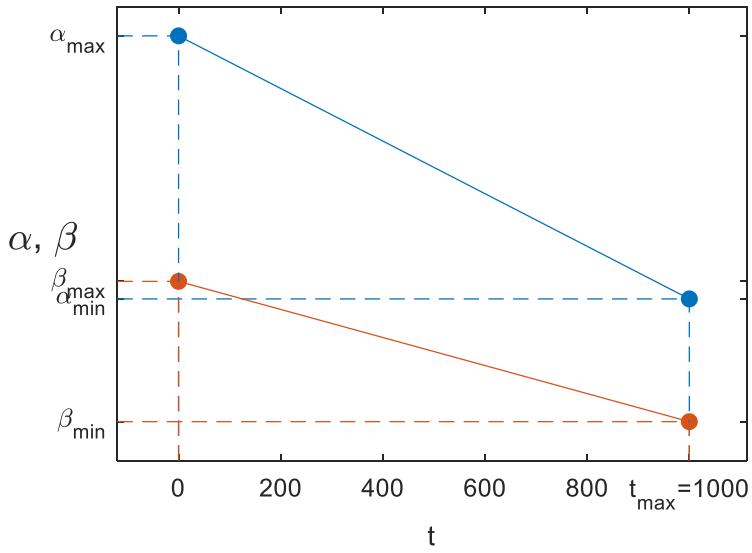
Ahhoz, hogy segítsük a hangyák munkáját, a feromonok fontosságát (az α paraméter értékét) folyamatosan csökkentjük az algoritmus futása során. Azt tapasztaltuk, hogy a lokális minimumból való kiugrást a β paraméter csökkentése is elősegíti, mivel előfordulhat olyan él-elrendeződés a gráfban, ahol egy nagyobb költségű él választása előnyösebb útvonalhoz vezet.

Az α és β paraméter értéke egy egyszerű lineáris függvény szerint változik:

$$\alpha(t) = \frac{\alpha_{min} - \alpha_{max}}{t_{max}} \cdot t + \alpha_{max} \tag{9}$$

$$\beta(t) = \frac{\beta_{min} - \beta_{max}}{t_{max}} \cdot t + \beta_{max} \tag{10}$$

ahol t a ciklusváltozó aktuális értéke, t_{max} pedig a ciklusváltozó maximális értéke, vagyis az iterációszám. A függvények illusztrációja a 4.3. ábrán látható, ahol az iterációszám $t_{max} = 1000$-re lett választva.



4.3. ábra: α és β lineárisan csökken az iterációk során

A következő alfejezetek 2 különböző koncepciót mutatnak be, melyekkel (külön-külön) felvértezve, az MMAS algoritmus képes megoldani a problémát.

4.4 S-MMAS

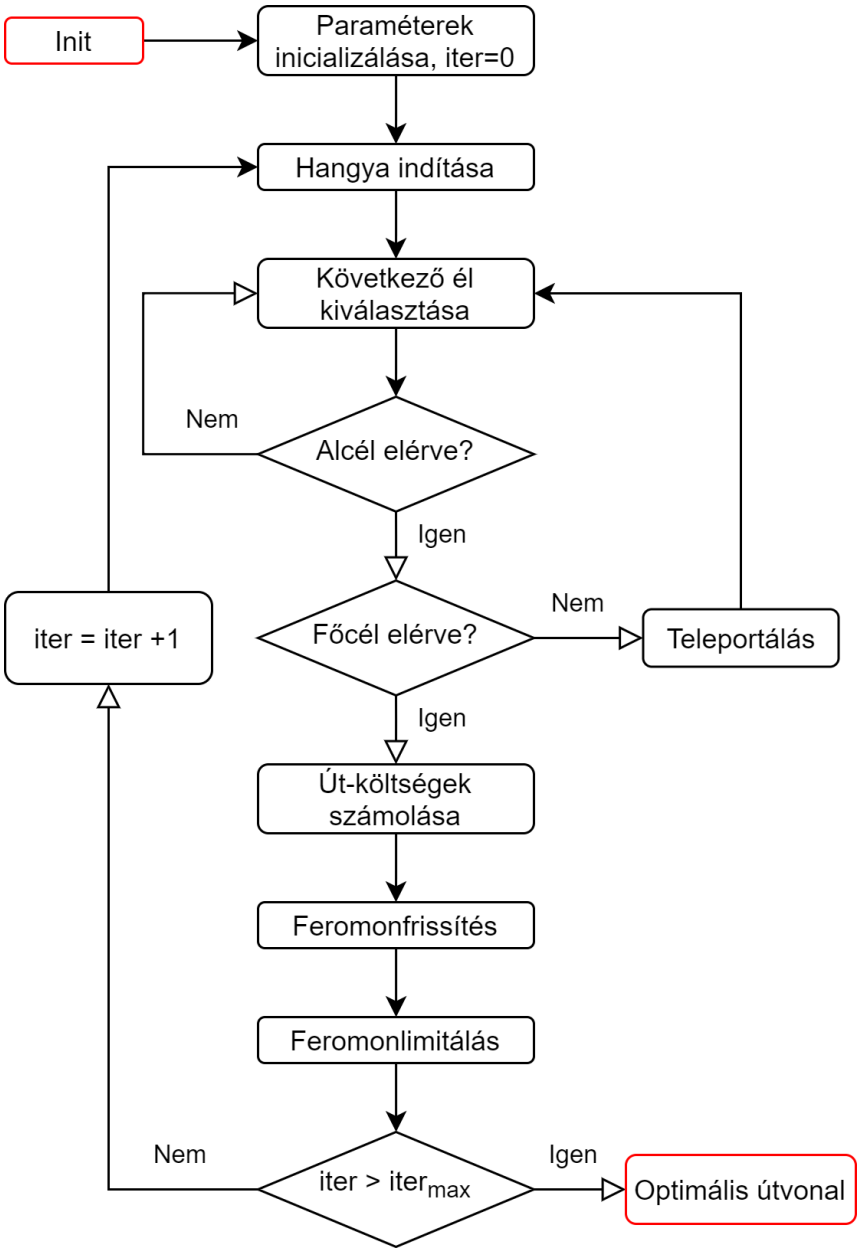
Ahhoz, hogy az elindított hangya több busz útvonalát adja vissza, definiáltunk egy teleportáló képességet, melyet a hangyák speciális feltételek mellett alkalmazhatnak. A használt modellben a buszokat reprezentáló hangyák egyenként vizsgálják az úthálózatot. A feltétel egyszerű: ha egy hangya (busz) az összes eddigi felvett utast letette, akkor költség nélkül átteleportálhat egy másik busz pozíciójára, és onnan folytathatja az útkeresést.

Legyen $t_z = (v_1, v_2, \dots, v_m)$ a z-edik hangya részmegoldása. Tételezzük fel, hogy minden felvett utas $h \in \{1, 2, \dots, n\}: P_h \subset t_z$ kezdő- $v_l = orig(P_h) \in t_z$ és cépozícióját $v_q = dest(P_h) \in t_z$ tartalmazza a részmegoldás $l < q$ sorrendben. Ha ez a feltétel teljesül, és az utolsó állomás az utolsó utas célpozíciójának felel meg $v_m = dest(P_{last})$, akkor a z-edik hangyának lehetősége van egyenlő valószínűséggel átugrani az eddig kihasználatlan buszok valamelyikére:

$$P_v(z) = \begin{cases} \frac{1}{n_{idle}}, & ha\ v \in I_{idle} \\ 0, & egyébként \end{cases} \tag{11}$$

ahol I_{idle} a kihasználatlan buszok halmaza, $n_{idle} = |I_{idle}|$ pedig a kihasználatlan buszok száma. Ezt az ugrást költség nélkül megteheti, tulajdonképpen erre a lépésre az útvonalának a részeként tekintünk.

Az algoritmus elején az összes hangya valamelyik busz pozíciójához van rendelve, mely tekinthető a kiindulási bolynak, innen indulva építik az útjukat. A hangyák külön-külön keresik a megoldást, ami azt jelenti, hogy egyszerre csak egy hangya van jelen a hálózatban. A megoldást kereső hangya az éleket iteratív módon választja a feromonkoncentráció térbeli eloszlásának megfelelően, melyet a (7) egyenlet határoz meg. Ha a hangya az összes felvett utas kezdő és végpontját is tartalmazza (a megfelelő sorrendben), akkor lehetősége van költség nélkül elteleportálni egy másik busz pozíciójába, melyet a (11)-es egyenlet ír le. Erre az eseményre úgy tekintünk, hogy a hangya visszatér a bolyba, majd azután új utat kezd. Azért ezt az értelmezést választottuk és nem azt, hogy visszatérés után egy másik hangya indul, mert a 4.5. fejezetben ismertetett módszer több hangyát használ, melyek párhuzamosan dolgoznak a gráfon. Ezt az algoritmus Serial-Max-Min Ant System-nek (S-MMAS) neveztük el, mivel az értelmezésünk szerint a megoldást kereső hangya sorosan dolgozza fel az úthálózatot. A hangya alkalmas útvonalat járt be, ha az összes utast elszállította a céljához. Ez az útvonal felosztható különálló buszok útvonalára, és tekinthető ezen útvonalak sorozatának vagy láncolatának. Ahogy halad az iteráció, a legkisebb költségfüggvényű út-láncolat fog kihangsúlyozódni.



4.4. ábra: S-MMAS algoritmus folyamatábrája

Az algoritmus folyamatábrája a 4.4. ábrán látható. Az alcél elérése ebben az esetben a fejezet elején ismertetett feltétel teljesülése, a főcél pedig az összes utas kielégítése. A cél alcélra és főcélra való bontása lehetőséget ad egyszerűen implementálni a teleportálás koncepcióját. Az ily módon általánosított módszer potenciálisan kiterjeszthető a VRP több-depós variánsaira és könnyedén adaptálható más probléma-környezetbe is.

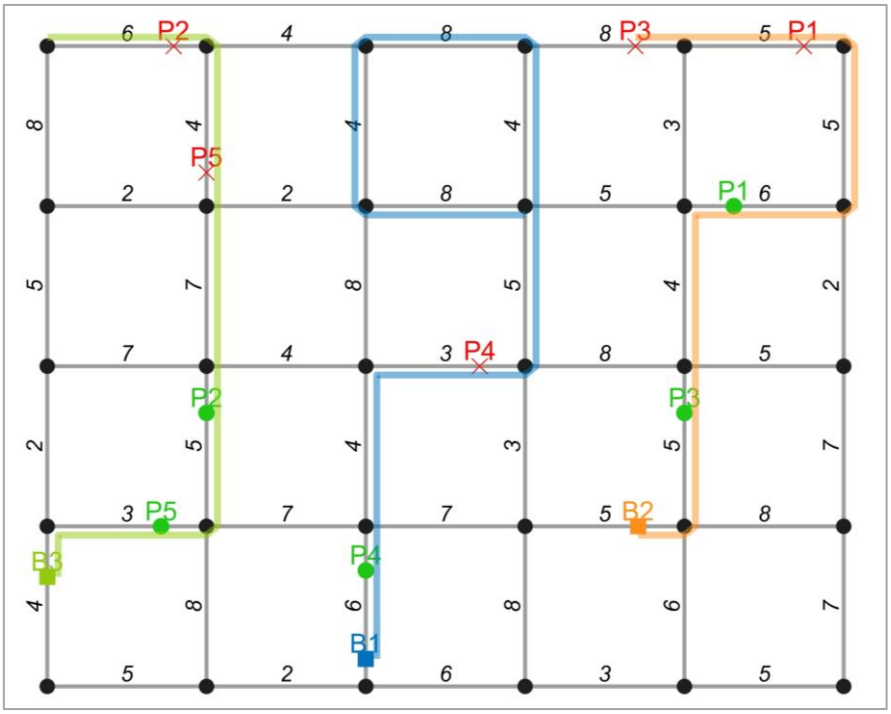
4.5 P-MMAS

Egy másik koncepció a gráf párhuzamos feldolgozása egyszerre több hangya segítségével. A 4.4-es fejezetben tárgyalt módszerben a hangyák egyenként, szekvenciálisan dolgozzák fel a gráfot, a valósághoz viszont közelebb áll a hálózat buszokból való párhuzamos feldolgozása több hangya segítségével.

Minden busz kiindulási pozícióját külön hangyabolynak tekintjük. Az ezekből kilépő hangyák élről-élre lépkedve egymással párhuzamosan építik az útvonalaikat, mellyel végeredményben közvetlenül a buszok útvonalát adják vissza. A hangyák csak a saját fajtáik (a saját bolyukból kilépő más hangyák) által hagyott feromonokra reagálnak, vagyis minden boly saját feromontérképpel rendelkezik. Egy iterációban a buszok számával megegyező $k = |I|$ hangya indul útra, minden bolyból egy, melyek a saját feromontérképükön hagynak feromonokat a (8)-as egyenletnek megfelelően. Az éleket egymással párhuzamosan választják a (7)-es egyenletnek megfelelően, egészen a cél eléréséig, mely továbbra is az összes utas elszállítását jelenti. A cél elérésekor a z_i-edik hangya útvonala $t_{z_i} = \left(v_{1_i}, v_{2_i}, \dots, v_{m_i}\right)$, ahol $m_i = m_j, \forall i, j \in \{1, 2, \dots, k\}$, azaz minden iterációbeli hangya útvonala ugyanannyi pontból áll. Ezeket az útvonalakat vissza kell „metszeni” a hangyák által felvett utolsó utas céljáig, mivel a hangyákat túlfuttatja az algoritmus az érvényes megoldáson:

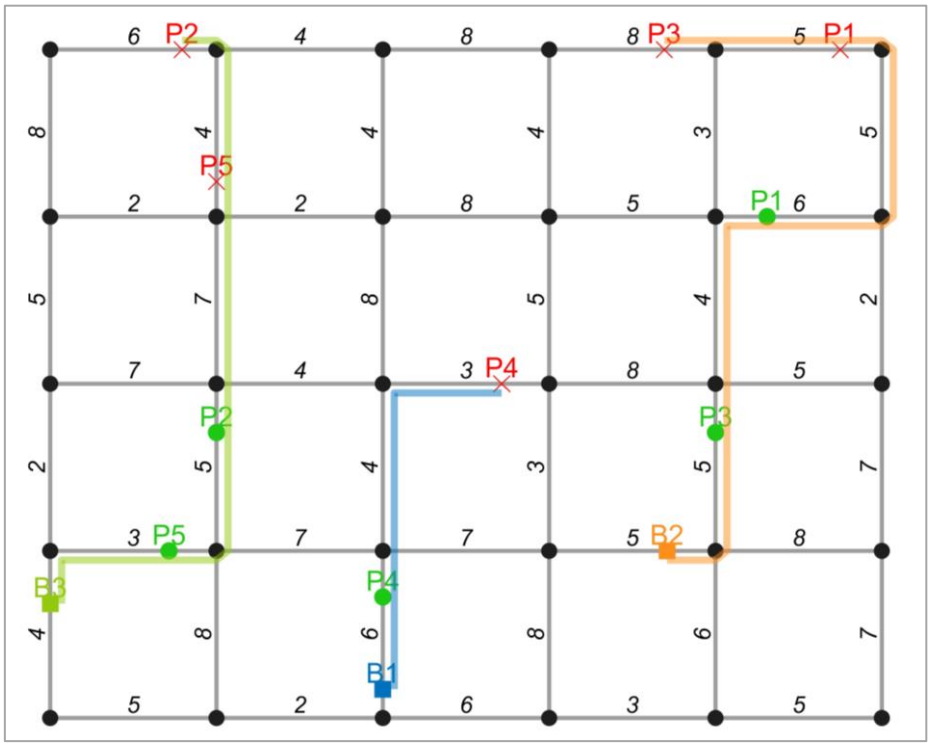
$$t_{z_i} \leftarrow \left(v_{1_i}, v_{2_i}, \dots, v_{q_i}\right), v_{j_i} \in t_{z_i}, \forall j \in \{1, 2, \dots q\} \tag{12}$$

ahol $v_{q_i} = dest(P_{last_i})$ a z_i-edik hangya utolsó utasának célpozíciója.



4.5. ábra: A hangyák túlfutása: B1 és B3 busz felesleges lépéseket tartalmaz

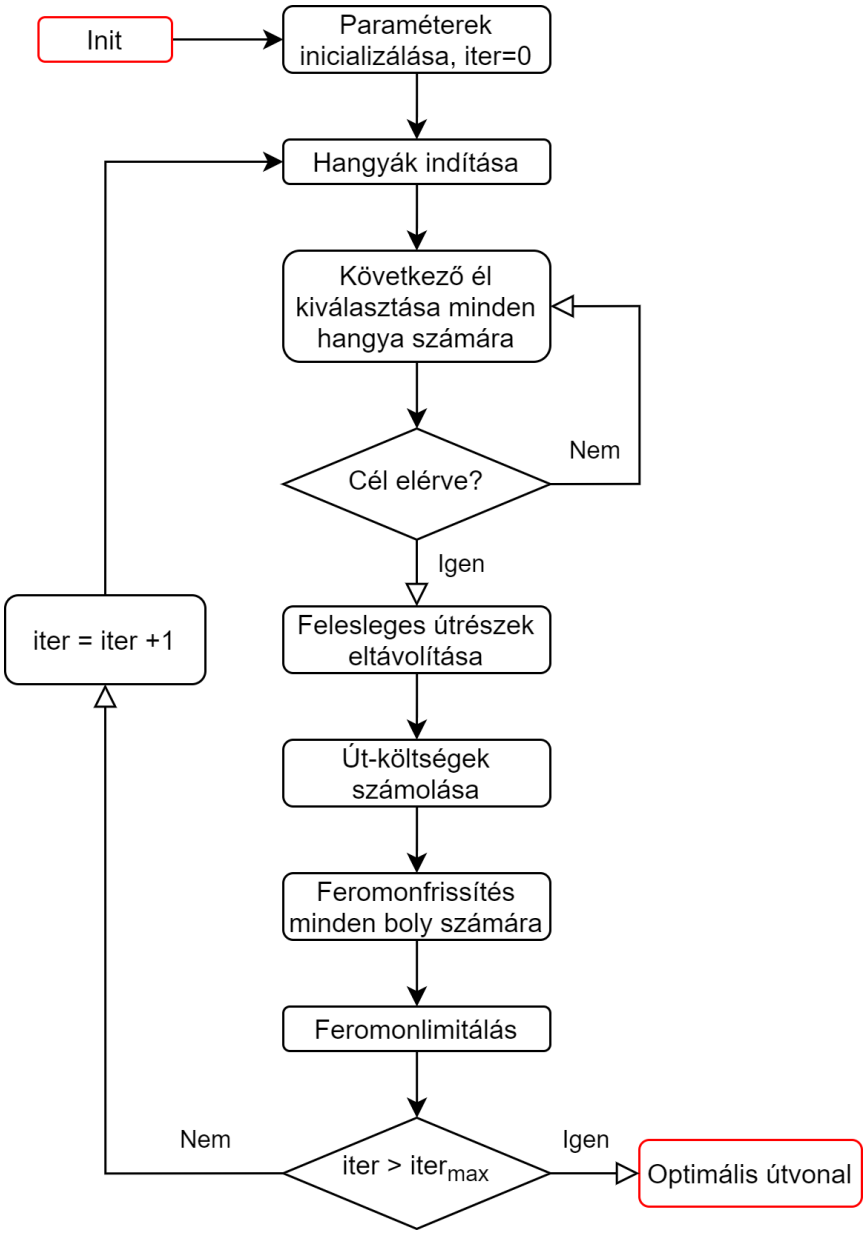
Ezt mutatja a 4.5. ábra, ahol a B2 busz 10 pontból álló útvonala a leghosszabb. Az algoritmus a B1 és B3 busz útvonalát is 10 pontra egészíti ki, ám ezek egy része haszontalan, így el kell távolítani. Az eltávolítás a (12) kifejezés szerint az utolsó utas letevése utáni felesleges lépések elhagyását jelenti.



4.6. ábra: A túlfutás eltávolítása

A 4.6. ábrán a túlfutás eltávolításával kapott eredményt látjuk. A hangyák csak ezután frissíthetik a saját feromontérképüket. A frissítés természetesen az érvényes útvonalak mentén történik. Az egyéni feromontérkép lehetőséget ad arra, hogy a különböző buszpozíciókból induló feromonösvények ne befolyásolják egyidejűleg a hangyák élválasztását, így csak az adott buszhoz tartozó releváns információt dolgozzák fel. Ez gyorsítja az MMAS konvergálását az optimális megoldáshoz.

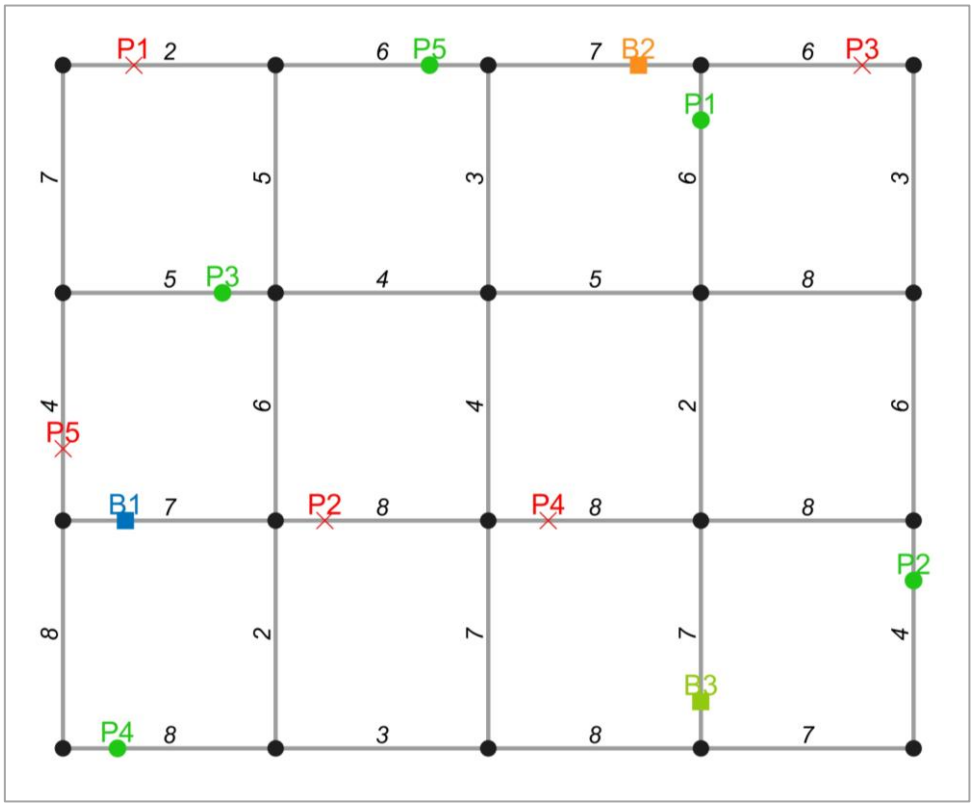
A 4.7. ábra az algoritmus folyamatábráját mutatja be, mely nagyon hasonló az eredeti MMAS folyamatábrájához (3.2. ábra), viszont lényeges különbség van a *Következő él kiválasztása* és a *Feromonfrissítés* lépések filozófiájában, ahogy azt fentebb kifejtettük.



4.7. ábra: P-MMAS algoritmus folyamatábrája

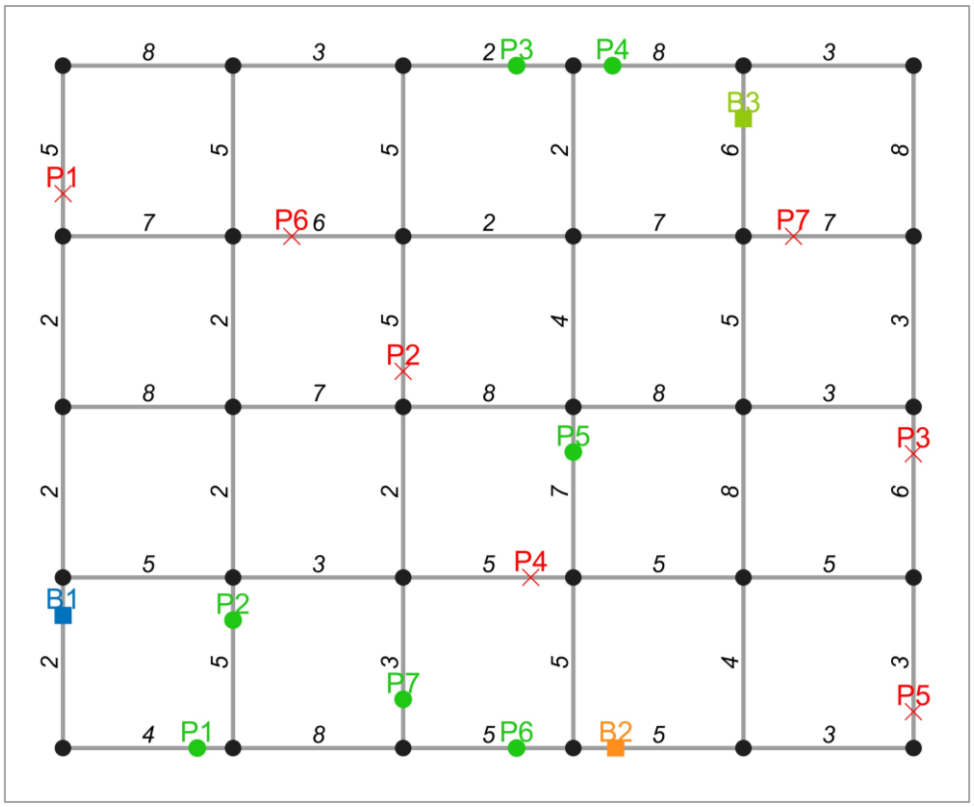
5 Teszt eredmények

Az S-MMAS és P-MMAS algoritmusokat különböző komplexitású problémákon teszteltük. Az úthálózat költségeit, a buszok és utasok helyzetét random generáltuk. Az eredmények az 5.1. táblázatban és az 5.4. táblázatban láthatók, ahol közzétettük az adott problémán elért legjobb és az átlagos eredményt (lásd: (5) kifejezés), valamint az átlagos futási időt. A tesztet egy átlagos számítási kapacitással és teljesítménnyel bíró laptopon futtattuk. A különböző problémákat sorszámmal láttuk el, megadtuk az úthálózat méretét, valamint a rajta elhelyezkedő buszok és utasok számát. Az azonos sorszámú problémák azonos feladatokhoz tartoznak mindkét táblázatban.



5.1. ábra: 1. probléma

Az 1. sorszámú probléma egy 4×5-ös úthálózat 3 busszal és 5 utassal (5.1. ábra). A nagyobb sorszámokhoz komplexebb feladat társul. A legnehezebb a 7. sorszámú feladat, ahol egy 5×6-os úthálózaton 3 busznak kell elszállítani 7 utast (5.2. ábra).



5.2. ábra: 7. probléma

5.1 S-MMAS eredményei

A 5.1. táblázatban különböző a sorszámmal ellátott problémák numerikus eredményei láthatók. Észrevehető, hogy a probléma komplexitásának növelése nagyobb futási idővel, valamint a legjobb útvonal és az átlagos útvonal költsége közötti különbség növekedésével jár.

5.1. táblázat: S-MMAS teljesítménye

	Úthálózat mérete	Buszok × utasok	Legjobb útvonal-költség	Átlagos útvonal-költség	Átlagos futási idő
1.	4 × 5	3 × 5	183	198.6	26.87 s
2.	4 × 5	3 × 6	201.2	238.25	38.7 s
3.	4 × 6	3 × 5	212	261.38	35.31 s
4.	4 × 6	3 × 6	238	283.43	45.17 s
5.	5 × 6	3 × 5	281.8	346.4	51.82 s
6.	5 × 6	3 × 6	324.2	421.57	64.45 s
7.	5 × 6	3 × 7	313	374.33	61.05 s

Az S-MMAS algoritmus paraméterkészletét tapasztalati úton határoztuk meg, mely az 1-4 probléma esetén az 5.2. táblázatban, az 5-7 probléma esetén pedig az 5.3. táblázatban található:

5.2. táblázat: S-MMAS paraméterkészlete 1-4 probléma esetén

$Iter_{max}$	Hangyák száma	α_{max}	α_{min}	β_{max}	β_{min}	ρ	Q	τ_{max}	τ_{min}
700	3	1.9	0.9	0.8	0.5	0.5	70	0.9	0.06

5.3. táblázat: S-MMAS paraméterkészlete 5-7 probléma esetén

$Iter_{max}$	Hangyák száma	α_{max}	α_{min}	β_{max}	β_{min}	ρ	Q	τ_{max}	τ_{min}
1000	3	2.1	1.0	0.7	0.4	0.5	90	0.45	0.03

5.2 P-MMAS eredményei

Az 5.4. táblázatban a P-MMAS algoritmus teljesítménye látható. A táblázat 1. oszlopában levő sorszámok ugyanazon problémákat jelölik, mint a 5.1. táblázat 1. oszlopában levők. Itt is észrevehető a futási idő növekedése, valamint az átlagos és legjobb útvonal-költség közötti különbség.

5.4. táblázat: P-MMAS teljesítménye

	Úthálózat mérete	Buszok × utasok	Legjobb útvonal-költség	Átlagos útvonal-költség	Átlagos futási idő
1.	4 × 5	3 × 5	183	192.22	10.13 s
2.	4 × 5	3 × 6	201.2	226.4	13.17 s
3.	4 × 6	3 × 5	212	249.33	15.31 s
4.	4 × 6	3 × 6	231.6	272.04	16.55 s
5.	5 × 6	3 × 5	258.4	339.65	23.92 s
6.	5 × 6	3 × 6	288.2	350.12	25.74 s
7.	5 × 6	3 × 7	273	319.52	27.52 s

A megoldások során használt paraméterszett az 5.5. táblázat és az 5.6. táblázatban találhatóak, értéküket a 5.2. táblázathoz és az 5.3. táblázathoz hasonlóan tapasztalati úton állítottuk be. Mindkét algoritmus ugyanazt a paraméterkészletet használta a megegyező problémákon.

5.5. táblázat: P-MMAS paraméterkészlete 1-4 probléma esetén

$Iter_{max}$	α_{max}	α_{min}	β_{max}	β_{min}	ρ	Q	τ_{max}	τ_{min}
700	1.9	0.9	0.8	0.5	0.5	70	0.9	0.06

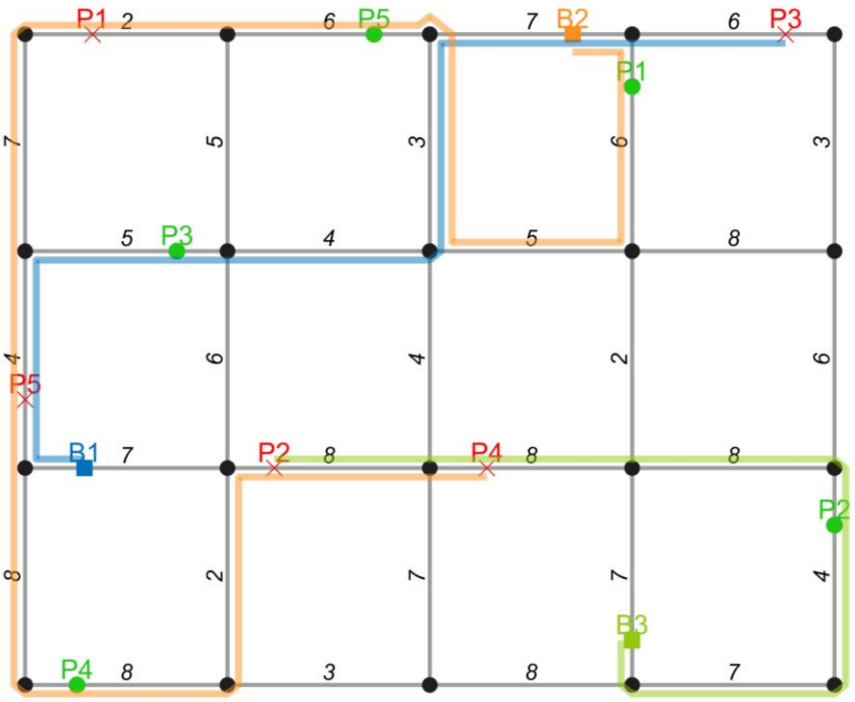
5.6. táblázat: P-MMAS paraméterkészlete 5-7 probléma esetén

$Iter_{max}$	α_{max}	α_{min}	β_{max}	β_{min}	ρ	Q	τ_{max}	τ_{min}
1000	2.1	1.0	0.7	0.4	0.5	90	0.45	0.03

5.3 Eredmények összevetése

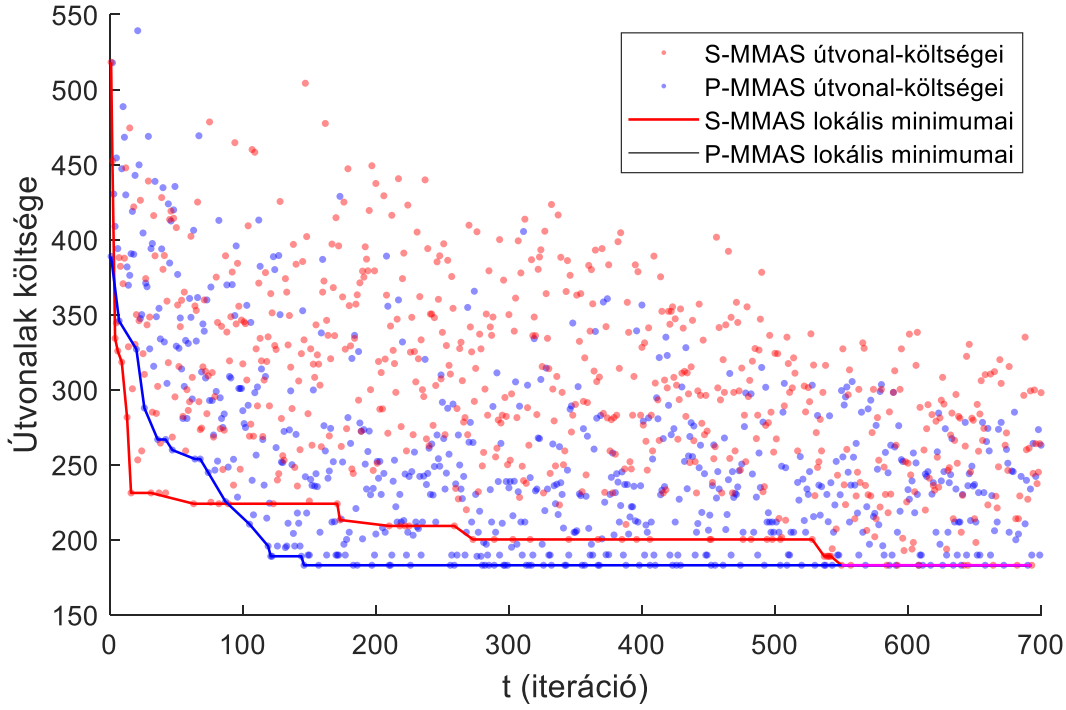
Az 5.1. táblázatból és az 5.4. táblázatból egyértelműen látszik, hogy a P-MMAS felülmúlja az S-MMAS-t mind a megoldások minőségében, mind az algoritmus futási idejében. Ez várható volt, hiszen a párhuzamos feldolgozás közelebb áll a valóságban tapasztaltakkal.

Az első 3 problémánál ugyanazt a legrövidebb úthalmazt találta meg mindkét algoritmus, a többinél viszont kivétel nélkül jobb megoldást nyújtott a P-MMAS. Az 1. problémára talált megoldás az 5.3. ábrán látható.



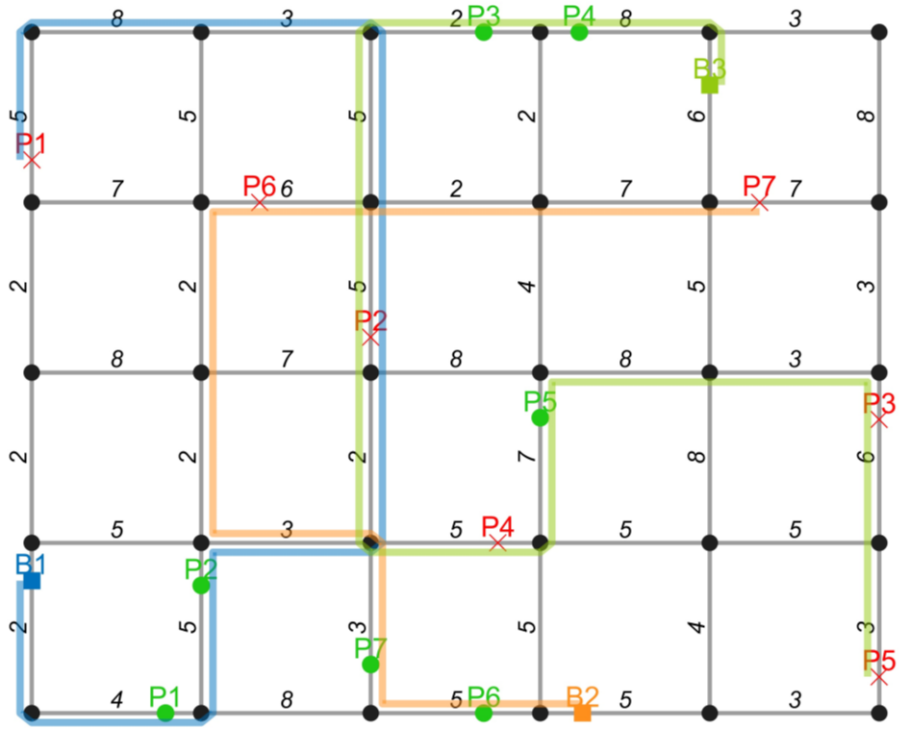
5.3. ábra: 1. probléma megoldása: B1{P3}, B2{P1, P5, P4}, B3{P2}

Az már látszik, hogy mindkét megoldás képes közel optimális megoldást adni, viszont érdekes lehet feltenni a kérdést, hogy vajon melyik algoritmus talált rá hamarabb erre az eredményre. Az algoritmusok konvergenciáját mutatja az 5.4. ábra, ahol az összes iterációbeli hangya útvonal-költségét egy pont reprezentálja.

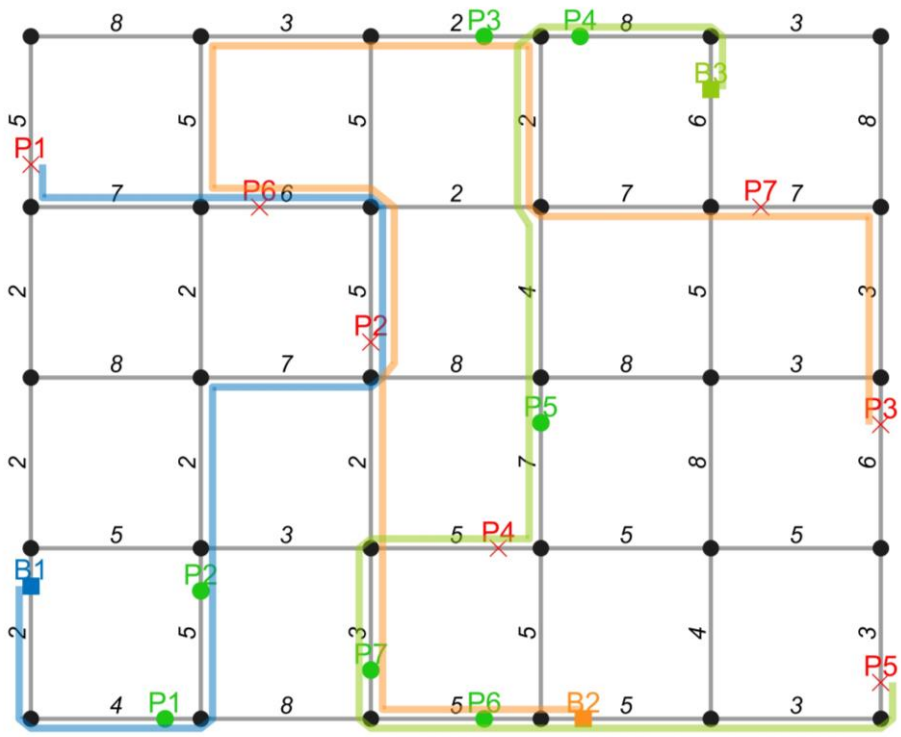


5.4. ábra: 1. probléma konvergenciája

Láthatjuk, hogy a P-MMAS már a 150. iterációnál megtalálta az optimális megoldást, míg az S-MMAS csak az 550-nél. Ez lényeges teljesítménybeli különbség. A további összehasonlítás végett a 7. (legnehezebb) feladatra adott megoldások az 5.5. és 5.6. ábrán láthatók.

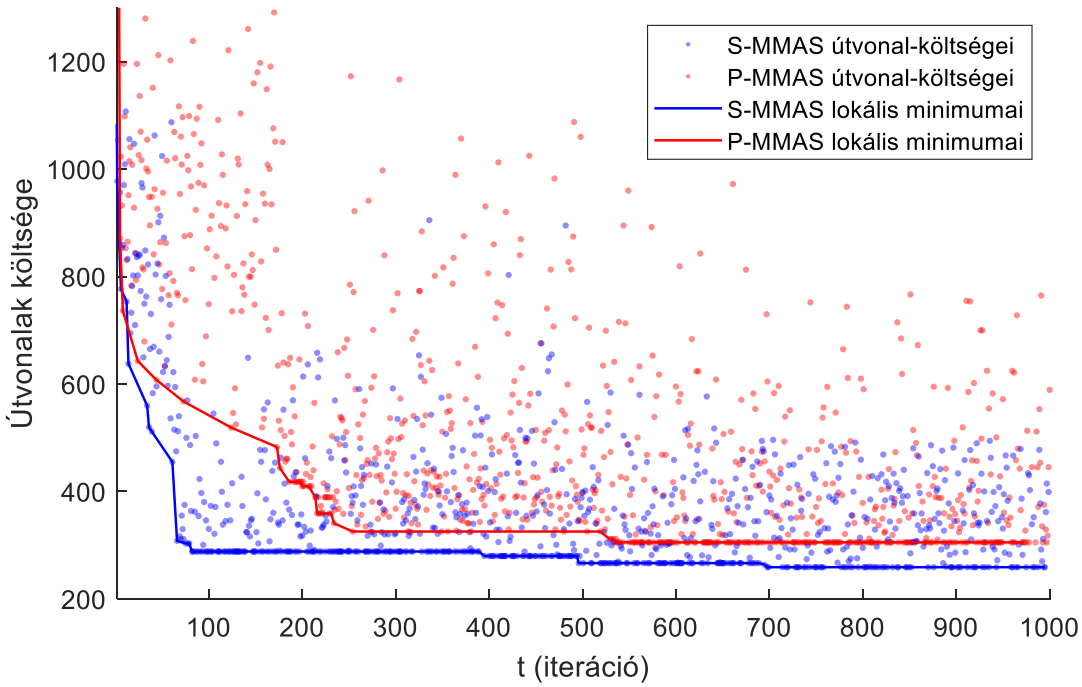


5.5. ábra: S-MMAS – 7. probléma megoldása: B1{P1, P2}, B2{P6, P7}, B3{P4, P3, P5}



5.6. ábra: P-MMAS – 7. probléma megoldása: B1{P1, P2}, B2{P6, P3, P7}, B3{P4, P5}

A buszok különböző utasokat vesznek fel a két megoldásban, ám az 5.6. ábra útvonalhalmaza kevesebb összköltséggel rendelkezik, mint az 5.5. ábráé. A hangyák útvonal-költségeinek az összehasonlítása az 5.7. ábrán látható.



5.7. ábra: 7. probléma konvergenciája

Mindkét algoritmus konvergenciája exponenciális jelleget mutat az összes probléma esetén, ám a P-MMAS kvalitatív és kvantitatív szempontból is túlszárnyalja az S-MMAS-t. A különbség a komplexebb problémák esetén jelentős, ugyanis amellett, hogy jobb megoldást talál, a futási ideje is alacsonyabb. A paraméterek helyes megválasztása fontos, mert nagy mértékben befolyásolja a konvergenciát. Egész valószínű, hogy létezik a fenti paraméterkészlettől eltérő beállítás, mellyel az algoritmus gyorsabban konvergál, így javul az eredmény minősége és a futási ideje. Stütze és Hoos közreadtak egy irányelvet a cikkükben [24], mely segítő kezet nyújt a paraméterek megválasztására, ám az útmutatás által nyújtott paraméterkészlet rosszabb teljesítménnyel rendelkezett, mint az általunk meghatározott paraméterszett.

6 Összefoglalás és kitekintés

A dolgozat fő célja egy olyan közlekedési rendszer felvázolása, melyben a városi buszok kötött pályás közlekedés helyett szabadon mozognak az úthálózaton. A koncepció lényege, hogy a buszok optimális útvonalakon közlekedve szállítják el az utasokat a kezdőpontjuktól közvetlenül a céljukig, háztól-házig való szállítást megvalósítva. A feladat olyan úthalmaz megtalálása, mely mentén az utasok a legrövidebb idő alatt jutnak el a kívánt céljukhoz. Megadtuk a felvetés matematikai modelljét, melyben úthálózati gráfot feltételeztünk és 2 módosított MMAS algoritmust, melyek optimális megoldást nyújtanak alacsony dimenziójú esetekben. Az S-MMAS és P-MMAS algoritmus egy olyan általánosításai az eredetinek, melyek alkalmazhatók olyan környezetben, ahol a probléma valamilyen több-kiindulópontú multi-ágens rendszerkoordinálás jelleget hordoz. Tipikusan ilyenek a DARP és VRP problémacsalád bonyolultabb variánsai. Az alacsony komplexitású esetekben kielégítő eredményeket kaptunk, azonban a magasabb dimenziójú problémák megoldásához további fejlesztés szükséges. A P-MMAS minőségben és futási időben is felülmúlja az S-MMAS algoritmust, így ez bizonyult az alkalmasabb elképzelésnek.

A P-MMAS legnagyobb hátránya a futási idő, melynek csökkentése a további fejlesztéseket tekintve központi jelentőségű. A paraméterek megfelelő megválasztásával egyértelműen javul az algoritmus teljesítménye, viszont ennek a meghatározása még további analíziseket igényel. A futási idő csökkentésére hatékony eszköz lehet a párhuzamosítás: a hangyákat ki lehet szervezni a CPU egyes magjaira, melyek ezután párhuzamosan futnak minden iterációban, majd az iteráció végén a legjobb hangyát választjuk ki. Ezzel kevesebb iteráció is elegendő ugyanannak az eredménynek az eléréséhez.

A problémát ki lehet egészíteni azzal, hogy a buszok egy előre egyeztett pontnál találkoznak és utasokat cserélnek, ha ezzel rövidebb idő alatt érnek a céljukhoz.

A későbbiekben további megkötéseket is figyelembe lehet venni, mint például a buszok kapacitása, idő-ablakok, buszok heterogenitása vagy a probléma dinamikus változata. A jövőben tervezzük leimplementálni a probléma ügyfél-alapú gráfon történő megoldását, melyhez rendelkezésre állnak már a dolgozatban szereplő S-MMAS és P-MMAS algoritmusok.

Felhasznált irodalom

[1] M. R. Garey and D. S. Johnson,Computers and intractability. freeman San Francisco, 1979, vol. 174.

[2] P. Toth and D. Vigo,The vehicle routing problem. SIAM, 2002.

[3] H. Hernández-Pérez and J.-J. Salazar-González, "Heuristics for the one-commodity pickup-and-delivery traveling salesman problem," Transportation Science, vol. 38, no. 2, pp. 245–255, 2004.

[4] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem (darp): Variants, modeling issues and algorithms, "Quarterly Journal of the Belgian, French and Italian Operations Research Societies, vol. 1, no. 2, pp. 89–101, 2003.

[5] H. N. Psaraftis, "A dynamic programming solution to the single vehiclemany-to-many immediate request dial-a-ride problem," Transportation Science, vol. 14, no. 2, pp. 130–154, 1980.

[6] J.-F. Cordeau and G. Laporte, "A tabu search heuristic for the static multi-vehicle dial-a-ride problem," Transportation Research Part B: Methodological, vol. 37, no. 6, pp. 579–594, 2003.

[7] J. Park and B.-I. Kim, "The school bus routing problem: A review," European Journal of operational research, vol. 202, no. 2, pp. 311–319, 2010.

[8] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," Transportation science, vol. 40, no. 4, pp. 455–472, 2006.

[9] Y. Huang, L. Zhao, T. Van Woensel, and J.-P. Gross, "Time-dependent vehicle routing problem with path flexibility," Transportation Research Part B: Methodological, vol. 95, pp. 169–195, 2017.

[10] S. Ropke, J.-F. Cordeau, and G. Laporte, "Models and branch-and-cut algorithms for pickup and delivery problems with time windows," Networks: An International Journal, vol. 49, no. 4, pp. 258–272, 2007.

[11] S. N. Parragh, "Introducing heterogeneous users and vehicles intomodels and algorithms for the dial-a-ride problem," Transportation Research Part C: Emerging Technologies, vol. 19, no. 5, pp. 912–930, 2011.

[12] G. Berbeglia, J.-F. Cordeau, and G. Laporte, "Dynamic pickup and delivery problems," European journal of operational research, vol. 202, no. 1, pp. 8–15, 2010.

[13] T. Garaix, C. Artigues, D. Feillet, and D. Josselin, "Vehicle routing problems with alternative paths: An application to on-demand transportation," European Journal of Operational Research, vol. 204, no. 1, pp. 62–75, 2010.

[14] H. Ben Ticha, N. Absi, D. Feillet, and A. Quilliot, "Vehicle routing problems with road-network information: State of the art," Networks, vol. 72, no. 3, pp. 393–406, 2018.

[15] J.-F. Cordeau, “A branch-and-cut algorithm for the dial-a-ride problem, ”Operations Research, vol. 54, no. 3, pp. 573–586, 2006.

[16] R. M. Jorgensen, J. Larsen, and K. B. Bergvinsdottir, “Solving the dial-a-ride problem using genetic algorithms,” Journal of the operational research society, vol. 58, no. 10, pp. 1321–1331, 2007.

[17] S. N. Parragh, K. F. Doerner, and R. F. Hartl, “Variable neighborhood search for the dial-a-ride problem,” Computers & Operations Research,vol. 37, no. 6, pp. 1129–1138, 2010.

[18] K. Braekers, A. Caris, and G. K. Janssens, “Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots,” Transportation Research Part B: Methodological, vol. 67, pp.166–186, 2014.

[19] P. Detti, F. Papalini, and G. Z. M. de Lara, “A multi-depot dial-a-ride problem with heterogeneous vehicles and compatibility constraints in healthcare,” Omega, vol. 70, pp. 1–14, 2017.

[20] T. Tripathy, S. C. Nagavarapu, K. Azizian, R. R. Pandi, and J. Dauwels, “Solving dial-a-ride problems using multiple ant colony system with fleet size minimisation,” in UK Workshop on Computational Intelligence. Springer, 2017, pp. 325–336.

[21] J. Kennedy, “Swarm intelligence,” in Handbook of nature-inspired and innovative computing. Springer, 2006, pp. 187–219.

[22] M. Dorigo, A. Colorni, and V. Maniezzo, “Distributed optimization by ant colonies,” 1991.

[23] M. Dorigo, M. Birattari, and T. Stutzle, “Ant colony optimization,” IEEE computational intelligence magazine, vol. 1, no. 4, pp. 28–39, 2006.

[24] T. Stützle and H. H. Hoos, “Max–min ant system,” Future generation computer systems, vol. 16, no. 8, pp. 889–914, 2000.

[25] A. E. Rizzoli, R. Montemanni, E. Lucibello, and L. M. Gambardella,“ Ant colony optimization for real-world vehicle routing problems,” Swarm Intelligence, vol. 1, no. 2, pp. 135–151, 2007.

[26] M. Dorigo, “Optimization, learning and natural algorithms, ”PhD Thesis, Politecnico di Milano, 1992.

[27] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: optimization by a colony of cooperating agents, ”IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 26, no. 1, pp. 29–41, 1996

[28] M. Dorigo and L. M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem, ”IEEE Transactions on evolutionary computation, vol. 1, no. 1, pp. 53–66, 1997.

[29] B. Bullnheimer, R. F. Hartl, and C. Strauss, “A new rank based version of the ant system.

[30] F. Lobo, C. F. Lima, and Z. Michalewicz, Parameter setting in evolutionary algorithms. Springer Science & Business Media, 2007, vol. 54.

[31] T. Stützle, M. López-Ibánez, P. Pellegrini, M. Maur, M. M. De Oca, M. Birattari, and M. Dorigo, “Parameter adaptation in ant colony optimization,” in Autonomous search. Springer, 2011, pp. 191–215.

[32] B. Yao, P. Hu, M. Zhang, and X. Tian, “Improved ant colony optimization for seafood product delivery routing problem, ”Promet-Traffic&Transportation, vol. 26, no. 1, pp. 1–10, 2014.

Ábrajegyzék

Táblázatjegyzék