SDU International Journal of Technological Sciences
Vol. 9, No 2, August 2017
pp. 50- 60

Computational Technologies

GA, AS, ACS VE MMAS ALGORİTMALARI PERFORMANSLARININ GEZGİN SATICI PROBLEMİ ÇÖZÜMÜ ÜZERİNDE DEĞERLENDİRİLMESİ

Raed AL-BADRİ *, Tuncay AYDOĞAN

Geliş Tarihi/ Received: 30.03.2017, Kabul tarihi/Accepted: 13.08.2017

Özet

Gezgin Satıcı Problemi (GSP) bir çok alanda kendisine uygulama bulmuş önemli bir optimizasyon problemidir. Bu çalışmada, sezgisel optimizasyon algoritmalarından Genetik Algoritma (GA), Karınca Sistemi (Ant System-AS/ANT), Karınca Koloni Sistemi (Ant Colony System-ACS) ve Max-Min Karınca Sistemi (Max-Min Ant System-MMAS) algoritmaları ile GSP çözülerek, çözümlerin performansları incelenmiştir. Algoritmaların tamamı bir arayüz üzerinde bulunmaktadır. Arayüzde istenilen sayıda rastgele oluşturulan noktalar (şehirler) ile haritalar oluşturulabilmekte veya hazır kütüphanelerden veri seti yüklenebilmektedir. Bu algoritmaların performansları maliyet (yol uzunluğu) ve tekrar sayısı olarak görülebilmektedir. Algoritmalar 36, 56, 76, 101 ve 150 nokta (şehir)'den oluşan 5 harita üzerinde denenmiştir. Her harita çözümünde en az maliyetli çözümü MMAS, en yüksek maliyetli çözümü GA'nın oluşturduğu görülmüştür. Sıralama azdan yükseğe doğru MMAS, AS, ACS ve GA biçiminde gerçekleşmiştir. Algoritmaların TSPLIB kütüphanesi içerindeki ch150 veri seti için performansları literatür ile karşılaştırılmış GA, AS ve MMAS'de daha düşük maliyetlere ulaşıldığı görülmüştür.

Anahtar Kelimeler: Genetik Algoritma, Karınca Kolonisi Optimizasyonu, Gezgin Satıcı Problemi

GA, AS, ACS AND MMAS ALGORITHMS PERFORMANCE EVALUATION ON TRAVELING SALESMAN PROBLEM SOLVING

Abstract

Travelling Salesman Problem (TSP) is an important optimization method that have been applied to various areas. In this study, TSP is solved by using several heuristic algorithms like Genetic Algorithm (GA), Ant System (AS/ANT), Ant Colony System (ACS) and Max-Min Ant System (MMAS), performances of these algorithms are then measured. Applied algorithms are implemented inside an interface. Using this interface, maps can be generated with as much as required random points (cities) or loaded from dataset. Performance criterion of the measurement may be seen as the cost (path length) and the number repetitions. Performance measurements of these algorithms are tested on 5 different maps consisting of 36, 56, 76, 101 and 150 points. At each test for solving TSP for each map, the least cost is observed for MMAS and the highest cost is observed for GA. Ascending sort of these algorithms based on their cost is observed as MMAS (least), AS, ACS and GA (highest). The performance of the algorithms for the ch150 dataset in the TSPLIB library was found to be lower in GA, AS and MMAS compared to the literature.

Key Words: Genetic Algorithm, Ant Colony Optimization, Travel Salesman Problem

*Süleyman Demirel Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, , Isparta, Türkiye
E-posta: raed.albadri@gmail.com

1. Giriş

Gezgin Satıcı Problemi (GSP), aralarında doğrudan yollar bulunan birden fazla nokta (şehiri) üzerinde, bir başlangıç noktasından başlayarak başladığı noktaya dönerken bir noktadan bir daha geçmemek ve tüm noktalara uğramak koşulu ile en kısa yolu, güzergahı hesaplayan bir optimizasyon problemidir. Bu problem 1800'lerde matematikçi W. R. Hamilton ve Thomas Kirkman tarafından tanımlanmıştır (Saiyed 2012).

Problemde noktalar arası mesafe maliyet olarak tanımlanarak en kısa yol en düşük maliyeti ortaya koymaktadır. GSP başta mühendislik ve istatistik olmak üzere hemen hemen tüm alanlardaki bazı problemlere uyarlanarak çözüm olmuştur. Geçmişten günümüze GSP için birçok çözüm metodolojileri geliştirilmiştir (Albayrak, vd. 2011; La Maire, vd. 2012; Karaboga, vd. 2011; Chena, vd. 2011; Erdem, vd. 2011; Kuşcu, vd. 2011).

Son yirmi yıl içerisindeki sezgisel yaklaşım metodolojilerindeki artışlar bu problemin çözümüne de katkı sağlamıştır. Literatürde Genetik Algoritmalar, Yapay Sinir Ağları, Karınca Kolonisi, Arı Kolonisi, Swarm, Kanguru Algoritması yöntemleri ve bu yöntemlerden geliştirilen yöntemler en çok çalışılanlar olmuştur (Albayrak, vd. 2011; La Maire, vd. 2012; Karaboga, vd. 2011; Chena, vd. 2011; Erdem, vd. 2011; Kuşcu, vd. 2011). Her yeni çalışma maliyeti ve hesaplama süresini daha da azaltmayı hedeflemiştir. GSP araştırmacıların geliştirdikleri yeni optimizasyon algoritmalarını sınayacakları bir araç olmuştur.

Bu çalışmada gezgin satıcı problemi GA, AS, ACS ve MMAS algoritmaları ile daha düşük maliyette sonuçlar üretmek amacıyla kodlanmış, performansları kendi aralarında ve literatür ile kıyaslanarak analiz edilmiştir. Önce, algoritmaların geçerliğini görmek için karmaşık sayılmayacak, az sayıda noktadan oluşan bir harita üzerinde algoritmalar çalıştırılmış ve tamamının aynı maliyette çözümler ürettiği görülmeye çalışılmıştır. Daha sonra da karmaşık sayılabilecek 36, 56, 76, 101 ve 150 nokta(şehir)'den oluşan farklı 5 harita üzerinde algoritmalar denenmiştir. En son ise, DataSet_ch150'yi kullanan diğer literatür ile kıyaslanmıştır.

2. Materyal ve Metot

Çalışmada Java programlama dili kullanılarak GA, AS, ACS ve MMAS algoritmaları kodlanmış, bir arayüz üzerinde birleştirilmiştir. Genetik Algoritma ve Karınca Sistemi algoritmaları günümüzde sıklıkla birçok alan ve optimizasyon problemlerinde kullanılmaktadır. Karınca Kolonisi algoritması bazı operatörleri geliştirilerek Karınca Koloni Sistem ve Max-Min Karınca Sistem olarak çeşitlendirilmiştir.

Genetik Algoritma (Genetic Algorithm-GA), "evrimsel süreç" olarak adlandırılan evrendeki canlıların çoğalma ve yaşam süreçlerinin gözlenmiş veya kabul edilmiş evrelerinin matematiksel bir modelidir. John Holland tarafından 1975'de ortaya atılmıştır. Algoritma, problemin parametrelerini en iyi çözüme ulaştırmak için gen, kromozom, popülasyon, çaprazlama, mutasyon, seçim gibi biyolojik süreçleri ve operatörleri kullanarak Tablo 1'deki kaba kodları yürütür (Holland, 1975; Biroğul, vd. 2007).

Tablo 1. Genetik Algoritma Kaba Kodları

```
{
 Başlangıç popülasyonunu oluştur
 Kriterleri belirle
 do {
        Kromozomları uygunluk fonksiyonuna göre değerlendir
        Kromozomları eşleştirme havuzuna gönder
        Kromozomlara çaprazlama operatörünü uygula
        Kromozomlara değişim operatörünü uygula
        Popülasyondaki tüm kromozomlara tamir operatörünü uygula
        Her bir kromozomun uygunluk değerine göre Rulet tekerleği yöntemini uygula
        Yeni popülasyonu bir önceki popülasyondan oluştur
 } while(!(iterasyon sonu mu) veya !(iyileşme durdu mu))
 En iyi kromozomu sonuç olarak al
}
```

Karınca Sistemi (Ant System-AS/ANT) algoritması, literatürdeki ilk temel Karınca Koloni Optimizasyonu (Ant Colony Optimization-ACO) algoritmasıdır (Dorigo, 1992; Dorigo, vd. 1997; Dorigo, vd. 1991). Karıncaların yuvalarından ayrılarak, yuvalarına dönene kadar yiyecek bulmak için feromon adı verilen hormonu kullanarak gösterdikleri arama davranışlarının matematiksel bir modelidir. Algoritmada karıncaların yuva ile yiyecek arasındaki en kısa yolu bulma yöntemi taklit edilmektedir. Tablo 2'de kaba kodu verilen algoritmanın belirgin özelliği, tur sonunda azalan feromon değerinin tüm karıncalar tarafından her tur sonunda güncellenmesidir.

Tablo 2. Karınca Sistemi Algoritması Kaba Kodları (Dereli, vd. 2010).

```
for her koloni
  for her karınca
        rotayı tamamla
        rotayı sezgisel doldurma algoritmasını kullanarak değerlendir
        feromon güncelle (lokal)
  end
  feromon güncelle (global)
end
```

Karınca Koloni Sistemi (Ant Colony System-ACS) algoritması ile temel Karınca Sistemi algoritması üzerinde ilk önemli gelişme sağlanmış oldu (Dorigo, vd. 1996). Bu algoritmadaki önemli farklılık en iyi karıncanın feromon güncellemesine izin verilmesidir.

Max-Min Karınca Sistemi (Max-Min Ant System-MMAS) algoritması da temel Karınca Sistemi algoritması üzerine kattığı özellikleri ile önemli gelişmeler sağlamıştır (Stützle, vd. 2000). Bu algoritmada da sadece en iyi karınca azalan feromonunu günceller ve feromonun en çok, en az değerleri sınırlıdır.

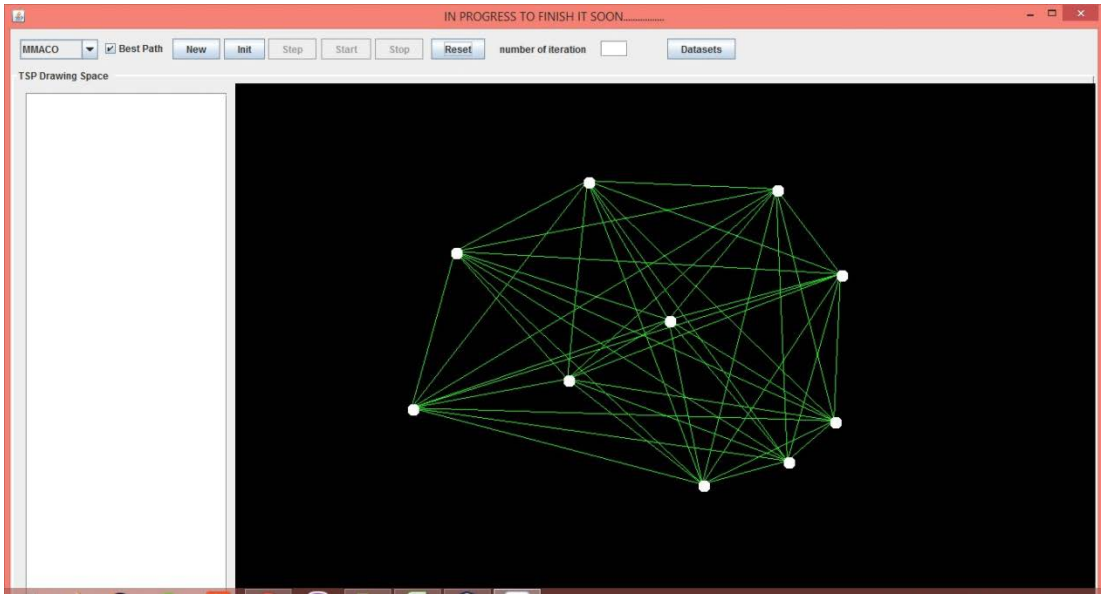
3. Bulgular

Şekil 1'de görülen program arayüzü üzerinde; algoritma seçim listesi, yeni harita oluşturma, haritayı başlangıç durumuna getirme, hesaplamayı başlatma, durdurma, adım adım çalıştırma, tekrar başlatma, hesaplama tekrar sayısı girişi ve DataSet yükleme işlevleri bulunmaktadır. Arayüzün sol tarafındaki alanda ise her adımdaki maliyet değişimi görülebilmektedir.

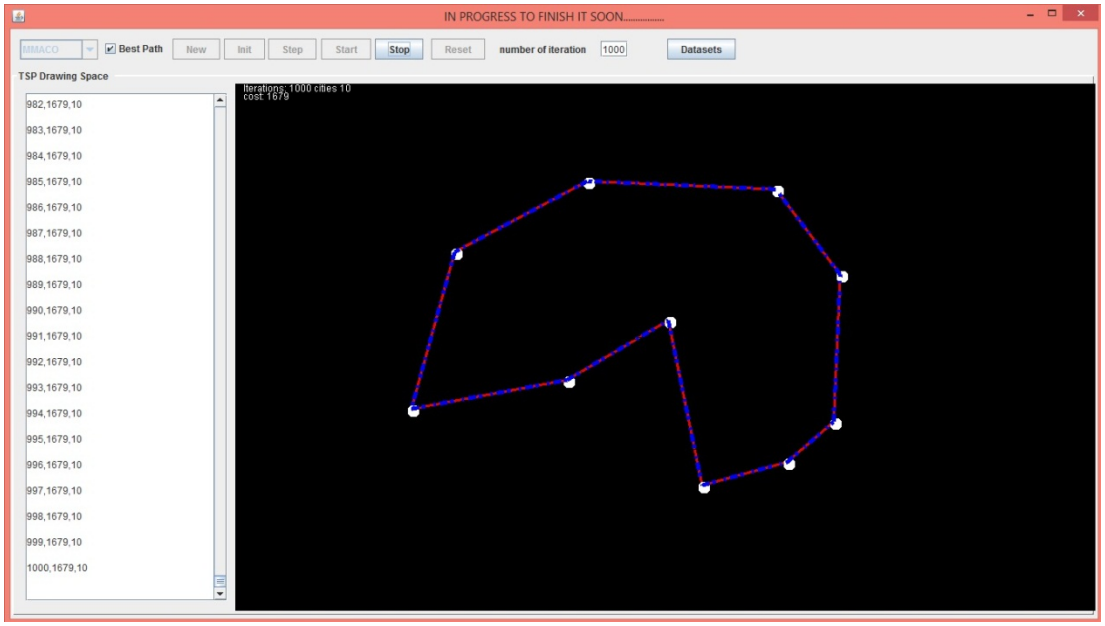
Çalışmada öncelikle kodlanan arayüzün ve algoritmaların geçerlilik ve doğrulaması denenmiştir. Daha sonra rastgele noktalar ve literatür ile kıyaslama amaçlı performans denemeleri yapılarak bulgulara ulaşılmıştır.

A. Geçerlilik/Doğrulama Denemesi

GA, AS, ACS ve MMAS algoritmaları kodlarının geçerliğini (validation) görebilmek için Şekil 1'deki az noktadan oluşan (10 şehirli) bir haritada kodlar çalıştırılmıştır. Algoritmaların tamamı Şekil 2'de görülen aynı yollardan oluşan çözümü, 1679 değerindeki aynı maliyetle hesaplayabilmişlerdir.



Şekil 1. Geçerlik haritasının deneme öncesi görüntüsü

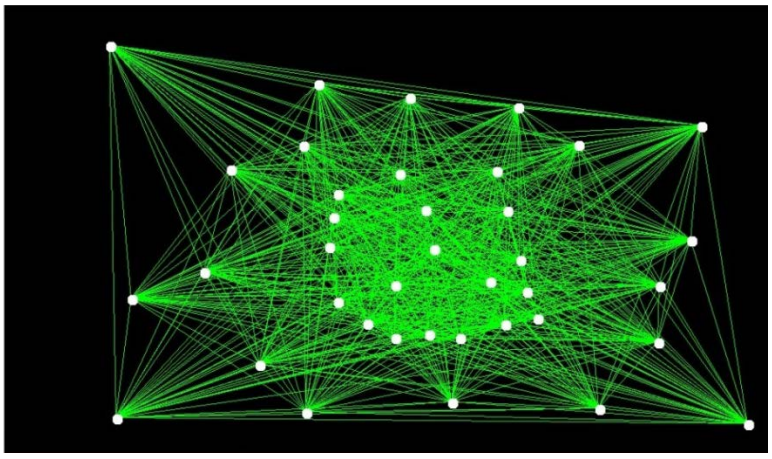


Şekil 2. Geçerlik haritasının deneme sonrası görüntüsü

B. Rastgele Noktalar ile Performans Denemeleri

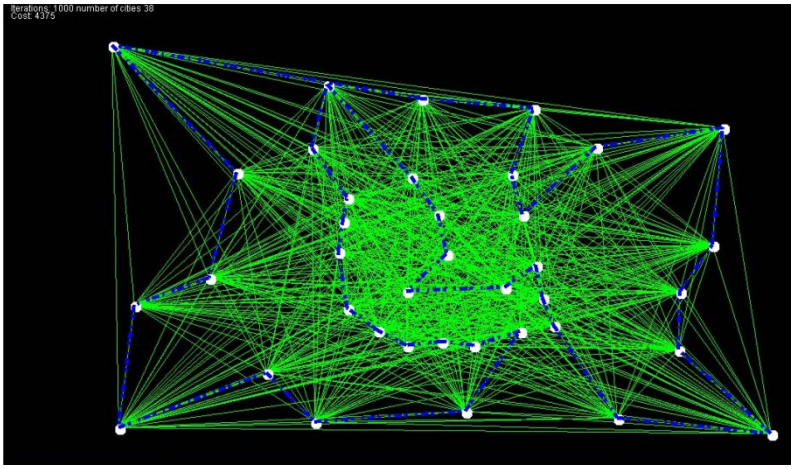
Kodlanan GA, AS, ACS ve MMAS algoritmaları, koordinatları rastgele oluşturulan 36, 56, 76, 101 ve 150 nokta (şehir)'den oluşan 5 harita üzerinde denenerek performansları incelenmiştir.

İlk deneme 36 noktadan oluşan Harita1 ile yapılmıştır. Şekil 3'de rastgele 36 şehirin oluşturulduğu ve aralarındaki tüm yol olasılıklarının çizildiği deneme öncesi görüntü görülmektedir.



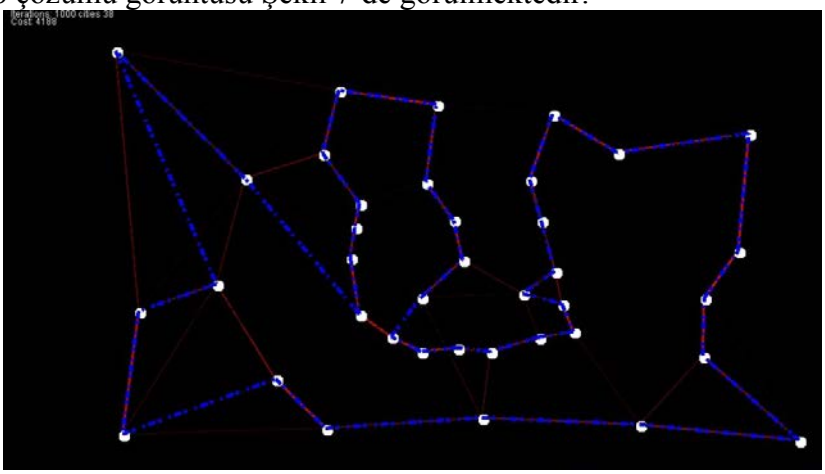
Şekil 3. Harita1'in deneme öncesi görüntüsü

Program arayüzünden GA algoritması seçilerek program çalıştırıldıktan sonra Şekil 4'deki GA çözümü elde edilmiştir. Mavi çizgiler çözümü, yeşil çizgiler olası diğer yolları ifade etmektedir.

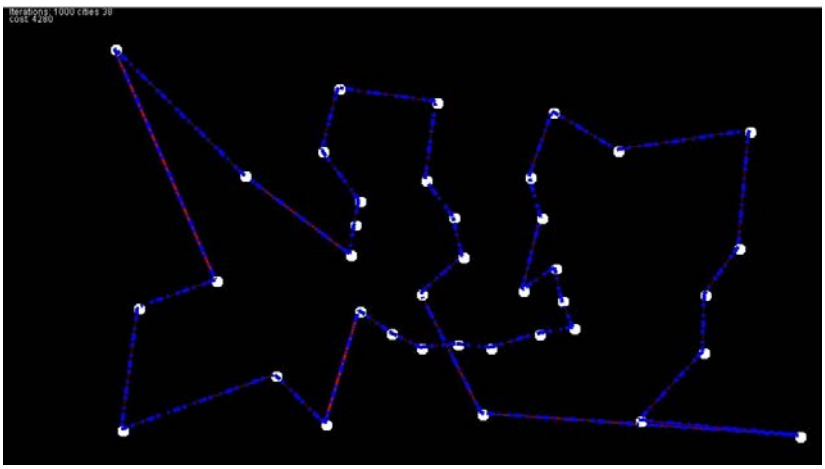


Şekil 4. GA'nın Harita1'e ürettiği çözüm görüntüsü

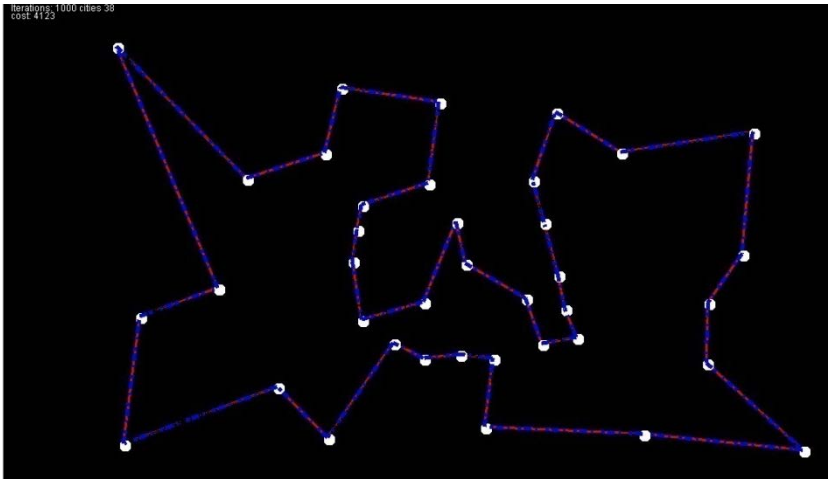
Program arayüzünden sırasıyla AS, ACS ve MMAS algoritmaları seçilerek program çalıştırıldığında elde edilen AS çözümü görüntüsü Şekil 5'de, ACS çözümü görüntüsü Şekil 6'de ve MMAS çözümü görüntüsü Şekil 7'de görülmektedir.



Şekil 5. AS'ın Harita1'e ürettiği çözüm görüntüsü



Şekil 6. ACS'un Harita1'e ürettiği çözüm görüntüsü



Şekil 7. MMAS'un Harita1'e ürettiği çözüm görüntüsü

Tablo 3'de GA, AS, ACS ve MMAS algoritmalarının Harita1 performans detayları görülmektedir.

Tablo 3. GA, AS, ACS ve MMAS algoritmalarının Harita1 için tekrarlama sayısı ve maliyet performansları

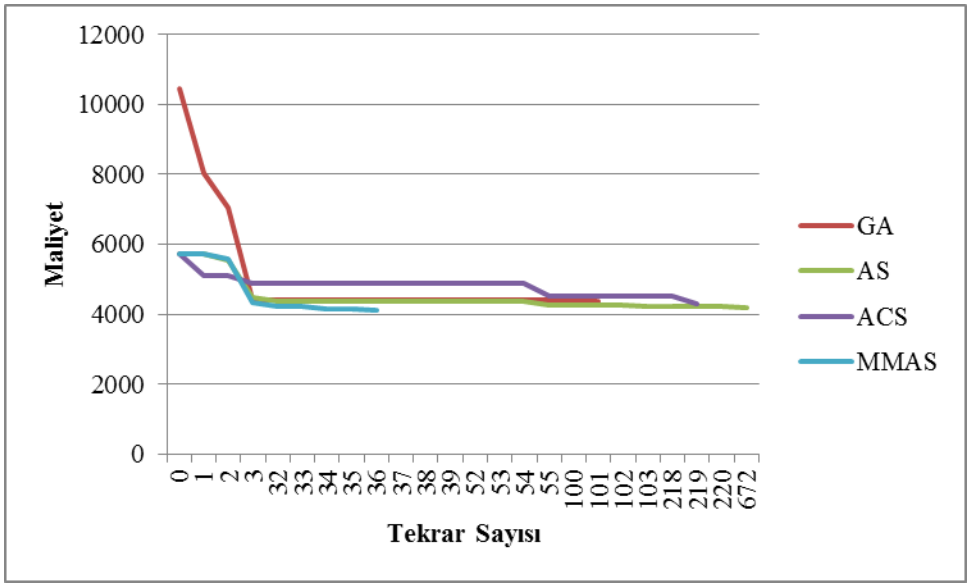
Harita1 performansı detaylı verileri				
#tekrarlama	**GA maliyet**	**AS maliyet**	**ACS maliyet**	**MMAS maliyet**
0	10460	5739	5739	5739
1	8051	5739	5114	5739
2	7055	5560	5114	5589
:	4395	4470	4886	4338
32	4395	4371	4886	4211
33	4395	4371	4886	4211
34	4395	4371	4886	4156
35	4395	4371	4886	4156
36	4395	4371	4886	4123
37	4395	4371	4886	4123
38	4395	4371	4886	4123
:	4395	4371	4886	4123
52	4395	4371	4886	4123
53	4395	4371	4886	4123
54	4395	4371	4886	4123
:	4388	4277	4531	4123
100	4388	4277	4531	4123
101	4375	4277	4531	4123
102	4375	4277	4531	4123
:	4375	4223	4531	4123
218	4375	4223	4531	4123
219	4375	4223	4280	4123
:	4375	4223	4280	4123
672	4375	4188	4280	4123
1000	4375	4188	4280	4123

ortalama maliyet	4398.754	4234.718	4360.243	4140.415
en yüksek maliyet	10460	5739	5739	5739
en az maliyet	4375	4188	4280	4123

GA 10460 maliyet ile çözüme başlarken, AS, ACS ve MMAS algoritmaları 5739 maliyet ile çözüme başlamıştır. GA 101. tekrarda 4375 maliyet ile en iyi çözümünü tamamlarken, AS algoritması 672. tekrarda 4188 maliyet ile, ACS algoritması 219. tekrarda 4280 maliyet ile ve MMAS algoritması 36. tekrarda 4123 maliyet ile en iyi çözümlerini tamamlamışlardır. Harrita1 için maliyet bazında en iyi çözümü 4123 ile MMAS hesaplamıştır. Tablo sonunda da ortalama maliyet, en yüksek maliyet ve en az maliyet bilgileri verilmiştir.

Ortalama maliyet, en yüksek maliyet ve en az maliyet hesaplanırken haritaların her birisi aynı harita üzerinde 10'ar defa çalıştırılarak aritmetik ortalaması alınarak elde edilmiştir.

Şekil 8'de GA, AS, ACS ve MMAS algoritmalarının Tablo 3'de verilen Harita1 için tekrarlama sayısı ve maliyet performansları grafiği görülmektedir.



Şekil 8. GA, AS, ACS ve MMAS algoritmalarının Harita1 için tekrarlama sayısı ve maliyet performansları

Tablo 4'de Harita2, Harita3, Harita4 ve Harita5 için GA, AS, ACS ve MMAS algoritmalarının ortalama maliyet, en yüksek maliyet ve en az maliyet performansları görülmektedir.

Tablo 4. GA, AS, ACS ve MMAS algoritmalarının Harita2, Harita3, Harita4 ve Harita5 için maliyet performansları

performanslar		GA	AS	ACS	MMAS
Harita2	ortalama maliyet	6270.4	5738.7	5819.165	5486.6
	en yüksek maliyet	10583	7101	7101	7101
	en az maliyet	6122	5659	5794	5448
Harita3	ortalama maliyet	7384.9	6526.5	6856.843	6155.8
	en yüksek maliyet	14110	7557	7557	7557
	en az maliyet	7161	6406	6752	6105
Harita4	ortalama maliyet	11440.3	7193.8	7630.288	6932.7
	en yüksek maliyet	18300	7934	7934	7934
	en az maliyet	9852	7131	7484	6873
Harita5	ortalama maliyet	11953.6	10018.9	10407.69	9174.5
	en yüksek maliyet	28776	10615	10615	10615
	en az maliyet	11583	9957	10272	9089

Haritaların nokta (şehir) sayısı arttıkça maliyetleri de artmaktadır. Ancak, haritaların hepsinde en az maliyetli çözüm MMAS algoritması tarafından hesaplanmaktadır.

C. Literatür ile Kıyaslama Denemeleri

Çalışmada incelenen GA, AS, ACS ve MMAS algoritma kodlamalarının, aynı algoritmaların literatürdeki diğer kodlamalar ile performans kıyaslaması da yapılmıştır. Kıyaslama için TSPLIB kütüphanesindeki DataSet_ch150'yi çalışmalarında kullanan Wang, (2014), Puris, vd. (2010) ve literatürde bilinen en iyi maliyet verileri kullanılmıştır (TSP Test Data. 2009; Puris, vd. 2010; Wang, Y. 2014).

Tablo 5. "DATASET ch150" için yapılan literatür kıyaslaması performans sonuçları

DATASET ch150'nin bilinen en iyi maliyeti=6528 (TSP Test Data. 2009; Puris, vd. 2010)	GA	AS	ACS	MMAS
Wang, Y. 2014	11908.5	-	-	-
Puris, vd. 2010	-	7219.8	6908.8	6867.6
Bu çalışmanın sonuçları	10225.5	7118.4	7250.6	6780.7

Tablo 5'de görülen kıyaslama verilerinde en iyi çözümün 6528 olduğu, Wang'ın çalşmasında GA, Puris, vd.'nin çalışmasında AS, ACS ve MMAS algoritmalarının çalışıldığı

görülmektedir. Bu çalışmadaki GA, AS ve MMAS algoritma maliyet performanslarının incelenen literature gore daha iyi olduğu, ancak en iyi çözüme ulaşamadığı görülmektedir.

4. Tartışma ve Sonuç

Bu çalışmada, sezgisel optimizasyon algoritmalarından GA, AS, ACS ve MMAS algoritmalarının performanslarını Gezgin Satıcı Problemi (GSP) üzerinde analiz etmek için bir yazılım geliştirilmiştir. Yazılım ile rastgele sayıda oluşturulan noktalar (şehirler) ilk oluşturulan nokta başlangıç kabul edilecek biçimde GSP algoritmasına göre çözülmek için hesaplanmaya tabii tutulmuştur. Algoritmaların hesapladıkları sonuçlar tekrar sayısı ve yol maliyeti olarak analiz edilmiştir.

Her harita çözümünde en az maliyetli çözümü MMAS, en yüksek maliyetli çözümü GA'nın oluşturduğu görülmektedir. Sıralama azdan yükseğe doğru MMAS, AS, ACS ve GA biçiminde gerçekleşmiştir. Bu çalışmadaki sıralamasının literatürdeki diğer çalışmalara paralellik gösterdiği ayrıca DataSet_ch150 veri setini literatürden daha düşük olan 6780 maliyet ile hesapladığı görülmektedir.

Kaynakça

Albayrak, M., Allahverdi, N. (2011). Development a new mutation operator to solve the Traveling Salesman Problem by aid of Genetic Algorithms, Expert Systems with Applications 38, 1313–1320.

Biroğul S., Güvenç U. (2007). Genetik Algoritma İle Çözümü Gerçekleştirilen Atölye Çizelgeleme Probleminde Ürün Sayısının Etkisi, No. 23, Akademik Bilişim.

Chena, S.M., Chiena, C.Y. (2011). Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques, Expert Systems with Applications, Volume 38, Issue 12, November–December, Pages 14439–14450.

Dereli T., Daş G. S. (2010). Konteyner Yükleme Problemleri İçin Karınca Kolonisi Optimizasyonu Yaklaşımı, Gazi Üniv. Müh. Mim. Fak. Der. Cilt 25, No 4, 881-894.

Dorigo, M., Socha, K., 2013, An Introduction to Ant Colony Optimization, IRIDIA 2013 Technical Report Series ISSN 1781-3794, TR/IRIDIA/2006-01.

Dorigo, M. (1992). Optimization, Learning and Natural Algorithms (in Italian). PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy.

Dorigo, M., Gambardella, L.M. (1997). Ant Colony System: A cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation, 1(1):53–66.

Dorigo,M., Maniezzo, V. and Colorni, A. (1991). Positive feedback as a search strategy, Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy.

Dorigo, M., Maniezzo V. and Colorni, A. (1996). Ant System: Optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics – Part B, 26(1):29–41.

Erdem, Y., Keskintürk, T. (2011). Kanguru Algoritması Ve Gezgin Satıcı Problemine Uygulanması, İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi Yıl:10 Sayı 19 Bahar s.51-63.

Holland, J. H. (1975). Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. U Michigan Press.

Karaboga, D., Gorkemli, B. (2011). A Combinatorial Artificial Bee Colony Algorithm for Traveling Salesman Problem, Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on, 15-18 June, 50 – 53, 978-1-61284-919-5.

Kuşcu, Ö., Küçüksille, E.U. (2011). Heuristic Methods in Vehicle Routing Systems, Elektronika ir Elektrotechnika, January, No.1, 65-70.

La Maire, B.F.J., Mladenov, V.M. (2012). Comparison of Neural Networks for Solving the Travelling Salesman Problem, Neural Network Applications in Electrical Engineering (NEUREL), 2012 11th Symposium on, 20-22 Sept., 21 – 24, 978-1-4673-1569-2.

Puris, A., Bello, R. & Herrea, F. (2010). Analysis of the efficacy of a Two-Stage methodology for ant colony optimization: Case of study with TSP and QAP, Expert Systems with Applications 37, 5443–5453.

Saiyed, A. R. (2012). The Traveling Salesman Problem, 1-15.

Stützle, T. and Hoos, H.H. (2000). MAX–MIN Ant System, Future Generation Computer Systems, 16(8):889–914.

TSP Test Data. (2009). http://www.math.uwaterloo.ca/tsp/data/ (Erişim: 01.08.2016)

Wang, Y. (2014). The Hybrid Genetic Algorithm with two Local Optimization Strategies for Traveling Salesman Problem, Computers & Industrial Engineering, Volume 70, April 2014, Pages 124–133.