2-OPT ALGORİTMASI VE BAŞLANGIÇ ÇÖZÜMÜNÜN ALGORİTMA SONUÇLARI ÜZERİNDEKİ ETKİSİ

Timur KESKİNTÜRK[1*], Barış KİREMİTCİ[2], Serap KİREMİTCİ[2]

[1]İstanbul Üniversitesi, İşletme Fakültesi, Sayısal Yöntemler Anabilim Dalı, Avcılar, İstanbul
tkturk2010@gmail.com

[2]İstanbul Üniversitesi, Ulaştırma ve Lojistik Fakültesi, Avcılar, İstanbul
baris@istanbul.edu.tr, serapy@istanbul.edu.tr

Geliş Tarihi: 17.05.2015; Kabul Ediliş Tarihi: 09.06.2016

ÖZ

Bu çalışmada, gezgin satıcı problemlerinin çözümü için Croes (1958) tarafından önerilen 2-opt algoritması tanıtılmış ve tur oluşturan sezgisellerle üretilen başlangıç çözümlerinin, algoritmanın performansı üzerindeki etkileri incelenmiştir. Yerel arama mantığı ile çalışan algoritma, turdaki iki kenarın turdan çıkarılması ve kalan kısımların farklı şekilde birbirlerine bağlanması şeklinde iyileştirmeler yapmaktadır. Tüm mümkün değişikliklerin yapılmasından sonra elde edilen çözüm, 2-optimal olarak adlandırılmaktadır.

Başlangıç çözümü tesadüfi olarak üretilen bir tur olabileceği gibi, farklı sezgisellerin ürettiği turlar da kullanılabilir. Çalışmada ayrıca, hangi başlangıç çözümünün 2-opt algoritmasının sonuçlarını iyileştirdiği de araştırılmıştır. Algoritma, farklı problemler üzerinde denenmiş, elde edilen sonuçlar da karşılaştırmalı olarak raporlanmıştır. Türkçe literatürde daha önce ele alınmamış olan algoritmanın detaylı anlatımıyla, anadili Türkçe olan araştırmacılar için bir boşluğu doldurması umulmuştur.

Anahtar Kelimeler: 2-opt algoritması, gezgin satıcı problemi, sezgisel, yerel arama

2-OPT ALGORITHM AND EFFECTS OF INITIAL SOLUTION ON RESULTS

ABSTRACT

In this study the 2-opt heuristic algorithm which was proposed by Croes (1958) for the travelling salesman problem is presented and the effect of the initial solutions produced by constructive heuristics on the performance of the algorithm is analyzed. The algorithm is based on a local search that works with the logic of removing two edges from the current solution and merging the two parts in a different way. After trying all possible changes the result is considered a 2-optimal solution.

The initial tour could be either a randomly created tour or a result of any other heuristics. In this work the best initial tour by the heuristics has also been studied. The algorithm has been tested on different benchmark problems and these results have been analyzed. Since this algorithm has not yet been studied in the Turkish literature this study will fill a gap on this topic.

Keywords: 2-opt algorithm, travelling salesman problem, heuristic, local search

* İletişim yazarı

1. GİRİŞ

Gezgin satıcı problemi (GSP) literatürde geniş olarak ele alınmış ve çözümü için birçok kesin, sezgisel ve metasezgisel yöntem geliştirilmiş, NP-zor sınıfına ait bir problemdir. Genel olarak tanımlamak gerekirse, birden çok noktaya, her birine bir kere uğrayacak şekilde ve başlangıç noktasına dönmek kaydıyla en kısa turun bulunmasıdır denilebilir.

Günlük hayatta çokça karşılaşılmakta olan problem, az nokta sayısı içerdiği durumda, kolayca çözülebilir ve hatta sezgisel olarak da en uygun rota belirlenebilir. Ancak nokta sayısı 4-5'i geçmeye başladığında problemin zorluk derecesi de üssel olarak artmaktadır. n nokta sayısı olmak üzere, simetrik bir gezgin satıcı probleminin alternatif çözüm sayısı $(n-1)!/2$ olmaktadır. Dağıtım, toplama, müşteri ziyaretleri, siyasi liderlerin miting gezileri gibi birçok alanda kullanılabilen problemin, zaman içerisinde yüzleştiği nokta sayıları da artmıştır. Böyle olunca, problemin çözümü için birçok yöntem geliştirilmiş ve yöntemler değişik modifikasyonlarla iyileştirilmeye çalışılmıştır. Tur geliştiren sezgiseller sınıfındaki 2-opt da bu alanda geliştirilmiş algoritmalardan biri olup, hızlı ve kabul edilebilir sonuçlar verdiğinden üzerinde çok çalışılmış bir algoritmadır. İki kenarın turdan silinip kalan parçaların farklı şekilde birleştirilmesi mantığı ile çalışan algoritmanın 3-opt, 4-opt, 5-opt ve her iterasyonda değişiklik yapılacak kenar sayısının bir mantık çerçevesinde belirlendiği k-opt (Lin-Kernighan Algoritması) versiyonları geliştirilmiştir.

Bu çalışmada, 2-opt algoritması detaylı olarak anlatılmış, farklı başlangıç çözümleri için farklı sonuçlar ürettiğinden başlangıç turunun sonuç üzerindeki etkisi incelenmiştir. Farklı sezgisellerin bulduğu çözümler başlangıç çözümü olarak belirlenmiş ve farklı versiyonlar literatür problemleri üzerinde denenerek karşılaştırma yapılmıştır. Çalışmanın ikinci bölümünde, gezgin satıcı problemi tanıtılmıştır. Üçüncü bölümde, 2-opt algoritması ayrıntılı olarak incelenmiş ve algoritmanın çalışma mantığı tanıtılmıştır. Bir sonraki bölümde, başlangıç çözümünün sonuç üzerindeki etkisi test problemleri ile analiz edilmiştir. Beşinci bölüm, başlangıç turunu oluşturabilecek sezgisellerin tanıtımına ayrılmıştır. Uygulama bölümünde, bu farklı sezgisellerin ürettiği başlangıç çözümlerinin test problemleri sonucuna etkisi karşılaştırmalı olarak ele alınmıştır. Son bölümde ise sonuçlar tartışılmış ve konu ile ilgili yapılabilecek diğer çalışmalardan söz edilmiştir.

2. GEZGİN SATICI PROBLEMİ

GSP (Gutin vd, 2002), 1800'lü yıllarda ünlü matematikçi W. R. Hamilton ve Thomas Kirkman tarafından Çizge Teorisi kitabında ilk olarak ele alınmıştır (Biggs vd., 1976). Karl Menger, GSP üzerine çalışma yapan ilk matematikçi olup, 1930 yılında Viyana'da gerçekleştirilen bir kolokyumda problem üzerine konuşmuştur (Fortini, 2007). Problemden ilk kez gezgin satıcı problemi adıyla H. Whitney ve M. Flood tarafından 1934 yılında Princeton Üniversitesi'nde gerçekleştirilen bir seminerde bahsedilmiştir (Flood, 1956). G. Dantzig, Fulkerson ve Johnson (1954) yaptıkları çalışmada, GSP'yi tam sayılı programlama yaklaşımı ile ifade etmişlerdir. Karp (1972), GSP'yi NP-zor sınıfına ait problem olarak tanımlamıştır. Literatürde yer alan sezgiseller incelendiğinde, genellikle tercih edilen 2-opt algoritması, Croes (1958) tarafından geliştirilmiştir. Verhoeven vd. (1995), 2-opt algoritmasından daha hızlı çözüm elde eden paralel 2-opt algoritmasını geliştirmişlerdir. Lin (1965), 2-opt algoritmasından daha iyi sonuç veren 3-opt algoritmasıyla literatürde yer almıştır. Nilsson (2003), 2-opt, 3-opt ve k-opt algoritmalarını karşılaştırmalı olarak incelemiştir. GSP'nin çözümünde metasezgiseller alternatif çözüm yöntemleri olarak kullanılmaktadır. Kuzu vd. (2014), literatürde geçen metasezgisellerin GSP'ye uygulanmasını incelemiş ve kullanılan metasezgiselleri tanıtmıştır.

$G = (V, A)$ olmak üzere $V = \{v_1, ..., v_n\}$ düğüm noktaları kümesi ve $A = \{(v_i, v_j) \mid v_i, v_j \in V, i \neq j\}$ kenarlar kümesidir. A kenarlar kümesi ile ilişkili olarak $C = (c_{ij})$ maliyetler matrisidir. Maliyetler, mesafe olabileceği gibi, süre veya yakıt da olabilir. Her $(v_i, v_j) \in A$ $c_{ij} = c_{ji}$ için ise problem simetrik adını almaktadır. $c_{ij} \neq c_{ji}$ durumunda ise problemler asimetri özelliği taşır.

i, j şehir indisi, *n* problemde bulunan şehir sayısı, X_{ij} *i.* şehirden *j.* şehire gidilmesi durumunda 1, diğer durumlarda 0 değerini alan ikili bir değişken olmak üzere, GSP'nin tam sayılı doğrusal programlama modeli aşağıdaki gibidir:

$$\text{Min} \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} c_{ij} \, X_{ij} \qquad (1)$$

$$\sum_{i=1, i \neq j}^{n} X_{ij} = 1 \qquad j = 1, \ldots, n \qquad (2)$$

$$\sum_{j=1, j \neq i}^{n} X_{ij} = 1 \qquad i = 1, \ldots, n \qquad (3)$$

$$\sum_{i \in S, j \in S} x_{ij} \leq |S| - 1 \qquad (S \subseteq V, |S| > 1) \qquad (4)$$

$$X_{ij} \in \{0,1\} \qquad (5)$$

Amaç fonksiyonu (1), probleme ait turdaki uzaklıklar toplamını minimize eder. Bir şehire yalnızca tek bir şehirden gelinebileceği Denklem (2) ve bir şehirden yalnızca bir tek şehire gidilebileceği Denklem (3) tarafından sağlanmaktadır. Denklem (4) alt tur engelleme kısıtıdır.

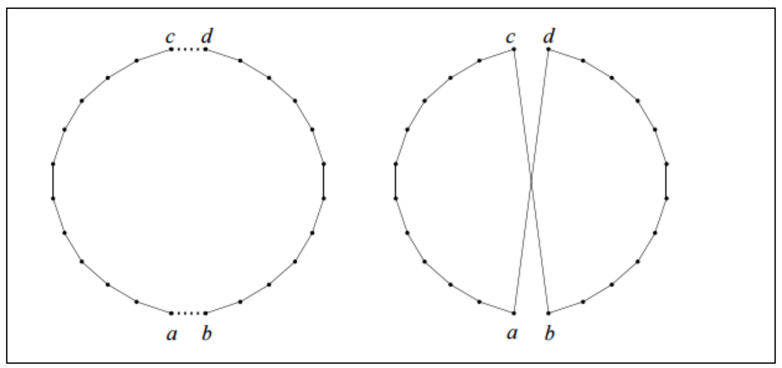
3. 2-OPT ALGORİTMASI

2-opt algoritması, GSP problemlerinin çözümüne yönelik, muhtemelen en temel ve çok geniş kullanım alanına sahip bir algoritmadır. İlk olarak Croes (1958) tarafından 1958 yılında geliştirilmiştir. Temel olarak, tur içerisindeki iki kenarın silinmesi ve iki parçaya ayrılan turun maliyetleri düşürecek şekilde, farklı olarak bağlanması şeklinde tanımlanabilir (Gutin ve Punnen, 2002). Bu işlem, ikili değişikliklerin artık çözümü iyileştiremediği noktaya kadar devam ettirilir. Şekil 1, her adımda yapılan değişikliği temsil etmektedir:

Burada yapılan değişiklik aslında, seçilen iki nokta arasındaki alt turun ters çevrilmesinden ibarettir. Şöyle ki turun a-b yönünde olduğunu ve a'dan başladığını düşündüğümüzde, turun aşağıdaki gibi olduğunu görebiliriz:

a b ------- d c ----------

2-opt ile a-b ve d-c kenarları turdan çıkarıldığında ve kalan iki parça farklı bir yolla birbirine bağlandığında, turun yeni rotası aşağıdaki gibi olmaktadır:



Şekil 1. 2-opt Algoritmasında Yapılan Rota Değişikliği (Johnson vd., 1997).

a d ------- b c ----------

Kırmızı ile gösterilen d-b aralığının, d ve b de dâhil olacak şekilde ters çevrildiği görülmektedir. Buradan hareketle aslında yöntemin, tüm mümkün ikililere ait yolları ters çevirerek daha iyi turlar elde etmeyi amaçladığı görülecektir. Burada dikkat edilmesi gereken nokta, sırası ile denemeler yaparken turu geliştiren bir değişiklik olduğunda ilgili denemeler tamamlanınca tekrar baştan başlanmasıdır. Örneğin 5 noktadan oluşan bir GSP probleminde yapılabilecek muhtemel değişiklikler aşağıdaki gibidir:

1-2, 1-3, 1-4, 1-5, 2-3, 2-4, 2-5, 3-4, 3-5, 4-5

Algoritma sırası ile ilgili değişiklikleri yaptığında, 2-3 aralığı ters çevrildiğinde toplam tur uzunluğunun kısaldığını farz edelim. Bu durumda algoritma, tüm diğer denemeleri tamamlayıp en uygun değişikliği kaydettikten sonra başa dönerek, tekrar 1-2'den başlayarak turu geliştirmeye çalışacaktır. Burada uygulanacak problemlerin simetrik olduğu bilindiğinden, çözüm süresini kısaltmak için ikililerin tersi ayrıca denenmemiştir. Bu yaklaşımla, *T* mevcut tur, *f* (*T*) mevcut turun uzunluğu, *T′* yeni bulunan tur, *f* (*T′*) yeni turun uzunluğu, *z* değişim ve *zmin* mevcut aramadaki en iyi değişim olmak üzere algoritmanın işleyiş adımları aşağıdaki şekilde olacaktır:

1. Adım: Başlangıç çözümünü ve tur uzunluğunu belirle: *T*, *f* (T)

2. Adım: *i* - *j* değerlerini sıfırla, *T″* = *T zmin* = 0, z = 0

3. Adım: *i* - *j* ikilisini belirle (*i* ≠ *j* ve *i* < *j*)

4. Adım: Tur içerisinde *i* - *j* aralığını ters çevir:*T′*

5. Adım: Turun yeni uzunluğunu belirle: *f (T′)*

6. Adım: *f*(*T′*) < *f*(*T*) ise *z*= *f*(*T′*) - *f*(*T*),

7. Adım: *z*<*zmin* ise *zmin* = *z*, *T″* =*T′*, denenmemiş *i-j* varsa 3. Adım'a git

8. Adım: *T*=*T″*; *zmin*<0 ise 2. Adım'a git, yoksa sonlandır

Aslında, yapılan bir değişiklikte toplam tur uzunluğunu hesaplamak yerine sadece değişiklik yapılan, tura ait kenar değişikliklerinden yola çıkarak da tur uzunluğunun geliştirilip geliştirilemediği anlaşılabilir. Tekrar yukarıdaki örneğe dönecek olursak, c ile a noktaları arası sabit kalmış ve b ile d arası da ters çevrilmiştir. Bu aralıklarda bir mesafe değişimi söz konusu değildir.

a b ------- d c ----------

a d ------- b c ----------

Mesafenin değiştiği (arttığı veya azaldığı) kenarlar a-b ve d-c ile a-d ve b-c kenarlarının değişmesi sonucu ortaya çıkmaktadır. Şöyle ki mevcut turdaki (*T*) a-b ve d-c uzunlukları yerine, a-d ve b-c uzunlukları eklenmiştir. Bu durumda *f* (*T*) ve *f* (*T′*) arasındaki fark aşağıdaki gibi hesaplanacaktır:

f (*T′*) - *f* (*T*)=*f* (*a, d*) + *f* (*b, c*) - *f* (*a, b*) - *f* (*d, c*)

f (*T*), *T* turuna ait toplam mesafe ve *f* (*a*, *b*), *a* - *b* kenarına ait uzaklıktır. Farkın bu şekilde hesaplanmasının avantajı, her değişiklik sonrasında yeni tura ait toplam uzunluğun tekrar belirlenmesine gerek olmamasıdır. Özellikle nokta sayısı (*n*) büyüdüğünde bu işlem çokça zaman alacaktır. Bu durumda, sadece farkın hesaplanması ile iyileşmelerin takip edildiği versiyona ait işlem adımları, *z* hesaplanan fark ve *zmin* mevcut en iyi iyileştirme olmak üzere aşağıdaki gibi olacaktır:

1. Adım: Başlangıç çözümünü ve tur uzunluğunu belirle: *T*, *f* (*T*)

2. Adım:(*a*, *b*) ve (*d, c*) değerlerini sıfırla, *z* = 0, *zmin* = 0

3. Adım: (*a*, *b*) ve (*d*, *c*) kenarlarını belirle (*a* ≠ *b* ≠ *d* ≠ *c* ve denenen (*a*, *b*) ve (*d*, *c*) ikililerinden farklı)

4. Adım: *z* = (*f* (*a*, *d*) + *f* (*b*, *c*)) - (*f* (*a*, *b*) + *f* (*d*, *c*))

5. Adım: *z* < *zmin* ise *zmin* = *z*, *bmin* = *b*, *dmin* = *d*, denenmemiş (*a*, *b*) ve (*d*, *c*) kombinasyonu varsa 3. Adım'a git

6. Adım: *zmin* < 0 ise *T* içerisinde *bmin-dmin* aralığını ters çevir, *f* (*T*)'yi hesapla ve 2. Adım'a git, yoksa sonlandır

Görüldüğü üzere, her iki mantıkla da algoritmaya ait adımlar bir programlama dili ile az satır sayısı ile yazılabilecek şekildedir. Ek 1'de birinci tip algoritmaya ait yazılmış olan Matlab kodları bulunmaktadır. Algoritmada Matlab fonksiyonları da kullanıldığından 25 satırlık kodla algoritma çalıştırılabilmektedir.

4. 2-OPT ALGORİTMASINDA BAŞLANGIÇ ÇÖZÜMÜNÜN SONUCA ETKİSİ

2-opt algoritması tur geliştiren sezgiseller sınıfında yer almaktadır. İteratif olarak mevcut çözümü iyileştirmeye çalışmaktadır. Bu tip sezgisellerin başlangıç çözümü genellikle tesadüfi olarak belirlenmektedir. Bir diğer yol ise diğer bazı tur oluşturan sezgisellerin sonucunu başlangıç çözümü olarak kullanmaktır. Bu bölümde, başlangıç çözümünün tesadüfi olarak belirlenmesinin sonuçlar üzerindeki etkisi incelenecektir.

Analizin yapılabilmesi için kullanılan problemlerden 8 tanesi TSPLIB'den alınmıştır (Reinelt, 1995). Ayrıca, Türkiye'nin **şehirlerarası mesafeleri kullanılarak oluşturulan GSP problemine de yer verilmiştir.** Türkiye problemine ait uzaklıklar ise Karayolları Genel Müdürlüğü'ne ait internet sitesinden alınmıştır (Karayolları Genel Müdürlüğü, 2014). Problemlere ait bilgiler Tablo 1'de verilmiştir.

Problemler, Matlab'de yazılan 2-opt algoritmasında, tesadüfi başlangıç çözümü ile 100'er kere çalıştırılmış ve sonuçlar ortalama, en küçük, en büyük değer, standart sapma, optimumdan ortalama sapma ve optimumdan yüzde ortalama sapma olarak Tablo 2'de verilmiştir.

Tablo 2'deki sonuçlar incelendiğinde, 3 problemde optimum değerlere ulaşıldığı görülmektedir. Optimumdan ortalama yüzde sapmalara bakıldığında ise iki problem hariç yaklaşık %6 civarındadır. Bir sonraki bölümde, tur oluşturan sezgisellerin sonuçlarının 2-opt algoritmasında başlangıç çözümü olarak kullanılmasıyla elde edilecek sonuçlar bu değerlerle karşılaştırılacaktır. Beşinci problemde (brazil58) sonuçlar optimuma oldukça yakın bir seyir izlemiştir. Sonuçların daha iyi anlaşılabilmesi için 100 çalıştırmaya ait bulunan değerler kutu bıyık diyagramı olarak, problem sırasına göre Şekil 2'de verilmiştir.

Kutu bıyık diyagramları incelendiğinde, 100 çalıştırmaya ait sonuçların genellikle ortalama etrafında

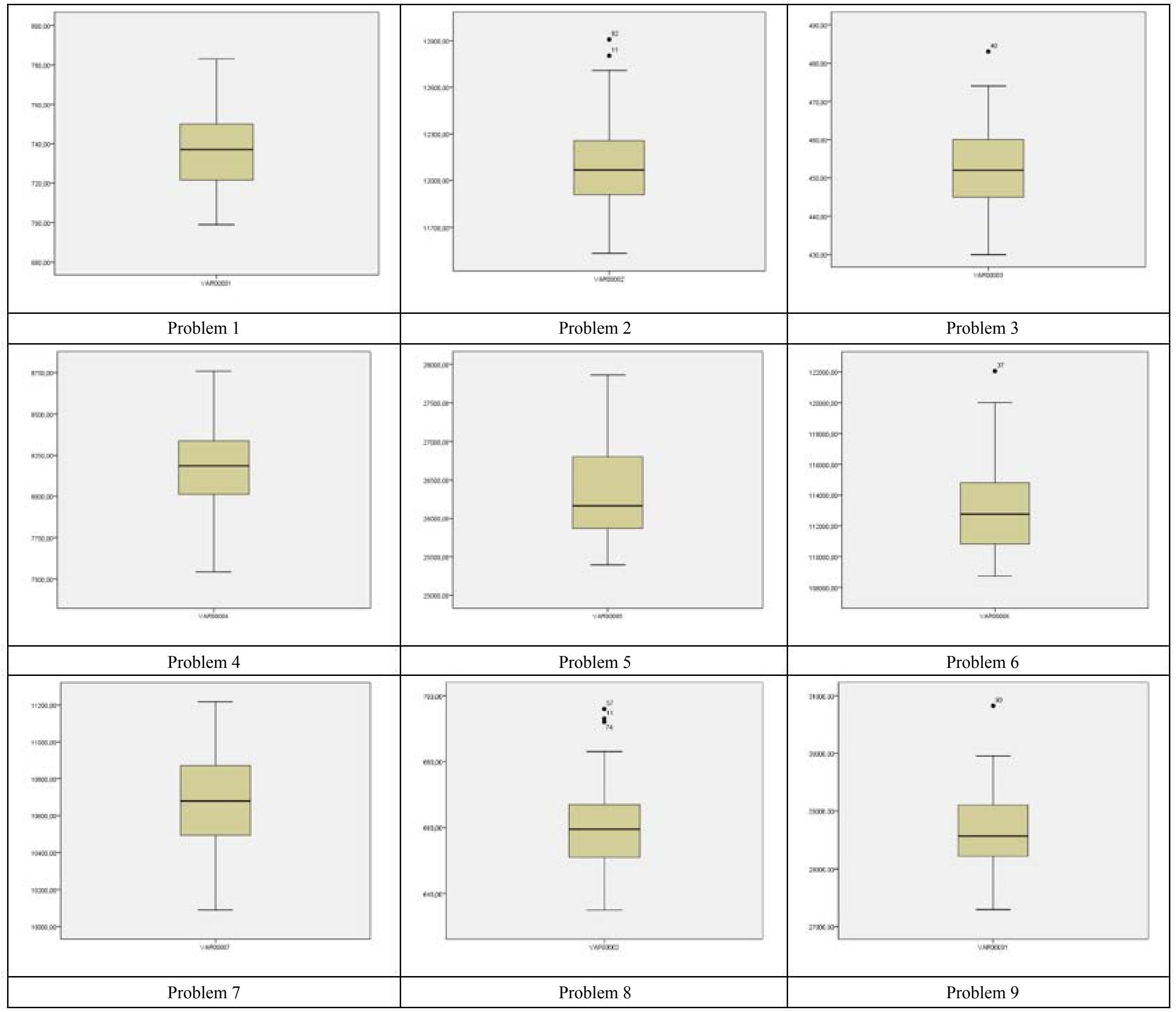
Tablo 1. Analizde Kullanılan GSP Problemleri

Problem	n	Optimum
dantzig 42	42	699
hk48	48	11.461
eil51	51	426
berlin52	52	7.542
brazil58	58	25.395
pr76	76	108.159
Türkiye81	81	9.920*
eil101	101	629
kroA150	150	26.130

* Önder ve arkadaşlarının (2013) çalışmasından alınan değerin optimum olduğu belirtilmemiştir.

Tablo 2. Tesadüfi Başlangıç Çözümlü 2-opt Sonuçları

No	Problem Adı	$\bar{x}$	min	max	σ	$\bar{\Delta}$	$\%\bar{\Delta}$
1	dantzig 42	736,69	699	783	20,25	37,69	5,39
2	hk48	12.094,29	11.532	12.909	269,83	633,29	5,53
3	eil51	452,28	430	483	9,97	26,28	6,17
4	berlin52	8.164,69	7542	8759	251,85	622,69	8,26
5	brazil58	26.311,78	25.395	27.862	587,03	916,78	0,13
6	pr76	113.187,84	108.757	122.067	2.809,51	5028,84	4,65
7	Türkiye81	10.687,41	10.089	11.217	253,86	767,41	7,74
8	eil101	659,62	635	696	12,31	30,62	4,87
9	kroA150	28605,99	27290	30826	616.89	2475.99	9,48



Şekil 2. Tesadüfi Başlangıç Çözümlü 2-opt Sonuçları – Kutu Bıyık Diyagramı

dağıldığı görülmektedir. Bu kanaate Tablo 1'deki standart sapmalara bakarak da varılabilir. Sadece Problem 2, 3, 6, 8 ve 9'da aykırı değerler bulunmuştur. Problem 5'te, ortalamanın üzerinde yer alan sonuçlar ağırlıkta görünmektedir ki bu da mesafelerin bu problemde daha yüksek rakamlardan oluşmasından kaynaklanmaktadır. Bu probleme ait optimumdan sapmaların ortalaması da %0,13 gibi çok küçük bir değer olarak bulunmuştur. Ayrıca yöntemin tesadüfi başlangıç çözümüyle, ilgili problemlere bulduğu optimumdan ortalama sapma değerlerinin ortalaması ise %5,2'dir.

5. 2-OPT ALGORİTMASINDA BAŞLANGIÇ ÇÖZÜMÜNÜN BELİRLENMESİNDE KULLANILACAK TUR OLUŞTURAN SEZGİSELLER

Tur oluşturan sezgisellerin amacı, bir başlangıç noktasından başlayarak probleme ait tüm noktalar tura dahil olana kadar her seferinde alt tura yeni bir nokta ekleyerek turu tamamlamaktır. Bu sezgiseller tur tamamlandığında çözüm aramayı durdururlar. Çalışmanın bu bölümünde, uygulanabilirliği açısından sıklıkla tercih edilen, tur oluşturan sezgisellere yer verilmiştir.

5.1 En Yakın Komşu Yöntemi (EYKY):

En Yakın Komşu Yöntemi, tur oluşturan sezgiseller arasında uygulanması en kolay sezgiseldir. GSP'de rastgele seçilen başlangıç düğümünden başlayarak daha önce ziyaret edilmemiş en yakın noktayı ziyaret etme prensibine dayanır. Tüm noktalar ziyaret edildiğinde tur tamamlanmış olur. Gutin ve Punnen (2002), EYKY'nin en iyi sonucu garanti etmemekle birlikte, çok hızlı sonuç vermesi açısından tercih edilebilir olduğunu belirtmişlerdir. Her bir noktadan başlayarak çözümler üretip, sonrasında bunlar içerisinden en iyi çözümü seçmek de bir yoldur. Bu durumda algoritma, Tekrarlı En Yakın Komşu Yöntemi (TEYKY) olarak adlandırılmaktadır.

Ekleme Sezgiselleri, seçilen başlangıç noktasına yeni noktalar eklenerek, başka bir ifadeyle, GSP'de ziyaret edilmeyen tüm noktaların alt tura eklenmesiyle tur oluşturan sezgisellerdir. Bu çalışmada, En Yakın Ekleme Yöntemi (EYEY), En Ucuz Ekleme Yöntemi (EUEY) ve En Uzak Ekleme Yöntemi'ne (EUzEY) yer verilmiştir.

5.2 En Yakın Ekleme Yöntemi (EYEY):

Bu yöntemde rastgele seçilen i başlangıç noktasına en yakın j noktasının seçilerek tura eklenmesi ile başlanmaktadır Daha sonra, oluşan alt turdaki noktalara en yakın olan bir k noktasının, belli bir maliyet hesabına göre belirlenen turdaki bir i, j aralığına eklenmesi ile çözüme devam edilir. Ekleme yapılırken, hangi i, j aralığına eklemek maliyet artışını minimum yapıyorsa yeni nokta belirlenen aralığa eklenmektedir. i ve j tura dahil olmayan bir k noktasının eklenebileceği kenarlara ait noktaları temsil etmek üzere, ekleme maliyeti (Δf) şu şekilde hesaplanmaktadır: $\Delta f = c_{ik} + c_{kj} - c_{ij}$. Tur tamamlanana kadar (başlangıç noktasına dönülene kadar) bu işlemlere devam edilir. Algoritmaya ait kodlar Jevtic'in (2014) kodlarından modifiye edilmiştir.

5.3 En Ucuz Ekleme Yöntemi (EUEY):

Rastgele seçilen i başlangıç noktasına en yakın j noktasının seçilerek eklenmesiyle bir alt tur oluşturulur. Turda olmayan k noktalarının alt turdaki tüm i - j aralıklarına eklenmesi durumundaki maliyetler hesaplanır. Bu maliyetlerden minimum olan k en uygun i ve j aralığına eklenir. Tur tamamlanana kadar (başlangıç noktasına dönülene kadar) bu işlemlere devam edilir. EUEY ile EYEY'nin temel farkı alt tura eklenecek olan k noktasının seçimidir. EYEY yönteminde, adından da anlaşılacağı üzere, alt tura en yakın nokta aday nokta olarak seçilmekte, tüm k noktaları için maliyet artışı hesaplanıp en uygun nokta en uygun aralığa eklenmektedir.

5.4 En Uzak Ekleme Yöntemi (EUzEY):

EUzEY yöntemi turda olmayan noktalardan tura eklenecek k noktasının belirlenmesi dışında, EUEY ve EYEY yöntemleri ile aynı prensipte çalışmaktadır. k noktası belirlenirken, mevcut alt tura en uzak olan nokta seçilmektedir. Maliyet artışını minimum yapan, alt turdaki i-j aralığına k noktası eklenir. Tur tamamlanana kadar (başlangıç noktasına dönülene kadar) bu işlemlere devam edilir.

5.5 Clarke ve Wright Yöntemi (CWY):

Tasarruf algoritması olarak da adlandırılan yöntemde, k başlangıç şehri olmak üzere tüm i-j ikilileri için tasarruflar hesaplanır. Tasarruflar, $s_{ij} = c_{ki} + c_{kj} - c_{ij}$ şeklinde hesaplanmaktadır. Daha sonra ikililer tasarruflarına göre büyükten küçüğe doğru sıralanmaktadır. En büyük tasarrufa sahip ikiliden başlanmak üzere, kenarlar alt tur oluşturmayacak ve aynı düğümden ikiden fazla geçiş olmayacak biçimde eklenir. Bu şekilde tur tamamlanarak nihai çözüme ulaşılır. Her noktanın ayrı ayrı başlangıç noktası belirlenerek kullanılması ile yöntem, Tekrarlı Clarke ve Wright Yöntemi (TCvWY) adını almaktadır.

6. UYGULAMA

Bu bölümde, bir önceki bölümde tanıtılan farklı tur oluşturan sezgisellerin ürettiği başlangıç çözümlerinin test problemlerinin sonucuna etkisi karşılaştırmalı olarak ele alınmıştır. İlk olarak, tur oluşturan sezgisellerin ilgili problemler için ürettikleri sonuçlar elde edilmiş ve Tablo 3'te tur uzunlukları, Tablo 4'te ise optimumdan yüzde sapma değerleri olarak raporlanmıştır.

Sonuçlar incelendiğinde, tesadüfi başlangıç çözümlerine göre optimuma çok daha yakın bir noktadan arama sürecinin başlayacağı anlaşılmaktadır. Bu aşamada, tur oluşturan sezgisellerin oluşturduğu turlar, 2-opt algoritmasının girdileri olarak kullanılmıştır. Tüm problemlerin çözümlerine ait algoritma, çalışma süreleri değişkenlik göstermekle birlikte, 10 saniyenin altındadır. Bu durum, tur oluşturan sezgisellerin eklendiği 2-opt algoritmaları için de geçerlidir. Dolayısıyla, süre açısından algoritmaları birlikte kullanmanın bir dezavantajı bulunmamaktadır. Elde edilen sonuçlar Tablo 5'te verilmiştir.

Öncelikle Tablo 3 ve Tablo 5'teki değerler karşılaştırıldığında görülecektir ki sadece 3 problem hariç tümünde bir iyileşme sağlanmıştır. İyileşme oranları farklı olmakla birlikte, genel olarak 2-opt algoritmasının, tur geliştiren sezgiselleri iyileştirdiği söylenebilir. Herhangi bir iyileşme olmayan 3 sonuç incelendiğinde, bunların EUzE+2opt ve TCvW+2opt sonuçlarında olduğu görülebilir. Bu değerlerden bir tanesi zaten EUzE yönteminin optimum sonucu bulduğu, berlin52 problemine ait sonuç olduğu görülecektir. Diğer iki sonucun ise zaten tur oluşturan sezgiseller tarafından optimum sonuçlara çok yakın bulunan sonuçlar olduğu görülecektir (dantzig42: 709 ve eil51: 43). Bulunan sonuçların optimumdan yüzde sapmalarına ait değerler Tablo 6'da verilmiştir.

Tablo 3. Tur Oluşturan Sezgisellerin Literatür Problemleri İçin Ürettikleri Sonuçlar

	TEYKY	EYEY	EUEY	EUzEY	TCvWY	Optimum
dantzig 42	864	776	759	717	709	699
hk48	12137	12551	12544	11879	12025	11468
eil51	482	472	464	443	440	426
berlin52	8181	8700	8632	7542	7946	7542
brazil58	27384	27921	28455	25941	26688	25395
pr76	130921	127483	122044	113319	114101	108159
Türkiye81	10651	11826	11796	10378	10468	9921
eil101	710	699	662	659	667	629
kroA150	30565	30308	29902	29011	29023	26130

Tablo 4. Tur Oluşturan Sezgisellerin Literatür Problemleri İçin Ürettikleri Sonuçların Optimumdan Yüzde Sapmaları

	TEYKY	EYEY	EUEY	EUzEY	TCvWY
dantzig 42	23,61	11,02	8,58	2,58	1,43
hk48	5,83	9,44	9,38	3,58	4,86
eil51	13,15	10,80	8,92	3,99	3,29
berlin52	8,47	15,35	14,45	0,00	5,36
brazil58	7,83	9,95	12,05	2,15	5,09
pr76	21,04	17,87	12,84	4,77	5,49
Türkiye81	7,36	19,20	18,90	4,61	5,51
eil101	12,88	11,13	5,25	4,77	6,04
kroA150	16,97	15,99	14,44	11,03	11,07
Ortalama	13,02	13,42	11,65	4,16	5,35

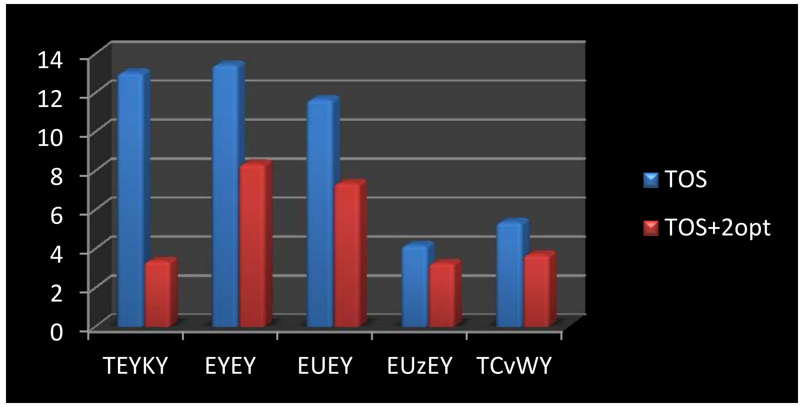
Tablo 5. Tur Oluşturan Sezgisellerle Başlangıç Çözümlü 2-Opt Algoritmasının Literatür Problemleri İçin Ürettikleri Sonuçlar

	TEYK+2opt	EYE+2opt	EUE+2opt	EUzE+2opt	TCvW+2opt	Optimum
dantzig 42	705	715	732	712	709	699
hk48	11544	11994	11942	11660	11752	11468
eil51	434	448	439	443	435	426
berlin52	7711	8599	8531	7542	7846	7542
brazil58	25704	27677	27181	25507	26096	25395
pr76	113404	117842	115370	111790	111548	108159
Türkiye81	10061	11136	11164	10377	10443	9921
eil101	673	663	652	655	651	629
kroA150	28782	29630	29077	28544	29023	26130

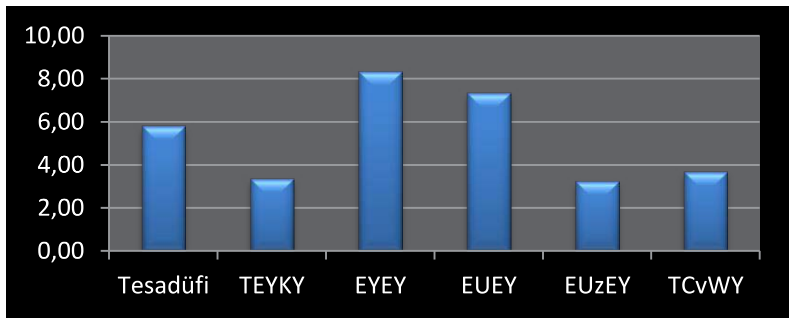
Tablo 6. Tur Oluşturan Sezgisellerle (TOS) Başlangıç Çözümlü 2-opt Algoritmasının Literatür Problemleri İçin Ürettikleri Sonuçların Optimumdan Yüzde Sapmaları

	TEYK+2opt	EYE+2opt	EUE+2opt	EUzE+2opt	TCvW+2opt
dantzig 42	0,86	2,29	4,72	1,86	1,43
hk48	0,66	4,59	4,13	1,67	2,48
eil51	1,88	5,16	3,05	3,99	2,11
berlin52	2,24	14,01	13,11	0,00	4,03
brazil58	1,22	8,99	7,03	0,44	2,76
pr76	4,85	8,95	6,67	3,36	3,13
Türkiye81	1,41	12,25	12,53	4,60	5,26
eil101	7,00	5,41	3,66	4,13	3,50
kroA150	10,15	13,39	11,28	9,24	11,07
Ortalama	**3,36**	**8,34**	**7,35**	**3,25**	**3,68**
Ortalama (TOS)	13,02	13,42	11,65	4,16	5,35

2-opt'la elde edilen sonuçların, tur oluşturan sezgisellere göre yüzde olarak da daha iyi sonuçlar ürettiği görülmektedir. Yüzde iyileşmeler doğal olarak, daha kötü sonuçlar veren TOS'larda daha yüksek, daha iyi sonuçlar veren TOS'larda daha düşüktür. Örneğin TEYK+2opt algoritması %13,02 olan oranı %3,36'ya indirmiştir. TCvW'de ise oran %5,35 iken 2-opt'la %3,68'e düşürülmüştür. İlginç olan sonuç ise 2-opt'lu TOS sonuçlarından en iyisinin TEYK+2opt'a ait sonucun olmasıdır. Sonucun böyle çıkmasının, 2-opt algoritmasının çözüm geliştirme alanının daha geniş olmasından kaynaklandığı düşünülmektedir. Sonuçların ortalama yüzde sapmalar için karşılaştırmalı olarak daha iyi görülebilmesi için Şekil 3 incelenebilir.



Şekil 3. TOS ve TOS+2opt Sonuçlarının Ortalamadan Yüzdelik Sapmaların Ortalamalarının Karşılaştırılması



Şekil 4. 2-opt Sonuçlarının, Başlangıç Çözümü Açısından Karşılaştırılması

Dördüncü bölümde, tesadüfi başlangıç çözümlü 2-opt algoritmasının sonuçları incelenmiş ve 100 çalıştırma sonucunda elde edilen değerler analiz edilmişti. Tüm problemler için optimumdan yüzde sapmaların ortalamalarının ortalaması %5,20 olarak hesaplanmıştı. 2-opt'un başlangıç çözümünü tesadüfi olarak belirlemek yerine, tur oluşturan sezgiselleri kullandığımızda elde ettiğimiz sonuçlar ise bu bölümde bulunmuştur. Bu değerlerin karşılaştırılmasına ise Şekil 4'te yer verilmiştir.

Şekil 4 incelendiğinde, başlangıç çözümlerinin 2-opt sonuçları üzerindeki etkisi rahatlıkla görülebilecektir. Öncelikle TEYKY, EUzEY ve TCvWY sezgisellerine ait sonuçlara başlandığında, sonuçların tesadüfi başlangıç çözümlü sonuçlara göre daha iyi sonuçlar verdiği söylenebilir. EYEY ve EUEY yöntemlerinin ise başlangıç çözümü açısından 2-opt'a katkı sağlayamadığı görülmektedir. En iyi iyileşme ise tekrarlı en uzak ekleme yöntemi başlangıç çözümü olarak kullanıldığında görülmüştür. Optimumdan sapma ortalamalarının ortalaması %5,8'den % 3,25'e düşmüştür ki bu da sonuçların %2,55 iyileşmesi demektir.

7. SONUÇ

Çalışmada, GSP problemleri için kullanılan ve yerel yazında henüz yer almayan 2-opt algoritması tanıtılmıştır. Klasik yöntemde tesadüfi olarak belirlenen başlangıç çözümünün sonuçlar üzerindeki etkisi, bilinen en iyi değerlerle karşılaştırılarak araştırılmıştır. Başlangıç çözümlerinin tesadüfi olarak belirlenmesi yerine, bilinen bazı tur oluşturan sezgiseller kullanıldığında, 2-opt'un sonuçlarının ne olacağı, gelişip gelişmeyeceği araştırılmıştır. Görülmüştür ki bazı tur oluşturan sezgisellerle başlandığında çözüm iyileştirilmiştir. Özellikle en yakın komşu yönteminin kullanılması ile iyileştirme en yüksek seviyeye çıkmıştır.

Test problemlerine ait nokta sayılarının 42 ile 150 arasında değiştiği ve belki farklı boyut ve zorluktaki problemler için farklı sonuçlar bulunabileceği unutulmamalıdır. Bir başka çalışmada daha farklı test problemleri ile analiz yapılabilir. Ayrıca 2-opt'ta iki kenar turdan çıkarılmakta ve ayrı iki parça yeni bir tur olacak şekilde birleştirilmektedir. Aynı mantıkla çalışan 3-opt, iki kenar yerine, turdan üç kenarın çıkarıldığı ve farklı şekillerde kalan parçaların birleştirildiği benzer bir algoritmadır. Bu yöntem de bir başka çalışmanın konusu olarak ele alınabilir.

KAYNAKÇA

1. **Biggs, N. L., LIyod, E. K., Wilson, R. J.** 1976. Graph Theory, Clarendon Press, Oxford.
2. **Blazinskas, A., Misevicius, A**. 2011. "Combining 2-opt, 3-opt And 4-Opt with K-Swap-Kick Perturbations for the Traveling Salesman Problem," Information Technologies' 2011: Proceedings of the 17th international Conference on Information and Software Technologies, (Eds. R. Butleris, R. Butkiene), 27-29 April 2011, Kaunas, Lithuania, Kaunas University of Technology, Technologija, Kaunas, p. 45-50.
3. **Croes, G. A.** 1958. "A Method for Solving Traveling-Salesman Problems," Operations Research, vol. 6 (6), p. 791-812.
4. **Dantzig, G., Fulkerson, R., Johnson, S.** 1954. "Solution of a Large-Scale Travelling-Salesman Problem," Journal of the Operations Research Society of America, vol. 2 (4), p. 393-410.
5. **Flood, M.** 1956. "The Traveling-Salesman Problem," Operations Research, vol. 4 (1), p. 61-75.
6. **Fortini, M.** 2007. "LP-Based Heuristics for the Traveling Salesman Problem," Doctoral Dissertation, Bologna University, Bologna.
7. **Gutin, G., Punnen, A. P.** 2002. The Traveling Salesman Problem and its Variations, Springer Science & Business Media, Chicago.
8. **Jevtic, A.** 2014. http://www.mathworks.com/matlabcentral/fileexchange/25542-nearest-neighbor-algorithm-for-the-travelling-salesman-problem/content/nn_tsp.m, son erişim tarihi: 11.09.2015.
9. **Johnson, D. S., Lyle, A. McGeoch.** 1997. "The Traveling Salesman Problem: A Case Study in Local Optimization," Local Search in Combinatorial Optimization, vol. 1, p. 215-310.
10. Karayolları Genel Müdürlüğü. 2014. http://www.kgm.gov.tr / SiteCollection Documents/KGMdocuments/Root/Uzakliklar/ilmesafe.xls, son erişim tarihi: 10.05.2014.
11. **Karp, R. M.** 1972. "Reducibility among Combinatorial Problems," In Miller, R. E., Thatcher, J. W. (Eds.), Complexity of Computer Computations, Plenum Press, New York.
12. **Kuzu, S., Önay, O., Şen, U., Tunçer, M., Yıldırım, B. F., Keskintürk, T.** 2014. "Gezgin Satıcı Problemlerinin Metasezgiseller ile Çözümü," İstanbul Üniversitesi İşletme Fakültesi Dergisi, sayı 43 (1), s. 1-27.
13. **Lin, S.** 1965. "Computer Solutions of the Traveling Salesman Problem," Bell System Technical Journal, vol. 44 (10), p. 2245–2269.
14. **Nilsson, C.** 2003. "Heuristics for the Traveling Salesman Problem," Tech. Report, Linköping University, Sweden.
15. **Önder, E., Özdemir, M., Yıldırım, B. F.** 2013. "Combinatorial Optimization Using Artificial Bee Colony Algorithm and Particle Swarm Optimization Supported Genetic Algorithm," Kafkas University, Journal of Economics and Administrative Sciences Faculty, vol. 4 (6).
16. **Reinelt, G.** 1995. "TSPLIB95," http://www.iwr.uni-heidelberg.de/groups/comopt/software/ TSPLIB95/tsp/, son erişim tarihi: 10.01.2015.
17. **Verhoeven, M. G., Aarts, E. H., Swinkels, P. C.** 1995. "A Parallel 2-opt Algorithm for the Traveling Salesman Problem," Future Generation Computer Systems, vol. 11 (2), p. 175-182.

Ek 1. 2-Opt Algoritmasının Matlab Kodları

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%         GSP için 2-opt Algoritması        %
%          Doç.Dr. Timur Keskintürk         %
%                 08.09.2015                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%  2-opt - Açıklamalar ve Girdiler

%Method GSP problemlerinin çözümü için geliştirilmiş iki kenarın
%turda olmayan iki kenar ile değiştirilmesine ve bu yolla turun
%iyileştirilmesine dayanmaktadır. (ab,cd)>(ac,bd). Tüm
%kombinasyonların denenmesi sonucu çıkan sonuca 2-optimal denir.

%Girdiler:
%distances: Probleme ait mesafeler matrisini içermektedir.
%initial: Modele dışarıdan girilen (varsa) başlangıç çözümü,
%yoksa "0" yazınız

%Referans verilerek kullanılabilir ve geliştirilebilir

function [best, besttour] = two_opt(distances,initial);

n = size(distances,1);
if initial==0;besttour=randperm(n);else;besttour=initial;end
best = totaldistance(distances,besttour);

combs=combnk(1:n,2);counter=0;z=-1;
while z<0
    while counter<size(combs,1)
        z=0;
        counter=counter+1;roadtabu=besttour;
roadtabu(combs(counter,1):combs(counter,2))=fliplr(roadtabu(comb
s(counter,1):combs(counter,2)));
        besttabu=totaldistance(distances,roadtabu);
        if besttabu<best
            z=besttabu-best;
            besttour=roadtabu;best=besttabu;
        end
    end
end
best
besttour

% Aşağıdaki fonksiyon GSP çözümüne ait toplam mesafeyi
vermektedir.

function totdistance=totaldistance(distances,T)
numofnodes=size(distances,1);
q = double(T);
ind = sub2ind([numofnodes,numofnodes],q,[q(2:numofnodes),q(1)]);
totdistance = sum(distances(ind));
```