




## 0-1 problem uzayları için kuantum tabanlı yerel arama ile güçlendirilmiş parçacık sürüsü optimizasyonu

### A quantum based local search enhanced particle swarm optimization for binary spaces

Fehmi Burçin ÖZSOYDAN\* 

<sup>1</sup>Endüstri Mühendisliği Bölümü, Mühendislik Fakültesi, Dokuz Eylül Üniversitesi, İzmir, Türkiye.  
burcin.ozsoydan@deu.edu.tr

Geliş Tarihi/Received: 08.11.2017, Kabul Tarihi/Accepted: 05.03.2017

\* Yazışılan yazar/Corresponding author

doi: 10.5505/pajes.2018.64614

Araştırma Makalesi/Research Article

#### Öz

Parçacık Sürüsü Optimizasyonu (PSO), problem çözmeye balık veya kuş sürülerinin hareketlerini taklit eden, oldukça bilinen sürü zekâsı tabanlı bir algoritmadır. İlk olarak kısıtsiz global optimizasyon problemlerini çözmek için önerilse de, çeşitli problem tiplerini içeren çok sayıda PSO çalışması mevcuttur. Fakat bununla birlikte, ilgili bilimsel yazından görülebileceği gibi, diğer uygulama türleriyle karşılaştırıldığında, kesikli ve 0-1 yapıdaki PSO uygulamaları görece daha az sayıdadır. Bu bağlamda, mevcut araştırmada, kuantum tabanlı yerel arama yordamı ile güçlendirilmiş bir 0-1 PSO modifikasyonu getirilmiştir. Bahsedilen kuantum tabanlı prosedür, algoritma tarafından bulunan eniyi çözüm etrafında üretilen bir küre içinde konumlanan ve kuantum parçacıkları olarak adlandırılan rastgele dağıtılmış parçacıklar üretir. Ardından bu parçacıklar, bulunan eniyi çözüm üzerinde olası iyileştirmeler sağlayabilmek için yerel arama amacıyla kullanılır. Önerilen yaklaşımın performansı, bu alanda sıkça kullanılan Bir-Enb, Aldatıcı, Plato ve Kral Yolu fonksiyonlarından oluşan bir 0-1 problem seti kullanılarak test edilmiştir. Deneysel çalışma, önerilen yaklaşımın 0-1 problemlerdeki etkinliğini göstermektedir.

**Anahtar kelimeler:** Metasezgiseller, Parçacık Sürüsü Optimizasyonu, 0-1 optimizasyon, Kuantum parçacıkları

#### Abstract

Particle Swarm Optimization (PSO) is a well-known swarm intelligence-based algorithm that simulates the movements of school or bird flocks in problem solving. Although it is first introduced to solve unconstrained global optimization problems, there are numerous reported publications of PSO involving various types of problems. However, as one can see from the related literature, compared to other types of implementations, discrete and binary PSO applications are relatively fewer in number. In this context, in the present work, a 0-1 PSO modification enhanced with a quantum-based local search procedure is developed. The mentioned quantum-based procedure generates randomly scattered particles referred to as quantum particles located within a sphere that is generated around the best-found solution by the algorithm. Next, these particles are used for local search to achieve possible improvements on the best-found solution. The performance of the proposed approach is tested by using a 0-1 problem suite consisting of the commonly used One-Max, Deceptive, Plateau and Royal Road functions. Experimental study shows the effectiveness of the proposed approach in 0-1 problems.

**Keywords:** Metaheuristics, Particle swarm optimization, Binary optimization, Quantum particles

## 1 Giriş

Doğadaki çeşitli organizmaların işleyişlerinden esinlenilerek oluşturulan algoritmalar, çözülmesi zor olan ve kesin çözüm yöntemlerinin çoğunlukla yetersiz kaldığı problemlerin çözümünde yeni fırsatlar ortaya çıkarmıştır. Metasezgiseller sınıfında yer alan bu algoritmaların ortak noktası, sözü edilen organizmaların karakteristik özelliklerini bilgisayar ortamında simüle etmektir. Bu alandaki çeşitlilik, özellikle son 10 yılda gözlenir bir şekilde artış göstermiştir. Çeşitliliğin artmasına rağmen dikkat edilecek olursa bu algoritmaların birçoğunda kullanılan operatörler, temelde Genetik Algoritmalar (GA), Diferansiyel Evrim (DE) Algoritması ve Parçacık Sürüsü Optimizasyonu' nun (PSO) operatörlerinden esinlenilmiştir. Bu bağlamda bu üç algoritmanın aslında metasezgisel algoritmaların temel taşları olduğu ileri sürülebilir.

Bu üç algoritmadan GA [1], bu alandaki ilk geliştirilen ve günümüzde hala güncelliğini koruyan bir algoritmadır. Mutasyon, çaprazlama ve doğal seleksiyon gibi evrimsel olayları bilgisayar ortamına simüle ederek çözülmesi zor problemlerin çözümünde sıklıkla kullanılmaktadır.

GA' nın yaygınlığının ve bilinir olmasının bir sebebi de, sözü edilen çaprazlama, mutasyon ve doğal seleksiyon gibi evrimsel operatörlerinin farklı algoritmalarda da kullanılabilir

olmasıdır. Örneğin Storn ve Price [2] tarafından önerilen DE, temelde GA'nın kullandığı çaprazlama ve mutasyon operatörlerini farklı şekilde kullanarak çözüm aramaktadır. GA, farklı çözümlerin kombinasyonları (çaprazlamaları) ve bunların üzerinde yapılan değişiklikler (mutasyon) ile yeni çözüm üretirken, DE, vektör farkı tabanlı bir yaklaşımla birlikte bu operatörleri kullanmaktadır.

PSO [3] ise bu algoritmaların bir ölçüde farklı olarak bir sürü zekâsı kullanımını simüle etmektedir. Bu yaklaşımda her bir parçacığın (çözüm vektörünün) kendi hafızası ve genel bir sürü hafızası vardır. Böylece her parçacık o ana kadar bulduğu eniyi çözümü saklayabilir. Benzer şekilde sürü tarafından bulunan eniyi çözüm de saklanır ve bu çözüm diğer parçacıklar tarafından kullanılabilir durumdadır. Bu yaklaşımın esin kaynağı toplu halde hareket eden kuş ya da balık sürülerinin sergiledikleri davranışlardır. Bu canlılar daha iyi yaşam koşullarına ve daha bol yiyecek kaynaklarına erişebilmek için sürü halinde hareket ederler.

Bu üç temel algoritmanın haricinde günümüze kadar pek çok benzeri yaklaşım önerilmiştir. Örneğin Karaboğa ve Baştürk [4] yapay arıları kullanarak fonksiyon optimizasyon problemlerini çözmüşlerdir. Yang [5] ateş böceklerinin iletişim davranışlarından esinlenmiş ve Ateş Böceği Algoritması'nı geliştirmiştir. Ardından Krishnanand ve Ghose [6], Yang' ın [5]

ateş böceği algoritmasına benzer ancak farklı yapıları kullanan bir algoritma sunmuşlardır. Kısa bir süre sonra Yang [7] yine popüler olduğu ileri sürülebilecek bir algoritma olan Yarasa Algoritması'nı geliştirmiştir. Bu algoritmada yarasaların yön bulma konusunda ses dalgalarından yararlanmasından esinlenilmiştir. Ardından aynı yazar, çiçeklerin polinasyon mekanizmalarını temele alan, kullanımı basit ama iyi sonuçlar üretebildiği söylenebilecek bir algoritma geliştirmiştir. Bu algoritmada çiçek polenlerinin canlılar ya da canlı olmayan faktörler aracılığıyla uzun (global arama) ya da kısa mesafe (yerel arama) taşınabilmesi durumlarını simüle edilmiştir [8]. Benzer şekilde kurt sürülerindeki bireylerin kendi içerisindeki hiyerarşik yapısı ve monark kelebeklerini göçü, görece yeni sayılabilecek doğa esinli diğer algoritmalara [9],[10] esin kaynağı olmuştur.

Mevcut çalışmanın amacı, transfer fonksiyonuna gerek kalmadan 0-1 çözüm uzaylarında arama yapabilen bir yöntem ve bu yöntemle uyum içerisinde çalışabilecek bir yerel arama yordamı geliştirmektir. Bu bağlamda, mevcut çalışmada, yukarıda sözü edilen algoritmaların temel taşlarını oluşturan GA'ların düzgün çaprazlama (uniformed-based) operatörünü kullanan bir 0-1 PSO önerilmiştir. Böylece bu yaklaşımda PSO'daki parçacık hareketleri gerçekleştirilirken transfer fonksiyonlarına gereksinim duyulmayacaktır. Ayrıca önerilen algoritma, atom modelinden ve atomik parçacıkların hareketlerinden esinlenilmiş bir kuantum tabanlı yerel arama yordamıyla güçlendirilmiştir.

İlgili bilimsel yazında kuantum davranışı tabanlı PSO uyarlamalarının [11]-[14] yer aldığı görülebilmektedir. Ancak bu çalışmalar, mevcut çalışmada sözü edilen kuantum tabanlı yerel arama yordamıyla yalnızca isim benzerliği taşımaktadır. Kuantum davranışı tabanlı PSO yaklaşımlarında [11]-[14], parçacıkların etkileşime geçeceği diğer bireyler ve aralarındaki operatörler, standart PSO operatörlerinden farklılık göstermektedir. Mevcut çalışmada kullanılan yerel arama yordamında ise algoritma tarafından bulunan eniyi çözüm etrafında belirli bir yarıçapa göre oluşturulan soyut bir küre içerisinde rassal dağıtık parçacıklar oluşturulmaktadır ve bu parçacıklar yerel arama amacıyla kullanılmaktadır. Önerilen yaklaşımın performansı, Bir-Enb, Aldatıcı, Plato ve Kral Yolu fonksiyonlarından oluşan bir 0-1 problem setinde test edilmiştir. Çalışmanın devamı şu şekilde organize edilmiştir.

Önerilen PSO modifikasyonu ile ilgili tüm detaylı bilgiler Bölüm 2'de sunulmuştur. Bölüm 3'te, kullanılan kıyaslama problemleri ve deneysel sonuçlar aktarılmıştır. Son olarak Bölüm 4'te, çalışmadan elde edilen sonuçlar özetlenmiştir.

## 2 Kullanılan çözüm yaklaşımları

Önerilen yaklaşımın ayrıntılarına geçmeden önce izleyen bölümde standart PSO'dan kısaca bahsedilmektedir.

### 2.1 PSO

Daha önce de belirtildiği üzere PSO [3], sürü zekâsı tabanlı bir metasezgisel algoritmadır. PSO'daki işleyiş genel olarak iki denklem üzerine kurgulanmıştır. Bunlardan ilki (1) hız vektörünün hesaplanması, diğeri (2) ise hesaplanan hız vektörüne bağlı olarak parçacık koordinatlarının yeni değerlerinin hesaplanması amacıyla kullanılmaktadır.

$$v_{ij} = wv_{ij} + c_1 \times rand \times (pbest_{ij} - x_{ij}) + c_2 \times rand \times (x_{best,j} - x_{ij}) \quad (1)$$

$$x_{ij} = x_{ij} + v_{ij} \quad (2)$$

Bu denklemlerde  $x_{ij}$ ,  $i$ . parçacığın  $j$ . boyuttaki değerini,  $pbest_{ij}$   $i$ . parçacığın  $j$ . boyuttaki parçacık eniyi çözüm vektörü değerini,  $x_{best,j}$  algoritma tarafından bulunan eniyi çözüm vektörünün  $j$ . boyuttaki değerini,  $v_{ij}$  ise  $i$ . parçacığın  $j$ . boyuttaki hızını temsil etmektedir. Bunların dışında  $w$ ,  $c_1$  ve  $c_2$  sırasıyla atalet (eylemsizlik) ve hızlandırma katsayılarını göstermektedir.

PSO, rassal ya da belirli bir kurala göre oluşturulmuş çözümlerden oluşan bir popülasyon ile başlatılır ve ilk olarak her çözümün çözüm değeri hesaplanır. Bunlar arasından eniyi çözüm vektörü ( $X_{best}$ ) bulunur. İlk iterasyon için  $v_{ij} = 0 \forall i, j$  ve  $pbest_{ij} = x_{ij} \forall i, j$  kabul edilebilir. Daha sonra sırasıyla Denklem (1) ve Denklem (2) aracılığıyla  $v_{ij}$  ve  $x_{ij}$  hesaplanır. İlgili parçacığın yeni koordinatları hesaplandıktan sonra ise yeni çözüm değeri bulunur ve gerekli ise (daha iyi bir çözüm bulduysa)  $pbest$  ve  $X_{best}$  güncellemeleri yapılır. Aynı işlem tüm parçacıklar için yapıldıktan sonra bir iterasyon tamamlanmış olur. Yeni iterasyonda yukarıdaki işlemler tüm parçacıklar için tekrarlanır ve önceden belirlenen bir tamamlanma koşulu sağlandığında algoritma sonlandırılır.

### 2.2 0-1 PSO

PSO ilk olarak sürekli problemlerin çözümü için önerilmiş, ardından farklı özelliklerdeki çeşitli problemler için uyarlamaları geliştirilmiştir [15],[16].

0-1 yapıdaki problemler için ise ilk olarak Kennedy ve Eberhart [17] sigmoid fonksiyonunu kullanmayı ve sürekli değişkenleri 0-1 değişkenlere çevirmeyi önermiştir. Bu teknik, PSO'nun orijinal işleyişini korumuş ancak sürekli problemlerdeki başarısının tam karşılığını yansıtamamıştır. Bu nedenle 0-1 ve kesikli yapıdaki problemler için daha farklı PSO uygulamaları önerilmiştir. Bunlardan en göze çarpanı evrimsel operatörleri PSO işleyişinde kullanmak olmuştur [18]-[26]. Tüm bunlar ile ilgili bir yazın taraması Kennedy [27] tarafından sunulmuştur.

#### 2.2.1 Önerilen 0-1 PSO ana işleyiş

Bu çalışmada önerilen PSO'nun tüm mekanizmaları Algoritma 1' de özetlenmiştir. Burada  $popSize$ ,  $dim$ , sırasıyla popülasyon büyüklüğünü, çözüm vektörü boyutunu temsil etmektedir. Ek olarak ayrıntıları aşağıda verilmiş olmak üzere,  $xover1$  ve  $xover2$  algoritma davranışını belirleyebilmek amacıyla kullanıcı tarafından tanımlanan parametrelerdir.

Algoritma parametreleri tanımlandıktan sonra buradan görülebileceği gibi öncelikle rassal olarak 0-1 popülasyonu oluşturulur ve başlangıç popülasyonu için çözüm değerleri hesaplanır. Başlangıçta bu çözüm vektörleri  $pbest$  olarak kullanılmıştır. Bunların arasından eniyi çözüm vektörü tespit edilir ve ardından birinci iterasyona geçilir.

İlk çözüm vektörü seçilir ve her bir boyutu için ayrı olmak üzere  $r=rand \in [0,1]$  düzgün rassal sayıları üretilir. Örnek olarak  $i$ . parçacığın  $j$ . boyutu için üretilen  $r$  sayısı  $xover1$  parametresinden küçük eşit ise  $x_{ij}$  elemanında bir değişiklik yapılmaz. Bu, orijinal PSO'da atalet teriminin bir karşılığıdır. Eğer aynı boyut için üretilen  $r$  sayısı  $xover1$ 'den büyük ve  $xover2$ 'den küçük eşit ise burada, parçacık eniyi çözüm vektöründen faydalanılır ve  $x_{ij} = pbest_{ij}$  ataması yapılır. Bu da orijinal PSO'daki bireysel zekâ mekanizmasının karşılığıdır. Benzer şekilde  $r$  sayısı  $xover2$ 'den büyük ise  $x_{ij} = x_{best,j}$  ataması yapılarak sosyal zekâdan faydalanılır ve  $i$ . parçacık  $j$ . boyutta eniyi parçacığın değerini almış olur. Buradan

görülebileceği üzere bu işleyiş, GA düzgün çaprazlama (uniform-based crossover) operatöründen esinlenilmiştir. Tüm boyutlar için bu işlem tamamlandıktan sonra ilgili parçacığın çözüm değeri hesaplanır ve parçacık eniyi çözüm değerinden daha iyi bir değer elde edildiyse  $pbest_{ij} = x_{ij} \forall j$  ataması yapılır. Benzer şekilde elde edilen yeni çözüm algoritma tarafından o ana kadar bulunan eniyi çözümden daha iyiyse  $x_{best,j} = x_{ij} \forall j$  ataması yapılır. Böylece  $i$ . parçacık için bu işlemler tamamlanır, aynı işlemler tüm parçacıklar için tekrarlanır ve 1 iterasyon tamamlanmış olur. Her iterasyonun sonunda ayrıntıları izleyen bölümde sunulacak olan kuantum tabanlı bir yerel arama prosedürü uygulanır.

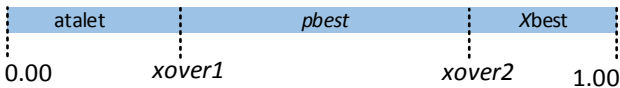
**Algoritma 1.** 0-1 PSO için sözde kod.

```

rassal olarak 0-1 popülasyonu oluştur
çözüm değerlerini hesapla
parçacık eniyi (pbest) ata ve global eniyi (Xbest) çözümü bul
while sonlandırma kriteri sağlanmıyorken do
  for i=1:popSize loop
    for j=1:dim loop
      r=rand ∈ [0,1]
      if r ≤ xover1
        xij = xij
      end
      if r > xover1 && r ≤ xover2
        xij = pbestij
      end
      if r > xover2
        xij = xbest,j
      end
    end
    yeni çözüm değerini hesapla
    gerekliyse i. parçacık eniyiyi güncelle
    gerekliyse eniyi parçacığı güncelle
  end
  kuantum tabanlı yerel aramayı uygula
end while
print bulunan eniyi çözümü yazdır

```

Dikkat edilecek olursa bu mekanizmanın kullanılabilmesi bazı şartlara bağlıdır. Öncelikle  $xover1$  ve  $xover2$  parametrelerinin  $[0,1]$  aralığında olmaları ve  $xover1 \leq xover2$  koşulunu sağlamaları gerekmektedir (Şekil 1). Ayrıca bu parametrelerin aldığı değerler algoritmanın davranışını belirleyecektir. Örneğin 0'a oldukça yakın bir  $xover1$  ile 1'e çok yakın  $xover2$  parametresi, yeni çözüm vektörleri oluşturulurken parçacık eniyi vektörlerinin kullanım olasılığını arttırmaktadır.



Şekil 1:  $xover1$  ve  $xover2$  parametreleri için görsel bir ölçek.

**2.2.2 Kuantum tabanlı yerel arama yordamı**

Blackwell ve Bentley [28] popülasyon çeşitliliğini sağlayabilmek amacıyla yüklü ve yüksüz parçacıkların eş zamanlı olarak kullanılmasını önermişlerdir. Bu parçacıklar popülasyonda farklı karakteristiklere sahip olarak tanımlanmıştır. Daha sonra Blackwell ve Branke [29], bu atomik yapıdan esinlenilmiş popülasyonu geliştirerek bir kuantum modeli sunmuşlardır. Klasik atom modelinde elektronlar çekirdek etrafında belirli bir uzaklıkta hareket etmektedir. Ancak Blackwell ve diğ. [30]'in kuantum modeline göre, elektronlar deterministik bir hareket değil, çekirdeğin etrafını kapsayan soyut bir kürenin içinde rassal dağıtık bir

şekilde hareket etmektedirler. Bu fikir burada önerilen PSO için bir yerel arama yordamı olarak kullanılmıştır. Kullanılan yöntemin detayları Algoritma 2'de sunulmaktadır.

**Algoritma 2.** Kuantum tabanlı yerel arama yordamı.

```

vj ~ N(0,1), 1 ≤ j ≤ dim olmak üzere rassal bir Gauss vektörü üret
bu vektörün orijine uzaysal uzaklığını (dist) belirle dist = √(∑j=1dim vj2)
u ∈ [0,1] düzgün rassal sayısını üret
for j=1:dim loop
  qj = xbest,j + vj × rcloud × √u / dist
end
if f(Q) > f(Xbest)
  Xbest = Q
  f(Xbest) = f(Q)
end
**rcloud: kuantum bulutu yarıçapı
**qj: kuantum parçacığının j. boyuttaki değeri
**Q: kuantum parçacığı

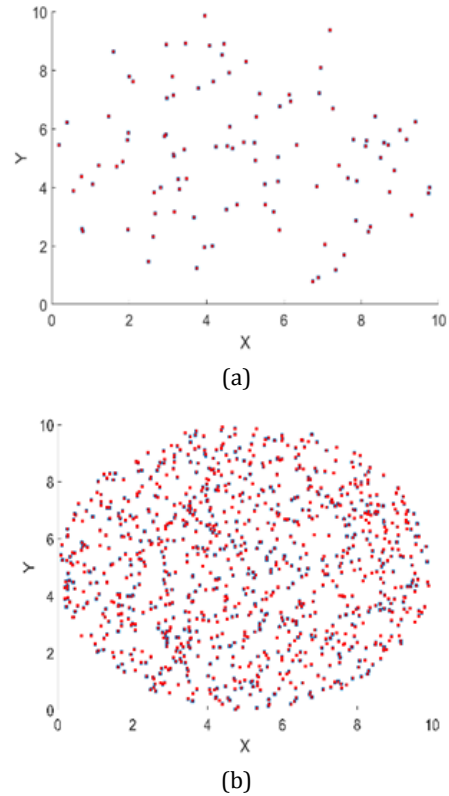
```

Algoritma 2'den açıkça görülebildiği üzere, kuantum parçacıkları 0-1 formatında değil, reel sayı olarak oluşturulmaktadır. Önerilen 0-1 PSO'nun ana akışında kullanılmasa bile burada bu sebeple transfer fonksiyonu kullanılmıştır. Denklem (3)'te gösterildiği gibi burada sigmoid (lojistik) transfer fonksiyonu tercih edilmiştir.

$$T(q_j) = \frac{1}{1 + e^{-q_j}} \quad (3)$$

$$b_j = \begin{cases} 0 & \text{if rand() < } T(q_j) \\ 1 & \text{if rand() } \geq T(q_j) \end{cases}$$

Son olarak rassal olarak dağıtılmış kuantum parçacıklarına ilişkin bir görsel ise Şekil 2'de sunulmuştur.



Şekil 2: Merkezi (5,5) noktasında ve  $r_{cloud}=5$  olan. (a): 100, (b):1000 kuantum parçacığı.

### 2.3 Örnek bir problem

Önceki bölümde sunulan çözüm yöntemi, burada küçük bir örnek problem üzerinde anlatılmıştır.

Öncelikle algoritmanın tüm gerekli parametreleri tanımlanır ve 0-1 popülasyonu rastgele oluşturulur. Başlangıç popülasyonunda çözüm vektörlerinin her bir boyutu 0.50 olasılıkla 0, 0.50 olasılıkla 1 değerini alır.  $xover1$  ve  $xover2$  parametreleri sırasıyla 0.20 ve 0.80 olarak tanımlanmış olsun. Başlangıç popülasyonu oluşturulduktan sonra tüm bireylerin amaç fonksiyonu değerleri hesaplanır,  $pbest$  ve  $X_{best}$  çözümler bulunur. İlk iterasyon için  $pbest_{ij} = x_{ij} \forall i, j$  olacağı için, algoritma operatörlerini daha iyi aktarabilmek amacıyla izleyen iterasyonlardan herhangi bir  $i$ . çözüm vektörü  $(x_{ij}, \forall j)$  0-0-1-1 olarak elde edilmiş olsun. Buradan problemin 4 boyutlu olduğu görülebilmektedir. Bu  $i$ . çözüm vektörü için  $pbest_{ij}, \forall j$  ise 1-0-1-0 olsun. Son olarak popülasyon eniyi çözümü olan  $X_{best}$  ise 1-1-1-1 olarak elde edilmiş olsun. Algoritma 1'den de görülebileceği üzere her bir boyut için  $r=rand \in [0,1]$  düzgün, sürekli rassal sayısı üretilir. Bunlar sırasıyla 0.18, 0.92, 0.56 ve 0.78 olsun. Bu durumda  $i$ . vektörün bir sonraki iterasyondaki boyutlarının değeri sırasıyla  $x_{i1}, x_{best,2}, pbest_{i3}$  ve  $pbest_{i4}$  olacaktır. Bir başka ifadeyle yeni vektör, 0-1-1-0 olarak elde edilmiş olacaktır. Bu işlemin ardından çözüm iyileştirmeleri sağlandıysa Algoritma 1'de tanımlanan güncellemeler yapılır. Bu işlem, popülasyondaki tüm çözüm vektörlerine uygulanır ve böylece bir sonraki popülasyon elde edilir. Son olarak kuantum tabanlı yerel arama yordamı (Algoritma 2) eniyi çözüme uygulanır.

Bu aşamada öncelikle problem boyutu kadar, ortalaması 0.00, standart sapması 1.00 olan normal dağılımı rassal değişkenler üretilir. Bu değişkenler ( $v_j$ ) sırasıyla -1.0329, -0.3233, 0.7665 ve 1.7447 olarak elde edilmiş olsun. Elde edilen Gauss vektörünün orijin noktası olan (0,0,0,0)'a uzaklığı hesaplanır. Bu uzaklık 2.1915'tir. Ardından,  $u \in [0,1]$  sürekli düzgün rassal sayısı üretilir ve Algoritma 2'deki işlem her bir boyut için uygulanır. Burada  $u=0.1635$ ,  $r_{cloud}=0.90$ ,  $X_{best}=(1,1,1,1)$  olarak alınır, kuantum vektörü (0.7302, 0.9155, 1.2001, 1.4556) olarak hesaplanır. İlgilenilen problemler 0-1 çözüm uzayına sahip olduğu için burada son olarak sigmoid transfer fonksiyonun yardımıyla 0-1 vektör, 0-1 vektöre dönüştürülür ve 0-1 kuantum vektörü 0-0-1-1 olarak elde edilir. Sigmoid fonksiyonunun rassal bir fonksiyon olduğu unutulmamalıdır. Ayrıca dikkat edilecek olursa, sigmoid fonksiyonundan geçirilmemiş vektörün tüm boyutlarının değeri, Şekil 2'de gösterildiği gibi  $X_{best} \mp r_{cloud}$  aralığında olacaktır. Yeni çözüm, eniyi çözümden daha iyiyse gerekli güncellemeler yapılır. Aynı işlem, kuantum tabanlı yerel aramada üretilecek çözüm sayısı kadar tekrarlanır.

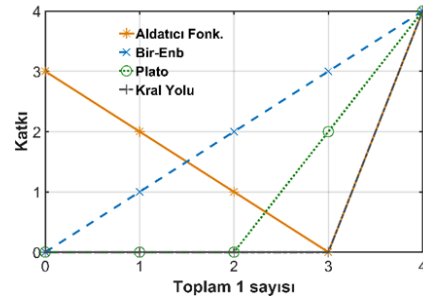
## 3 Deneysel sonuçlar

Bu bölümde, öncelikle deneysel çalışmalarda kullanılmış olan problem setine yer verilmiş, ardından algoritma parametrelerinin uygun değerlerinin bulunabilmesi amacıyla bir ön çalışma yapılmış ve son olarak elde edilen sonuçlar aktarılmıştır.

### 3.1 Kullanılan 0-1 problemler

Tüm detayları Wang ve diğ. [31] ile Calderin ve diğ. [32] tarafından da belirtilen problemler mevcut çalışmada da kullanılmıştır. Bu problem grubu Bir-Enb (One-Max), Plato (Plateau), Kral Yolu (Royal Road) ve Aldatıcı (Deceptive) fonksiyonlardan oluşmaktadır.

Bu problemler her birisi 25 bloktan oluşan çözüm vektörleri ile temsil edilen kısıtsız enbüyükleme problemleridir. Her bir blok 4 elemanın (bit) birleşiminden oluşmakta ve böylece çözüm vektörü toplamda 100 elemandan oluşmaktadır. Bir bloktaki toplam 1 sayısı (unitation number), o bloğun probleme bağlı olarak amaç fonksiyonuna katkısını belirtmektedir. Örneğin Plato fonksiyonunda bir bloktaki toplam 1 sayısı 3 ise o bloğun amaç fonksiyonuna katkısı 2, 4 ise katkısı 4 olmaktadır. Geriye kalan tüm durumlarda katkı 0'dır. Kral Yolu probleminde bir bloğun katkısı, yalnızca bloktaki toplam 1 sayısının 4 olduğu durumlarda 4'e eşit, diğer durumlarda ise 0 olmaktadır. Aldatıcı fonksiyonlar için bir bloktaki toplam 1 sayısı 0, 1, 2, 3 ve 4 iken, o bloğun katkısı sırasıyla 3, 2, 1, 0 ve 4 olmaktadır. Son olarak Bir-Enb probleminde amaç basitçe çözüm vektöründeki toplam 1 sayısını enbüyükleme. Bu problemlere ilişkin bir görsel Şekil 3'te sunulmaktadır. Burada bir bloğun alabileceği toplam 1 sayısı ve probleme göre, amaç fonksiyonuna sağlanan katkı özetlenmiştir.



Şekil 3: Kullanılan 0-1 problemlere ilişkin bir görsel [31],[32].

### 3.2 Algoritma parametrelerinin ayarlanması

Algoritmaların başarısı, kullanılan parametrelerin değerleri ile oldukça yakından ilişkilidir. Bu nedenle testlere geçilmeden önce mevcut çalışmada bu değerlerin uygun seviyeleri belirlenmeye çalışılmıştır. Bu bağlamda öncelikli olarak PSO'nun davranışını belirleyen  $xover1$  ve  $xover2$  parametrelerinin çeşitli kombinasyonları test edilmiş ve Tablo 1'de sunulmuştur. Bu parametrelerin etkisinin daha net görülebilmesi için kuantum yerel araması kapatılmıştır. Buradaki tüm deneyler 100 popülasyon büyüklüğü ve 100 iterasyon değerleri altında yapılmıştır. Performans gösterge değeri olan  $\bar{F}_{BG}$  ise Denklem (4)'te formüle edilmiştir. Burada  $itMaks$ ,  $N$  ve  $F_{BG_{ij}}$  sırasıyla izin verilen maksimum iterasyon sayısı (algoritma sonlandırma kriteri), toplam koşum sayısı ve  $i$ . iterasyon  $j$ . koşumdaki algoritma tarafından bulunabilen eniyi çözüm değerini göstermektedir.

$$\bar{F}_{BG} = \frac{1}{itMaks} \sum_{i=1}^{itMaks} \left( \frac{1}{N} \sum_{j=1}^N F_{BG_{ij}} \right) \quad (4)$$

Tablo 1:  $xover1$  ve  $xover2$  parametrelerinin ayarlanması.

$xover1=0.10$	$xover1=0.20$	$xover1=0.30$	$xover1=0.40$	$xover1=0.50$					
$xover2$	$\bar{F}_{BG}$	$xover2$	$\bar{F}_{BG}$	$xover2$	$\bar{F}_{BG}$	$xover2$	$\bar{F}_{BG}$	$xover2$	$\bar{F}_{BG}$
0.20	90.19	0.30	88.96	0.40	88.70	0.50	89.97	0.60	90.67
0.30	89.54	0.40	89.51	0.50	90.36	0.60	90.71	0.70	91.79
0.40	89.30	0.50	90.94	0.60	91.36	0.70	93.01	0.80	93.74
0.50	90.80	0.60	92.33	0.70	93.67	0.80	94.23	0.90	92.08
0.60	92.94	0.70	94.31	0.80	94.90	0.90	92.42		
0.70	94.73	<b>0.80</b>	<b>95.59</b>	0.90	92.74				
0.80	95.55	0.90	92.85						
0.90	92.87								

Bu analizde görüldüğü üzere 0-1 PSO,  $xover1=0.20$  ve  $xover2=0.80$  için eniyi performansı elde etmiştir. Benzer bir uygulama  $r_{cloud}$  parametresinin uygun değerini tespit etmek için yapılmış ve 0.01 ile 20.00 arasında değerler test edilmiştir. Bu sonuçlara göre  $r_{cloud}=1.00$  için iyi değerlerin elde edilebildiği gözlenmiştir.

### 3.3 Deneysel sonuçlar

Kuantum yerel aramalı PSO ile (0-1 kPSO) yerel arama yordamsız 0-1 PSO'nun adil bir şekilde karşılaştırılabilmesi için her iki algoritmanın da bir iterasyonda eşit sayıda fonksiyon hesaplama (FH) yapmasına izin verilmiştir. 0-1 PSO ve 0-1 kPSO algoritmalarında popülasyon büyüklüğü eşit seçilirse, 0-1 kPSO yerel arama yordamı sebebiyle daha fazla çözüm değerlendirecek ve avantajlı hale geçecektir. Bu nedenle 0-1 kPSO'nun popülasyon büyüklüğü kuantum tabanlı yerel aramada değerlendireceği çözüm kadar azaltılmıştır. Buna ilişkin konfigürasyon Tablo 2'de sunulmuştur.

Tablo 2:  $popSize+kuantum$  yerel arama konfigürasyonu.

1 iteryonda izin verilen FH	0-1 PSO	0-1 kPSO
FH/t=120	120+0	100+20
FH/t=100	100+0	85+15
FH/t=80	80+0	65+15
FH/t=50	50+0	40+10

Parametre ayarlama testlerinde olduğu gibi burada da  $itMaks$  100'e eşitlenmiştir. Popülasyon büyüklüğüne ve kuantum yerel arama sayılarına ise Tablo 2'den kolayca erişilebilmektedir. Örnek olarak 1 iterasyonda 120 FH'ye izin verilen testlerde 0-1 PSO'nun popülasyon büyüklüğü 120 iken, 0-1 kPSO'nun popülasyon büyüklüğü 100 ve kuantum yerel arama sayısı 20 olarak belirlenmiştir. Tüm bu parametreler altında elde edilen sonuçlar Tablo 3'te sunulmaktadır.

Tablo 3: Deneysel sonuçlar.

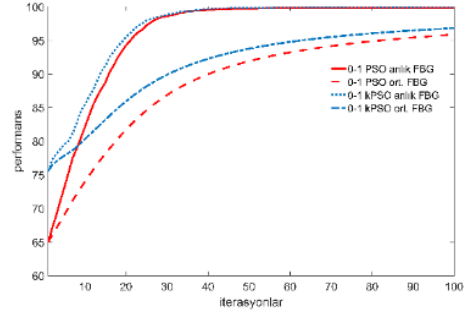
		0-1 PSO	0-1 kPSO
FH/t=120	Bir-Enb	95.85±0.07	<b>96.80±0.09</b>
	Plato	88.90±0.51	<b>93.07±0.32</b>
	Aldatıcı Fonk.	79.57±0.38	<b>80.06±0.46</b>
	Kral Yolu	72.85±0.92	<b>81.56±0.95</b>
FH/t=100	Bir-Enb	95.46±0.12	<b>96.52±0.11</b>
	Plato	88.72±0.53	<b>92.56±0.31</b>
	Aldatıcı Fonk.	79.13±0.39	<b>79.29±0.49</b>
	Kral Yolu	72.21±0.77	<b>80.23±1.07</b>
FH/t=80	Bir-Enb	95.07±0.12	<b>95.88±0.17</b>
	Plato	87.20±0.47	<b>91.57±0.40</b>
	Aldatıcı Fonk.	78.04±0.34	<b>78.66±0.47</b>
	Kral Yolu	69.12±0.71	<b>79.47±0.85</b>
FH/t=50	Bir-Enb	92.87±0.19	<b>94.62±0.22</b>
	Plato	83.71±0.67	<b>90.04±0.46</b>
	Aldatıcı Fonk.	76.45±0.43	<b>76.49±0.49</b>
	Kral Yolu	66.52±0.83	<b>74.23±1.18</b>

Tablo 3'ten de görülebileceği üzere 0-1 kPSO, 0-1 PSO'dan daha başarılı sonuçlar elde etmiştir. Bu fark özellikle FH/t değerinin azaldığı testlerde daha belirgin hale gelmektedir. Bu da kuantum tabanlı yerel aramanın bu problemlerde başarılı sonuçlar üretebildiğini göstermektedir.

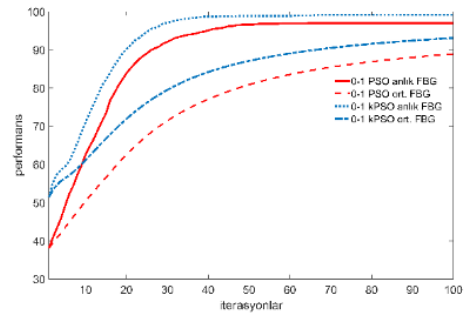
Dikkat edilecek olursa Aldatıcı Fonksiyonlardaki başarı başabaş olarak değerlendirilebilir. Ancak bu problemde az farkla da olsa 0-1 kPSO'nun bir derece önde olduğu söylenebilir. Benzer durum Bir-Enb problemi için de geçerlidir. Burada ise 0-1 kPSO'nun 0-1 PSO'ya göre Aldatıcı Fonksiyonlarda

olduğundan daha önde olduğu söylenebilir. Bu fark, Plato ve Kral Yolu problemlerinde daha belirgin bir şekilde göze çarpmakta ve bu problemlerde 0-1 kPSO'nun 0-1 PSO'ya üstünlüğü daha net gözlemlenebilmektedir.

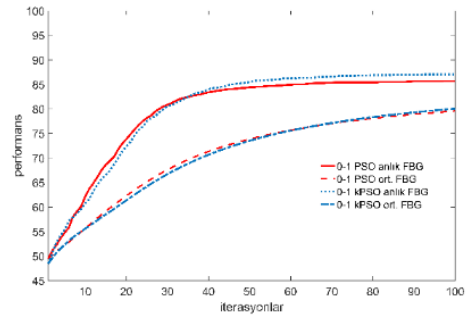
Test edilen algoritmaların yakınsama eğrileri ise Şekil 4'te verilmiştir.



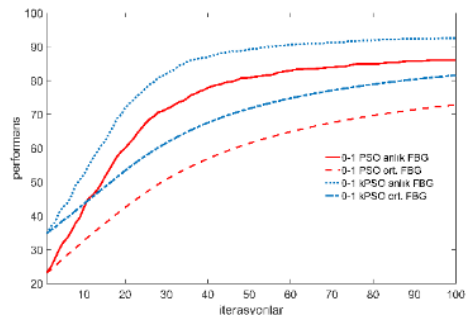
(a)



(b)



(c)



(d)

Şekil 4: 0-1 PSO ve 0-1 kPSO algoritmalarının. (a): Bir-Maks, (b)ç Plato, (c): Aldatıcı Fonksiyon, (d): Kral Yolu problemlerindeki yakınsama eğrileri.

Buradan da görülebileceği üzere kuantum tabanlı yerel arama ile güçlendirilmiş 0-1 kPSO eğrileri 0-1 PSO eğrilerinin üzerinde seyretmektedir. Bu durum, önerilen algoritmanın başarısını göstermektedir. Aldatıcı fonksiyonlardaki başabaş durumu burada da görülebilmektedir. Benzer şekilde Plato ve Kral Yolu problemlerindeki performans farkı bu görsellerden açıkça izlenebilmektedir.

Elde edilen bulgular 0-1 kPSO'nun sözü edilen problemlerde 0-1 PSO'ya göre üstünlüğüne işaret etmektedir. Bu durum özette popülasyon büyüklüğü arttırmak yerine fonksiyon hesaplama kapasitesinin daha verimli ve daha etkin bir şekilde kullanılmasının çözüm kalitesine katkı sağlayabileceğini göstermektedir. Bu da mevcut çalışmada kuantum tabanlı yerel arama yordamı tarafından gerçekleştirilmektedir. Ancak unutulmamalıdır ki 0-1 PSO'nun işleyişinde transfer fonksiyonuna ihtiyaç yoktur. Bu, 0-1 PSO'nun avantajı olarak görülebilir. Bununla birlikte kuantum tabanlı yerel arama sonrası 0-1 çözüm vektörlerine geçiş için 0-1 kPSO'da transfer fonksiyonu kullanılmıştır. Bu durum, işleyiş açısından dezavantaj olarak değerlendirilebilse de, çözüm kalitesine olumlu katkı sağlanabildiği ileri sürülebilir.

#### 4 Sonuçlar

Mevcut çalışmada 0-1 uzaylı problemlerde kullanılabilirlik üzere bir Parçacık Sürü Optimizasyonu (PSO) Algoritması geliştirilmiştir. Sözü edilen algoritma, transfer fonksiyonu kullanılmayıp bunun yerine sık kullanılan bir evrimsel operatör olan çarpazlama operatörünü kullanmaktadır.

Önerilen algoritma kuantum tabanlı bir yerel arama yordamı ile güçlendirilmiş ve yeni bir modifikasyon (0-1 kPSO) geliştirilmiştir. Böylece algoritma tarafından bulunan eniyi çözümün etrafında rastgele dağılmış olan kuantum parçacıkları olarak adlandırılan rassal dağıtık çözümler ile detaylı bir arama yapılabilmektedir.

Önerilen algoritmaların performansları, ilgili alanda kıyaslama problemi olarak sıkça kullanılan Bir-Enb, Plato, Aldatıcı ve Kral Yolu fonksiyonlarından oluşan bir 0-1 problem setinde test edilmiştir. Deneysel sonuçlar kuantum yerel arama yordamı ile güçlendirilmiş 0-1 kPSO'nun 0-1 uzaylı problemlerdeki etkinliğine işaret etmektedir. Geliştirilen yaklaşımların farklı problemler için uyarlaması, gelecek çalışma olarak düşünülmektedir.

#### 5 Kaynaklar

- [1] Goldberg DE. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Massachusetts, USA, Addison-Wesley, 1989.
- [2] Storn R, Price K. "Differential Evolution-A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces". International Computer Science Institute, Muenchen, Germany, Scientific Report TR-95-012, ICSI, 1995.
- [3] Eberhart RC, Kennedy J. "A new optimizer using particle swarm theory". *IEEE 1995 International Symposium on Micro Machine and Human Science*, Nagoya, Japan, 4-6 October 1995.
- [4] Karaboga D, Basturk B. "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm". *Journal of Global Optimization*, 39(3), 459-471, 2007.

- [5] Yang XS. *Firefly Algorithms for Multimodal Optimization*. Editors: Watanabe O, Zeugmann T. Lecture Notes in Computer Sciences. 169-178, Berlin, Germany, Springer Heidelberg, 2009.
- [6] Krishnanand KN, Ghose D. "Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions". *Swarm Intelligence*, 3(2), 87-124, 2009.
- [7] Yang XS. *A New Metaheuristic Bat-Inspired Algorithm*. Editors: González JR, Pelta DA, Cruz C, Terrazas G, Krasnogor N. Studies in Computational Intelligence. 65-74, Berlin, Germany, Springer Heidelberg, 2010.
- [8] Yang X-S. *Flower Pollination Algorithm for Global Optimization*. Editors: Durand-Lose J, Jonoska N. Lecture Notes in Computer Science. 240-249, Berlin, Germany, Springer Heidelberg, 2012.
- [9] Mirjalili S, Mirjalili SM, Lewis A. "Grey wolf optimizer". *Advances in Engineering Software*, 69, 46-61, 2014.
- [10] Wang GG, Deb S, Cui Z. "Monarch butterfly optimization". *Neural Computing and Applications*, in press, 2015.
- [11] Sun J, Feng B, Xu W. "Particle swarm optimization with particles having quantum behavior". *IEEE Congress on Evolutionary Computation*, Portland, OR, USA, 19-23 June, 2004.
- [12] Sun J, Xu W, Feng B. "A global search strategy of quantum-behaved particle swarm optimization". *IEEE Conference on Cybernetics and Intelligent Systems*, Singapore, Singapore, 1-3 December 2004.
- [13] Xi, M, Sun J, Xu W. "An improved quantum-behaved particle swarm optimization algorithm with weighted mean best position". *Applied Mathematics and Computation*, 205(2), 751-759, 2008.
- [14] Dong H, Yang X, Teng X, Sha Y. "A diversity reserved quantum particle swarm optimization algorithm for MMKP". *International Conference on Computer and Information Science*, Okayama, Japan, 26-29 June, 2016.
- [15] Banks A, Vincent J, Anyakoha C. "A review of particle swarm optimization. Part I: background and development". *Natural Computing*, 6(4), 467-484, 2007.
- [16] Banks A, Vincent J, Anyakoha C. "A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications". *Natural Computing*, 7(1), 109-124, 2008.
- [17] Kennedy J, Eberhart RC. "A discrete binary version of the particle swarm algorithm". *IEEE 1997 International Conference on Systems, Man, and Cybernetics*, Orlando, FL, USA, 12-15 October 1997.
- [18] Ozsoydan FB, Sipahioglu A. "Heuristic solution approaches for the cumulative capacitated vehicle routing problem". *Optimization*, 62(10), 1321-1340, 2013.
- [19] Pan QK, Tasgetiren MF, Liang YC. "A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem". *Computers & Operations Research*, 35(9), 2807-2839, 2008.
- [20] Park JB, Jeong YW, Shin JR, Lee KY, Kim JH. "A hybrid particle swarm optimization employing crossover operation for economic dispatch problems with valve-point effects". *IEEE 2007 International Conference on Intelligent Systems Applications to Power Systems*, Toki Messe, Niigata, Japan, 5-8 November 2007.
- [21] Tseng CT, Liao CJ. "A discrete particle swarm optimization for lot-streaming flowshop scheduling problem". *European Journal of Operational Research*, 191(2), 360-373, 2008.

- [22] Marinakis Y, Marinaki M, Dounias G. "A hybrid particle swarm optimization algorithm for the vehicle routing problem". *Engineering Applications of Artificial Intelligence*, 23(4), 463-472, 2010.
- [23] Marinakis Y, Marinaki M. "A hybrid genetic-particle swarm optimization algorithm for the vehicle routing problem". *Expert Systems with Applications*, 37(2), 1446-1455, 2010.
- [24] Lian Z, Gu X, Jiao B. "A similar particle swarm optimization algorithm for permutation flowshop scheduling to minimize makespan". *Applied Mathematics and Computation*, 175(1), 773-785, 2006.
- [25] Zhang G, Dou M, Wang S. "Hybrid genetic algorithm with particle swarm optimization technique". *IEEE 2009 International Conference on Computational Intelligence and Security*, Beijing, China, 11-14 December 2009.
- [26] Su D, Xu W, Sun J. "Quantum-behaved particle swarm optimization with crossover operator". *IEEE 2009 International Conference on Wireless Networks and Information Systems*, Shanghai, China, 28-29 December 2009.
- [27] Kennedy J. *Particle Swarm Optimization*. Editors: Sammut C, Webb GI. Encyclopedia of Machine Learning. 760-766, US, Springer, 2011.
- [28] Blackwell TM, Bentley PJ. "Dynamic search with charged swarms". *GECCO 2002 Genetic and Evolutionary Computation Conference*, New York, USA, 9-13 July, 2002.
- [29] Blackwell T, Branke J. *Multi-Swarm Optimization in Dynamic Environments*. Editors: Raidl G, Cagnoni S, Branke J, Corne D, Drechsler R, Jin Y, Johnson C, Machado P, Marchiori E, Rothlauf F, Smith G, Squillero G. Applications of Evolutionary Computing. 489-500, Heidelberg, Berlin, Germany, Springer, 2004.
- [30] Blackwell T, Branke J, Li X. *Particle swarms for dynamic optimization problems*. Editors: Blum C, Merkle D. Swarm Intelligence. 193-217, Heidelberg, Berlin, Germany, Springer, 2008.
- [31] Wang H, Wang D, Yang S. "A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems." *Soft Computing*, 13(8-9), 763-780, 2009.
- [32] Calderín JF, Masegosa AD, Pelta DA. "Algorithm portfolio based scheme for dynamic optimization problems". *International Journal of Computational Intelligence Systems*, 8(4), 667-689, 2015.