T.C
OSMANGAZİ ÜNİVERSİTESİ
ENDÜSTRİ MÜHENDİSLİĞİ BÖLÜMÜ

SOSYAL ZEKA TABANLI OPTİMİZASYON YAKLAŞIMI ILE MODEL ÇÖZÜM UYGULAMALARI

Bilgehan DOĞAÇ
Onur KURT
Demet SOLA

Danışman: Yrd. Doç. Muzaffer KAPANOĞLU

ESKİŞEHİR 2006

ÖNSÖZ VE TEŞEKKÜR

Doğadaki canlıların geçirdikleri evrim süreci içerisinde geliştirdikleri ve bu süreç içerisinde nesillerinin tükenmesine engel olup ayakta kalmalarını sağlayan sosyal zekalarını, Endüstri Mühendisliğinin en önemli silahlarından olan "Optimizasyon" alanına uyarlaması üzerinde durduğumuz bu çalışmanın, ele aldığımız konularda kayda değer bir ilerleme kaydedebilmiş olması en büyük beklentimizdir.

"Yerel Sorunlara Endüstri Mühendisliği Yaklaşımları ve Sosyal Sorumluluk Projeleri" kapsamında ele alındığında, çalışma iki açıdan bu başlığa tam uyum sağlamaktadır. Birincisi açıkça üzerinde durduğumuz "Üretim Planlaması Optimizasyonu"'nun yerel ekonomimiz üzerindeki etkilerinin araştırılması iken, bir diğer açı ise sosyal zekanın avantajlarını ortaya koyup, insanlar arasında uygulanabilirliğinin düşündürülmesidir.

Umarız çalışmamız gerek Endüstri Mühendisliğinin bir adım daha ileremesine, gerek ise yerel ekonomimizin güçlenmesine katkıda bulunabilecek şekilde amacına ulaşır.

Bizlere çalışmamız boyunca yol gösteren ve destek olan sayın danışman hocamız Yrd.Doç. Muzaffer KAPANOĞLU'na, eğitimimiz boyunca gerekli bilgi birikimini sağlamamız için çaba sarfeden değerli hocalarımıza, ve tabi varlığımızın mimarları ailelerimize teşekkürü borç biliriz.

ÖZET

Bu çalışma, hayvanlarda gözlenen sosyal zekanın bir optimizasyon aracı olarak ele alınmasını, çeşitli optimizasyon problemlerini çözebilecek şekilde uyarlanmasını ve sonuç olarak ortaya çıkan ürünün (yazılımın) işletmelerimize kazandırabileceği iyileştirmenin boyutunun analiz edilmesini içermektedir. Başka bir açıdan bakıldığında ise bu çalışma, sosyal zekanın ne kadar büyük bir potansiyel yarattığını, ve birbirleriyle etkileşim içerisinde olan bireylerin oluşturduğu bu sosyal modelin insanlar üzerine uygulanabilirliğini düşündürmesi açısından önemlidir.

Öncelikle doğadaki sosyal zeka modelleri tanıtılmış, daha sonra sosyal zekanın algoritmik yapısına değinilerek bu yapının optimizasyon alanında nasıl kullanılabileceği anlatılmış, kısıtlı model çözümü için geliştirdiğimiz uyarlamalar ve yazılımımız açıklanmıştır. Son olarak sosyal zeka tabanlı optimizasyon tekniğini kullanarak işletmelerimizin ve Endüstri Mühendisliğinin önemli problemlerinden biri olan "Dinamik Talep Altında Sipariş Zamanı ve Miktarı Optimizasyonu" problemini nasıl çözdüğümüz anlatılmış, bunun yerel ekonomimize ne gibi katkılarda bulunabileceğine değinilmiştir.

Anahtar sözcükler: Sosyal Zeka Tabanlı Optimizasyon, Penaltı Yöntemi, Huni Etkisi, Dinamik Talep Altında Sipariş Zamanı ve Miktarı Optimizasyonu.

ABSTRACT

This study includes swarm intelligence used to solve various optimization problems and analysis of winnings of developed software (as a result) for our enterprises. On the other hand this study is about what a big potential swarm intelligence creates and feasibility of adapting, this social model of individuals in interaction with each other, on humans.

First of all, we defined swarm intelligence models in nature. After that we touched on algoritmik structure of swarm intelligence and told how to use this structure on optimization problems, our adaptations to solve constricted model and our software. At last, we told how we solved "Lotsize and Order Time Problems with Dynamic Demand", which is an important problem of industrial engineering and our enterprises, by using Particle Swarm Optimization technique and gains of our local economy.

Key Words: Particle Swarm Optimization, Penalty Method, Funnel Effect, Lotsize and Order Time Problems with Dynamic Demand.

İÇİNDEKİLER

Sayfa No

Önsöz ve Teşekkür ..... **I**

Özet / Abstract ..... **II**

İçindekiler ..... **IV**

Şekiller Dizini ..... **V**

1. GİRİŞ **6**

2. SOSYAL ZEKA **8**

2.1. Sürü (Grup, kültür) **8**

2.2. Doğadan Sosyal Zeka Örnekleri **9**
2.2.1. Karınca kolonileri 9
2.2.2. Kuş sürüleri 9
2.2.3. Avını çevreleyen kurt sürüleri 10

3. SOSYAL ZEKA TABANLI OPTİMİZASYON (SZTO) **11**

3.1. Günümüzdeki Uygulama Alanları **12**

3.2. Klasik Algoritma **12**

3.3. SZTO'nun Kısıtlı Modellere Uygulanabilirliği ve Karşılaşılan Güçlükler **15**

3.4. Geliştirdiğimiz Teknikler **17**
3.4.1. Hassasiyetin kademeli olarak arttırılması 17
3.4.2. Penaltı Yöntemi 18
3.4.3. Huni Etkisi 21

3.5. Genel Amaçlı Model Çözüm Uygulaması **26**

4. YEREL PROBLEMLERE SZTO YAKLAŞIMI **29**

4.1. İşletmelerimizde Optimizasyon Açıkları **29**

4.2. Üretim Planlamasında Optimizasyon Eksikliği **29**
4.2.1. Problemin tanımı 29
4.2.2. Dinamik talep altında sipariş zamanı ve miktarı optimizasyonu uygulaması 30

4.3. Üretim Planlamasında Optimizasyonun Yerel Ekonomi Üzerindeki Etkisi **35**

5. KAYNAKÇA **37**

ŞEKİLLER DİZİNİ

Şekil 1- Çözüm kümesi 22
Şekil 2 – Huni etkisi 23
Şekil 3 – Seçim mantığı 25
Şekil 4 – Bir model ve çözümü 27
Şekil 5 – Değişkenlerin grafiği 27
Şekil 6 – Amaç fonksiyonu grafiği 28
Şekil 7 – Dinamik talep altında sipariş zamanlarının ve miktarlarının bulunması 32
Şekil 8 - 4 siparişe kadar toplam stokta bulundurma maliyetinin değişimi 33
Şekil 9 - 4 Siparişe kadar toplam maliyetin değişimi 33
Şekil 10 - Sistemin amaç fonksiyonu olan toplam stokta bulundurma maliyetinin ilgili iterasyonlara göre değişimi 34

1. GİRİŞ

Günümüz iş dünyasında hala el ile ve üstünkörü yapılmakta olan optimizasyon konusu, aslında üzerinde çok önemle durulması gereken mühendislik konularının başında gelmektedir. Özellikle Endüstri Mühendislerinin uzmanlık alanı olan optimizasyon teknikleri, ya ne kadar yarar sağlayabileceğinin tam kavranamaması ile, ya da etkin bir şekilde uygulayabilecek kalifiye personelin bulunmaması nedeni ile birçok işletmede uygulanmamaktadır. Tesis planlamasından, üretim planlamasına kadar birçok alanda yeri olan optimizasyon teknikleri, etkin bir şekilde uygulandığı taktirde işletmeye hayal bile edilemeyecek iyileştirmeler sağlayabilmektedir.

Optmizasyon yapmanın çeşitli yolları vardır. Problem bir optimizasyon modeli formuna dönüştürüldükten sonra optimum değerler elle veya çeşitli bilgisayar yazılımlarıyla yapılabilir. Optimizasyon problemleri çok çeşitli olabildiği için, modelleri çözmekte bir çok değişik algoritmadan faydalanılmaktadır. Günümüzde bu modelleri çözmek için Excel, Matlab gibi genel amaçlı programların yanında yalnızca optimizasyon için geliştirilmiş Lingo, Lindo, Cplex gibi programlar da bilgisayarların gelişmiş işlemci gücü sayesinde en zorlu problemlere bile çözümler üretebilmektedirler. Ancak bu programların da çözemedği bazı problemler vardır ki bunları çözmekte klasik yöneylem araştırması teknikleri yetersiz kalabilmektedir.

Bu proje, klasik yöneylem araştırması algoritmalarından farklı olarak, doğadaki canlı türlerinin organizasyon yapılarından esinlenilerek geliştirilen yapay zeka tekniklerinin özel optimizasyon problemlerine uyarlanabilmesi açısından önem taşımaktadır. Zira ele alınan “Dinamik Talep Altında Sipariş Miktarı ve Zamanı” problemini simplex algoritması gibi klasik teknikler çözmekte yetersiz kalmaktadırlar. Özellikle ele aldığımız “Sosyal Zeka Tabanlı Optimizasyon” (SZTO) tekniğinin kısıtlı modellere uyarlamasının örneklerine çok nadir raslanmaktadır.

SZTO tekniğini kısıtlı modeller üzerinde denediğimiz ve özellikle ele aldığımız “Dinamik Talep Altında Sipariş Zamanı ve Miktarı Optimizasyonu” problemlerinde oldukça olumlu sonuçlar elde ettiğimiz bu projenin, henüz bakir kabul edilen bu alanda

bir ilerleme kaydedebilmiş olmasını, ve yaklaşımlarımızın getirdiği olumlu sonuçların, işletmelerimizin optimizasyon açıklarını kapatabilecek düzeyde olmasını umuyoruz.

2. SOSYAL ZEKA

Henüz literatürlerde yeni bir paradigma olan sosyal zeka, ülkemizde de yeni araştırmalara konu olmakta ve bu alanda yeni gelişmeler kaydedilmektedir. Çalışmamızda, İngilizce kaynaklarda "swarm" olarak geçen, Türkçe kaynaklarda da "sürü" olarak çevrilmiş olan terim, anlamı daha iyi karşılayabilmek ve çözümlemek adına "sosyal" olarak değerlendirilmiştir.

İlkel bir sosyalleşmeden yola çıkarak, çeşitli grupların araştırma yeteneğini simüle etmemizden dolayı, "sürü zakası" yerine "sosyal zeka" adlandırmasını çalışmamız içerisinde tercih etmekteyiz.

2.1. Sürü (Grup, kültür)

Son günlerde biyologlar ve bilgisayar bilimcileri doğadaki sinerjik zeka örnekleri üzerindeki araştırmalarını sıklıklaştırmışlardır. Nasıl organize olduklarını, ne çeşit sosyal yapıları olduğunu ve amaçlarını nasıl gerçekleştirdiklerini, birbirlerinden ne ölçüde etkilendiklerini araştıma konularına dahil etmişlerdir. Aynı hayvan türlerinin bireylerinin bir araya gelerek oluşturdukları bu kümelere "sürü", bu kümelerin bireylerinin birbirlerinden etkileşimleri sonucunda ortaya çıkan zeka potansiyeline ise "sosyal zeka" diyoruz.

Sürülerde N adet temsilci bir amaca yönelik davranışı gerçekleştirmek ve hedefe ulaşmak için birlikte çalışmaktadır. Kolaylıkla gözlenebilen bu "kollektif zeka" temsilciler arasında sık tekrarlanan davranışlardan doğar. Temsilciler faaliyetlerini idare etmek için basit bireysel kurallar kullanır ve gurubun kalan kısmıyla etkileşim yolu ile sürü amaçlarına ulaşır. Grup faaliyetlerinin toplamından bir çeşit kendini örgütleme doğar.

2.2. Doğadan Sosyal Zeka Örnekleri

2.2.1. Karınca kolonileri

Yeryüzünde en kalabalık nüfusa sahip olan canlılar, karıncalardır. Her yeni doğan 40 insana karşılık, 700 milyon karınca dünyaya gelir. Ve bu canlılar hakkında öğrenebileceğimiz çok fazla bilgi vardır.

Böcek türlerinin en "sosyal"lerinden biri olan karıncalar, son derece iyi "örgütlenmiş" bir düzen içinde, "koloniler" denen topluluklar halinde yaşarlar. Örgütlenmeleri öyle gelişmiş bir düzen içindedir ki, bu açıdan insanlarınkine benzer bir uygarlığa sahip oldukları bile söylenebilir. Karıncalar kendileri açısından en ideal olan sosyal sistemi milyonlarca sene öncesinden günümüze kadar hiçbir aksaklığa meydan vermeden sürdüregelmişlerdir.

Bir milyon tanesinin sinir hücrelerinin toplamı ancak 20 gram olan bu canlıların bir araya gelerek oluşturdukları bu sosyal zeka örneği yıllardır çeşitli bilim dallarından insanlar tarafından incelenmektedir. Ve günümüzde “Karınca Kolonisi Optimizasyonu” olarak bilinen bir yapay zeka tekniği, karıncaların gıda arama tekniklerinin incelenmesi sonucu ortaya çıkmıştır.

2.2.2. Kuş sürüleri

Kuş sürüleri, basit bir sürü algoritması izleyerek, birlikte durur, dönüşleri koordine eder ve birbirlerinden sakınırlar. Benzer problemler hava trafik kontrolünde ve gemi konvoylarında görülmekte ve benzer metodlar kullanılarak bu problemler çözülebilmektedir. Kuş sürülerinin bu koordine davranışları, her kuşun kendisine en yakın kuştan belli bir uzaklıkta ve etrafında kuşların hızında uçmayı başarmaya çalıştığı varsayımıyla açıklanabilmektedir. Sürü, her varlığın bireysel faaliyetlerinin eşzamanlı olarak yanıt gördüğü ve bütünü değiştirdiği, kendini zorlayan bir yapıdır. Aslında her kuş yalnızca en yakınında bulunan eşinin hareketlerini hissetmektedir. Ancak onun bu hareketlere tepkileri diğerlerine yayılmakta ve böylece sistem bir bütün olarak grupsal bir koordinasyon sergilemektedir.

2.2.3. Avını çevreleyen kurt sürüleri

Doğadaki başka bir sosyal zeka örneğini kurtlar avlanırlarken sergilemektedirler. Öyle ki, bir geyik yakalamak için kurt sürüsünün koordine bir şekilde hareket etmesi ve avını çevrelemesi gerekmektedir. Onların koordine davranışı, uzun vadeli iletişim mekanizmalarının veya karmaşık zekice stratejilerin varlığı varsayımı olmaksızın sosyal zeka ile açıklanabilir. Kurtlar basitçe, diğer kurtlarla uygun uzaklığı korumaya çalışarak avlanırlar. Bu basit stratejinin simülasyonları, hareket edebildiği kadar hızlı en yakın kurttan kaçan bir geyikle gösterir ki avını çevrelemek uygulanabilir ve başarılı bir çözümdür.

3. SOSYAL ZEKA TABANLI OPTİMİZASYON (SZTO)

Dr Eberhart ve Dr Kennedy tarafından 1995'de, kuş ve balık sürülerinin sosyal davranışlarından esinlenilerek geliştirilen SZTO, popülasyon temelli, stokastik bir optimizasyon tekniğidir.

SZTO, genetik algoritma (GA) gibi evrimsel hesaplama teknikleriyle birçok benzerlik taşımaktadır. Genetik algoritmadan farklı olarak, SZTO'de mutasyon ve genetik değişim (crossover) gibi evrim işleçleri bulunmaz. SZTO'de parçacık adı verilen potansiyel çözümler, problem uzayı içerisinde güncel optimal parçaları takip ederek uçuşmaktadır.

Her parçacık problem uzayı üzerindeki koordinatlarını önceden ulaştığı en iyi çözümle ilişkilendirilmiş olarak kaydetmektedir (En iyi çözüm değeri ayrıca kaydedilir.) . Bu değer bireysel en iyi olarak adlandırılır. Diğer bir "en iyi" değer sosyal zeka eniyileyicisi tarafından izi sürülen parçacığın komşuluğunda ulaşılan en iyi değerdir. Bu konum bireysel en iyi olarak adlandırılır. Bir parçacık bütün popülasyonu kendi topolojik komşuluğu olarak alırsa, en iyi değer grupsal en iyi olarak adlandırılır.

SZTO konsepti, her aşamada, bireysel en iyi ve grupsal en iyi konumlarına doğru hızın artmasından oluşmaktadır. İvme rastsal bir terim tarafından ağırlandırılmaktadır.

Geçmiş yıllarda, SZTO birçok uygulama alanı ve araştırmaya başarıyla uygulanmış; diğer metotlarla karşılaştırıldığında, daha hızlı ve daha ucuz olan SZTO'nun daha iyi sonuçlar verdiği kanıtlanmıştır.

SZTO'yu çekici kılan diğer bir unsur düzenlenecek parametre sayısının azlığıdır. SZTO geniş bir uygulama alanına sahip olduğu gibi spesifik gereksinimlere cevap veren spesifik uygulama alanlarında da kullanılabilir.

3.1. Günümüzdeki Uygulama Alanları

SZTO, mantığının basitliğinin sağladığı avantaj ile birçok farklı yapıya uyarlanabilmektedir. Günümüzde farklı alanlarda SZTO yaklaşımlarının denendiği uygulamaları görmekteyiz. Bu uygulamalar henüz çözüm getirilememiş problemlerin uyarlamaları olmaktan çok, SZTO'nin geliştirilmesine yönelik yapılan çalışmaların ne ölçüde etkili olduğunu göstermek için hazırlanmış uygulamalardır. Fakat bununla birlikte, yalnızca bazı spesifik problemlerin çözümüne yönelik SZTO uygulamalarınada rastlanmaktadır. 1999 yılında Rio de Janeiro kentinde gerçekleştirilen, International Conference on Intelligent System Applications to Power Systems (ISAP'99), isimli konferansta tanıtılan bir çalışma (Yoshida et. al., 1999), "Voltajda Durağanlık Sağlayan Güç ve Voltaj Kontrolü İçin Sosyal Zeka Tabanlı Optimizasyon" başlığı altında tanıtılmıştır. Bu çalışmanın en önemli yanlarından birisi, ele alınan problemin kısıtlı bir non-linear model olmasıdır. Geliştirilen yaklaşım ise, bizim de kullanmış olduğumuz penaltı yöntemi ile yakın benzerlik göstermekte, temelde aynı mantığa dayanmaktadır.

3.2. Klasik Algoritma

SZTO n parçacıktan oluşan bir bütünün bir elemanının kendi eniyisi ve grupsal eniyisinden etkilenmesi ile grupta d boyutta hareket etmesinden oluşur.

Amaç fonksiyonunun herbir boyutu parçacığın d düzlemde gezinmesiyle elde edilen değerin minumum ya da maksimum yapılmasıyla elde edilir.

Örnek vermek gerekirse; n. parçacık 2 boyutlu bir düzlemde hareketini x ve y koordinatlarında gezerek sağlar. Her bulunduğu nokta amaç fonksiyonunu bir değere taşır ki amaca göre bu değer etkin veya değildir.

Sosyal yapının içerdiği herbir parçacık, kendi en iyisini hafızasında tutarak bunu hızında bir etmen olarak kullanır. Bireysel en iyisinden ne kadar uzaksa o noktaya doğru hızında bir artış gösterir. Bu davranışı aynı şekilde grubun en iyisine göre de gerçekleştirmektedir. Yani yüzeyi tararken grubun en iyisinden etkilenerek hızını ayarlama durumunda kalır. Grupsal eniyiden ne kadar uzaksa hızındaki ve dolayısıyla

yönündeki değişiklik o ölçüde çok olur. Yani sürü içerisindeki bireyler aşağıdaki bilgileri kullanarak hareketlerini değiştirme eğilimi gösterirler: [11],[15]

-Geçerli posizyonu (x, y)
-Geçerli hızı vx, vy
-Bireysel eniyi pozisyonu (pbestx, pbesty)
-Grupsal eniyi pozisyonu (gbestx, gbesty)

Her bir bireyin hızı aşağıdaki formulle değiştirilir:

$$V_i^{k+1} = X(w * V_i^k + C_1 * (Rnd) * (pBest_i - x_i) + C_2 * (Rnd) * (gBest_i - x_i))$$

V_i^k: i. bireyin k. iterasyondaki hızı
x_i : i. bireyin pozisyonu
w : Eylemsizlik fonksiyonu
C_i: Eylemsizlik faktörü
Rnd: Türetilen rassal sayı
$pbest_i$: i. Bireyin eniyi pozisyonu
gbest: Grup eneyisi pozisyonu
X: Daraltma faktörü (Constriction factor)

Hız birim yer değiştirme miktarı olduğundan, her birim zamana SZTO içerisinde bir iterasyon demekteyiz. Bir iterasyonda, parçacıkların grubun eniyisinden ve kendi en iyilerinden ne kadar uzakta olduklarına bağlı olarak hızları, dolayısıyla konumları, ve eylemsizlik değerleri gibi temel özelliklerinde değişimler olmaktadır. Parçacık bu değişimlere göre canlandırılmaya çalışılmaktadır.

Bu canlandırılmanın yapılabilmesi, her iterasyonda parçacığın hızında meydana gelen değişimlerle sağlanmaktadır. Parçacığın grupsal eniyi ile bireysel eniyiden ne ölçüde etkileneceğini C_i faktörleri belirler. C faktörlerini 0.5 olarak düşünürsek ve ortak paranteze alırsak eğer, iki farkın ortalamasını aldığımızı farkedebiliriz. Yani C

faktörlerinin kullanılma amaçları, parçacıkların etkilendikleri konumlar arasında bir ağırlıklandırma gerçekleştirmektir.

Denklem içerisindeki eylemsizlik değeri her iterasyonda değiştmektedir. Bu değişiklik, eylemsizlik fonksiyonuna göre belirlenen değerden minimum değerine doğru düşme mantığına dayanmaktadır. Amacı ise türetilen hızın ileriki iterasyonlarda küçültülerek yakınsamasının sağlanabilmesi, böylece de daha yakın sonuçların bulunabilmesinin sağlanabilmesi temeline dayanır.

Eylemsizlik fonksiyonu aşağıdaki gibi bulunur:

$$w = w_{max} - \frac{w_{max} - w_{min}}{iter_{min}}$$

W_{max}= İlk eylemsizlik kuvveti
W_{min}= Minimum eylemsizlik kuvveti
$İter_{max}$=Maksimum iterasyon sayısı

Meydana gelen yer değiştirme, birim iterasyonda sahip olunan konumlara eklenerek bir sonraki konumlar bulunur.

Hızlar ile parçacıkların pozisyonları,

$$x_i^k = x_i + vx_i^k$$

'şeklinde değiştirilir. Aynı işlem her bir boyut için tekrarlanır.

C faktörlerini ve eylemsizlik faktörünü Shi ve Eberhart, yaptıkları araştırmalar ve deneyler suretinde, probleme göre değişmediğini ve sabit olarak c_j=2, w_{max}=0.9 ve w_{min}=0.4 olarak bulmuşlardır. Özellikle uygulamalarımızda bu buluguları sabit olarak kabul edip çalışmalarımızı sürüdürdük..[2],[12]

Daraltma faktörü (X) temel SZTO denkleminde bir farklılık denklemidir. Bundan dolayı arama işlemi özdeğer analizi ile analiz edilir. Bu analizde daraltma faktörü özdeğer analizini kullanır ve aşağıdaki özellikleri kontrol eder:

a) Sistemin gerçek değer alanından uzaklaşmaması ve sonunda yakınsaması,
b) Sistemin farklı alanları da verimli bir şekilde araştırması.

Daraltma faktörü aşağıdaki gibi bulunur:

$$X = \frac{2}{\left|2 - Q - \sqrt{Q^2 - 4Q}\right|}; \text{ eğer } C_1 + C_2 > 4$$

$$Q = C_1 + C_2$$

Daraltma faktörünün indirici etkisi daha göze çarpıcıdır ve parçacıkların amaç değerine daha fazla yakınsayabilmesini sağlayabilir. Daraltma faktörü yaklaşımı, herbir parçacığın yayılımını en iyi değere odaklanarak düşürerek en yüksek değerde amaç değerleri elde etmeyi sağlar. Bir başka değişle eylemsizlik faktörünün bir alternatifi veya dengeleyicisi rolündedir. Biz bu çalışmada daraltma faktörünü hız denklemimizde kullanmadık.

3.3. SZTO'nun Kısıtlı Modellere Uygulanabilirliği ve Karşılaşılan Güçlükler

SZTO belirli sınırlar içerisinde herhangi bir fonksiyonun en büyük veya en küçük değerini kolayca saptayabilmektedir. Ancak incelenen noktaların belirli kısıt denklemlerini sağlaması istendiğinde durum biraz güçleşmektedir. SZTO açısından kısıtlı bir model ile tek bir fonksiyon arasındaki fark, seçilecek herhangi bir noktanın kısıt denklemleri ile çelişme olasılığının oldukça yüksek oluşundan kaynaklanmaktadır. Klasik SZTO algoritması üzerinde öncelikle yapılması gereken en önemli değişiklik; kısıt denklemlerinden herhangi biri ile çelişen bir noktanın, grupsal en iyi veya bireysel en iyi olarak atanmasına izin verilmemesi olacaktır. Yalnızca bu değişiklik bile SZTO'nun çok basit bazı modeller üzerinde başarılı sonuçlar elde etmesini

sağlayabilmektedir. Ancak sözü geçen basit modellerin, verilen sınırlar içerisinde geniş bir çözüm kümesine sahip olması, ve dolayısı ile eşitlik kısıtı içermemesi gerekmektedir. Örneğin,

$$f(x) = 3 * X1 - 2 * X2$$

fonksiyonunu ele alalım, ve bu fonksiyonu X1 = 0,1,2..30 ve X2 = 0,1,2..30 sınırları içinde inceleyelim. Kısıt olmadığı durumda fonksiyonun bu sınırlar içerisindeki maksimum değeri X1 = 30 ve X2 = 0 noktasında bulunur ve 90'dır. Bu fonksiyonun maksimum değerini

$$X1 + X2 < 15$$

kısıtını sağlayan noktalar arasından seçmek istediğimizde, eski çözüm kümemiz içerisinden bu denklemi sağlamayan noktaları bir şekilde çıkarmamız gerekir. Böyle bir durumda, yukarıda bahsedilen küçük değişiklik SZTO'nun aradığımız noktayı yakalamasını sağlamaktadır. Ancak kısıtımız

$$X1 + X2 < 15$$

yerine

$$X1 + X2 = 15$$

olursa durum değişir. Bu durumda çözüm kümemizden

$$X1 + X2 = 15$$

doğrusunun üzerindeki noktalar ile birlikte altındaki noktaları da çıkarmamız gerekir. Çözüm kümemizin sürekli olduğu gözönüne alınırsa, ataması rassallık unsuru içeren bir algoritma tarafından yapılan herhangi bir noktanın bu doğru üzerinde olma olasılığı çok küçüktür. Eğer X1'in değeri 4,9999'e ulaşmış ise, X2'nin değerininde 10,1111 değerine

ulaşmış olması gerekir. Bu olasılığı yükseltmek için değerler virgülden sonra dört yerine bir veya iki haneye yuvarlatılabilir. Ancak eğer onbinde birlik değişimler karar verici için önem arz ediyorsa, değerlerin yuvarlanması sonuçların hassasiyetini düşürecektir. Bu sorunun bir diğer etkisi ise ilk iterasyonda noktaların hiçbirinin kısıtlara uymaması ve dolayısı ile grupsal en iyinin bulunamamasıdır. Eğer ilk iterasyonda bir grupsal en iyi saptanabilirse daha sonra algoritma bu nokta çevresinde daha iyi noktalara ulaşabilir. Fakat ilk iterasyonda bireysel en iyiler ve grupsal en iyi saptanamaz ise SZTO algoritması etkinliğini yitirir.

Diğer bir yöntem olan penaltı yöntemi için aynı durum geçerli değildir. Herhangi bir noktanın grupsal veya bireysel en iyi olabileceği göz önüne alındığında cezaların ileriki iterasyonlar doğrultusunda, kısıtlara uyma ölçüleri oranında azaltılarak kısıtlara uyması zorunlu hale getirilmiştir. Bunun açıklaması ileriki başlıklarda "penaltı yöntemi" başlığı altında incelenecektir.

3.4. Geliştirdiğimiz Teknikler

Bölüm 3.3'de bahsedilen güçlüklerle başedebilmek için bazı teknikler geliştirilmiş ve klasik SZTO algeritmasına entegre edilmiştir. Böylelikle klasik algoritmanın çözemediği bazı modeller tamamen çözülebilir hale gelmiş, bazılarında ise yakın sonuçlar alınabilmiştir.

3.4.1. Hassasiyetin kademeli olarak arttırılması

Modelin içerdiği herbir kısıt çözüm kümesinin biraz daha küçülmesine neden olmaktadır. Kısıtlar gözardı edildiğinde çözüm kümesi belirlenen sınırlar dahilindeki bütün noktaları kapsar. Ancak her bir kısıt bu noktaların bir kısmının çözüm kümesi dışında kalmasına neden olur. Örneğin X1 değişkeni 0 ile 30 aralığında incelenirken, X1 <= 15 kısıtı çözüm kümesinin 0 ile 15 aralığında sınırlanmasına neden olur. Eğer binde birler mertebesindeki noktalar değerlendiriliyor ise bu durumda 15,0001 ile 30 aralığında değerlendirilecek 149.999 adet kısıta uymayan nokta var demektir. Aynı şekilde çözüm kümesinde ise 160.000 parça değerlendirilmeyi beklemektedir. Çözümde kullanılan parçacık sayısının 100 olacağını varsaydığımızda bu rakamlar parçacıkların sınırlı iterasyonda tarayabileceğinin çok çok üzeridedir. Eğer yalnızca tamsayılar ile

ilgilenecek olursak bu durumda 0 ile 15 aralığındaki 16 nokta çözüm kümemizi ve 16 ile 30 arasındaki 15 nokta ise kısıta uymayan noktaları temsil eder. Dolayısıyla bu noktalar arasında kısıta uyan noktaların yakalanma olasılığı oldukça yüksektir. Tabi noktalarla birlikte kısıtlarında tamsayıya yuvarlanmış olmaları gerekir.

Tüm bu hususlar dikkate alındığında hassasiyeti ilk iterasyonlarda düşük tutmak, daha sonra gittikçe arttırmak SZTO'nun bu yöntemde verimliliğini arttıracaktır. İlk iterasyonlarda tamsayılar arasından en iyisi seçilecek, daha sonra onlar mertebesine inilerek diğer noktalar araştırılacak ve bu süreç son iterasyonlara gelindiğinde arzu edilen hassasiyete ulaşacaktır. Böylelikle ilk iterasyonda en iyi noktaların saptanabilme olasılığı yükselecek ve parçacık sayısının aşırı arttırılmasına ve dolayısı ile çözüm süresinin uzatılmasına ihtiyaç duyulmayacaktır.

Bu teknik yazılımımızın genel model çözme algoritmasına uygulanmış ve olumlu sonuçlar elde edilmiştir.

3.4.2. Penaltı Yöntemi

SZTO'nun kısıtlı modelleri çözebilmesi için geliştirdiğimiz tekniklerden bir tanesi de penaltı model çözüm yöntemidir.

Ana hatlarıyla yöntem, belirtilen kısıtlar doğrultusunda çözüm kümesinde kalmalarını sağlayacak cezalarla parçacıkları yönlendirmek ve karar verebilmelerini sağlamaktır. Bu etki, amaç fonksiyonuna kısıtları sağlama becerisine göre minimum bir model ise ekleme yaparak, maksimum ise de eksiltme yaparak gerçekleşir.

Çözüm uzayında rastsal dağılan parçacıklar, amaç fonksiyonlarını modelin içeriğine göre minimum ya da maximum yapma eğilimindedirler. Bunun sağlanabilmesi için normal SZTO mantığında olduğu gibi, herbir parçacığın hızı grubun en iyisi ve bireysel en iyinin pozisyonundan etkilenerek değişir. Bu değişim ölçüsü ise az önce de bahsettiğimiz gibi, amaç fonksiyounun amaca ne kadar yakın olduğuna bağlıdır.

Penaltı yönteminin uygulandığı algoritmanın başlangıcında herbir parçacığın belirlenen kısıtları ne ölçüde sağladığı belirlenerek amaç fonksiyonuna yansıtılır.

Aşağıdaki örnek üzerinden gidecek olursak:

Min X1 + X2 + X3 ;

X1 + X2 > 5
X2 + X3 = 4

‘ X1 + X2 + X3 minimum yapılmaya çalışılmaktadır. Bunu yaparken sağlanması gerekli iki kısıtımız mevcuttur.

İterasyonlara geçmeden önce rastsal olarak çözüm uzayı üzerinde konumlandırılarak yaratılmış parçacıklar, kısıtları sağlayıp sağlamamalarına göre araştırılmaktadır. Bunun yapılması simplex algoritmasındakine benzer bir dönüştürme işlemi ile gerçekleştirilir.

Min X1 + X2 + X3 ;

X1 + X2 + S1 = 5
X2 + X3 + S2 = 4

Noktanın koordinatları kısıtları sağlamıyorsa ne kadar sağlamadığının tanımlanabilmesi için bir bolluk değişkeni (S) kısıtlara eklenerek eşitlik sağlanır. Çünkü parçacıkların hızlarının değişiminde bir etken olan rastsallık, kısıtların dışında hareket etmelerine olanak sağlamıştır. Bunun önlenebilmesi ise eklenen değişkenlerin amaç fonksiyonunda da değerlendirilmesine bağlıdır. Böylece parçacıklar kısıtlardan etkilenir.

Böylelikle, süreç içerisinde uyulması zorunlu kısıtları sağlamayan parçacıklar, sağlamama değerlerine göre daha büyük bir amaç fonksiyonu değerine (amaç minimuma ulaşmak olduğu için) sahip olacaklardır.

Min X1 + X2 + X3 + S1 + S2 ;

X1 + X2 + S1 = 5
X2 + X3+ S2 = 4

Dönüştürme işleminin ardından iterasyonlara geçilir. Bir iterasyon boyunca normal sürü SZTO dışında, süreçte bazı farklılıklar mevcuttur. Bunlardan ilki, hesapanan amaç fonksiyonu değerinin bir geçici değişkene aktarılarak, bu değişkenin kısıtları sağlama ölçüsü oranında arttırılması suretiyle, herbir parçacığın grupsal en iyi ve bireysel en iyi değerlerinin bu değişken üzerinden yapılmasına gidilmesiyken, bir diğeri ise, kısıtlara ve amaç fonksiyonuna eklenen değişkenlerin çözüm uzayında bir başka boyut olarak tanımlanmasıdır. Yani örneğimizde, 3 boyut varken , boyut sayısı 5 e çıkartılarak çözüm aranır.

Boyutsal değişiklik yapıldıktan sonra, iterasyonlar başlatılarak her iterasyonda algoritma, mevcut amaç fonksiyonu değerini kısıtların sağlanamamasına göre belli bir oranda arttırır. Böylece kısıtların sağlanması bir zorunluluk haline getirilebilmiştir.

Algoritmanın temel mantığını açıklayacak olursak:

Dummy= f(X1,X2,X3)
İf X1 + X2 < 5 then dummy = dummy + dummy * S1 / $S1_{başlangıç}$
İf X2 + X3 <> then dummy = dummy + dummy * S2 / $S2_{Başlangıç}$.
…
İf dummy < GrupsalEniyiDeğeri then .....
…
GrupsalEniyiDeğeri=dummy
…
End if
İf dummy< bireyselEniyiDeğeri then.....
…
BireyselEniyiDeğeri = dummy

…
End if,

şeklindedir.

Bu şekilde en iyi nokta seçimi için kısıtları sağlamaya zorlanan parçacıklar, verilen fonksiyonu ilgili kısıtlar doğrultusunda çözüme ulaştırabilir. Bulunan sonuçlardaki hatalar ise yaklaşık 10000 de 1 ler seviyesinde olup bu hatanın da iterasyon sayısı ve grup sayısının arttırılması ile azaltılabileceği düşünülebilir.
Penaltı yöntemi ile SZTO algoritmasındaki zorluk çekilen bir husus ise, kısıtlara verilen cezalar ile amaç fonksiyonunda katsayıları düşük olan (minimum problemi olduğunu düşünürsek), yani atanması kuvvetle muhtemel değişkenler arasında bir ödünleşmenin gerçekleşerek sonucu etkilemesidir. Bu ödünleşmenin engellenmesi, amaç fonksiyonuna yüklenen bolluk değişkenlerin katsayılarının arttırılması ile mümkün olabilir. Böylece kısıtların sağlanabilmesi daha olası bir durum olacaktır.

3.4.3. Huni Etkisi

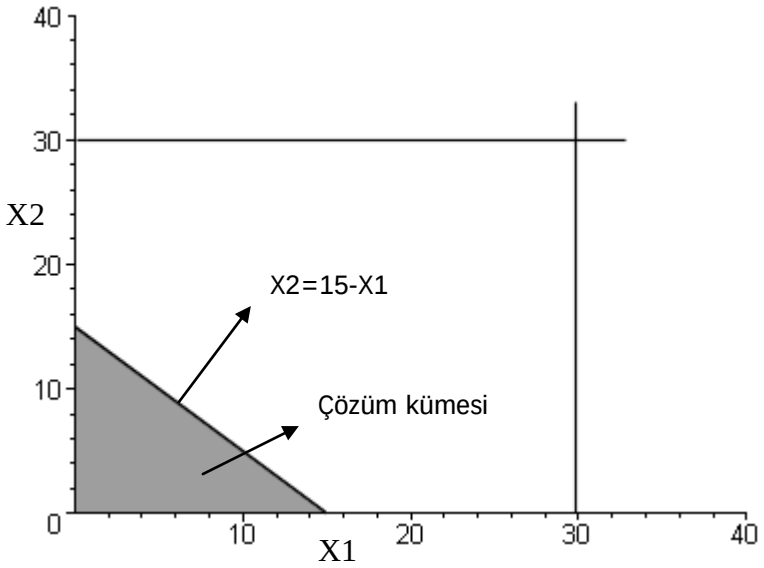
Çözüm kümesinin çok küçük olduğu durumlarda bu uygulanabilecek bir diğer yöntem bu çözüm kümesini komşu noktaların bir kısmını daha kapsayacak şekilde genişletmektir. Ancak bu genişlik iterasyon sayısı ilerledikçe daralmalıdır ki kısıtlardan sapma olabildiğince küçülsün. Yine Bölüm 3.3'deki örneği ele alacak olursak, amaç fonksiyonumuz

$$3 * X1 - 2 * X2$$

ve kısıtımız ise

$$X1 + X2 < 15$$

iken, çözüm kümesi şekil 1'de görülmektedir.



Şekil 1- Çözüm kümesi

Bu durumda klasik algoritmayı, kısıtı sağlamayan noktaların en iyi nokta olarak atanmasına engel olacak şekilde değiştirerek gri bölge ile temsil edilen çözüm kümesi içerisinde sonuç bulunabilir. Ancak eğer kısıt

$$X1+X2 < 15$$

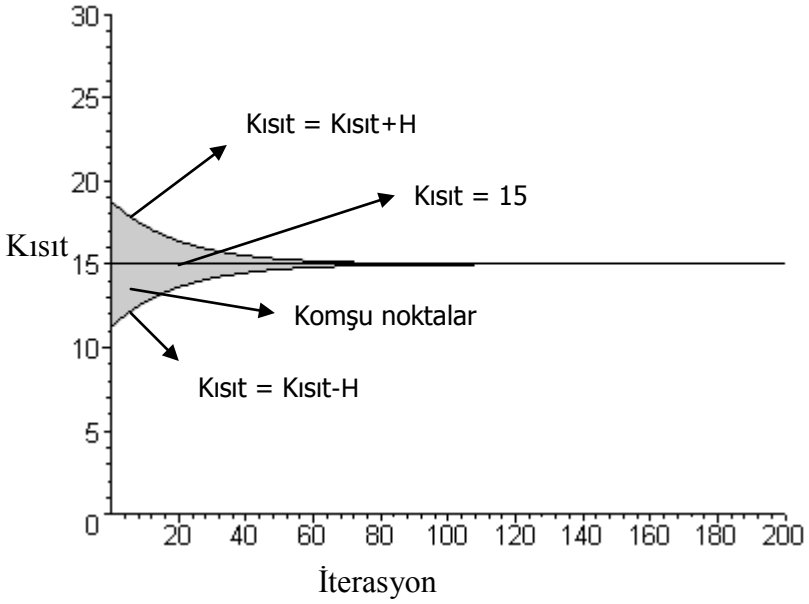
yerine

$$X1+X2 = 15$$

olarak değiştirilir ise, bu durumda gri bölgedeki noktalar çözüm kümesinden çıkar ve yalnızca kısıtı oluşturan denklem üzerindeki noktalar çözüm kümesi olur. Bu durumda klasik algoritmanın kısıta uyan noktaları yakalama olasılığı çok düşük olduğundan uygun noktalar ve dolayısı ile en iyi noktalar saptanamaz ve algoritma işlevselliğini yitirir.

Bu sorunu aşabilmek için kısıtı esneterek, aslında kısıtın orijinal haline uymayan fakat yine de belli bir yakınlıkta olan noktaları da çözüm kümesine dahil etmemiz faydalı

olur. Fakat bu esneklik kalıcı olmamalı, maksimum iterasyona doğru yaklaşıldıkça kaybolmalıdır ki kısıtı gerçekten sağlayan noktalara doğru gidilsin. Şekil 2'de görülen grafikte bu etki sağlanmıştır. Buna göre her iterasyonda orijinal kısıt değeri belirli bir miktar esnetilir. Bu esneklik kısıtın yönüne göre de değişmektedir. Eşitlik kısıtlarında orijinal kısıt değeri hem pozitif, hem de negatif yönde esnetilerek tek bir değerden bir aralığa götürülür. Kısıt denklemin bir değerden büyük olmasını şart koşan kısıtlarda bu değer olduğundan daha yukarı; küçük olmasını şart koşan kısıtlarda ise olduğundan daha aşağı çekilir.



Şekil 2 – Huni etkisi

Grafikte görülen +/- H değeri her iterasyonda,

M_{ep} = Maksimum esneme payı,

M_{itr} = Maksimum iterasyon sayısı, olmak üzere,

$$H = Kısıt * M_{ep} * \exp(- İtr / M_{itr} * 0{,}1)$$

formülü ile sağlanmaktadır. Böylelikle kısıt ilk iterasyonda gerçek değerinden M_{ep} kadar esnemiş olacaktır. Eğer model bize çözüm sağlayan noktanın

$$X1+X2 = 15$$

kısıtına uymasını şart koşuyor ise huni etkisi sayesinde bu kısıt iki ayrı kısıta bölünmüş oluyor. İlk modelimiz

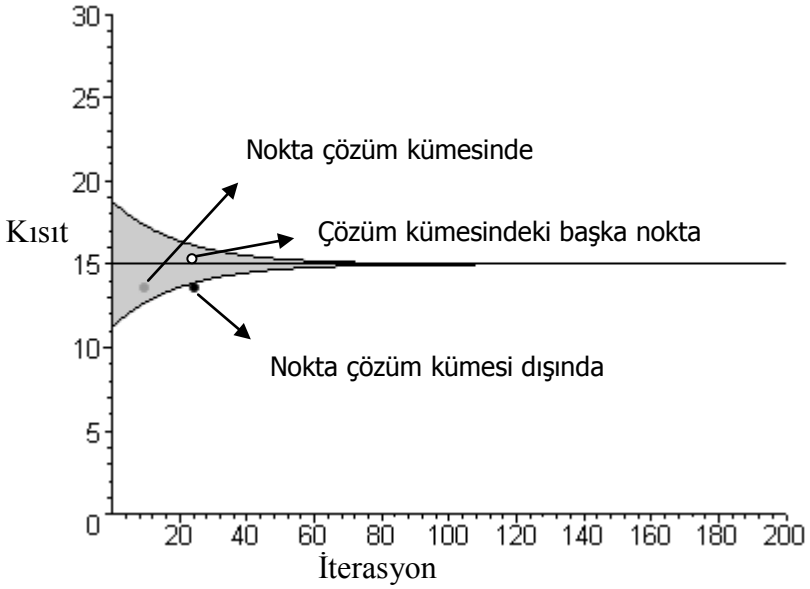
$$\text{Max } 3 * X1 - 2 * X2$$
$$\text{s.t}$$
$$X1+X2 = 15$$

iken, huni etkisi sayesinde

$$\text{Max } 3 * X1 - 2 * X2$$
$$\text{s.t}$$
$$X1+X2 \geq 15 - H$$
$$X1+X2 \leq 15 + H$$

halini almaktadır. İlk iterasyonda H değeri M_{ep} = 0,3 alınırsa 4,5 olacak, daha sonra üstel olarak azalarak son iterasyona gelindiğinde 0 olacaktır. Böylelikle SZTO'nun ilk iterasyonlarda en iyi noktaları seçebilme olasılığı yükselecek ve son iterasyonlara doğru en iyi noktaların kısıtlara giderek yaklaşması sağlanacaktır.

Klasik SZTO algoritmasının bu tekniğe uyarlanabilmesi için en iyi noktanın seçilmesini sağlayan kriterlerde bazı değişiklikler yapılması gerekmektedir. Buna göre herhangi bir noktanın değeri en iyi ise ve nokta çözüm kümesi içerisinde ise en iyi olmayı sürdürecektir. Ancak eğer nokta çözüm kümesi dışında kalırsa, değeri hala en iyi olsa bile, bu nokta yerine çözüm kümesi içerisinde yakalanan ilk nokta en iyi nokta seçilecektir. Yakalanan ilk nokta en iyi seçildiği için diğer noktalar içerisinde çözüm kümesinde olan daha yüksek değerli bir noktanın kaçırılması da engellenmiş olacaktır. Şekil 3'te bu seçim mantığı görülmektedir.



Şekil 3 – Seçim mantığı

Nokta 10. iterasyonda kısıta verilen esneklik sayesinde çözüm kümesi içinde bulunmuş ve değeri en iyi olduğu için en iyi nokta seçilmiştir. Ancak 25. iterasyona gelindiğinde nokta hala en yüksek değerli nokta olduğu halde çözüm kümesi dışında kaldığı için taranan noktalar içerisinde çözüm kümesine dahil olduğu saptanan ilk nokta en iyi olarak atanmıştır. Daha sonra değerlendirilen parçacıklar içerisinde daha iyi bir noktada olan ve çözm kümesinde bulunan başka bir noktaya rastlanırsa bu nokta en iyi olarak atanacaktır.

Huni etkisi etkinleştirildiği taktirde algoritmada bir değişiklik daha yapılmaktadır. C1 ve C2 katsayıları dinamikleştirilmekte ve parçaıkların yönelimleri bireysel en iyiden grupsal en iyiye doğru ağırlıklandırlımaktadır. C1 ve C2 katsayıları SZTO'nun bireysel en iyilere mi yoksa grupsal en iyiye mi ağırlıklı olarak yöneleceğini belirler. C1 katsayısı yüksek ise parçacık bireysel en iyiye gitmeyi daha çok isteyecektir. C2 yüksek olduğunda ise parçacık grupsal en iyiye doğru yönelmeyi daha fazla isteyecektir.
Algoritma içerisinde yapılan değişiklik ile C1 ve C2 katsayıları dinamikleştirilmiştir. Buna göre C1 ve C2 katsayılarının başlangıçta 2 olması, iterasyon sayısı ilerledikçe düzgün bir şekilde C1'in 1'e, C2'nin ise 4'e gitmesini sağlanmıştır. Böylece ilk iterasyonlarda kendi deneyimlerinden ve grubun deneyimlerinden eşit ağırlıkta

etkilenen parçacık, iterasyon sayısı maksimum düzeye yaklaştıkça grubun deneyimlerine daha çok ağırlık vermeye başlayacaktır.

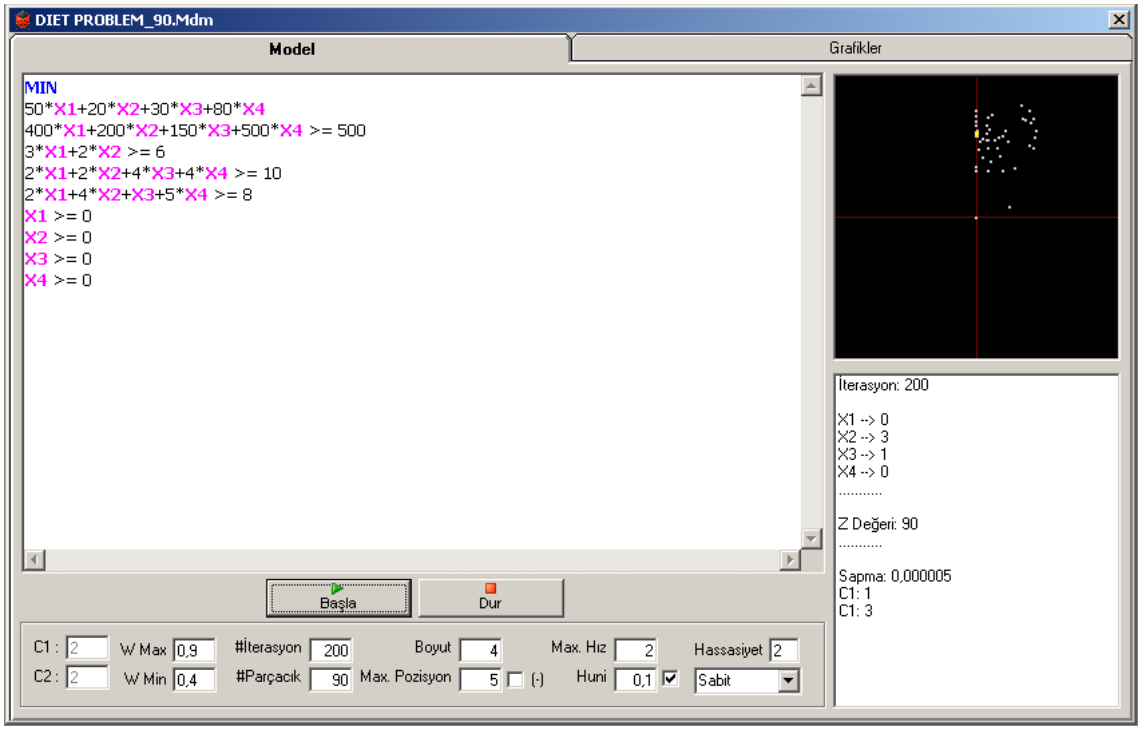
3.5. Genel Amaçlı Model Çözüm Uygulaması

Bölüm 3.4.1 ve 3.4.3'te ele alınan uyarlamalar sayesinde birçok modelin geliştirmiş olduğumuz yazılım ile çözülebilmesi mümkün hale gelmiştir. Genel amaçlı model çözmek için geliştirdiğimiz arayüzün tasarımında, kullanıcıya mümkün olan en fazla esnekliği sağlayabilmek için, oluşturulan modelin belirli kurallar çerçevesinde direkt olarak yazılabileceği bir ekran oluşturduk. Böylece kullanıcı, ne tür problem olursa olsun, modelini doğru yapıda ve kurallara uygun bir şekilde girdiği taktirde algoritma en iyi çözümü bulmaya çalışacaktır. Fakat dikkat edilmesi gereken en önemli husus algoritmanın çalışmasını belirleyen parametrelerin doğru seçilmesidir. Örneğin en iyi koordinatlarının 0 ile 10 arasında değerler aldığı önceden kestirilen bir problem için, maksimum pozisyon değerinin (araştırmanın yapılacağı maksimum değer) 100 gibi gereğinden fazla büyük seçilmesi yazılımın en iyiyi bulmasını güçleştirecek, ve hatta belkide bulamamasına neden olacaktır. Zira bu yazılım grupsal optimum noktaları değil de, verilen sınırlar içerisindeki bireysel optimum noktaları bulabilecek şekilde geliştirilmiştir.

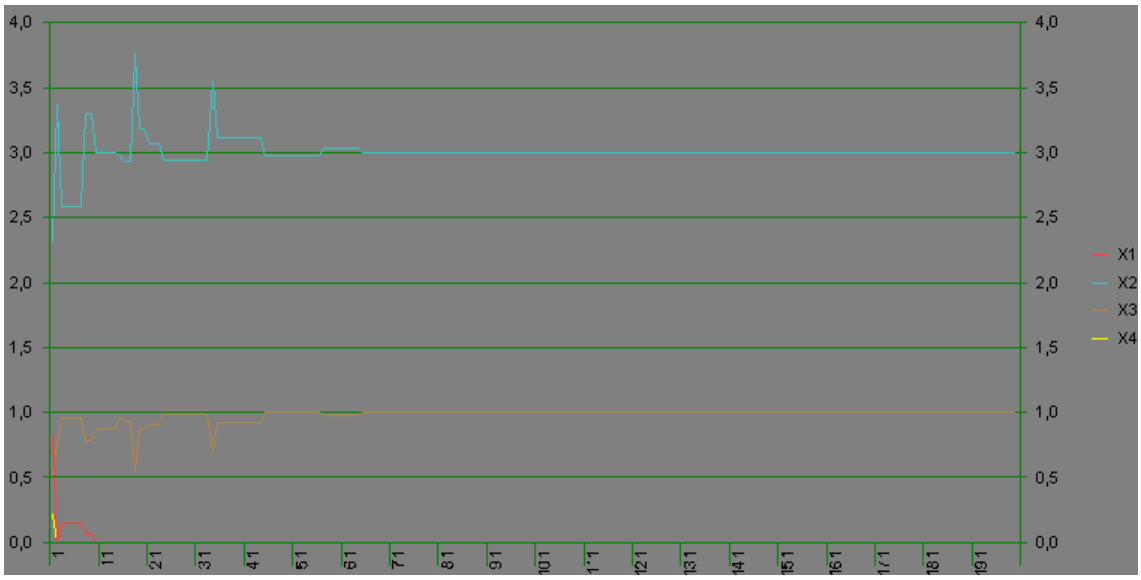
Eşitlik kısıtı içeren modellerde huni etkisinin ve kademeli hassasiyet tekniklerinin kullanılması algoritmanın etkenliğini önemli ölçüde arttırmaktadır. Hatta birçok model bu teknikler kullanılmaksızın çözülememektedir.

Bu tekniklerin, özellikle de huni tekniğinin kullanılması Z değerinin dalgalı bir seyir izlemesine neden olmaktadır. Klasik algoritmada ise sürekli iyileştirme olduğu için Z değerinin grafiği sürekli artan veya sürekli azalan olmaktadır.

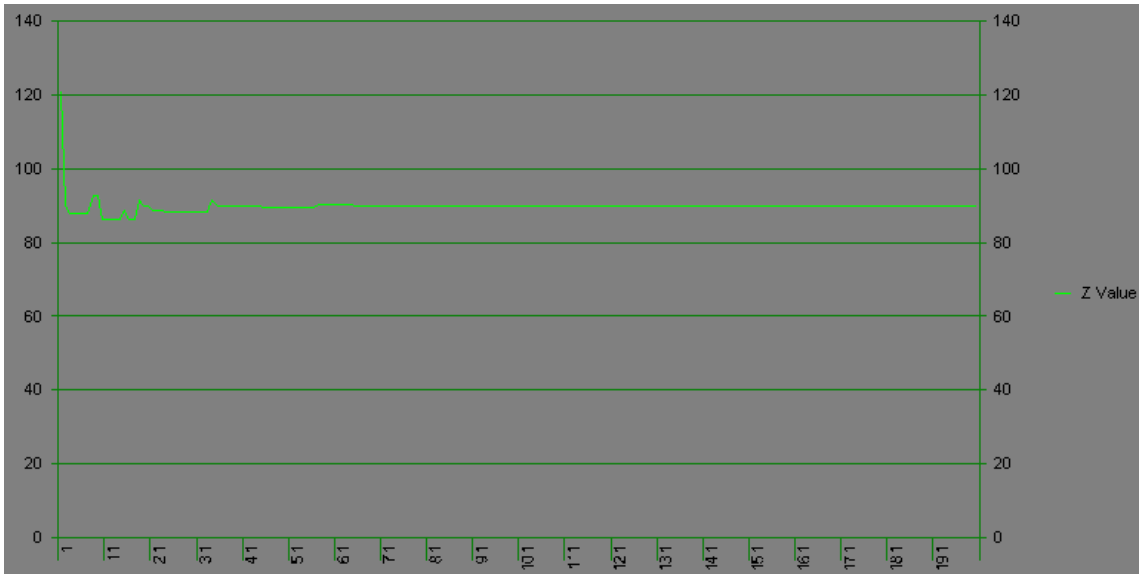
Şekil 4'te doğru parametreler sayesinde başarı ile çözülmüş bir model; şekil 5'te modeldeki değişkenlerin,ve şekil 6'da ise Z değerinin süreç boyunca aldıkları değerleri gösteren grafikler görülmektedir.



Şekil 4 – Bir model ve çözümü



Şekil 5 – Değişkenlerin grafiği



Şekil 6 – Amaç fonksiyonu grafiği

4. YEREL PROBLEMLERE SZTO YAKLAŞIMI

4.1. İşletmelerimizde Optimizasyon Açıkları

Teknoloji ve bilimde kaydedilen ilerlemeler sonucunda, artık iş yapmak için kullanılan yöntemler değişmeye başlamış, ayakta kalabilmek için globalleşen dünya ile uyum sağlamak zorunlu olmuştur. Eskiden haklarında üstünkörü karar verilebilen parmetreler, artık değişik mühendislik teknikleri ile enine boyuna incelenerek, mümkün olan en iyi şekilde hesaplanmalıdırlar. Aksi taktirde bu parametrelerin optimumdan uzak olmalarının getireceği ekonomik yük birikerek işletmenin gelişiminin önüne bir engel olarak çıkmakta, sektördeki rekabet gücünün zayıflamasına neden olmaktadır.

Genel olarak bakıldığında; bir işletmede tesisin etkin bir şekilde planlanmasından, üretimin minimum maliyetlerle yapılmasına kadar birçok alanda, en iyi değerinin hesaplanması gereken bir çok parametre vardır. Bu en iyi değerler, maliyeti küçülten, işgücünden tasarruf sağlayan, sonuçta işletmenin kaynaklarını en verimli şekilde kullanmasına olanak veren değerlerdir. Bu anlamda bakıldığında bu değerlerin hesaplanmasının gerekliliği kaçınılmazdır. Ancak çoğu işletmede bu ince ayarlar gerekli özen gösterilerek yapılamamakta, karar vericiler daha iyi seçeneklerinin varlığından haberdar olamamaktadırlar.

Tek bir işletmenin kaynaklarını daha verimli kullanması tek başına yerel ekonomi ve ülke ekonomisi üzerinde pek etki yaratamasa da, olaya bir bütün olarak baktığımızda optimizasyonun ne denli önemli bir rolü olduğunu anlayabiliriz. Bütün işletmelerin kaynaklarını en verimli şekilde kullanabilmesi, yerel ekonomimizin ve dolayısıyla ülke ekonomisinin kalkınmasında çok önemli bir etki yaratmaktadır.

4.2. Üretim Planlamasında Optimizasyon Eksikliği

4.2.1. Problemin tanımı

Bazı modellerin çözümünde, mevcut yöntemlerin etkisiz kaldığından bahsetmiştik.

Endüstri Mühendisliğinin sıklıkla üzerinde durduğu bu tip problemlerden bir tanesi de dinamik talepli devrelerde sipariş zamanı ve miktarının optimizasyonu problemidir. Tek kısıtlı bir amaç fonksiyonunun çözümünü içeren bu problem, toplam sipariş maliyetini

en küçükleyecek stokta bulundurma maliyetinin tesbiti için sipariş sayısı bazında optimizasyonu gerektirir.

Simplex algoritma gibi model çözüm yöntemleri ile çözülememesi problemin optimum iyileştirilmesinin gerçekleştirilmesini engellemektedir.

Yapılan araştırmalarda yapay zeka tekniklerinin (Genetik algoritma, tabu,difizyon... gibi) liner olmayan sürekli fonksiyonları içeren sistemlerde etkin çözümler ürettiği farkedilmiştir. [6].

Problem model formuna sokulmak istendiğinde, incelenmek istenen her sipariş sayısında minimum yapılmak istenen bir amaç fonksiyonu ve bunun yanında devre toplam talebine eşit olan belirleyici bir kısıt içeren bir optimizasyon problemine dönüşmektedir.

Yani;

$$\text{Min } i * c\left[t_1 * S_{başaşlang} + \sum_{i=1}^{n} ((t_{i+1} - t_i) * D(t_{i+1})) - \int_0^{t_{son}} D(t_{son})dt \right]$$

$$D(t_2) + \sum_{i=1}^{n} ((t_{i+1} - t_i) * D(t_{i+1})) = D(t_{son})$$

$$0 \le t_i \le 1 \qquad \text{i =1,2,3,.....n(Sipariş miktarı)}$$

Görüldüğü gibi ardışık bir yapıya sahip olan amaç fonksiyonu ve kısıtlar kolaylıkla algoritmaya dönüştürülerek her sipariş miktarı için problemin analizi yapılabilmekte ve modelin çözümü de SZTO ile gerçekleştirilebilmektedir.
İncelenmek istenen sipariş sayısı arttıkça modelin çözümü zorlaşabilmektedir.

4.2.2. Dinamik talep altında sipariş zamanı ve miktarı optimizasyonu uygulaması

SZTO ile model çözümünün yerel problemlere uygulanabilir olmasının anlaşılabilirliği açısından, penaltı yöntemi dinamik devre sipariş miktarının bulunması problemine uygulanmaktadır.

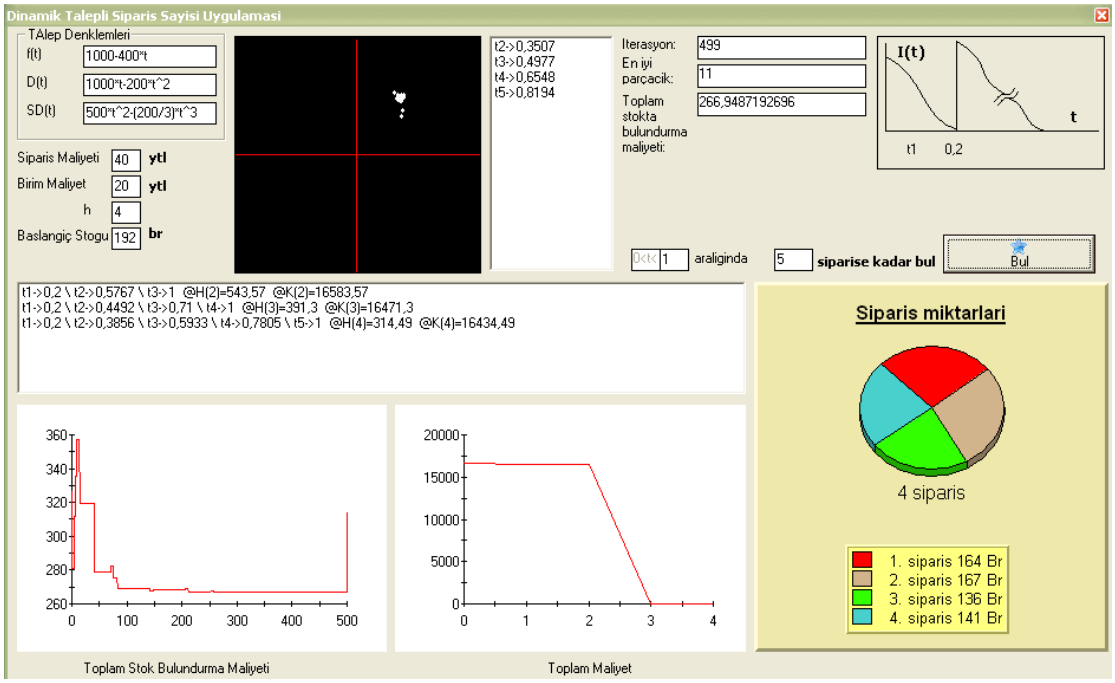
Bu problem, incelenmek istenen talep denklemi ile sipariş maliyeti (A), stokta bulundurma oranı (i), birim maliyet (C), devre uzunluğu (t_{son}) ve başlangıç stoğu parametreleri ile modelin kurulması ve çözülmesi olarak düşünülebilir.

Her sipariş sayısı için sipariş zamanlarının optimum olarak bulunması, toplam stokta bulundurma maliyetinin minimum olmasını sağlayan önemli bir unsurdur. Sipariş sürelerinin bulunması da kurulan modelin çözümüne bağlıdır ve bulunan sonuç toplam stokta bulundurma maliyetinin optimizasyonunu sağlamaktadır. Yani sipariş zamanlarının bulunması stokta bulundurma maliyetini en az yapar. Böylece sipariş sayıları arasında karşılaştırma yapılarak toplam maliyetin değişimini gözlemleme imkanı sağlanmış olur.

Simplex algoritma gibi klasik yöneylem araştırması yöntemleri bu modellerin çözülmesinde yetersiz kalmaktadır. Fakat SZTO model çözüm yöntemi ile bu modeller etkin ve hızlı bir seyirde çözülebilmekte ve sonuca yansıtılmaktadır.

Çoğu literatürde 3 sipariş, ve sonrası (n>2) için en az toplam stokta bulunudurma maliyetinin bulunması garanti edilememektedir. [7]. Fakat uygulamamızda görülebileceği gibi SZTO kullanılarak sipariş süreleri en optimum şekilde bulunmuş ve sonuca yansıtılmıştır.

Uygulamanın çalıştırılması için kullanıcı tarafından, talebin zamana göre değiştiği devresel talep denklemi (f(t)), birikimli talep denklemi ve birikimli talep denkleminin integrali arayüzdeki ilgili alanlara girilmektedir. Bu denklemler ve diğer ilgili üretim parametreleri ile model kurulmakta ve inecelenmek istenen sipariş sayısına kadar çözülmektedir.



Şekil 7 – Dinamik talep altında sipariş zamanlarının ve miktarlarının bulunması

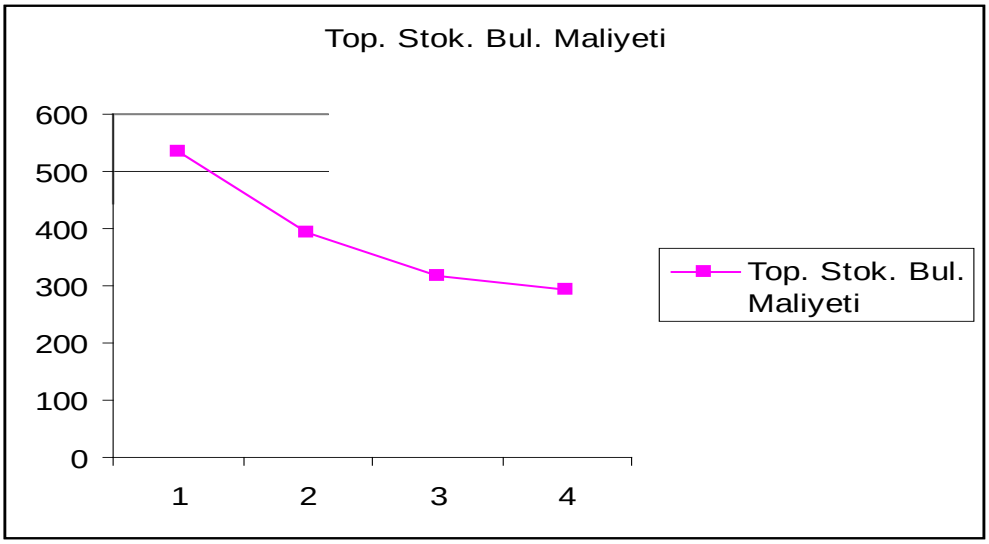
Uygulama başlatıldığında ilk olarak (t_1) başlangıç stoğunun bitiş zamanı 100 iterasyonda hesaplanarak, n = 1 sipariş ve sonrası durumlarda sipariş süreleri ($t_2,t_3,...t_{son-1}$) ve stokta bulundurma maliyetleri (H(n)) ile toplam maliyetlerin (K(n)) hesaplanması sağlanır.

Uygulama f(t)=1000-400*t talep denklemi ile, A = 40 (sabit sipariş maliyeti), c = 20 (birim maliyet), i = %20 (yıllık stokta bulundurma-maliyet oranı) ve I(0) = 192 (başlangıç stoğu) değerlerine sahip olan basit bir problem üzerinde incelenir ise, $t_{1=}$ 0.2 bulunduktan sonra n = 5 sipariş sayısına ilişkin değerler tablo 1'deki gibi bulunur.

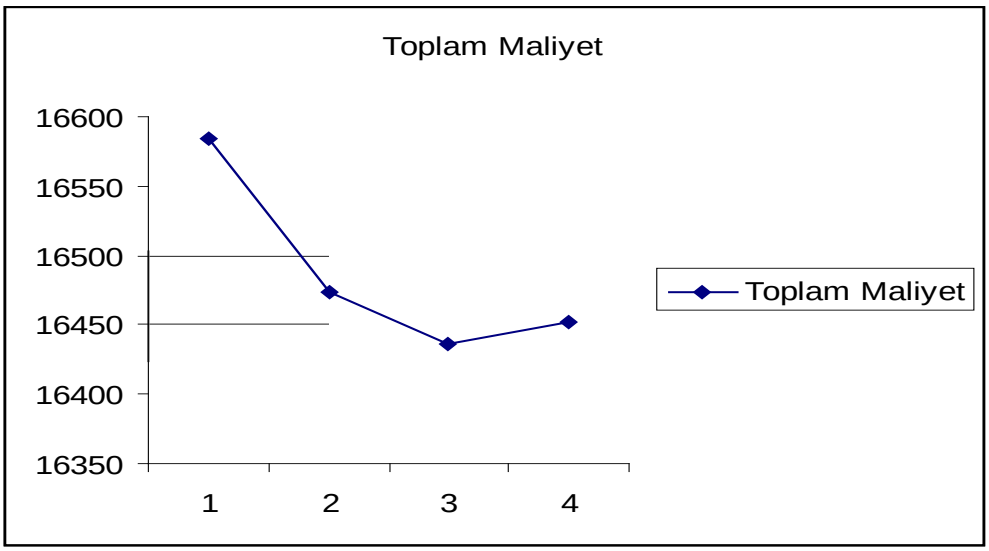
	Stokta bulundurma Maliyeti	Top. Maliyet
1	534,57	16583,57
2	393,43	16473,43
3	315,61	16435,61
4	292,29	16452,29

Tablo 1

Bu şekilde toplam stokta bulundurma maliyeti üstel olarak azalırken, toplam maliyet de 3 siparişe kadar azalma gösterebilmiştir. Burada modellerin çözülmesi etkin bir şekilde maliyetlerin araştırılmasını sağlayabilmektedir ve hangi miktarda siparişin hangi sürelerde tayin edilmesi gerektiğinin bulunmasını kolaylaştırmıştır.



Şekil 8 - 4 siparişe kadar toplam stokta bulundurma maliyetinin değişimi



Şekil 9 - 4 Siparişe kadar toplam maliyetin değişimi

Toplam stokta bulundurma maliyetindeki azalma miktarının, sipariş maliyetinin altına düştüğü durumda, sipariş miktarında optimizasyon sağlanarak en uygun sipariş sayısının bulunması ve dolayısıyla sipariş miktarının da belirlenmesi sağlanmıştır.

Şekil 10'daki grafikte de görüldüğü gibi, her sipariş sayısı 500 iterasyonda bir,

$$Min 4*(38,4+(X_2-0,2)*(1000*(X_2)-200*(X2)^2)+(X_3-X_2)*(1000*(X_3)-200*(X_3)^2)+(1-(X_3))*800-433,3333)$$

$$1000*(X_2)-200*(X_2)^2+((X_2)-(0,2))*(1000-400*(X_2))+((X_3)-(X_2))*(1000-400*(X_3))=800$$

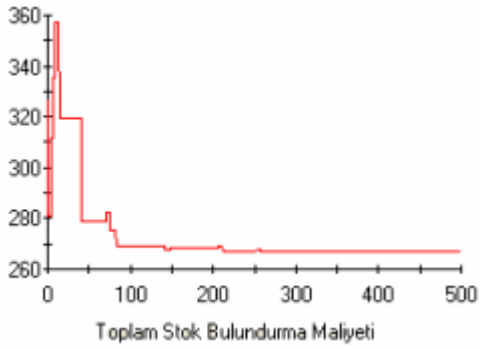
$$X_i-X_{i-1}> 0$$

$$X_i>0$$

$$X_i<1$$

$$i = 1 \text{ to } n$$

gibi modeller çözülerek sonuçlar bulunup, sipariş sayısının optimum olabilmesi için karşılaştırma imkanı sağlanabilmektedir. Yapılan hata ise % 0,001 ler mertebesindedir.



Şekil 10 - Sistemin amaç fonksiyonu olan toplam stokta bulundurma maliyetinin ilgili iterasyonlara göre değişimi

Sonuçlar bulunduktan sonra, çözülen modeller ve çözümleri de uygulama klasörü içerisinde rapor.txt olarak kaydedilmektedir.

4.3. Üretim Planlamasında Optimizasyonun Yerel Ekonomi Üzerindeki Etkisi

Bilindiği gibi Eskişehir Organize Sanayi Bölgesi 32 milyon metrekarelik büyüklüğü ile ülkemizin en büyük OSB'sidir. Halen Eskişehir OSB'de 250 firma bulunmakta olup, bunlardan 210 tanesi faaliyette, geriye kalan 40 firma ise proje ve inşaat aşamasındadır. Faal firmalarda çalışanların sayısı yaklaşık 20 bin kişi, 2003 yılında gerçekleştirilen ihracat miktarı ise 350 milyon dolardır.

Mühendislik biliminin işletmelerimize sağladığı yöntemlerin etkin bir şekilde kullanılmaması, bütün gözetildiğinde, ciddi miktarların Eskişehir ekonomisine kazandırılamamasına yol açmaktadır. Yerel ekonomi, zarar olarak kayıtlara geçirilmeyen optimizasyon kayıpları düşünüldüğünde büyük ölçüde zarar görmektedir.

Çalışmamızda incelediğimiz "Dinamik Talep Altında Sipariş Zamanı ve Miktarı Optimizasyonu" örnek probleminde yapılan optimizasyon sonucunda, optimum bulunan spariş süreleri yardımıyla rutin yönteme nazaran yıllık ortalama 100 YTL 'lik bir toplam stokta bulundurma maliyeti iyileşmesi sağlanabilmektedir. Bir işletmenin en az 200 kalem malzeme siparişi yaptığını düşünelim – ki bu değer bile oldukça sembolik niteliktedir – işletmenin bu durumdan yıllık kazancı 20.000 YTL olabilmektedir. Belirttiğimiz gibi OSB de 210 adet işletme faaliyet göstermektedir. O halde yuvarlak bir hesap yaptığımız taktirde, yılda ortalama 4.200.000 YTL' lik bir kazancı yerel ekonomimize kazandırmış olmaktayız. Ve bu yalnızca Eskişehir için hesaplanmış basit bir örnektir.

Yne örneğimiz üzerinden devam eder ve Eskişehir OSB de 20.000 çalışanımız olduğunu düşünürsek, SZTO ile bulunan sipariş miktarları ile her çalışanımızın bütçesine yılda 210 YTL yaklaşık değer katabilmek mümkün olabilir. Bu miktar sadece bir fikir vermek için düşünülmüştür. Bireysel bazda az gözüken fakat yerel ekonomi bazında ise büyük etkiler sağlayabilecek bir sonuç olduğu açıkça görülmektedir.

Eknomik sıkıntılar şu günlerde en büyük sorunlarımızdan bir tanesidir. Kaynaklarımızdan en iyi şekilde istifade etmek artık geleceğimize ve halkımıza borçlu olduğumuz bir zorunluluktur.

5. KAYNAKÇA

[1] Kennedy and R. Eberhart, 1995, "Particle Swarm Optimization", Proceedings of IEEE International Conference on Neural Networks (ICNN'95), Vol. IV, pp.1942-1948, Perth, Australia.

[2] Y. Shi and R. Eberhart, May 1998, "A Modified Particle Swarm Optimizer", Proceedings of IEEE International Conference on Evolutionary Computation (ICEC'98), pp.69-73, Anchorage.

[3] M. Clerc, "The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization", Proc. of IEEE International

[4] R. Eberhart and Y. Shi, 2000, "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization", Proc. of the Congress on Evolutionary Computation (CEC2000), pp.84-88.

[5] N. Hansen and S. Kern , 2004, "Evaluating the CMA Evolution Strategy on Multimodal Test Functions Parallel Problem Solving from Nature", PPSN

[6] T. Krink, R. Thomsen, **"**Self-Organized Criticality and Mass Extinction in Evolutionary Algorithms Institute for Advanced Study", Berlin

[7] L. A. Johnson ve D. C. Montgomery, 1974, "Operations research in production planning, scheduling, and inventory control" ,Georgia Institude of Technology ,John Wiley & Sons Inc.; New york.

[8] A. Salman, I. Ahmad, S. Al-Madani, "Particial swarm optmization for task assignment problem".

[9] A. İşlier, Üretim Süreçlerinin "Bir Yetenek Algoritmasıyla Oluşturulması", 658.533. i75,Anadolu University Journal of science and technology

[10] M. Kapanoglu, F. Utkan, and M. Alikalfa , 2005, "Particle swarm optimization for faicilty Layout Problems" ,Proceedings of the 10th Annual International Conference on Industrial Engineering – Theory, Applications and Practice, Clearwater, Florida, USA December 4-7, ISBN: 0-9654506-1-9 552

[11] K.E. Parsopuulos ve M.N. Vrahatıs ,"Recent approaches to global optimization problems through Particle Swarm Optimization", Department of Mathematics, University of Patras Artificial Intelligence Research

[12] Y. Shi and R. Eberhart, 1998, "Parameter Selection in Particle Swarm Optimization", *Proc. of the 1998 Annual Conference on Evolutionary Programming*, San Diego.

[13] F., Mark , "Foundations of Swarm Intelligence:From Principles to Practice, Institute for Systems Research", University of Maryland

College Park, Maryland 20742

[14] O., Kurt, 2006, "Tesis içi Yerleşim Probleminin Genetik Algoritmalarla Çözüm Uygulamaları, Sosyal Zeka Optimizasyon Tekiniğinde ve Fonksiyonel Uygulamaları", Osmangazi Üniversitesi Endüstri Mühendisliği, Eskişehir.

[15] A. P. Engelbrecht**, "**Fundamentals of Computational Swarm Intelligence" ISBN: 0-470-09191-6