YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DEMİRYOLU TRAFİK KONTROLÜ PROBLEMİNİN GENETİK ALGORİTMALARLA ÇÖZÜMÜ

İnşaat Yüksek Mühendisi Selim DÜNDAR

FBE İnşaat Mühendisliği Anabilim Dalı Ulaştırma Programında Hazırlanan

DOKTORA TEZİ

Tez Savunma Tarihi : 29 Ocak 2010
Tez Danışmanı : Doç.Dr. İsmail ŞAHİN (Yıldız Teknik Üniversitesi)
Jüri Üyeleri : Prof.Dr. Haluk GERÇEK (İstanbul Teknik Üniversitesi)
: Prof.Dr. Güngör EVREN (Okan Üniversitesi)
: Prof.Dr. Zekai ŞEN (İstanbul Teknik Üniversitesi)
: Yrd.Doç.Dr. Nevzat ERSELCAN (İstanbul Teknik Üniversitesi)

İSTANBUL, 2009

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

a_i^m	Tren *i*'nin buluşma noktası *m*'ye gerçekleşen varış zamanı
b_{ik}^m	Tren *i* ve tren *k*'nin buluşma noktası m'den kalkış sırasını belirleyen ikili karar değişkeni
c_{ij}^m	Tren *i* ve tren *k*'nin buluşma noktası m'den kalkış sırasını belirleyen ikili karar değişkeni
d_i^m	Tren *i*'nin buluşma noktası *m*'den gerçekleşen kalkış zamanı
e_i^m	Tren *i*'nin buluşma noktası *m*'ye kadarki yığışımlı gecikme süresi
i	Giden trenlerin indeksi
I	Giden trenlerin kümesi
j	Dönen trenlerin indeksi
J	Dönen trenlerin kümesi
M	Buluşma noktalarının indeksi
S_i	Tren *i*'nin varması planlanmış ve/veya durması planlanmış buluşma noktalarının kümesi
w_i	Tren *i*'nin temel öncelik katsayısına bağlı olarak atanan ağırlık değeri
μ^m	Buluşma noktası *m*'deki hatların uzunluğu
σ_i	Tren *i*'nin ilk kalkış buluşma noktasının indeksi
ε_i	Tren *i*'nin son varış buluşma noktasının indeksi
α_i^m	Tren *i*'nin buluşma noktası *m*'ye planlanmış varış zamanı
δ_i^m	Tren *i*'nin buluşma noktası *m*'den planlanmış kalkış zamanı
λ_i	Tren *i*'nin uzunluğu
τ_i^m	Tren *i*'nin buluşma kesimi *m*'deki (*m* ve *m+1* arasındaki) tabii seyir süresi
η_{ik}^m	Tren *i* ve tren *k* arasında buluşma kesimi *m*'deki en küçük izleme süresi
ρ_{ij}^m	Tren *i*'nin *m* buluşma noktasına varış zamanıyla tren *j*'nin kalkış zamanı arasındaki en küçük (karşılaşma) emniyet süresi
ξ	Yeteri kadar büyük bir pozitif sayı
Ψ	Yeteri kadar büyük bir pozitif sayı

KISALTMA LİSTESİ

AMCC	Avoid Most Critical Computation (Time) (En Kritik Hesaplama Süresinin Önlenmesi)
CBS	Coğrafi Bilgi Sistemleri
CTC	Centralized Traffic Control (Merkezi Trafik Kontrolü)
FIFO	First In First Out (İlk Giren İlk Çıkar)
FOFI	First Out First In (İlk Çıkacak Olan İlk Girer)
GA	Genetik Algoritmalar
NP	Nondeterministic Polynomial (Deterministik Olmayan Polinom)
TCDD	Türkiye Cumhuriyeti Devlet Demiryolları
TGV	Train à Grande Vitesse (Fransız Hızlı Trenleri)
TWS	Train Working Simulator (Tren Çalıştırma Benzeticisi)
YSA	Yapay Sinir Ağları

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

2001 yılında tamamladığım "Trenlerarası Çatışmaların Çözümünde Dispeçer Kararlarının Yapay Sinir Ağları ile Modellenmesi" konulu yüksek lisans tezimde trenlerarası çatışmalara çözüm getiren dispeçerlerin karar değişkenlerini incelemiş ve bu kararları taklit eden bir yapay sinir ağı modeli geliştirmiştim. Bu çalışma sırasında karşılaştığım "Yapay Zeka" konuları bende büyük bir merak ve ilgi uyandırdı. Dolayısıyla farklı yapay zeka tekniklerini de inceleme fırsatı buldum.

Doktora eğitimime başladığımda, mümkün olduğunca yapay zeka teknikleri üzerine dersler almayı tercih ettim. Aldığım dersler ve incelediğim kaynaklarda özellikle genetik algoritmalar dikkatimi çeken bir yapay zeka tekniği oldu. Probleme arama uzayının farklı noktalarından başlaması, rasgele arama tekniği uygulayarak, kısa süre içerisinde en iyiye yakın derecede iyi sonuçlar elde etmesi genetik algoritmaların dikkat çekici özellikleri arasındaydı. Demiryolu trafik kontrolü probleminin de büyük boyutlu yapısı ve kısa zaman içinde karar verme zorunluluğu bende genetik algoritmaların bu probleme uygulanması ile iyi sonuçların elde edilebileceği fikrini uyandırdı.

Doktora derslerini tamamlayıp, yeterlilik sınavlarını da başarıyla geçtikten sonra, kendime doktora tez konusu olarak "Demiryolu Trafik Kontrolü Probleminin Genetik Algoritmalarla Çözümü"nü seçtim. Bu kapsamda, öncelikle 117 adet kaynak içeren kapsamlı bir kaynak araştırması gerçekleştirdim. Bu sayede probleme getirebileceğim yeni yaklaşımlar da belirginleşmeye başlamış oldu.

Daha sonra, genetik algoritmayı oluşturmak için Matlab paket programını seçtim. Matlab'in içerisinde bulunan genetik algoritma eklentisi, matematiksel bir uyum fonksiyonu tanımlanması gerektiği için, benim için kullanışsızdı. Bu nedenle Matlab'in programlama dilini kullanarak kendi genetik algoritmamı oluşturdum. Bunu gerçekleştirebilmek için programcılık bilgimi ve becerimi de geliştirmem gerekti.

Kullanılabilir bir genetik algoritma oluşturduktan sonra, bu algoritmayı demiryolu trafik kontrolü problemine uygulayabilmek için bir ara yüz geliştirdim. Ara yüz geliştirme konusunda değerli çalışma arkadaşım Arş.Gör. Ufuk Kırbaş'ın kıymetli yardımlarına ihtiyaç duydum. Kendisine yardımlarından ötürü teşekkürü borç bilirim.

Algoritma geliştirildikten sonra sıra algoritmanın test edilmesine gelmişti. Bunun için 51 adet örnek problem oluşturdum. Her bir problemi en az 10 defa çalıştırdım. Bu çalıştırmaların tamamlanması yaklaşık 4-5 ay gibi bir süre aldı. Ancak algoritma mantığında yaptığım bazı hatalar nedeniyle bu çalıştırmaları 4 kez tekrarlamak zorunda kaldım. Bu da 1,5 sene gibi önemli bir sürenin kaybına neden oldu. Ancak sonuçta geliştirdiğim algoritmanın sonuçları gerçekçi ve beni tatmin edici seyivede olmuştu.

Elimde uygulanabilir çizelgeler oluşturan bir program mevcuttu, ancak bu çizelgelerin ne derece iyi olduğu konusunda bir değerlendirme yapmak gerekiyordu. Öncelikle her bir problemin en iyi çözümünü bulup, algoritmanın çözümü ile en iyi çözümü karşılaştırmayı düşündüm. Ancak küçük boyutlu problemlerin en uygun çözümünün elde edilmesi dahi oldukça uzun sürdüğü için farklı bir yöntem seçmek kaçınılmaz hale geldi. Bu yüzden, yüksek lisans tezimde geliştirmiş olduğum yapay sinir ağını kullanmayı düşündüm. Geliştirdiğim ağ, dispeçer kararlarını %99 başarı ile taklit edebilmekteydi. Böylece dispeçer kararları ile genetik algoritma kararlarının karşılaştırmasını yapabilecektim. Sonuçlarda da görüleceği üzere, genetik algoritma (toplam tren gecikmelerini en küçükleme ölçütüne göre) örnek problemlerin hepsinde ya dispeçerle aynı kararları vermiş ya da daha iyi bir çözüm elde etmiştir.

Son olarak, geliştirdiğim algoritmanın başarımını arttırmak için, dispeçer çözümünü genetik algoritmanın başlangıç çözümleri arasında dahil etmeyi denedim (melez algoritma). Ancak bu deneme uyguladığım örnek problemlerde kayda değer bir fayda sağlamadı.

Artık doktora tezimin son kısımlarına gelmiştim. 6,5 yıllık çalışma süresi boyunca karşılaştığım onca sorunu kararlılıkla aştıktan sonra, gerçekleştirdiğim tezin yazımını gerçekleştirdim ve değerli hocalarımın görüşüne sundum.

Doktora çalışmam boyunca değerli görüş ve önerilerinden sıkça yararlandığım danışmanım Doç.Dr. İsmail Şahin'e;

30 yıllık yaşantım boyunca hepsinden ayrı ayrı kıymetli bilgiler aldığım, isimlerini listelemeye kalktığımda belki de apayrı bir tez oluşturacak hocalarımın tümüne;

En önemlisi, bana hayatın her alanında destek olan, doktora çalışması gerçekleştirmeme ön ayak olan çok sevgili aile bireylerim, anneannem Bedriye Ünkan, babam Sadık Ender Dündar ve annem Gülen Dündar'a;

teşekkür etmeyi bir borç bilirim.

ÖZET

Ulaştırma alt sistemlerinden biri olan demiryolu, diğer ulaştırma alt sistemleriyle yoğun bir rekabet halinde bulunmaktadır. Yürütüle gelen yanlış politikalar sonucu ülkemizde demiryolu ulaştırmasına olan talep, yolcu ve yük taşımacılığında karayolunun oldukça gerisinde kalmıştır. Demiryolunun pazar payını arttırması ve rekabetini devam ettirebilmesi için hizmet kalitesini arttırması gerekmektedir. Dakiklik ve güvenilirlik bir ulaştırma alt sisteminin kalitesini belirleyen ölçütlerin başında gelmektedir. Bu ölçütlerin istenilen seviyede tutulabilmesi de kısmen etkin trafik kontrolü ile sağlanabilir.

Trenler önceden hazırlanmış bir hareket planına uygun biçimde hareket etmektedir. Ancak beklenmedik bazı olayların gerçekleşmesi sonucu gecikmeler ve dolayısıyla trenlerarası çatışmalar meydana gelebilmektedir. Trafik kontrolü, trenlerarası çatışmaları, gecikmeleri mümkün olduğunca azaltacak şekilde çözüp, yeni bir uygulanabilir çizelge hazırlamak için uygulanır. Problemin zorluk derecesi nedeniyle, problemin en az gecikme içeren çözümüne kabul edilebilir bir süre içerisinde ulaşılması imkânsızdır. Bu çalışmada, 5 dakika gibi kısa bir süre içerisinde uygulanabilir ve gecikme toplamının olabildiğince en küçüklendiği bir çizelge hazırlamak için, genetik algoritmalar kullanılmıştır. Geliştirilen algoritmanın çözümleri, belirli boyuttaki problemlerin kesin ve dispeçer çözümleri (yapay sinir ağı) ile karşılaştırıldığında, algoritma kısa sürede yeteri kadar iyi sonuçlar vermektedir. Algoritmanın uygulanması için geliştirilen bilgisayar programı, tren dispeçerleri için bir karar destek sistemi olarak da kullanılabilir.

Anahtar Kelimeler: Demiryolu trafik kontrolü, trenlerarası çatışmalar, yeniden çizelgeleme, genetik algoritmalar, yapay sinir ağları, optimizasyon

JÜRİ:

1. Doç.Dr. İsmail ŞAHİN
2. Prof.Dr. Haluk GERÇEK
3. Prof.Dr. Güngör EVREN
4. Prof.Dr. Zekai ŞEN
5. Yrd.Doç.Dr. Nevzat ERSELCAN

Kabul tarihi:29.01.2010
Sayfa Sayısı: 132

ABSTRACT

Railways, which is a sub-system of transportation system, is in intense competition with other subsystems. As a result of wrong policies that have been carried out in our country, demand on railway transportation for freight and passenger has been outnumbered by highway transportation. To increase the market share and continue to compete with other systems, railway transportation has to increase its service quality. Timeliness and reliability are two of the most important criteria that determine the quality of a transportation subsystem. Keeping these criteria at a desired level can be partially achieved by performing effective traffic control.

Train movements are coordinated on according to a planned schedule. But train conflicts may occur in case of train delays, which happen because of unplanned events. Traffic control is carried out for rescheduling in a way which tries to minimize the train delays in conflict resolution. Reaching the optimal solution within an acceptable time is impossible because of the complexity of the problem. In this study, genetic algorithms are used to develop a schedule, which maintains delay minimization objective within 5 minutes solution time. In comparison of the solutions of the algorithm developed with the solutions of the exact and train dispatcher (artificial neural network model) for problems with certain size, the algorithm provided good enough solutions in reasonable amount of time. Computer program which is an implementation of the algorithm can also be used as a decision support tool for train dispatchers.

Keywords: Railway traffic control, intertrain conflicts, rescheduling, genetic algorithms, artificial neural networks, optimization

JÜRİ:

1. Doç.Dr. İsmail ŞAHİN
2. Prof.Dr. Haluk GERÇEK
3. Prof.Dr. Güngör EVREN
4. Prof.Dr. Zekai ŞEN
5. Yrd.Doç.Dr. Nevzat ERSELCAN

Kabul tarihi:29.01.2010
Sayfa Sayısı: 132

1. GİRİŞ

1.1 Amaç

Günümüzde demiryolları gerek yüksek hızlı ulaşım imkânları ve yüksek taşımacılık kapasitesi gerekse de düşük işletme maliyetleri sayesinde tekrar önem kazanmaktadır. Ancak ülkemizde özellikle 1950'lerden sonra karayolu yatırımlarına önem verilmiş olması, yeni demiryollarının yapımını yavaşlatmış, Cumhuriyet'in ilanından 1950'ye kadar geçen süre (27 yıl) içerisinde 3955 km yeni demiryolu hattı yapılırken, 1950'den 1997'ye kadar (47 yıl) yalnızca 936 km yeni demiryolu hattı inşa edilmiştir. Teknolojideki ve sanayideki gelişmeler sonucu özel araç maliyetlerinin de düşmesi sayesinde taşıt sahiplilik oranı gitgide artmış, bu da insanların karayolunu kullanmalarını adeta teşvik etmiştir.

Özellikle son yıllarda ülkemizde demiryollarına verilen önem nispeten artmış, yeni projeler hayata geçirilmiş ve geçirilmektedir. Demiryollarının kendisine olan talebi arttırması ve toplu taşımadaki payını %2'den çok daha yukarılara çıkarabilmesi için sunulan hizmetin kalitesinin artması da büyük önem taşımaktadır. Küreselleşen dünyada zamanın parasal değerinin artması seyahat sürelerini en aza indirmeyi zorunlu kılmakla birlikte yolculuk sırasında konfor olabildiğince yüksek tutularak, bilgisayar/internet ile çalışabilme imkânları sunulmaktadır.

Demiryollarının *erişebilirliğinin* karayoluna nazaran daha düşük olması, kapıdan-kapıya yolculuklarda olumsuz bir özellik olarak görülse de, karayolundaki araç yoğunluklarından dolayı oluşabilecek gecikme sorunlarına karşı avantaj sağlamaktadır. Ulaştırmada *hizmet düzeyini* belirleyen faktörlerden iki tanesi *dakiklik* ve *güvenilirliktir*. Dakiklik araçların önceden işletme tarafından hazırlanıp, kamuoyuna duyurulmuş olan çizelgedeki hareket saatlerine uygun hareket etmesi ile ifade edilmektedir. Güvenilirlik ise planlanan varış-kalkış zamanlarından sapmanın bir ölçüsü olarak tanımlanmaktadır. Özellikle ev-iş ve ev-okul yolculuklarında bu iki faktör büyük önem taşımaktadır.

Demiryolu, hareketi dikey ve yanal yönde sınırlanmış, yalnızca hat boyunca düşey yönde harekete izin verilen bir *kılavuzlanmış sistemdir*. Karayolu ve denizyolunda iki boyutta hareket imkanı varken, hava yolu kullanıldığında her üç boyutta da serbest hareket imkanı (belirli kurallar çerçevesine) mevcuttur. Demiryolunun bu özelliğinden dolayı, *hat kapasitesi* diğer sistemlere oranla düşük kalmakta, bu da demiryolu hatlarının *etkin* ve *verimli* kullanımını gerektirmektedir. Etkinlik, önceden belirlenmiş hedeflere kıt kaynakları olabildiğince az kullanarak erişme, verimlilik ise eldeki kaynaklar (araç, personel, yakıt,

zaman, vs.) ile en yüksek çıktıyı (taşınan yolcu/yük miktarı) elde etme olarak tanımlanabilir.

Özellikle tek hatlı demiryollarında, her iki yöne hareket eden trenlerin aynı yolu kullanma zorunlulukları, hat kapasitesinin etkin ve verimli kullanımını zorunlu hale getirmektedir. Tek hatlı demiryollarında iki komşu istasyon ya da yan hat aralığında aynı anda sadece bir yönde hareket imkânı vardır. Bu da zıt yönde hareket eden trenlerden birine, diğer tren hattı boşaltıp, makas ve sinyaller düzenleninceye değin istasyon ya da yan hatta bekleme zorunluluğunu getirmektedir.

İşletmeler tarafından hazırlanıp, kullanıcılara duyurulmuş olan çizelgelerde (orer) zıt yönde hareket eden trenlerin karşılaşacağı ve aynı yönde hareket eden trenlerden hızlı olanın yavaş olanı geçeceği yer ve zamanlar belirlidir. Trenler önceden hazırlanmış çizelgeden saptıkları takdirde, bu karşılaşma/öne geçme yer ve zamanlarında da sapmalar olmaktadır. Bu da iki trenin demiryolu hattı üzerinde aynı yerde aynı zamanda bulunması durumunlarını, yani diğer bir deyişle *çatışmaları* doğurmaktadır. Örneğin tek hatlı bir demiryolunda zıt yönde hareket eden iki trenin iki komşu istasyon arasında karşılaşması bir çatışma durumudur. *Demiryolu trafik kontrolü* problemi trenlerin önceden hazırlanmış çizelgeden sapmaları durumunda ortaya çıkmaktadır. Şahin (1996)'e göre, demiryolu trafik kontrolünün tanımı aşağıdaki gibi yapılabilir:

"Belirli bir demiryolu hattı üzerinde, gerçek işletme koşullarında (aynı ve zıt yönde) hareket etmekte olan trenlerin her birinin ilk kalkış istasyonu ve bu istasyondan en erken kalkış zamanı, (varsa) ara istasyon/yan hat (istasyon ve/veya yan hat)'lara varış ve buralardan kalkış zamanları, son varış istasyonu ve bu istasyona planlanmış varış zamanı ve trafik kontrolünün amaç ya da amaçları (hedefleri) bilinmektedir. Bu amaçlara erişmek üzere demiryolu işletmesinin fiziksel kısıtlarına uygun olan, trenlerin ilk kalkış ve son varış istasyonları arasındaki hareketlerini belirleyen hareket planlarının (ya da genel olarak çizelgenin) hazırlanmasıdır."

Bir başka ifadeyle;

Çatışan tren hareketlerinin çözüleceği istasyon/yan hatların (ya da karşılaşma/öne geçme yerlerinin) belirlenmesi, ve herhangi bir anda hangi trenin hangi hatta bulunması gerektiğinin saptanmasıdır."

Demiryolu trafik kontrolü *NP-complete* (Nondeterministic Polynomial-complete – deterministik olmayan polinom/kombinatoryal) sınıfına giren bir en iyileme problemidir. Bu tarz bir problemin zorluğu (ya da başka bir deyişle çözüm seçeneklerinin sayısı) problemin

boyutuyla üstel olarak artmaktadır. Bu nedenle, çözüme ulaşmak için harcanması gereken süre ya da gerekli bilgisayar hafızası, problemin boyutuna bağlı olarak hızla artmaktadır. Küçük boyutlu problemler geleneksel en iyileme teknikleri (ör. doğrusal programlama) sayesinde birkaç saniyede çözülebilirken; daha büyük problemlerin çözümü günümüz bilgisayar teknolojisiyle dahi milyon yılları bulabilmektedir.

Demiryolu trafik kontrolü günümüzde deneyimli trafik kontrolörleri (*dispeçerler*) vasıtasıyla yürütülmektedir. Dispeçerler, yıllar boyunca kazandıkları bilgi ve deneyimleri vasıtasıyla, karar verme yeteneklerini kullanarak, trenlerarası çatışmaları çözmektedirler. Bunu yaparken de doğal olarak problemi basitleştirme yolunu seçmektedirler. Örneğin, trenlere öncelik sırasına göre hat kesimlerini kullanma izni vermektedirler. Öncelik kuralları, belirli ölçüde kolaylık sağlarken, en iyi çözümden uzaklaşma gibi bir mahsuru da vardır (Şahin, 1996).

Çatışması aleyhinde çözülen bir trenin gecikmesi artmakta, bu sebeple çizdelgeden daha fazla uzaklaşmakta ve potansiyel çatışmaların sayısı artmaktadır. Bu da gecikmelerin ağ kesimi boyunca yayılarak diğer trenleri de etkilemesine sebep olmaktadır. Bu nedenle bir çatışmanın çözümü, mikro düzeyde en uygun çözüm olarak görünürken, makro düzeyde, yani belirli bir ağ kesimi bazında, gecikmelerin toplamlarını en iyileyecek çözümden uzaklaşmaya sebep olabilir. Şüphesiz ki sevk görevlileri çatışma çözümlerini yaparken probleme makro değil, mikro düzeyde bakmaktadırlar. Bu yüzden bir trenlerarası çatışma için kusursuz olarak görülebilecek bir çözümün bile ne denli iyi olduğunun farkına varamamalarına neden olmaktadır.

Bu çalışmanın amacı, işletim sırasında, tren hareketlerinde mevcut çizelgeye (orere) göre sapmalar meydana gelmesi durumunda, belirli bir ağ kesiminde, belirli bir zaman aralığında gerçekleşebilecek tüm çatışmaların en iyi çözümlerini bulup, toplam (ağırlıklı) gecikmeleri en aza indiren yeni bir çizelge elde etmektir. Yeni çizelge elde etmek için, karmaşık ve büyük boyutlu problemlere dakikalar içerisinde en iyiye yakın çözümler bulabilen *genetik algoritmalar* kullanılmıştır. Elde edilen sonuçlar, sevk görevlilerinin kararları ile karşılaştırıldığında ağ kesimindeki (ağırlıklı) gecikme toplamlarını azaltarak daha iyi sonuçlar sağlamaktadır.

1.2 Kapsam

Çalışmanın ikinci bölümünde, tren çizelgeleme ve yeniden çizelgeleme problemleri üzerine yapılmış olan bazı çalışmalara ilişkin kaynak araştırması sunulmaktadır.

Üçüncü bölümde, demiryolu trafik yönetiminin bazı özellikleri açıklanmıştır. Burada, trafik yönetimi probleminin tanımı yapılmakta; yönetimin eleman ve alt elemanları tanıtılmakta ve aralarındaki hiyerarşik ilişki ortaya konmaktadır. Daha sonra, trafik yönetiminin en önemli işlevlerinden biri olan 'yeniden çizelgeleme' ve 'trafik kontrolü' ele alınarak, kurulan modelin daha iyi anlaşılabilmesi için bir hazırlık yapılmaktadır.

Dördüncü bölümde, genetik algoritmalara değinilmiş ve temel çalışma ilkeleri sunulmuştur. Genetik algoritmalar, geleneksel yöntemler kullanıldığında problemin boyutu nedeniyle çözüm bulunması çok uzun süre alması durumunda kullanılan, en iyi sonucu elde etmekten ziyade, kısa sürede, en iyiye yakın derecede yeteri kadar iyi sonuçlar sunan bir yapay zeka yöntemidir.

Beşinci bölümde, trenlerarası çatışmaların çözümünü gerçekleştiren genetik algoritma sunulmuş ve değişik problemler için algoritmanın ürettiği çözümler gösterilmiştir. Geliştirilen algoritma en çok 5 dakika gibi kısa bir süre içerisinde verilen kesimde oluşacak trenlerarası çatışmaların çözümünü yapmakta ve elde ettiği en iyi (ağırlıklı gecikme toplamı en küçük olan) sonucu kullanıcıya sunmaktadır. Genetik algoritmanın çözümlerini karşılaştırmak için problemin bir matematik modeli oluşturulmuş ve en uygun (optimum) çözümü elde eden bir model çözümü bulunmuştur. Ancak matematik modelin çözüm süresinin uzunluğu nedeniyle yalnızca küçük boyutlu problemlerde kullanılabilmektedir. Problem boyutu büyüdükçe, karşılaştırma yapabilmek için, dispeçer kararlarını taklit eden bir yapay sinir ağı modeli kullanılmıştır. Genetik algoritma çözümleri, çalışma süresi ve gecikme toplamları bakımından dispeçer çözümleri ile karşılaştırıldığında, (ağırlıklı) gecikme toplamlarını azaltarak daha iyi sonuçlar sağlamaktadır.

Altıncı ve son bölümde, çalışmanın genel bir değerlendirmesi yapılmakta ve problemin kapsamının genişletilmesine ilişkin bazı öneriler sunulmaktadır.

2. KAYNAK ARAŞTIRMASI

Literatürde, demiryolu trafik kontrolü probleminin çözümüne ilişkin çalışma sayısı sınırlıdır. Bunun en önemli nedenlerinden biri, problemin en iyi çözümünün klasik matematik teknikleri kullanılarak bulunmasının neredeyse imkânsız olmasıdır. Sezgisel algoritmalar ile bulunan çözümler ise yaklaşık olmaktadır. Aşağıda, demiryolu trafik kontrolü ve yeniden çizelgeleme ile ilgili yapılmış çalışmalar incelenmiş, böylece problemin çözümüne ilişkin yaklaşımlar belirlenmiştir.

Cheryavskii (1971a, 1971b ve 1972) yolcu trenleri için hazırlanmış bir çizelgeye, yük trenleri eklemek için bir ileri bakış yöntemi kullanmıştır. Bunun için bir sezgisel algoritma geliştirmiş, geliştirdiği algoritma *iş-atölye* yaklaşımı ile çatışmaları çözmekte ve uygun bir çizelge elde etmektedir.

Amit ve Goldfarb (1971) İsrail Demiryolları'nda demiryolu trafik kontrolü üzerine çalışmışlardır. Geliştirilen sezgisel algoritmada, hat kullanımı ve zamansal kısıtlar bulunmaktadır. Hat kullanımına ilişkin kısıtlar bir yöndeki trenlere öncelik verilmediği taktirde doğrusal olmamaktadır.

Szpigel (1973) demiryolu trafik kontrolü problemini bir matematiksel program olarak ele alan ve sayısal sınamalar sunan ilk araştırmacılar arasındadır. Bu çalışmada, demiryolu trafik kontrolü problemi bir iş-atölye problemi olarak ele alınmakta, trenlerin işleri, hat kesimlerinin de makineleri temsil ettiği kabul edilmektedir. Szpigel, ağırlıklı seyir süreleri toplamını en küçüklemeyi hedeflemekte, bunun için de standart *dal-sınır algoritması* içinde kısıt yaratma yöntemini uygulamaktadır. Bu yöntemle ancak çok küçük boyutlu problemler çözülebilmekte ve bu çözümler dahi oldukça uzun süreler almaktadır (5 hat kesimi ve 10 tren için IBM/360 bilgisayar kullanılarak 30 dakikadan fazla).

Petersen (1974) tek hatlı bir demiryolu kesiminde gerçekleşebilecek karşılaşma ve öne geçme çatışmaları nedeniyle oluşacak gecikmeleri hesaplamakta ve farklı tren sayıları içeren problemlere ait seyahat sürelerini belirlemektedir. Petersen'in bu çalışması hat kapasitesi ile ilgili yapılmış çalışmaların da başında gelmektedir. Petersen (1975) çalışmasında, demiryolu hattı üzerine kısmen çift hatlı kesimler eklemekte ve istasyonlar arası mesafeleri de göz önüne almaktadır. Buna karşılık bu çalışmada yalnızca karşılaşma çatışmalarından dolayı oluşan gecikmeler göz önünde bulundurulmaktadır. Petersen (1977) bu kez hat kapasitesini Petersen (1974a ve 1974b) çalışmalarına dayanarak hesaplamaktadır.

Thomas (1974), uygulanabilir bir çizelge elde edecek bir model geliştirmiştir. Geliştirilen modelde tren hareketleri sisteme tanımlanmakta ve bir başlangıç çizelge oluşturulmaktadır. Oluşturulan başlangıç çizelge, trenlerarası çatışmaları da içeren uygulanamaz bir çizelgedir. Sistem saati 1'er dakika arttırılarak, sistemde o anda herhangi bir çatışma bulunup, bulunmadığı incelenmektedir. Çatışma tespit edildiğinde önceliği olan trenin geçmesine izin verilerek, diğer tren bir önceki istasyon ya da yan hatta bekletilmektedir. Geliştirilen modelde en fazla 20 tren ve 36 hat kesimi bulunmakta ve model en fazla 98 iterasyon yapabilmektedir. Çizelgeleme problemi olmadığı için model çevrim içi (online) çalışmamakta, dolayısıyla da oldukça hızlı çözüm elde etme zorunluluğu bulunmamaktadır. Buna karşın elde edilen çözümlerin ne derece iyi olduğuna ilişkin bir karşılaştırma sunulmamıştır.

Petersen ve Fullerton (1975) vagon atama problemi ile çizelgeleme problemini bir arada sunmaktadırlar. Geliştirilen algoritma ile toplam vagon-gün sayısı en aza indirilmeye çalışılmaktadır. Gelen vagonlar trenlere atanmakta, bu sayede farklı tren tertipleri oluşturulmaktadır. Oluşturulan trenler için çizelgeler oluşturulmakta ve gecikme değerleri hesaplanmaktadır. Bu çalışmada da ne denli iyi sonuçlar elde edildiğine dair bir veri bulunmamaktadır.

Peat, Marwick, Mitchell & Co. (1975) demiryolu trafik kontrolü için bir benzetim modeli geliştirmiştir. Her bir trenin sevk edilmesi için planlanmış zamanının gelmesi ya da hat kesiminin boşalması beklenmektedir. Sistemde oluşacak çatışmalar benzetim programı tarafından ilk giren ilk çıkar mantığı ile çözülmektedir ve herhangi bir en iyileme hedefi söz konusu değildir.

Belshaw (1976) hat kapasitesi üzerine bir çalışma sunmaktadır. Bu çalışmada, trenlerin gireceği çatışma sayısı tahmin edilmekte ve buna bağlı olarak gecikme süreleri kestirilmeye çalışılmaktadır. Elde edilen gecikme değerleri kullanılarak bir benzetim programı çalıştırılmaktadır. Benzetim sonucuna göre hat kapasitesi ve hatta çalışan tren sayısına bağlı olarak sistemde oluşacak gecikme miktarı tahmin edilmektedir.

Rudd ve Storry (1976) çizelgeleme problemi için kullanılmakta olan yöntemleri incelemekte ve yeni bir yöntem sunmaktadır. Çalışmada incelenen yöntemlerden TWS (Train Working Simulator) yöntemi, tren önceliklerini göz önüne alan bir yöntemdir. Bu özelliği nedeniyle Thomas (1974)'ın çalışmasını hatırlatmaktadır. Çatışma durumunda daima önceliği olan trenin geçmesine izin verilmekte, diğer tren bir önceki istasyon ya da yan hatta bekletilmektedir. Trenlerin farklı öncelik değerine sahip olduğu bu yöntemde, bir tren diğerini uzunca bir süre beklemek zorunda kalabilmektedir.

Çalışmada incelen diğer bir model, bir benzetim modeli olan SIMTRAC'tır. Oldukça gerçekçi olarak geliştirilen bu model, çok fazla değişken içermektedir. Bu nedenle tren sayısı arttığında SIMTRAC da sonuç elde etmede zorlanmaktadır. Ayrıca modelde *tıkanmalar* oluşabilmekte ve model farklı hat yapıları ile tren türleri için uyum sağlayamamaktadır. Diğer bir benzetim modeli olan MOST olay tabanlı bir ileri bakış yöntemi kullanmakta buna karşın farklı hat yapıları ve tren türleri için yine uyum gösterememektedir. Ayrıca modelde kullanılan 2 blok ileri ya da geri bakış yöntemi zaman zaman yeterli olmamaktadır. FIFO (first in first out – ilk giren ilk çıkar) yöntemi, çizelgelemede kullanılan bir diğer yöntemdir. Bu yöntemde çatışma saptandığında, hat kesimine ilk giren trenin hareketine izin verilmekte, diğer tren ise bekletilmektedir. Bu yöntem kullanıldığında her zaman uygulanabilir çizelgeler elde edilememekte, zaman zaman sistemde tıkanmalar da oluşabilmektedir. FRISCO adı verilen bir diğer yöntem ise temel olarak TWS yöntemine benzemektedir. Bu yöntemde de sistem saati 1'er dakika arttırılarak sistemde çatışma bulunup, bulunmadığı incelenmekte, çatışma saptandığında tren öncelikleri ve çatışma çözümü sonucunda elde edilecek gecikme değeri göz önüne alınarak çatışma çözülmektedir.

Araştırmacıların geliştirdiği yöntem olan STS'de olay tabanlı bir benzetim modeli kullanmaktadır. Modelde, her trenin karşılaşacağı bir sonraki olay (çatışma) saptanmakta ve bu olaylar depolanmaktadır. Depolanan olaylar çözüldüğünde dengeli bir çizelge hızlı bir biçimde elde edilebilmektedir. Oluşan çizelgeler, diğer yöntemlerle kıyaslandığında daha az gecikme toplamları içermektedir.

Jelaska (1976) tren çizelgeleme problemi üzerinde durmaktadır. Bilgisayar kullanılarak bir başlangıç tren çizelgesi oluşturulmakta, oluşan çizelge üzerindeki tren hareketleri değiştirilerek iyileştirilmeye çalışılmaktadır. Kullanılan yöntem, en iyi çizelgenin elde edilmesini garanti etmemekte, buna karşın uygulanabilir ve iyi çizelgeler sunmaktadır.

Dube ve Belshaw (1977) ile Belshaw ve Hallonquist (1978) çalışmalarında çizelgeleme problemini Kanada için uygulamaktadırlar. Bunun için bir benzetim modeli sunulmaktadır. Modelde 3 farklı tür tren bulunmaktadır ve her bir tren türünün bir diğeriyle çatışmaya girmesi durumunda bekleme süreleri bir matris halinde önceden tanımlanmaktadır. Modelde hat ve istasyon yapıları da farklılaştırılabilmektedir. Benzetim modeli, uygulanabilir bir çizelge sunmakta ancak sunduğu çizelgenin ne kadar iyi bir çizelge olduğuna dair bir ölçüt bulunmamaktadır.

Wong ve Rosser (1978) tren çizelgelemesi problemine farklı bir yaklaşım sunmaktadır. Yeni Zelanda demiryolu ağı için geliştirilen benzetim modelinde, beklemelerden dolayı ağırlıklı

gecikme değeri en büyük olan trenin kalkış saati değiştirilerek, gecikmeler azaltılmaya çalışılmaktadır. Bir tren için sonuç elde edildiğinde, bir sonraki tren için aynı işlemler tekrarlanmaktadır. Daha iyi sonuç elde edilemeyene kadar bu işlem tekrarlanıp, daha iyi bir çizelge oluşturulmaktadır.

Purdon ve diğ. (1978) 3 ayrı olay için, sistemde bir sorun meydana geldiğinde oluşabilecek gecikmeleri incelemektedir. Bu işlem için bir benzetim modeli çalışmada sunulmaktadır. Benzetim modelinin sonuçları, gerçek sonuç ve kuyruk kuramının sonuçları ile karşılaştırılmaktadır.

Elbrond (1978) tek hatlı bir demiryolu kesiminin kapasitesini çeşitli kısıtlar altında belirlemek için bir algoritma sunmaktadır.

Steiner (1978) yeni gelişmekte olan bilgisayar teknolojilerinin tren dispeçerlerine yardımcı olmak için nasıl kullanılabileceğini inceleyen teorik bir çalışma sunmaktadır.

Yokota (1979) çift hatlı demiryolu kesimlerinde bulunan yan hatların dizaynına bağlı olarak verimini matematiksel olarak incelemekte ve (1980) aynı araştırmayı tek hatlı kesimler için sunmaktadır.

Ohtsu (1979) Tokyo-Shinkansen hattı için bir benzetim yazılımı gerçekleştirmektedir. Çalışmada mevcut tren çizelgelerinin ne kadar iyi oldukları incelenmektedir. Çizelgelerin iyiliğini belirleyen ölçüt ise, beklenmeyen bir olay gerçekleştiğinde, ya da diğer bir deyişle trenlerarası çatışma meydana geldiğinde, çizelgelerin eski haline dönebilme hızıdır.

Assad (1980) literatürdeki demiryolu ulaşım modellerini iki hedef açısından incelemektedir. Bunlardan ilki demiryolu hattının modellenmesi, ikincisi ise demiryolu ulaşım modellerini literatürdeki diğer ulaştırma modellerinin içindeki yerini saptayan bir kılavuz oluşturmaktır. En iyileme, kuyruk ve benzetim modellerinin tamamı en iyileme amacıyla kullanılmaktadır. Her bir model türünün yeri, demiryolunun planlama etkinliklerindeki işlevine bağlı olarak incelenmektedir.

Hasegawa ve diğ. (1981) tren gecikmelerinin diğer trenlere yayılımını inceleyen bir makro model sunmaktadırlar. Modelde, tren gecikmelerinin diğer trenlere yayılımı, trafik akım kuramındaki şok dalgalarına bezetilmektedir. Geliştirilen modelin sonuçları ile benzetim sonuçlarının birbirine oldukça yakın olduğundan bahsedilmesine karşın, çalışmada sunulan sonuçlar oldukça sınırlıdır.

Petersen ve Taylor (1982) bir demiryolu hattı için geliştirilen bir benzetim ya da en iyileme modelinin çerçevesini, ya da diğer bir deyişle içermesi gereken olguları tanımlamaktadır.

Tanım yapıldıktan sonra, davranışsal bir sevk etme modeli sunmaktadır. Model farklı blok uzunlukları bulunan, hem tek hem de çift hatlı kesimleri bulunan 104 mil uzunluğundaki bir demiryolu hattında sınanmıştır. Bu hat üzerinde, bir gün içerisinde, 3 farklı öncelik değerine sahip 51 adet tren sevk edilmektedir. Çalışmada 30 ayrı gün için sınama sunulmakta, tren başına ortalama gecikme 55,3 dakika, gecikmelerin standart sapması da 42,1 dakika olmaktadır.

Lusicic (1983) mevcut bir çizelgeye tren eklenmesi problemini ele almaktadır. Çalışmada yakıt sarfiyatının en aza indirilmesi için eklenecek trenlerin en uygun duruş yer ve zamanları saptanmaktadır.

Sauder ve Westerman (1983) tarafından yayınlanan çalışma, demiryolu trafik kontrolünün uygulamaya yönelik ilk çalışması olarak göze çarpmaktadır. Çalışmada, tren dispeçerleri için bir karar destek sistemi sunulmaktadır. Sunulan karar destek sisteminin amacı tren gecikmelerinin ağırlıklı toplamını en küçüklemektir. Araştırmacılar, en kısa yol algoritması ve doğrusal programlama teknikleri kullanıldığında en uygun sonucun elde edilemediğini, ancak dal ve sınır algoritması kullanıldığında en uygun sonucun bulunabildiğini öne sürmektedirler. Çözümdeki zaman aralığı altı ve sekiz saat olarak seçilmiş ve yirmi saatlik bir ileri bakış zaman aralığı kullanılmıştır. Bu yöntemle ağırlıklı gecikme toplamları 457 dakikadan 300 dakikaya kadar indirilebilmiş ve yıllık 3 milyon Amerikan Doları tasarruf sağlanabildiği belirtilmiştir. Ancak bu yöntem yüksek trafik hacimlerinde ve geniş zaman aralıklarında kullanılamamaktadır.

Demiryolu trafik kontrolü problemi için Japon araştırmacılar uzman sistemler tekniğini kullanarak çözüm sunan ESTRAC (Expert System for Train Traffic Control – Tren Trafik Kontrolü için Uzman Sistem) sistemi geliştirmeye 1982 yılında başlamışlardır. ESTRAC, bilgi mühendisliği teknolojisinin, demiryolu şebekelerindeki yeniden çizelgeleme problemine ilk uygulamasıdır. Bu sistemin üç farklı sürümü bulunmaktadır. ESTRAC-I (Araya ve ark., 1983), ESTRAC-II (Araya ve Fukumori, 1985) ve ESTRAC-III (Komaya ve Fukuda, 1989).

İlk versiyon olan ESTRAC-I'de istasyonlardaki trenlerin sıralaması değiştirilmekte, bunun için de sezgisel kurallar kullanılmaktadır. Eğer-sonra (if-then) kuralları kullanan ve tren sıralamalarını dal-sınır yöntemiyle gecikmeleri en küçüklemeyi hedefleyen ESTRAC-I'de 10 istasyon ve 10 tren için 1 dakika içerisinde sonuç elde edilebilmektedir. ESTRAC-I'de sadece küçük ve basit demiryolu şebekeleri için çözüm elde edilebilmesine karşın, yeni bir teknoloji olan bilgi mühendisliğinin, yeniden çizelgeleme problemine uygulanması açısından önemlidir.

İkinci versiyon olan ESTRAC-II, tren sevk görevlilerinin gerçek zamanlı işletme koşullarında kullandıkları yeniden çizelgeleme kurallarının bir bilgisayar uygulamasıdır. Geliştirilen yazılımda tren çizelgeleri bilgisayar ekranı üzerinde görüntülenmekte ve çizelge üzerine fare yardımıyla değişiklikler yapılabilmektedir. ESTRAC-II'de tren sevk görevlilerinin uyguladığı yöntemler bir kural tabanında tanımlanmış, bu kurallar kullanılarak, yeniden çizelgeleme etkili bir biçimde yapılmıştır. Program 14 istasyon ve 30 tren için 2 dakika içerisinde çözüm üretebilmektedir.

Son versiyon olan ESTRAC-III'de yeni bir benzetim yöntemi kullanılmaktadır. Bu yöntemde birbiriyle ilişkili olan trenlerin hareketleri benzetim ünitesiyle öngörülmekte, yeniden çizelgeleme için basit eğer-sonra kuralları kullanılmaktadır. Böylece tren sevk görevlilerinin problem çözme teknikleri bilgisayar ortamında taklit edilebilmektedir. ESTRAC-III'de 40 istasyon ve 60 tren içeren büyük boyutlu ve karmaşık demiryolu şebekeleri için 5 saniye gibi kısa bir süre içerisinde çözüm elde edilmiş, bu sayede tren sevk görevlilerini verdikleri kararlarda etkin ve etkili bir şekilde destekleyebileceği kanıtlanmıştır. ESTRAC-III yalnızca karşılaşma çatışmalarını değerlendirmekte, öne geçme çatışmalarını dikkate almamaktadır.

Quinchon (1984) çizelgeleme problemini Fransız Demiryolları'na uygulamıştır. Öncelikle farklı lokomotiflerin özellikleri göz önünde bulundurulmaktadır. Çizelgeleme yapılırken trenlerin tabii ve en düşük seyir süreleri ile kaybedilen zamanı geri kazanma payı değerlendirilmektedir. Trenler için seyir diyagramları oluşturulmakta, oluşturulan diyagramlarda trenlerin diğer trenlerle bağlantıları da göz önüne alınarak çizelge oluşturulmaktadır.

Kimura ve Koga (1985) farklı tren türleri için seyir diyagramlarının nasıl olması gerektiğini incelemekte, bunu yaparken de enerji tüketiminin en küçüklenmesini hedeflemektedir.

Bronzini ve Clarke (1985), trenlerin hat boyunca sevk işlemini kendiliğinden gerçekleştiren bir yazılım sunmaktadır. Bu yazılım hat tıkanıklıklarını da hesaba katarak, uygulanabilir bir çizelge oluşturmaktadır. Oluşturulan çizelgeye uygun olarak hat kapasitesi kontrolü yapılmaktadır.

Welch ve Gussov (1986) hat kapasitesini Kanada Demiryolları için incelemektedir. Gelecekte oluşabilecek talebi karşılayabilmek için, kapasiteyi ölçecek bir yazılım gerçekleştirilmiştir. Çalışmada farklı senaryolar için çözümler sunulmaktadır.

Crainic ve Guerin (1987) tek hatlı bir demiryolu kesiminde her istasyon aralığı için ayrı ayrı kapasiteyi belirlemektedir. Belirlenen kapasitelere göre seyahat süreleri ve tren gecikmeleri

saptanmaktadır. Yapılan çalışmanın yük trafiği için kullanılması hedeflenmektedir.

Petersen ve Taylor (1987) hızlı trenler için gerekli yan hatların yerini ve sayısını matematiksel olarak belirleyen bir çalışma sunmaktadır. Daha esnek çizelgeler elde edebilmek için, bu verilerde serbestlik değerleri kullanılmaktadır. Tek hatlı bir demiryolu kesiminde oluşacak gecikmeler incelenmekte ve bu değerler kesimin çift hatlı olması durumunda oluşacak gecikme değerleri ile karşılaştırılmaktadır.

Kraft (1987) tek hatlı bir demiryolu kesiminde oluşacak gecikmeleri en aza indirmek için geliştirdiği algoritmada dal ve sınır yöntemini kullanmaktadır. Yerel bir en iyileme yöntemi kullanıldığından ilk 20 – 30 döngüde iyi sonuç elde edilebilmektedir. Bu yöntem kullanıldığında, sistemde oluşan toplam gecikmelerin %15 - %55 oranında azaltılabildiğinden bahsedilmektedir. Örnek problemlerde az sayıda tren çalışmaktadır ve büyük boyutlu problemlerde çalışması için gerekecek süre bilinmemektedir.

Greenberg ve diğ. (1988) tek hatlı demiryolu kesimlerinde ve buna alternatif olarak uygulanabilecek farklı kesimlerde oluşacak sevk etme gecikmelerini kestirmektedir. Trenlerin istasyona Poisson Dağılımı'na uygun olarak geldikleri kabul edilmekte ve problem kuyruk kuramı ile modellenmektedir.

Petersen ve Taylor (1988) Welland Kanalı'ndaki teknelerin çizelgeleme problemi için bir çözüm yöntemi sunmaktadırlar. Bu problem doğası nedeniyle yalnızca karşılaşma çatışmaları içeren bir demiryolu trafik kontrolü problemine benzemekte, kanalda kot değişimi yapılan kesimler, demiryolundaki tek hatlı kesimlere benzetilebilmektedir. Geliştirilen modelin amacı, tekne gecikmeleri toplamının en küçüklenmesidir. Yazarlar öncelikle yaklaşık bir çözüm yöntemi geliştirmekte, daha sonra gemilerin serbestlik derecelerini de göz önüne alarak, dinamik programlama tekniği kullanıp, gecikme toplamlarını en aza indirmeye çalışmaktadırlar. Geliştirilen sezgisel algoritma, yeteri kadar hızlı sonuçlar üretebilmesine karşın, sunulan çözümlerin ne kadar iyi olduğuna dair bir sayısal bulgu ortaya konulmamaktadır.

Jovanovic ve Harker (1989) çizelgeleme problemi için SCAN adında bir karar destek yazılımı geliştirmişlerdir. Yazılım 3 ayrı birimden oluşmaktadır. İlk birim, önceden tanımlanmış demiryolu hattı için oluşturulan çizelgelerin uygulanabilirliğini ölçmektedir. İkinci birim, uygulanabilir bir çizelge elde edilene kadar, oluşturulan çizelgede değişiklik yapmaktadır. Üçüncü ve son birim ise çizelgelerin güvenilirliğini ve esnekliğini ölçmektedir. Uygulanabilir çizelge oluşturma probleminin tamamen karmaşık (NP-Complete) sınıfına girdiği ve iş-atölye problemine oldukça benzediğinden bahsedilmektedir. Örnek problemler 24 saatlik bir zaman

penceresinde çalışmakta ve 30 dakika içerisinde çözüm elde edilebilmektedir.

Mills ve Perkins (1989), çizelgeleme problemi için farklı bir yöntem sunmaktadır. Hat boyunca her bir trenin istasyonlara ne zaman varması gerektiğini incelemiş, bunu gerçekleştirebilmek için de her hat kesiminde uygulaması gereken hızlar belirleyen bir model geliştirmiştir. Enerji tüketiminin en küçüklemeyi amaçlayan model dinamik programlama yöntemini kullanmaktadır.

Harker ve Hong (1990) kısmen çift hatlı kesimler de içeren tek hatlı bir demiryolunda gerçekleşebilecek gecikme olasılıklarını ölçmekte ve gecikme miktarlarını bu olasılıklara göre belirlemektedir. Gecikme miktarları hesaplandıktan sonra, çizelgenin güvenilirliği, esnekliği baz alınarak ölçülmekte ve daha esnek çizelge oluşturmak için değişiklikler yapılmaktadır.

Jovanovic ve Harker (1990), demiryolu trafik kontrolü probleminin amacının ağırlıklı gecikmelerin yanında enerji tüketiminin de en küçüklenmesi olması gerektiğinden bahsetmektedir. Karar destek sisteminin bileşenleri olarak, enerji maliyeti, gecikme maliyeti, taşıt değeri, nakliye değeri ve personel maliyetinin kullanılması gerektiği, ayrıca oluşturulan çizelgelerin esnek olmasının zorunlu olduğunu ve hızlı elde edilebilmesi gerektiği ileri sürülmektedir. Hat bakımını gerçekleştiren araçların da yavaş hareket eden bir tren olarak düşünülüp, hareketlerinin çizelgeye eklenmesi gerektiği ve çizelgeye sonradan ilave edilebilecek trenlerin de kolaylıkla karar destek sistemine tanıtılabilmesi gerektiğinden bahsedilmektedir.

Kraay ve diğ. (1991), enerji tüketimini göz önüne alan bir model sunmaktadır. Geliştirilen modelin amacı, tren gecikmelerinin en küçüklenmesinin yanında, trenin hat boyunca seyir hızının farklılaştırılması yoluyla seyir diyagramını oluşturmaktır. En uygun seyir diyagramı oluşturma problemi, gecikmelerin en küçüklenmesini amaçlayan geleneksel trafik kontrolü probleminden daha zordur, çünkü problem trenlerin komşu istasyonlar arasındaki seyir sürelerinin artması ile enerji maliyeti azaltılırken, diğer yandan gecikmelerin de en aza indirilmesinin bir arada hedeflendiği bir çok amaçlı karar verme problemi halini almaktadır. Kraay ve diğ. doğrusal olmayan karışık tamsayılı matematik program için üç farklı çözüm yöntemi geliştirmişlerdir; tam iteratif algoritma, uygulanabilir küme üretimine dayalı kesin algoritma ve üretilmiş uygulanabilir küme filtrelemesine dayalı bir sezgisel algoritma. Çalışmada sunulan sezgizel algoritmanın ilk uygulamaları, çözümü olan küçük boyutlu problemler için en uygun çözümün yeterli bir yaklaşıklıkla tahmin edilebileceğini göstermektedir.

Jovanovic ve Harker (1991), SCAN yazılımını geliştirerek, dal ve sınır yöntemi ile oldukça hızlı bir biçimde çizelge oluşturmaktadır. Geliştirilen yazılım, karşılaşma çatışmalarının yanı sıra öne geçme çatışmalarını da çözerek uygulanabilir bir çizelge oluşturmaktadır. Hattın tıkanması ve yan hat kapasitesi de yazılımda göz önüne alınan değişkenler arasında bulunmaktadır. Dal ve sınır yöntemindeki dallar, trenleri ve trenlerin karşılaşacakları istasyonları temsil etmektedir. Uygun bir çizelge elde edildikten sonra, daha iyi çözümler araştırılmaktadır. Uygulanabilir bir çizelge elde edilemediği takdirde, en fazla çatışmanın çözülebildiği dal görüntülenip, çözülememiş çatışmalar ile her bir trenin gecikme miktarları görüntülenmektedir. Dispeçerler, görüntülenen çizelge üzerinde değişiklikleri gerçekleştirip, çözülememiş çatışmaları çözme imkanına sahiptir. Çalışmada son olarak, oluşturulan çizelgelerin güvenilirliği ölçülmektedir.

Mills ve diğ. (1991) küçük boyutlu demiryolu trafik kontrolü problemleri için, çatışmaların çözülmesi gereken istasyon ya da yan hattı belirleyen ve bunu sağlamak için tren hızlarını farklılaştıran bir matematik model geliştirmişlerdir. Matematik model, çok fazla sayıda değişken içerdiği için büyük boyutlu problemler için ve yeniden çizelgelemede uygulanamamaktadır.

Mess (1991) matematiksel yöntemler kullanarak, iyi çizelgeler oluşturan bir algoritma sunmaktadır. Çalışmada herhangi bir en iyileme hedefi bulunmamakta ve başka bir yöntem ile karşılaştırma sunulmamaktadır.

Jiaxin ve Howlett (1992) enerji sarfiyatının en aza indirilmesi için, kritik hızların belirlendiği bir matematik model geliştirmişlerdir.

Petersen ve Merchant (1992), yük trenleri için, gerekli vagon sayısını belirleyip, oluşturulan her bir trenin başlangıç ve son noktalarını belirleyen ve geliştirilen çizelgelerdeki her bir çatışmanın çözüldüğü bir model sunmaktadırlar. Dinamik programlama ile geliştirilen modelde, dal ve sınır yöntemi kullanılarak çizelgelerin uygunluğu incelenmektedir. Geliştirilen modelin hesap yükünü azaltmak için, bir takım sezgisel yöntemler kullanılmaktadır.

Cai ve Goh (1994) tren çizelgeleme problemi için bir 0 ve 1 tam sayılı programlama algoritması sunmaktadır. Geliştirilen algoritma bir amaca yönelik en iyi çizelgeyi sunmak yerine, en kısa yolları inceleyerek, kısa zamanda uygulanabilir bir çizelge elde etmeyi amaçlamaktadır.

Carey ve Kwiecinski (1994) çift hatlı kesimlerdeki öne geçme çatışmalarından doğan tren gecikmelerini incelemektedir. Tren takip aralığı matematik bir modelle belirlenip, bu model sonucuna göre trenlerin gecikme miktarları hesaplanarak, tahmini varış süreleri saptanmaktadır.

Carey (1994a ve b) tek hatlı kesimlerde tren rotalama problemi için bir matematik model sunmaktadır. 4 tren ve 6 çift hatlı kesim içeren bir modelin çözümünün çok uzun sürdüğünden bahsedilmektedir. En uygun çözümün bulunmasına alternatif olarak trenler ayrı ayrı rotalanıp, çizelgede meydana gelen çatışmalar çözülmektedir. Model temel olarak, trenlerin kullanacağı hat kesimlerini belirlemektedir. Algoritmayı hızlandırmak için sezgisel algoritmalar kullanılıp, dal ve sınır yöntemi ile çözüm elde edilmektedir.

Özekici ve Şengör (1994), tekrarlı bir çizelgeye sahip hatlar için, en iyi sevk yöntemini saptayan bir çalışma sunmaktadır. Gecikme durumunda trenlerin hızlandırılması ve tamamen iptal edilmesi gibi alternatif çözümler göz önüne alınmaktadır. Karar destek sistemi olarak kullanılabilen modelin çıktısına göre sevk etme konusunda alınacak kararlar saptanmaktadır.

Salim ve Cai (1995), genetik algoritmaları trafik kontrolü problemine uygulayan ilk çalışmadır. Geliştirilen genetik algoritmadaki genlerin her biri iki tabanlı bir sayı halındedir. Kromozomlar, n istasyon sayısını, m tren adedini göstermek üzere $n*m$ boyutludur. Her bir gen, trenin istasyonlarda yaptığı duruşları temsil etmektedir. 0 değeri trenin bekleme yaptığını gösterirken, 1 değeri ise herhangi bir bekleme olmadan geçtiğini göstermektedir. 12 istasyon ve 9 tren içeren bir problem için, 100 bireyden oluşan bir toplum ve 10000 hesaplama adımı için ancak 1.5 saatte çözüm elde edilebilmektedir.

Kraay ve Harker (1995), yük trenlerinin çizelgelerinin en iyilenmesi üzerine bir model sunmaktadır. Oluşturulmuş trenlerin, istasyonlarda vagon bırakıp alması da söz konusu olmaktadır. Problemin doğrusal programlama ile çözümü oldukça uzun sürdüğü için bir sezgisel algoritma geliştirilmiştir. Uygulanabilir bir çizelge elde edildikten sonra, yerel arama teknikleri ile daha iyi bir sonuç aranmaktadır. Bu yöntemde, tüm çatışmaların çözüldüğü yer sabit tutulup, her seferinde sadece bir çatışmanın çözüldüğü yer değiştirilip, daha iyi bir çizelge elde edilmeye çalışılmaktadır. Bu yöntem kullanıldığında 1 dakika içerisinde iyi çizelgeler elde edilebilmektedir.

Pyrgidis (1995), hat kapasitesini inceleyen bir çalışma sunmaktadır. Çalışmada hat kapasitesine etkili olan her bir etken detaylı olarak incelenmiş ve çeşitli grafikleri çizilmiştir.

Shen ve diğ. (1995), bir kargo sevk görevlisinin kararlarını öğrenip, taklit etme yeteneğine sahip bir yapay sinir ağı sunmaktadır.Yapay sinir ağının eğitilmesi 24 saat gibi uzun süreler alabilmekte, ancak öğrenmiş bir ağ, gelen isteklere hızlıca iyi kararlar verebilmektedir.

Higgins ve diğ. (1996), çizelgeleme probleminde farklı tren ve istasyon/yan hat sayıları için kullanılabilen, tek hatlı kesimlerde en uygun çözümü elde eden bir model sunmaktadır. Sunulan modelin ana amacı, işletme maliyetlerini en aza indirmektir. Modelde dal ve sınır algoritması, tabu araştırması ile birlikte kullanılmaktadır.

Howlett (1996) da tıpkı Kraay (1991) gibi yakıt tüketimini en küçüklemeyi amaçlamaktadır. Bunun için, tren seyir diyagramlarının düzenlemesi konusunu incelemiştir. Bunun yanı sıra, yakıt tüketiminin en küçüklenmesi için her bir hat kesiminde korunması gereken en uygun hızı saptamaktadır.

Chang ve Tia (1996), toplu taşıma sistemlerinde beklenmedik ve ani bir talep artışı olduğunda devreye giren çizelgeye yeni bir taşıt yerleştirme konusunda bir karar destek sistemi olarak çalışan bir bulanık mantık modeli sunmaktadır. Modelin giriş değişkenleri olan süreklilik, riskler ve tıkanıklık, bulanık sayılar halinde tanımlanırken, modelin çıktısını bulanık olmayan taşıt ekleme kararı oluşturmaktadır.

Nachtigall ve Voget (1996), demiryolu hatları arasında aktarma yapacak yolcuların bekleme zamanları toplamını en küçükleyen bir genetik algoritma sunmaktadır. Geliştirilen programda öncelikle bir sezgisel kullanılarak uygun bir çözüm elde edilmekte ve elde edilen çözüm genetik algoritma ile iyileştirilmektedir.

Ferreira ve Higgins (1996), trenlerin toplam seyahat sürelerini azaltırken, varış zamanlarının da güvenilirliğini arttıran bir çok amaçlı çizelgeleme problemi üzerinde çalışmaktadır. Çalışmada, önceki çalışmalardan farklı olarak gecikme riski değişkeni de göz önünde bulundurulmaktadır. Sunulan modelde toplam seyahat süresi ve trenlerin gecikme riski en aza indirilmektedir. Model, 14 istasyon/yan hat üzerinde çalışan 31 tren için sınanmaktadır. Model sonucu elde edilen çizelgede, ilk çizelgeye oranla %50 daha az gecikme riski bulunmaktadır.

Bussieck ve diğ. (1996), yolcu trenlerine hizmet veren hatlarda oluşan yolcu talebini en iyi karşılayabilecek hat yapısını ve sefer sıklığını inceleyen bir çalışma sunmaktadır.

Frank (1996), tek hatlı bir demiryolu kesiminde tek ve çift yönlü demiryolu trafiğinin matematik modelini sunmaktadır. Trenlerin devir süreleri ve gerekli tren sayısı elde edilip, hat kapasitesi hesaplanmaktadır.

Chiu ve diğ. (1996), demiryolu trafik kontrolü problemi için bir sezgisel algoritma sunmaktadır. Geliştirilen algoritma gecikmelerin en küçüklenmesini hedeflemektedir. Algoritmanın gerçek hayat problemleri için sınanıp iyi sonuçlar verdiği öne sürülmektedir ancak bu sonuçlar çalışmada gösterilmemiştir.

Hallowell ve Harker (1996), çift hatlı kesimleri de içeren tek hatlı bir demiryolu için çizelgelerin başarımını ölçmektedir. Öncelikle her bir trenin gecikme olasılıkları hesaplanmakta, daha sonra karşılaşma ve öne geçme çatışmaları sonucu oluşacak gecikme olasılıkları hesaplanmaktadır. Sunulan matematik model ile, her trenin gecikme değeri hesaplanıp, çizelge başarımı ölçülmektedir. Çalışma sonunda modelde elde edilen gecikme değerleri ile gerçekleşen gecikme değerleri karşılaştırılmaktadır. Modelin, tüm trenlerin gecikmelerini 20 dakikalık bir hata payı ile tahmin edebildiğinden bahsedilmektedir. Hallowell ve Harker (1998) bir diğer çalışmalarında, geliştirdikleri bu modelin başarımını sınamaktadır. Modelde 3 günlük bir zaman penceresi içerisinde çalışılmakta, 4'er saatlik zaman dilimleri için çizelge oluşturulmaktadır. Elde edilen çizelgeler, geleneksel yöntemlerle elde edilen çizelgeler ile karşılaştırılarak başarımı ölçülmektedir. Trenlere belirli bir gecikme payı değeri atandığı için, yolcu trenleri dışında kalan trenler genellikle erken ulaşmaktadır.

Şahin (1996 ve 1999), demiryolu trafik kontrolü probleminin matematiksel modelini oluşturmuştur. Çalışmada, dispeçerlerin karar davranışlarına ilişkin bir çok nitelikli seçim modeli sunulmaktadır. Değişik niteliklerin ağırlıklarının belirlenmesi için doğrusal programlama kullanılmaktadır. Çalışmada ayrıca, trenlerarası çatışma yönetimine dayalı trafik kontrolü için bir sezgisel algoritma sunulmaktadır. Algoritmanın oluşturulmasında bir sistem yaklaşımı kullanılmıştır. Algoritmanın çekirdeğinde, en erken çatışma bulunmaktadır. Algoritma, bu çatışmanın alternatif çözümlerini bir sistem yaklaşımıyla değerlendirerek, her bir çözümün uygulanması durumunda sistemde meydana gelecek beklenen toplam gecikmeleri hesaplamaktadır. Daha sonra, sistemde en az gecikme (ya da çizelgeden en az sapma) oluşturulacak alternatif seçilmektedir. Bu işlem, en erken çatışmaların sırasıyla çözülmesiyle devam etmektedir. Algoritma, etkin trafik kontrolü için gerekli olan "dinamik öncelik sayısı" kavramını da kullanmaktadır. Üretim planlamasında "kritik oran" olarak bilinen öncelik kuralı, her tren için belirlenen ve trenin seyri boyunca yenilenen dinamik öncelik sayısının (bekleme olasılığının) belirlenmesinde kullanılmaktadır.

Kesin çözümü elde eden doğrusal programlama yönteminde, 10 buluşma nokası ve 8 tren ila 19 buluşma noktası ve 6 tren arasında olan örnek problemlere 30 saniye ila 125 dakika arasında çözüm bulunabilmektedir. Dispeçerin kararlarını taklit etmeyi hedefleyen sezgisel

algoritma %80 başarıyla çalışmaktadır.

Salim ve Cai (1997), çevresel etkileri en aza indirmek için, doluluğu fazla olan trenleri daha az bekletmeyi hedefleyerek çizelge oluşturan bir genetik algoritma sunmaktadır. Algoritmadaki genler, yazarların 1995 yılındaki çalışmasında olduğu gibi trenlerin istasyonlardaki bekleme durumlarını temsil etmektedir. 12 hat kesimi için 9 tren içeren problemlerden 15 tren içeren problemlere kadar denemeler sunulmuştur. 10000 ila 20000 tekrarlama adımı sonucunda çözüm elde edilebilmektedir.

Higgins ve diğ. (1997), ağırlıklı tren gecikmelerini en küçükleyen bir çalışma sunmaktadır. Yerel arama, genetik algoritma, tabu araması (tabu search) ve iki adet melez model geliştirilmiştir:

- Yerel arama: Dal ve sınır yöntemi kullanılarak uygun bir başlangıç çözümü geliştirilmektedir. Aramaya geliştirilen uygun çözümden başlanmaktadır. Çatışmalar, sıralanarak çözülmektedir. Arama esnasında her çatışma çözümü komşu istasyon/yan hatta kaydırılmaktadır.

- Genetik Algoritma: Kromozomdaki gen değerleri bekleyen tren, geçen tren ve çatışmanın çözüldüğü aralık olarak 3'lü gruplanarak düzenlenmektedir. Çaprazlama işlemi sonucunda eğer aynı çatışma kromozomda tekrarlanmışsa, o kromozomdan çıkarılmaktadır. Bu yöntemle tüm çatışmalar çözümlenmemiş olabilmektedir. Başkalaşım işleminde çatışmanın çözüldüğü istasyon rasgele değiştirilmektedir. Çok fazla sayıda uygunsuz çözüm sağlama riski bulunmaktadır. Başkalaşım işleminin, çaprazlamadan önce gerçekleştirilmesi doğru bir yaklaşım değildir.

- Tabu arama: Yerel aramaya benzer bir yöntem kullanılmaktadır. Eğer algoritma yerel en küçük noktalara takılmışsa, eğime ters yönde aramaya devam edilmektedir. Önceden elde edilmiş çözümlere (tabu çözümler) geri dönülmemektedir.

- Melez1: Genetik algoritmaya bir işlemci olarak yerel arama eklenmektedir. Genetik algoritmanın en iyi çözümlerinin %5'ine her çaprazlama sonrası yerel arama uygulanmaktadır.

- Melez2: Genetik algoritmaya işlemci olarak bu kez tabu arama eklenmektedir. Bir birey ebeveyn olarak seçildiğinde, onunla en iyi çiftlenebilecek birey aranmaktadır. Bu sayede bir sonraki nesilde tabu bireye rastlanmamaktadır.

50 trene kadar, 113 çatışma içeren problemler için çözüm sunulmaktadır. En iyi sonuçları sırasıyla genetik algoritma, melez1 ve melez2 sağlamaktadır. Melez modellerde çözüm süresi 5 katına kadar çıkabilmektedir. Belli bir zaman aralığında (5-50 sn arası) çözüm elde edilmek istendiğinde genetik algoritma ve melezler yetersiz kalmaktadır.

Leclerc ve Potvin (1997), kurye servisi problemi üzerinde durmaktadır. Gelen istek için en uygun araç genetik algoritma ile belirlenmektedir. En iyi uyum değerine sahip araç seçilip, istenilen yere sevk edilmektedir. Gösterim dallar değil, kromozomlar halinde yapılmaktadır. Sevk görevlisinin kararlarıyla uyum göstermeyen sonuçlar elde edilmektedir.

Ho ve diğ. (1997), demiryolu hat kesişimlerinde oluşan trafik kontrolünü ele almaktadır. Geliştirilen algoritma bir olay tabanlı trafik akım modeli içermekte ve dinamik programlama kullanılmaktadır. Yeniden çizelgeleme işlemi boyunca yeni bir trenin sisteme dahil olmadığı yalıtılmış çatışmalarda model en uygun çözüme oldukça yakın çözümler sunabilmektedir. Yeni trenlerin dahil olabildiği durumlarda olasılıksal yaklaşımlar kullanılmaktadır. Algoritma tarafından geliştirilen çizelgelerdeki toplam gecikmeler, ilk giren ilk çıkar yöntemi gibi geleneksel çözümlere nazaran %10 daha iyi sonuç vermektedir.

Ferreira (1997), yük trenlerinin işletme koşullarını inceleyen bir çalışma sunmaktadır. Çalışmada uygulanılan çizelgeleme ve yeniden çizelgeleme yöntemleri de incelenmektedir.

Andersson ve diğ. (1997), demiryolu trafik kontrol problemine bir sistem analizi yaklaşımıyla bakmaktadır. Bir sistemi kontrol edebilmek için, dört gereksinimin karşılanmış olması gerektiğinden bahsedilmektedir:

- Hedefin durumu
- Modelin durumu
- Gözlemlenebilme durumu
- Kontrol edilebilme durumu

amana bağımlı bir sistemin gerçek zamanlı kontrolünün analizinde, karar ve işlem arasındaki ayrımın önemli olduğundan bahsedilmektedir. Kararların etkileri, sevk görevlisinin hedeflediği kuramsal etkilerdir. Kontrol işlemlerinin etkileri ise, sevk görevlisinin hedeflediklerini gerçekte ne kadar uygulayabildiğidir. Eğer sevk görevlisinin zamanı az ise, kötü eşleme ya da beklenmedik bir durumu fark etmesi durumunda, duraklaması (eğer mümkünse) ve durumu tekrar gözden geçirmesi gereklidir.

Anderrson ve diğ. (1998), dispeçerler ile yapılan görüşmeler ya da anketler sonucunda, geliştirilen bir trafik kontrol sisteminin arayüzünde bulunması gerekenleri incelemekte ve bunun ışığında yeni bir ilk örnek sunmaktadır.

Chiang ve diğ. (1998), Tayvan demiryolları için çizelge hazırlama problemi için bir uzman sistem sunmaktadır. Öncelikle demiryolu ağının geniş bir kesimi için oldukça büyük bir çizelge oluşturulmakta, oluşan çizelgede bulunan çatışmalar ise yerel olarak çözülmektedir. Uzman sistem yardımıyla 1 saat içerisinde oluşturulan çizelgelerin, böyle bir sistem kullanılmadığı takdirde, ancak 4 ila 5 ay arası bir çalışma sonucunda elde edilebileceği iddia edilmektedir.

Leilich (1998), kapasite kısıtı önem taşıyan hat kesimleri için oluşturulmuş yazılımları inceleyip, başarımlarını karşılaştıran bir derleme çalışması sunmaktadır.

Hooghiemstra ve Teunisse (1998), Hollanda Demiryolları'ndaki yolcu talebini karşılamak için 5 adımdan oluşan bir döngü sunmaktadır:

1. Ekonomik senaryolara bağlı olarak talep tahmini gerçekleştirilmektedir.
2. Demiryolu hatları, ya da sunulan hizmet planlanmaktadır.
3. Sunulan hizmetin uygulanabilirliği incelenmektedir. Bunun için eşit takip süreli (homojen) bir çizelge oluşturulmaktadır.
4. Ağın başarımı ve diğer etkenler incelenmektedir.
5. Altyapı tanımlanmaktadır.

Geliştirilen benzetim yazılımı öncelikle bir çizelge oluşturmaktadır. Daha sonra, trenler özellikle yoğun istasyonlarda rotalanmaktadır. Aktarmalar sırasında fazla beklenilmemesi için 5 dakikadan fazla bekleme süreleri kabul edilmemektedir. Bu nedenle trenlerin 2 dakikalık gecikmeleri bile büyük risk oluşturabilmektedir. Modelin ve yazılımın geliştirme aşamasında olduğu için test edilemediğinden bahsedilmektedir.

Benyahia ve Potvin (1998), kurye servisi problemine genetik algoritmalar ile çözüm sunmaktadır. Sevk görevlileri, stresli bir işle uğraştıkları için meslek hayatlarının da kısa sürdüğü ileri sürülmektedir. Bu yüzden karar destek sistemlernin büyük önem taşıdığı üzerinde durulmaktadır. Kurye probleminin doğası gereği, belirli bir zaman penceresi içerisinde hizmet sunulması zorunluluğu mevcuttur. Bu çalışmada, genetik algoritma, dallar halinde kodlanmaktadır. Seçim yöntemi olarak, sıralamaya bağlı seçim tercih edilmektedir.

İki ayrı uyum fonksiyonu mevcuttur ve bunların toplamını en büyükleyen kromozom seçilmektedir. Algoritmanın çalıştırılması 1 saat sürmektedir. Oldukça yavaş sonuç vermektedir.

Cordeau ve diğ. (1998), rotalama ve çizelgeleme problemleri hakkında oldukça kapsamlı bir derleme çalışması sunmaktadır.

Roth ve diğ. (1998), demiryolu trafik kontrolü problemini bilişsel yönden ele alıp, tren sevk görevlilerinin stratejilerini incelemektedir.

Carey (1999), öne geçme çatışmalarının nedenlerini incelemekte ve öne geçme çatışmaları durumunda çizelgelerin güvenilirliğini ölçen bir sezgisel algoritma sunmaktadır.

Van Egmond (1999), demiryolu hat kapasitesini kuyruk kuramına dayanan bir yaklaşımla ölçmektedir. Çalışmada tren hareketleri yol-zaman grafiğinde basamaklar olarak tanımlanmaktadır. Bu nedenle problem iş-atölye problemine benzetilmektedir. Oluşturulan çizelgelere bazı güvenlik süreleri (tampon süreler) de eklenmiştir. Daha esnek çizelgeler oluşturulduğundan bahsedilmektedir.

Vukadinovic ve diğ. (1999), kargo servisleri için taşıt atama problemine sinirsel bulanık yaklaşım getirmektedir. Dispeçerlerin bulanık karar verme ölçütleri incelenmekte, bu ölçütlerin önemini modellemek için de bir yapay sinir ağı kullanılmaktadır. Geliştirilen 3 yapay sinir ağı modeli sevk görevlilerinin 100 ayrı sevk kararı kullanılarak eğitilmiş, eğitim esnasında kullanılmayan 60 karar kullanılarak da sınanmıştır. Eğitim kümesinin %97'si, sınama kümesinin de %95'i sevk görevlilerinin kararını taklit edebilmiştir.

Adenso-Diaz ve diğ. (1999), sistemde 3 türde beklenmeyen olay meydana gelebileceğinden bahsetmektedir. Bunlardan birincisi herhangi bir dispeçer kararına ihtiyaç duyulmayan, sadece hizmette oluşan önemsiz aksamalar; ikincisi trenlerden sadece bir tanesinin etkilendiği aksamalar (sistemdeki diğer trenler aksamadan herhangi bir şekilde etkilenmemektedir); üçüncüsü ise diğer trenleri de etkileyen aksaklıklar şeklindedir. Beklenmeyen olaylar karşısında alınabilecek kararlar da 3'e ayrılmış durumdadır. Birinci durumda herhangi bir işlem yapmaya gerek duyulmamaktadır. Bu durum sadece gecikme süresi çok düşük olduğunda söz konusu olabilmektedir. İkinci durumda gecikme fazla ise tren iptali söz konusu olurken, üçüncü durumda ise hizmeti sürdürmek için başka bir tren gönderilmektedir. Oluşturulan modelde taşınan yolcu sayısının en büyüklenmesi amaçlanmaktadır. Bu üç karardan hangisinin alınması gerektiği dal ve sınır yöntemi ile çözülmektedir. Ancak, çözüm uzayını daraltmak için bazı geri çözümleme (backtracking) algoritmaları kullanılmaktadır.

Sistem Asturia, İspanya'da çalışmakta ve uygun çözümleri 5 dakikada sağlamaktadır. Önce uygun bir çözüm elde edip, dal ve sınır ile iyileştirilmeye çalışılmaktadır. Çalışmada diğer herhangi bir yöntemle karşılaştırma sunulmamakta ve bazı durumlar için 5 dakika çok uzun bir süre olabilmektedir.

Isaai ve Singh (2000), tren çizelgeleme problemi için bir sezgisel algoritma sunmaktadır. Oluşturulan çizelgelerdeki hedef, trenlerin beklemelerinin, toplam seyahat süresine olan oranını en aza indirmek olarak belirlenmiştir. Geliştirilen algoritmanın 1 saniye içerisinde uygulanabilir iyi bir çizelge oluşturabildiğinden bahsedilmektedir.

Fransoo ve Bertrand (2000), Hollanda Demiryolları için trafik hacmini iki katına çıkarmayı hedefleyen bir çalışma sunmaktadır. Demiryolu şebekesi çift hatlı olduğu için yalnızca öne geçme çatışmaları söz konusudur. Öne geçme çatışmalarının çözülmesi için yeni yan hatlar inşa edilmektedir. Çalışmada yan hatların inşa edilmesi gereken yeri belirleyecek bir matematiksel model sunulmaktadır.

Fay (2000), tren çatışmalarının çözümü için bulanık mantık tabanlı bir algoritma sunmaktadır. Tren sevk görevlilerinin 10 saatlik kararlarını temel alıp, kural tabanı geliştirilmiştir. Geliştiren sistemde, trenler ikili olarak değerlendirilmektedir. Geliştirilen sistemin, en iyi sonuçları bulmaktan uzak olduğu göze çarpmaktadır. Sistemin amacı daha çok sevk görevlisi kararlarını taklit etmek olarak görülmektedir. Benzetim sonuçları bir tablo halinde sunulmaktadır. Bu sonuçlarda geçen tren, bekleyen tren ve çatışma çözümü sonucunda oluşan gecikme değerleri gösterilmektedir.

He ve diğ (2000), gelen vagonları farklı yük trenlerine atayan bir genetik algoritma geliştirmiştir. Problemin tamamen karmaşık yapıda olduğundan bahsedilmektedir ve iş-atölye yaklaşımında bulunulmaktadır. Trenlerin istasyondan ayrılış zamanları bulanık rakamlar olarak kabul edilmektedir. Bulanık sayılar erken, geç ve tercih edilen olarak belirlenmektedir. Amaç fonksiyonu olarak, vagonların istasyonda kalış sürelerini en aza indirecek şekilde, istasyon çıktısını en büyükleyecek bir fonksiyon geliştirilmiştir. Kromozomlar ise trenlerin kodunu temsil etmektedir. Kodların sırasına göre trenler organize edilmektedir. Modelin sonuçları hakkında çok az bilgi bulunmaktadır ve karşılaştırma çok kısıtlı sunulmaktadır. Modelin ne derece iyi sonuçlar verdiği anlaşılamamaktadır.

Hansen (2001), coğrafi bilgi sistemi (CBS) verilerinden yararlanarak, tren hızlarının gerçek zamanlı olarak belirlenmesi sayesinde, gecikmelerin oldukça azaltılabildiğinden bahsetmektedir. Çalışma herhangi bir sınama verisi içermemektedir.

Dalal ve Jensen (2001) üç aşamalı bir benzetim modeli sunmaktadır. İlk aşamada, trenler istasyondaki peronlara kuyruk kuramından faydalanılarak atanmaktadır. Sonraki aşamada, oluşturulan trenlere personel ataması yapılmaktadır. Modelin son aşaması da uygun bir çizelge yaratmaktadır. Ancak çizelge oluşturma yöntemine ait detaylı bilgi sunulmamaktadır.

Huisman ve Boucherie (2001), demiryolu hattının her kesiminde farklı trenlerin farklı hızlarda seyrettiğinden yavaş trenin hızlı treni geciktirebileceğinden bahsetmektedir. Daha dengeli bir çizelge oluşturabilmek için tesadüfi değişkenlerin kullanıldığı (stokastik) bir model geliştirilmiştir. Modelde 2 ya da daha fazla sayıda hat bulunmaktadır. Bu nedenle yalnızca öne geçme çatışmaları söz konusudur. Farklı senaryolar için ortalama gecikme değerleri ve gecikme olasılıkları hesaplanmıştır.

Middelkoop ve Bouwman (2001), geliştirilen SIMONE adlı paket program sayesinde esnek çizelgeler oluşturulabildiğinden, hat kararlılığının ölçülüp, çizelgelerin iyileştirilebildiğinden ve şişe boyunlarının saptanabildiğinden bahsetmektedir.

Ping ve diğ. (2001), tren sevk problemine genetik algoritma yaklaşımı sunmaktadır. Çalışmada toplam gecikme süresinin en küçüklenmesi amaçlanmaktadır. Trenlerin istasyonlardan ayrılma zamanları belirlenerek, bu gecikmeler en küçüklemeye çalışılmaktadır. Çift hatlı bir yol kesiminde, seçilen bir istasyondan her trenin ayrılış sırası kromozom haline dönüştürülmektedir.

Amaç fonksiyonunu oluşturan gecikmelerin 3 bileşeni bulunmaktadır:

1. 1. Trenin istasyondan ayrılış zamanı
2. Diğer trenlerin istasyondan ayrılış zamanı (iki tren arasındaki takip süresi de hesaba katılmaktadır)
3. Trenlerin bir sonraki istasyona varış zamanları

Bu yöntem kullanıldığında, bazı gecikmeler iki defa hesaba katılmaktadır. Toplam gecikme değeri, gerçek gecikmeyi göstermekten uzak kalmaktadır.

Kromozomlar, trenlerin istasyondan çıkış sırası olarak belirlenmektedir. Dolayısıyla problem, sıralama problemi şekline dönüşmektedir. Sonuçlar kısmı çok kısa tutulmuştur. Yalnızca en yoğun istasyon için benzetim gerçekleştirmiştir. Herhangi bir karşılaştırma sunulmamaktadır.

D'Ariano ve diğ. (2002), geciken bir trenin planlanmış yol-zaman grafiğinden sapmasıyla diğer trenleri de etkilediğinden bahsetmektedir. Bu sayede geciken bir trenin gecikme değeri, diğer trenlere de yayılmaktadır. Otomatik sevk destek sistemleri dispeçerlere aşağıdaki

konularda yardımcı olmaktadır:

- Tren hareketlerini tahmin etme
- Beklenen çatışmaları saptama
- Bu çatışmalara çözüm önerileri sunma

Sistemler, sabit ve değişken hız modelleri olmak üzere ikiye ayrılmaktadır. Sabit hız modellerinde, trenlerin mümkün olan en büyük hız ile hareket ettikleri kabul edilmektedir. Değişken hız modellerinde ise sinyal sistemlerinin durumlarına göre tren hızları değişkendir. Bu çalışmada değişken hızlı bir model geliştirilmiştir. Model, belirli bir zaman aralığını göz önünde bulundurarak çalışmaktadır. Model, iki alt problemden oluşmaktadır:

- Çatışmaların bulunmadığı bir çizelge oluşturmak
- Hız profillerini koruyarak, trenler arasındaki en düşük takip mesafesini korumak

Algoritma çalışırken tüm trenleri göz önüne almaktadır. Gelecekteki durum, hat işgali ve dinamik tren karakteristiklerine göre belirlenmektedir. Çalışmanın ana amacı tren gecikmelerini en küçüklemek olarak belirlenmiştir. Hareket süresine hızlanma, yavaşlama ve iki blok arasındaki hız farklılıkları da yansıtılmaktadır. Sistem, 3 ayrı algoritma üzerinde çalışmaktadır:

- Basit sevk etme kuralları (ilk giren ilk çıkar (FIFO) ve ilk çıkacak olan ilk girer (FOFI))
- Alternatif çizelge düzenlemesi üzerine bir sezgisel algoritma (AMCC-Avoid most critical computation time algoritması. Her yinelemede en fazla gecikme yaratan çözümün tersi yaratılmaktadır. Bu sayede büyük boyutlu problemlerin bile birkaç saniye içerisinde çözülebildiğinden bahsedilmektedir)
- Dal ve sınır yöntemi

Hızlar farklı sürücü davranış şekillerine göre farklılaştırılmaktadır. Bu çalışmada üç ayrı senaryo mevcuttur:

- Takip eden tren tamamen duruyor, sinyal sarıya döndüğünde, yaklaşma hızı ile yoluna devam ediyor.
- Takip eden trenin ikinci fren evresi esnasında sinyal sarıya dönüyor ve yaklaşma hızı ile tren hızlanıyor.
- Takip eden tren görüş mesafesine ulaştığı anda sinyal yeşile dönüyor ve tren proje hızı ile devam ediyor.

Uygulama, Amsterdam, Schiphol'de 10 km'lik bir ana koridorda gerçekleştirilmiştir. Bu koridorda iki istasyon mecvuttur. Hoofdorp istasyonu yük, Schiphol istasyonu ise yolcu trenlerine hizmet vermektedir. 4 tren için çözüm yapılmıştır. FIFO yöntemi ile en fazla 176, ortalama 67,75 sn'lik gecikmeler elde edilmiştir. Dal ve sınır yöntemi kullanıldığında ise en fazla gecikme 127, ortalama gecikme 38,5 sn olmuştur.

Geliştirilen algoritmalar, Leiden-Amsterdam arasındaki 20 km'lik kesimde sınanmıştır. Bu kesimde 5 istasyon mevcuttur. 14 ve 27 tren içeren senaryolar için çalışma yapılmıştır. Genelde dal ve sınır yöntemi daha iyi sonuçlar sunmasına rağmen çözümü elde etmek diğerlerinden daha uzun sürmüştür. Daha sonra sırasıyla AMCC, FOFI ve FIFO çözümleri gelmektedir. Hızların değiştirildiği yöntemde, sabit hızlıdan daha büyük gecikme değerleri gözükmekte ama bu değerlerin daha gerçekçi olduğundan bahsedilmektedir.

Dündar (2003) ile Dündar ve Şahin (2003), dispeçerlerin kararlarını taklit eden bir yapay sinir ağı sunmaktadır. Geliştirilen model, çatışmaya giren trenleri ikili olarak karşılaştırmakta ve trenlerden hangisinin bekletilip, hangisinin geçeceği kararlarını modellemektedir. Sevk görevlisi kararları modellenirken 4 ölçüt göz önüne alınmıştır. Bunlar, trenlerin diğerlerine karşı önemini temsil eden "temel öncelik sayısı"; mecvut çatışma aleyhine çözülen trenin kaybedeceği süreyi gösteren "miyopik çözümdeki gecikme"; trenler için bir gecikmişlik ölçütü olan "kritik oran" ve çatışma çözüldükten sonra trenlerin maruz kalabileceği "potansiyel çatışmaların sayısı"dır.

Geliştirilen yapay sinir ağı, 331 çatışmanın 178'i kullanılarak eğitilmiş, eğitme esnasında kullanılmayan 153 çatışma kullanılarak da sınanmıştır. Geliştirilen ağ, 331 çatışmanın %96'sını başarı ile taklit edebilmektedir. Bu nedenle sevk görevlileri için bir karar destek sistemi olarak kullanılabileceğinden bahsedilmektedir.

Carey ve Carville (2003), özellikle yoğun tren trafiği yaşayan istasyonlarda görülen trenlerin platformlara atanması problemi üzerine bir sezgisel algoritma sunmaktadır. Geliştirilen algoritma çevrimiçi atama yapmamakta, sadece tekrarlı çizelgeler için atama problemini çözmektedir. Geliştirilen algoritma Leeds istasyonu için planlamacıların tercihleri ile karşılaştırılarak sınanmıştır. Çalışmada herhangi bir en iyileme söz konusu değildir, yalnızca uygulanabilir bir atama gerçekleştirilmektedir.

Pachl ve White (2003), Alman Demiryolları'nın işletme stratejileri hakkında bilgi vermektedir. Demiryolu hatları farklı firmalara kiraya verilmekte, firmalar talep ettikleri tren çizelgelerini oluşturup, Alman Demiryolları'na göndermektedir. Farklı firmalar tarafından

oluşturulan çizelgelerde bulunan çatışmalar, firmalar arası görüşmeler ile çözülmeye çalışılmakta, çözüm sağlanamayan durumlarda en yüksek ücreti ödeyen firmanın istediği çizelge uygulanmaktadır. Her firmadan gelen talebe uygun olarak tren çizelgeleri hazırlanmakta, çizelge hazırlanırken blok işgalleri göz önüne alınmaktadır. Çizelge hazırlandıktan sonra oluşacak tren talepleri de çizelgedeki boşluklar göz önüne alınarak yanıtlanmaktadır.

Chen ve diğ. (2003), demiryolu işletmelerinin güvenilirliğini arttırmak için hat kapasitesini incelemekte ve işletilen tren sayısının azaltılması gereken hatları saptamaktadır. Çalışmada Monte Carlo benzetimi kullanılmakta ve yedek kapasite tanımlanmaktadır.

Mattson (2004), tren gecikmelerini birincil ve ikincil olarak iki sınıfa ayırmıştır. Birincil gecikmeler, çizelgede mevcut bulunan beklemeler nedeniyle oluşan gecikmeler, ikincil gecikmeler de beklenmedik olaylar sonucunda gerçekleşen çatışmaların çözülmesi sonucu oluşan gecikmelerdir. Çalışmada, birincil ve ikincil gecikmelere ait bir kaynak araştırması sunulmakta ve uygulanan yöntemlerin yararları ile zararları incelenmektedir. İkincil gecikmelerin modellenmesi için en uygun yöntem makro benzetim olarak görülmektedir.

Dorfman ve Medanic (2004), çizelgeleme problemine kesikli olay yaklaşımı sunmaktadır. Geliştirilen çizelgede oluşan çatışmalar yerel olarak saptanmakta ve çözülmektedir. Çatışmalar çözüldükten sonra hat tıkanıklığı kontrol edilmektedir. Çalışmada, hattı boşaltma süresi, trenlerin toplam gecikme süresi ve en büyük gecikme süresi olmak üzere 3 adet değerlendirme ölçütü kullanılmaktadır. 3 farklı türde problem için denemeler gerçekleştirilmiştir. Birinci problemde, tek hatlı bir demiryolu kesimi üzerinde sabit hızla hareket eden trenler bulunmaktadır. İkinci problemde, bu kez çift hatlı bir demiryolu kesimi üzerinde sabit hızla hareket eden trenler bulunmaktadır. Üçüncü ve son problemde ise, çift hatlı bir demiryolu kesiminde, trenler farklı hızlarda hareket edebilmektedir. Her üç problem için de iyi çizelgeler elde edilebilmektedir ancak en iyileme durumu söz konusu değildir.

Ghoseiri ve diğ. (2004), çizelge oluşturma problemi için, yolcu memnuniyetini en büyükleyecek ve yakıt tüketimini en küçükleyecek şekilde çok amaçlı bir model sunmaktadır. Direnimler hesap edilip, matematiksel bir model haline getirilerek, yük trenlerindeki yakıt tüketimi için amaç fonksiyonu belirlenmektedir.

Yolcu memnuniyeti için ise 6 kısıt kullanılmaktadır:

- Tren hareketlerinin sürekliliği
- Olayların sürekliliği
- Bağlardaki seyahat süreleri ve platformların işgali
- Bağların aynı anda yalnızca bir tren tarafından kullanılabilmesi
- Takip süresi
- Alt ve üst zaman sınırları.

Çok amaçlı karar vermede Pareto en iyiliği kullanılmaktadır. Bir ödünleşme (trade-off) sınırı belirlenmekte ve bu sınırdaki değerler değerlendirmeye alınmaktadır. Daha sonra 6 farklı en iyileme kısıtı belirlenerek bunlar arasındaki değerlendirme sunulmaktadır.

Feng ve diğ. (2004), hazır beton karıştırıcılarının şantiyelere sevk edilmesi için bir genetik algoritma sunmaktadır. Algoritmada, şantiyelerdeki kamyon bekleme sürelerinin en küçüklenmesi hedeflenmekte ve kromozomlar, kamyonların şantiyelere sevk sırası olarak tanımlanmaktadır. Geliştirilen algoritma, küçük boyutlu problemlerde en iyi sonucu saniyeler içerisinde bulabilmekte, problem boyutu büyüdükçe en iyi sonuca yakınsama değeri ölçülmektedir.

Lu ve diğ. (2004), tek ve çok hatlı demiryolları için bir benzetim modeli sunmaktadır. Geliştirilen model, herhangi bir zamanda trenin uygulayabileceği en yüksek hızı da hesaplayabilmektedir. Trenlerin toplam seyahat süresi hız kısıtlarının bir fonksiyonu olarak hesaplanmaktadır. Geliştirilen benzetim modelinin Los Angeles'daki yolcu ve yük trenleri için sınanıp, yeterli görüldüğünden bahsedilmektedir ancak bir karşılaştırma mevcut değildir.

Mazzarello ve Oliviani (2005), Avrupa Birliği Projeleri kapsamında bir trafik yönetim sistemi geliştirmiştir. Sistem iki parçadan oluşmaktadır. Birinci parça çatışmaları saptayıp, çözümler üretmekte, ikinci parça da her tren için en iyi hız değerini bulmaktadır. Demiryolu şebekeleri bölümlere ayrılmış durumdadır. Bu sistem her alt bölge için ayrı ayrı çalışmaktadır. Sistemin nasıl çalıştığı basit olarak anlatılmakta ancak herhangi bir benzetime ya da sonuçlara yer verilmemektedir. Çatışma çözümlerinde sezgisel algoritmalar kullanılmaktadır. Geciken ve önceliği olan trenlere öncelik verilmektedir. Çatışmalar yerel olarak çözümlenip, yeni çizelgeler üretilmektedir. Yerel çözümler içerdiğinden en iyi sonucu elde etmekten uzak bir çalışma olduğu göze çarpmaktadır.

Rodriguez (2005), demiryolu trafik kontrolü problemini çok iyi tanımlayan bir çalışma olarak göze çarpmaktadır. Kısa ve net olarak demiryolu trafik kontrolü probleminden

bahsedilmektedir. Karar destek sistemlerinde, kararların iyilik derecesi ile hesaplama süresinin çatıştığından bahsedilmektedir. Çalışma sadece kesişim bölgelerindeki sevk problemini ele almaktadır. Çatışma çözümünde bir takım kısıtlar kullanılmaktadır. Problem, bir Gantt şemasına benzetilmiş ve iş-atölye yöntemi ile bağlantı kurulmuştur. Kullanılan kısıtlar; kaynak atama kısıtı, kapasite kısıtı, zaman kısıtları olarak görülmektedir. Geliştirilen model, Pierrefitte-Gonesse kesişiminde (Fransa) denenmiştir. Bu kesimde sadece 6 tren bulunmaktadır. Bunlardan üçü TGV, ikisi banliyö, biri de yük trenidir. Çözümde 25920 farklı olasılık olabileceğinden bahsedilmektedir. Bir benzetim programı kullandığında ortalama gecikme 1210 saniye iken, geliştirilen modelde 20 saniyeye kadar düşürülmüştür. Tek bir tren geciktiğinde ise 300 saniyelik bir gecikme süresi elde edilmektedir. Modelin uzun hatlarda ve fazla sayıda tren çalıştığında nasıl sonuçlar sağlayacağı (özellikle kısa zaman içerisinde) belli değildir.

Wegele ve Schneider (2005) demiryolu trafik kontrolü problemi üzerine bir genetik algoritma sunmaktadır. Genetik algoritmanın amaç fonksiyonunda her bir trenin her bir istasyonda maruz kalacağı gecikmenin yanında, gecikmelerden, sinyal ve makas ayarlamalarından ve iki tren arasındaki aktarmanın kopmasından ötürü de cezalandırmalar kullanılmaktadır. İki ayrı yöntemle modelleme yapılmaktadır. Birinci yöntemde 260 tren ve 12 istasyon için bir bireyde 17000 gen bulunmaktadır. Benzetimde, tüm trenlerin 0 ila 15 dakika arası düzenli dağılan ilk gecikme değeleri bulunmaktadır. Benzetimde tıkanmalar (deadlock) oluşabilmektedir. İkinci yöntemde, kromozomlar, tren sıraları olarak tasarlanmıştır. Çalışma, bu açıdan Ping ve diğ.'nin (2001) çalışmasını andırmaktadır. Birinciye göre daha hızlı bir yöntem sunulmakta ve tıkanmalar daha kolay atlatılabilmektedir. Sonuç olarak sadece 3 dakika çalışan algoritmanın %11,2'lik bir en iyileme başarısına ulaştığından bahsedilmektedir.

Takeuchi ve Tomii (2005), demiryolu trafik kontrolü işleminin esnekliğini denetleyen bir çalışma sunmaktadır. Tren iptalinden, tren türünü değiştirmeye kadar 10 ayrı tekrar çizelgeleme probleminden bahsedilmektedir. Bunlar 5 ayrı sınıfa ayrılmaktadır. En kolay gerçekleştirilen karara 0, en zor gerçekleştirilen karara da 4 değeri atanmıştır. Geliştirilen bir algoritma ile, mümkün olan en az gecikmeyi sağlayacak çözüm sağlanmaya çalışılmaktadır. Bu çalışma kapsamında kullanıcı istekleri ön planda tutulmuştur. Bu yüzden tren iptali en son durumda göz önüne alınmaktadır. İki ayrı plan değerlendirmeye alınmaktadır. Yol üzerindeki çalışma kısa sürdüğü sürece taşıt değişimi iptalden daha faydalı olmakta, ancak çalışma süresi uzadıkça iptal daha uygun çözümler sağlamaktadır. Çalışmanın sonuçları hakkında çok fazla bilgi sunulmamaktadır.

Chang ve Chung (2005), tren çizelgeleme problemi için bir genetik algoritma sunmaktadır. Gerek duyulduğu durumlarda, tekrar çizelgeleme işlemi de benzer şekilde gerçekleştirilmektedir. Ancak çalışmanın odaklaştığı konu tren çizelgeleme konusudur. Genetik algoritmada, kromozom gösterimi yerine, matris gösterimi tercih edilmiştir. Matrislerde satırlar trenleri, sütunlar da istasyonları temsil etmektedir. Katı ve yumuşak (esnek) kısıtlar bulunmaktadır. Bu kısıtların toplamını sağlayacak tren çizelgesinin seçimi yapılmaktadır. Katı kısıtlar olarak kalkış-varış zamanları, takip mesafesi, seyahat süresi, bekleme süresi, en küçük takip mesafesi, yolcuların seyahat süresi, kapasite ve yükleme etkeni kullanılmaktadır. Yumuşak kısıtlar ise, yolcuların ortalama seyahat süresi ile tüm trenlerin ortalama doluluk etkeni olarak belirlenmektedir. Çizelgeleme işleminin genetik algoritma ile nasıl yapıldığından bahsedilmemekte ancak yeniden çizelgelemenin genetik algoritma ile nasıl yapıldığından bahsedilmektedir. Orijinal çizelgeden sapmalar en aza indirilmeye çalışılmaktadır. Planlanan çizelge, yeniden çizelgelemenin ilk adımı olarak kabul edilmektedir. Çizelgelemede makul sonuç bulmak 20 dakikayı bulmaktadır. Yeniden çizelgelemede ise, en iyiye ulaşmaktan çok, uygun sonuçlar elde etmek daha öncelikli bir amaç olmaktadır.

White (2005), Kuzey Amerika'da kullanılmakta olan benzetim yazılımları hakkında genel bilgiler sunmaktadır.

Şahin ve diğ. (2005), tren sevk problemi için 3 adet sezgisel algoritma sunmaktadır. Çalışma sadece yük trenleri için gerçekleştirilmektedir. 48 saatlik bir zaman penceresi içerisinde çalışılmaktadır. Bu nedenle, çözümlerin 5-10 dakika arasında elde edilmesi makul gözükmektedir. Trenlerarası çatışmalar, zaman-mekan uzayında tanımlanmakta ve probleme tam sayılı programlama ile çözüm getirilmektedir. Geliştirilen ilk sezgisel algoritmada, CPLEX paket programı kullanılmaktadır. Trenler için önceden tanımlanan "en fazla toplam kabul edilebilir gecikme süresi" içerisinde kalarak, çatışmalara uygun çözümler getirilmesi amaçlanmaktadır. Ancak tren sayısı arttıkça, makul bir sürede çözüm elde edilememektedir. İkinci sezgisel algoritmada, trenlerarası çatışmalar sırasıyla tespit edilmekte, çatışmaya bir çözüm getirilmekte ve bu çözüm neticesinde trenler hareketlerine devam etmektedirler. Çatışma çözümleri neticesinde uygun olmayan bir çözüm elde edildiği takdirde (trenlerin en fazla toplam kabul edilebilir gecikme süresi aşıldığında), gecikme süresini aşan trenin dahil olduğu çatışmalar, bu tren lehine çözülerek, bu süre azaltılmaya çalışılmaktadır. Üçüncü sezgisel algoritmada ise, bir ileri bakış (look-ahead) değişkeni ilave edilerek, çatışma çözümlenmesi esnasında, trenlerin ileride girecekleri çatışmalar da hesaba katılarak karar

alınmaktadır. Çatışma çözümlemeleri esnasında, en iyi sonuca ulaşmak için, dal ve sınır yöntemi kullanılmaktadır. Farklı test problemlerinin yanı sıra, bir de gerçek probleme çözüm getirilerek, algoritmalar denenmektedir. Gerçek problemde, sevk görevlisi kararları sonucu toplam seyahat süresi 14,119 dakika ve toplam gecikme 6,311 dakika olarak gerçekleşmiştir. Geliştirilen algoritmalar ile bu süreler sırasıyla 8,245 dakika ve 437 dakikaya çekilebilmektedir.

Vromans ve diğ. (2006), çizelgelerin güvenilirliğinin arttırılması için çizelgelerin dengeli olması gerektiğinden bahsedilmektedir. Dengeli çizelgelerde, çizelgedeki tren hareketleri birbirine paralel yapıdayken, dengesiz çizelgelerde aynı hat kesiminde farklı hızlar kullanılabilmektedir. Dengesiz bir yapıda, bir gecikmenin yayılımı, dengeli bir yapıya göre çok daha fazla olabilmektedir. Çizelgeleri dengeli hale getirmek için, şehirlerarası trenlerin belirli kesimlerde yavaşlatılması, şehir içindeki trenlerin de hızlandırılması, hat kısaltması ve her trene eşit sayıda durak atanması gibi öneriler getirilmektedir.

Dessouky ve diğ. (2006), iki ve üç hatlı kesimler içeren bir bölge için çizelgeleme problemini ele almaktadır. Geliştirilen algoritmada dal ve sınır yöntemi kullanılmaktadır. Algoritmanın 2 saat çalışması sonucunda en uygun çözümler elde edilebilmektedir. Büyük boyutlu problemlerde, kullanılmakta olan CPLEX adlı paket programın yetersiz kaldığı ancak geliştirilen algoritmanın sonuç verdiği görülmektedir.

Kaynak araştırması kapsamında toplam 116 adet kaynak incelenmiştir. Bu kaynaklardan 79 adedi çizelgeleme problemi üzerinde durmaktadır. Çizelgeleme problemlerinden de 18 adedi demiryolu trafik kontrolü ve yeniden çizelgeleme problemi ile ilgilidir. 79 adet çizelgeleme ve yeniden çizelgeleme probleminden 7 adedi çözüm elde etme sürecinde genetik algoritmalardan yararlanmaktadır.

Bu tez kapsamında, demiryolu trafik kontrolü ve yeniden çizelgeleme probleminde genetik algoritmalar kullanılmıştır. Dispeçerlerin birkaç dakika bazen de bir dakikadan daha kısa süreler içerisinde yeniden çizelgeleme kararları verip, uygulamaya koyması gerektiğinden, diğer çalışmalardan farklı olarak, hızlı bir biçimde probleme mümkün olduğunca en iyi çözümü bulmayı hedefleyen ve elde ettiği çözümü de görsel olarak kullanıcıya sunan bir algoritma geliştirilmiştir. Geliştirilen algoritma en çok 5 dakika içerisinde uygulanabilir iyi bir çizelgeyi dispeçerin kullanımına sunmaktadır. Bu bakımdan, bir karar destek sistemi olarak da kullanılabileceği söylenebilir.

3. DEMİRYOLU TRAFİK KONTROLÜ VE YENİDEN ÇİZELGELEME

3.1 Giriş

Demiryollarında trenler, genellikle işletme tarafından önceden hazırlanarak ilan edilmiş bir çizelgeye uygun olarak hareket ederler. Çizelge (orer), tren hareketlerinin şematik olarak gösterildiği bir yol-zaman grafiğidir. Yaygın kullanımında grafiğin düşey ekseninde yol (ve istasyonlar ile yan hatlar) temsili olarak gösterilirken, yatay eksende ise zaman bilgileri bulunur. Tren hareketleri bu çizelge üzerinde eğik olarak çizilmiş doğru parçaları ile gösterilirken, yatay çizgiler de istasyon ya da yan hatlardaki beklemeleri göstermektedir.

Ulaştırma hizmetini periyodik olarak sürdüren işletmelerde, bu hizmet önceden hazırlanmış çizelgeye uygun olarak yürütülür. Bu yöntem, *plan duyarlı işletmecilik* olarak adlandırılır (Şahin,1996). Trenler önceden hazırlanmış çizelgedeki hareket planlarına uygun hareket ettikleri sürece plan duyarlı işletmecilik koşulları devam etmektedir. Ancak herhangi bir nedenden ötürü tren hareketleri önceden planlanmış çizelgenin dışına çıktığında, trenler için önceden belirlenmiş yer ve zaman bilgilerinde de değişiklikler olur. Bu da trenlerarası çatışmaları doğurmaktadır. Trenlerarası çatışmaların yeteri kadar önceden tespit edilip, çözülmesi sonucu trenler güvenli hareketlerine devam edebilirler. Ancak, çatışma çözümü sonucunda oluşan gecikmeler ağ kesimi boyunca yayılarak, diğer trenleri de etkilemektedir. Bu da ağ kesimi bazında tren gecikmelerinin artmasına neden olmaktadır. Sunulan hizmetin etkinlik düzeyi tren hareketlerinin, önceden hazırlanmış çizelgeye uygunluğunun bir göstergesidir ve tren gecikmeleri ile ölçülmektedir.

3.2 Demiryolu Trafik Yönetimi

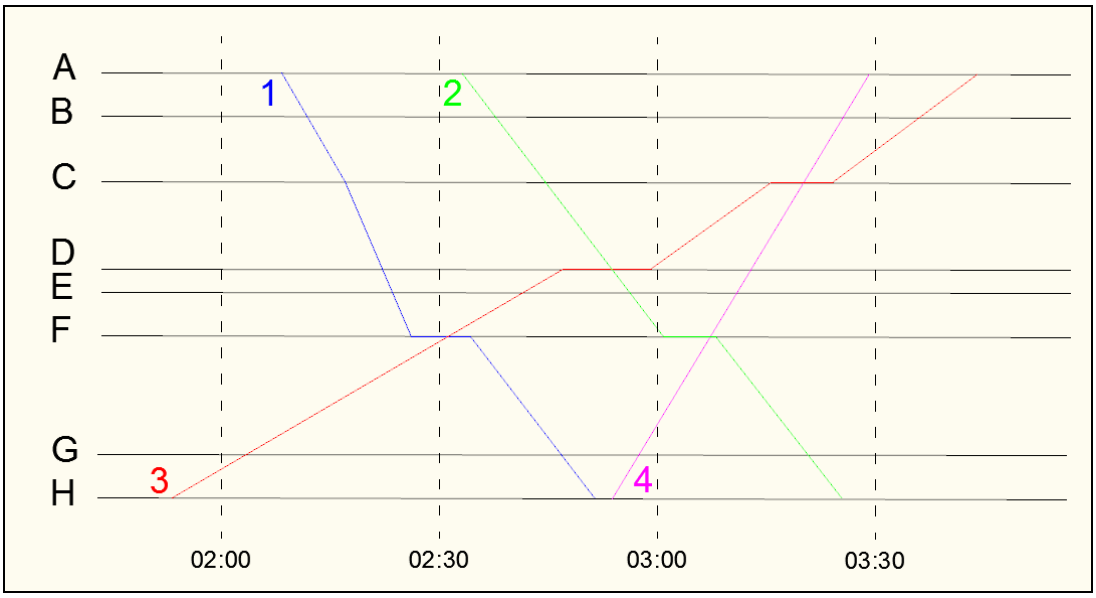
Bir işletmeye ait demiryolu şebekesi belirli sınırlara sahip alt bölgelere ayrılır. Her bir alt bölge yönetim açısından birbirinden bağımsız olmasına karşın, bölgeler arasında seyahat eden trenlerin bulunması, ya da bir başka deyişle bir trenin ilk kalkış istasyonundan son varış istasyonuna varıncaya dek süren seyri boyunca birden fazla alt bölgeden geçebilmesi nedeniyle bölgeler arasında sürekli bir bilgi iletişimi zorunluluğu vardır.

Herhangi bir nedenden ötürü bir trenin hareketi önceden planlanmış çizelgenin dışında gerçekleştiğinde, diğer trenlerle olan, önceden belirlenmiş karşılaşma/öne geçme yer ve zamanları değişmekte, bu da trenler arası çatışmaları doğurmaya neden olabilir. Trenler arası çatışmalar yeteri kadar önceden saptanmalı ve gerekli kararlar alınarak, uygulanabilir yeni bir

çizelge oluşturulmalıdır. Burada, çizelgeden sapmaların saptanması ve çizelgede yapılan değişikliğin uygulanması *kontrol*; çizelgenin, değişiklik yapılarak uygulanması ise, *yeniden çizelgeleme* işlemlerini oluşturmaktadır. Tren hareketlerinin yeniden çizelgelemesi, belirlenen güvenlik ve etkinlik hedefleri doğrultusunda, trenlerin karşılaşma/öne geçme yer ve zamanlarının belirlenmesi, hangi trenin hangi hatta ve ne kadar süreyle bulunması gerektiğinin saptanması anlamına gelmektedir (Şahin, 1996). Demiryolu trafik kontrolünde etkinlik ölçütü trenlerin çizelgeden sapma miktarıdır. Bu nedenle tren gecikmelerinin (ağırlıklı) toplamlarının en küçüklenmesi demiryolu trafik kontrolünün temel hedefidir.

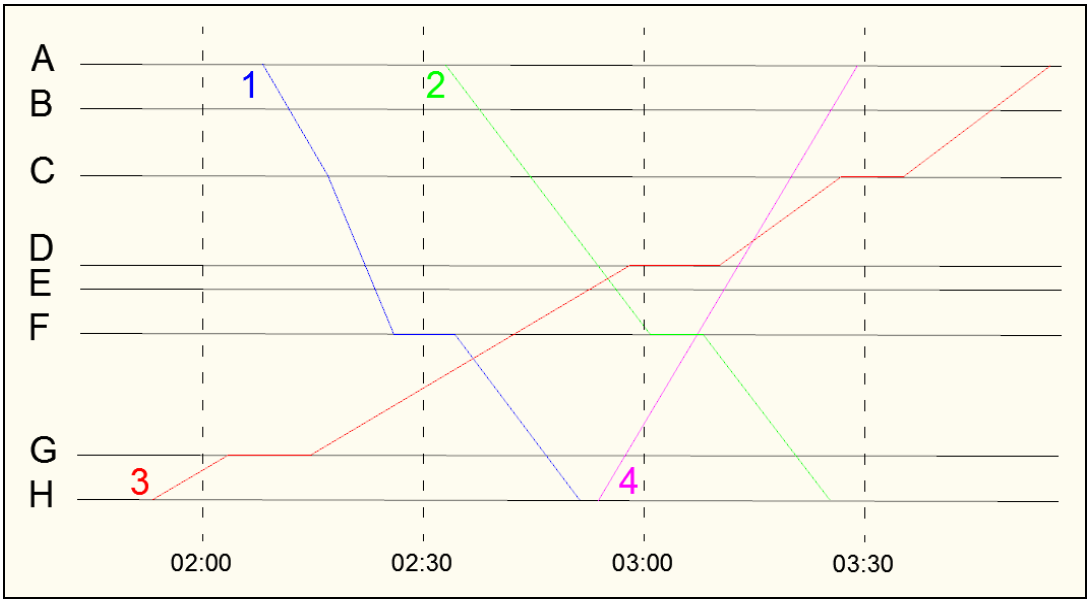
3.3 Yeniden Çizelgeleme ve Trafik Kontrolü

Şekil 3.1'de 8 istasyon (A-H) ya da yan hat içeren tek hatlı bir demiryolu kesimi için önceden hazırlanmış çizelge görülmektedir. Şekilden de görülebileceği üzere, önceden hazırlanmış olan çizelge trenler arası çatışma içermeyen, uygulanabilir bir çizelgedir. 1 numaralı tren, F istasyonuna vardığında karşı yönden gelmekte olan 3 numaralı trenin de aynı istasyona varmasını bekler. Bu tren, G-F istasyon aralığını boşaltıp, F istasyonundan ayrıldıktan sonra, gerekli makas düzenlemelerinin de yapılmasının ardından yoluna devam eder. Daha sonra, 3 numaralı tren D istasyonuna vardığında, karşı yönden gelmekte olan 2 numaralı trenin aynı istasyona varmasını bekler. 3 numaralı tren, 2 numaralı tren D istasyonundan çıktıktan sonra, gerekli düzenlemeler yapıldıktan sonra yoluna devam eder. 2 numaralı tren ise F istasyonuna vardığında 4 numaralı trenin aynı istasyona varışını bekler; 4 numaralı tren de yoluna devam ettikten ve gerekli düzenlemeler yapıldıktan sonra yoluna devam eder. Son olarak 3 numaralı tren C istasyonuna vardığında, aynı yönden seyretmekte ve daha hızlı olan 4 numaralı trenin de aynı istasyona varıp kendisinin önüne geçmesini bekler. 4 numaralı tren, 3 numaralı treni geçtikten ve bir sonraki blok kesimini de terk ettikten sonra 3 numaralı trenin yoluna devam etmesine izin verilir.



Şekil 3.1 A-H demiryolu kesimine ait önceden hazırlanmış çatışma bulunmayan çizelge

3 numaralı tren beklenmedik bir biçimde G istasyonunda bir süre durmak zorunda kaldığında, Şekil 3.2'de görülen durum söz konusu olacaktır. Bu durumda, diğer trenlerin tamamen çizelgeye uygun hareket ettikleri kabul edildiğinde, önce 3 numaralı tren, 1 numaralı trenle F ve G istasyonları arasında karşılaşma çatışmasına girecektir. Daha sonra 3 numaralı trenle 2 numaralı tren D ve E istasyonları arasında karşılaşma çatışmasına girecek ve son olarak da 4 numaralı tren ile 3 numaralı tren C ve D istasyonları arasında karşılaşma çatışmasına girecektir.



Şekil 3.2 3 numaralı trenin G'de planlanmamış duruşu sonucu ortaya çıkan trenler arası çatışmalar

Bahsedilen olayda, 3 numaralı trenin planlanmamış duruşu sonucunda G istasyonundan çıkması, bunun sonucunda da gerçekleşecek trenler arası çatışmaların saptanması “kontrol” işleminin başlangıç bölümünü oluşturur. Gerçekleşecek çatışmaların çözümü sonucu yeni bir uygulanabilir çizelgenin oluşturulması da “yeniden çizelgeleme” işlemidir.

4. GENETİK ALGORİTMALAR

Genetik algoritmalar, temeli doğal seçilim ve doğal genetik üzerine kurulmuş arama algoritmalarıdır. Dizi halindeki yapılara uygulanan en iyinin hayatta kalması ilkesini, yapısal ancak rasgele bir bilgi değişimi yöntemi ile birleştirerek, bir arama algoritması oluşturur (Goldberg, 1989). Genetik algoritmalar problemlere tek bir çözüm üretmek yerine farklı çözümlerden oluşan bir çözüm kümesi üretir [1]. Her yeni nesilde, yeni bir yapay canlılar (diziler) kümesi oluşturarak, bunların hayatta kalmak için ne kadar yeterli olduğunu ölçer. Elde bulunan bilgilerden faydalanarak, başarımı arttırıcı yönde, yeni arama noktalarını verimli bir biçimde araştırır (Goldberg, 1989). Böylelikle, arama uzayında aynı anda birçok nokta değerlendirilmekte ve sonuçta bütünsel çözüme ulaşma olasılığı yükselmektedir [1].

Genetik algoritma (GA) terimi ilk olarak John Holland tarafından *Adaptation in Natural and Artificial Systems* (Doğal ve Yapay Sistemlerde Uyum) isimli 1975 yılında yayımladığı kitabında kullanılmıştır (Reeves, 2003). İlk olarak Holland evrim yasalarını genetik algoritmalar içinde en iyileme problemleri için kullanmıştır [1]. Yine 1975 yılında, Holland'ın doktora öğrencilerinden olan Ken DeJong, *An Analysis of the Behavior of a Class of Genetic Adaptive Systems* (Genetik Uyum Sağlayan Sistemler Sınıfının Davranışının Çözümlemesi) konulu doktora tezini yayımlamıştır. Bu alanda daha önce Holland'ın diğer öğrencileri tarafından tezler yapılmasına karşın, bu çalışma GA'ın en iyileme kapasitesi hakkındaki en kapsamlı çalışma olduğu için önem taşımaktadır. GA ile ilgili ilk konferans 1985 yılında düzenlenmiştir. Holland'ın diğer bir doktora öğrencisi, David Goldberg ilk önce doğal gaz boru hattı en iyilemesi konusundaki doktora tezi ile ödül kazanmış, bunu takiben 1989 yılında, GA konusundaki başucu kitaplarından olan *Genetic Algorithms in Search, Optimization and Machine Learning* (Arama, En İyileme ve Makine Öğrenmesinde Genetik Algoritmalar) adlı kitabını yayımlamıştır. Bu kitabın yayımlanmasından sonra GA'dan yararlanan bilimsel çalışma sayısı hızla artmıştır. Günümüzde, pek çok en iyileme probleminin uygulanmasında GA'dan yararlanılmaktadır (Reeves, 2003).

Algoritma terimi ismini büyük İslam alimlerinden cebirin kurucusu ve babası El Harezmi (780-840)'den almıştır. Alimin Arapça ismi Al-Khwarizmi'dir. Latince'de 'kh' telaffuzu olmadığından g harfine dönüşerek bu alimin adı Algoritm telaffuzunu almıştır. El-Harezmi belki de farkına varmadan günümüz matematiğinin temel bir dalına kitaplarından birinin adını vermiştir. Bugün bilgisayarların başlıca teorik ve uygulamalı etkinliklerinden birinin temeli olan ve algoritma denen bilime de kendi adını vermiştir. Böylece El-Harezmi tüm dünyaya hatırı sayılır bir iz ve tükenmeyecek bir etki bırakmıştır (Şen, 2004).

4.1 Genetik Algoritmaların Temelleri

Genetik algoritma adı, karmaşık bir yapının, bileşenler vektörü halinde gösterilmesi ve bir kromozomun genetik yapısının benzeştirilmesinden doğmuştur. Bitki ve hayvanların üretilmesi esnasında yetişen yavruların belirli özelliklere sahip olması istenir. Bu özellikler de genetik bilimine göre, ebeveynlerin sahip oldukları kromozomların birleşimi tarafından belirlenir. Benzer şekilde, karmaşık problemlere daha iyi çözümler elde etmek istendiğinde, mevcut çözümlerin parçaları farklı biçimlerde birleştirilerek, yeni çözümler elde edilir (Reeves, 1996).

GA, problemlerin çözümü için evrimsel süreci bilgisayar ortamında taklit ederler. Diğer en iyileme yöntemlerinde olduğu gibi çözüm için tek bir yapının geliştirilmesi yerine, böyle yapılardan meydana gelen bir küme oluştururlar. Problem için olası pek çok çözümü temsil eden bu küme genetik algoritma terminolojisinde nüfus adını alır. Nüfuslar vektör, kromozom veya birey adı verilen sayı dizilerinden oluşur. Birey içindeki her bir elemana gen adı verilir. Nüfustaki bireyler evrimsel süreç içinde GA işlemcileri tarafından belirlenirler.

GA, diğer en iyileme yöntemleri kullanılırken büyük zorluklarla karşılaşılan, oldukça büyük arama uzayına sahip problemlerin çözümünde başarı göstermektedir. Bir problemin bütünsel en iyi çözümünü bulmak için garanti vermezler. Ancak problemlere makul bir süre içinde, kabul edilebilir, iyi çözümler bulurlar. GA'ın asıl amacı, hiçbir çözüm tekniği bulunmayan problemlere çözüm aramaktır. Kendilerine has çözüm teknikleri olan özel problemlerin çözümü için mutlak sonuca erişmenin hızı ve kesinliği açısından GA kullanılması uygun değildir. GA ancak;

- Arama uzayının büyük ve karmaşık olduğu,
- Mevcut bilgiyle sınırlı arama uzayında çözümün zor olduğu,
- Problemin belirli bir matematiksel modelle ifade edilemediği,
- Geleneksel en iyileme yöntemlerinden istenen sonucun alınmadığı

alanlarda etkili ve kullanışlıdır.

GA'ın, diğer geleneksel en iyileme yöntemlerinden temel farkları şöyle sıralanabilir:

1. GA, problemlerin çözümünü değişkenlerin değerleriyle değil, kodlarıyla arar. Değişkenler kodlanabildiği sürece çözüm üretilebilir. Bu sebeple GA ne yaptığı konusunda bilgi içermez, ancak nasıl yaptığını bilir.
2. GA, aramaya tek bir noktadan değil, noktalar kümesinden başlar. Bu nedenle çoğunlukla yerel en iyi çözümde sıkışıp kalmazlar.
3. GA, türev yerine uyum fonksiyonunun değerini kullanır. Bu değerin kullanılması ayrıca yardımcı bir bilginin kullanılmasını gerektirmez.
4. GA, gerekirci kuralları değil olasılıksal kuralları kullanır [1].

Bir problemin çözümünde kullanılacak herhangi bir GA için aşağıdaki beş elemana ihtiyaç duyulmaktadır:

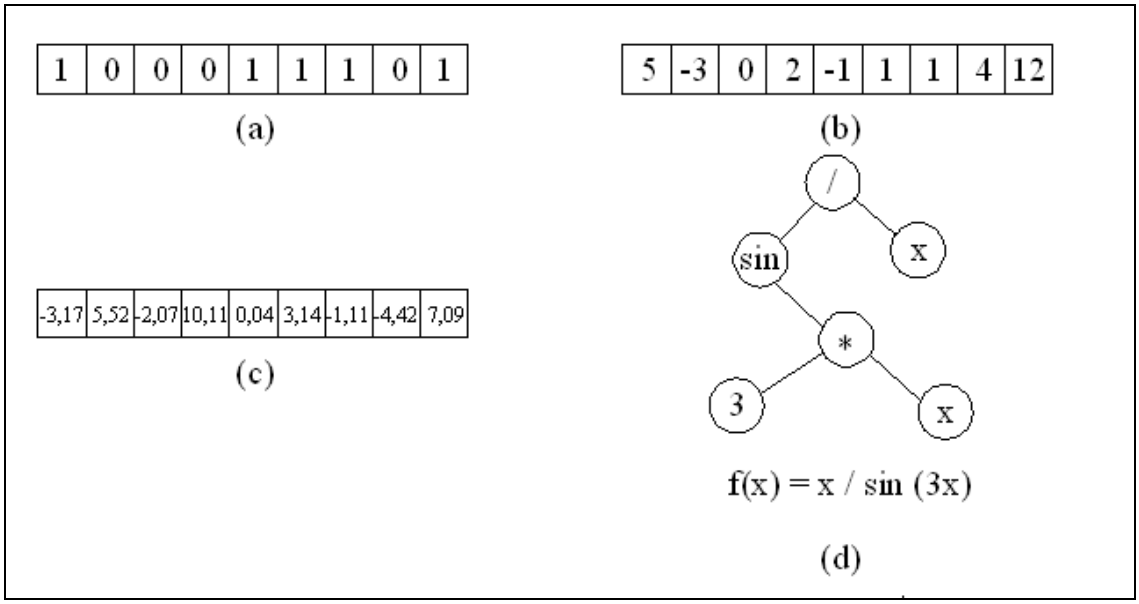
1. Problem için çözümlerin genetik temsili,
2. Çözümlerin başlangıç nüfusunu oluşturacak bir yöntem,
3. Çözümleri uygunluk açısından değerlendirmeye tabi tutacak değerlendirme fonksiyonu,
4. Genetik bileşimi değiştirecek işlemciler,
5. Kontrol değişkenlerinin değerleri (nüfus büyüklüğü, işlemcilerin uygunlama olasılıkları, vd.) (Karaboğa, 2004).

4.1.1 Gösterim Şekli

GA'da karmaşık yapıların gösteriminde ilk ve en çok kullanılan gösterim şekli (0,1) alfabesini kullanan iki tabanlı sayı dizisidir. Doğrudan iki tabanlı kodlama en yaygın kullanılan kodlama tekniğidir. Bu tekniği kullanan GA ikili kodlanmış GA olarak da adlandırılır. Bununla birlikte ikili sistemde tekdüze kodlama ve Gray kodlama gibi farklı gösterim şekilleri de kullanılmaktadır (Karaboğa, 2004). Problemlerin yapısına uygun olarak gösterim şekilleri de farklılık göstermektedir. İkili kodlamanın yeterli olmadığı, ya da çok uzun iki tabanlı sayı dizileri oluştuğu durumlarda tam sayılı kodlama kullanılmaktadır. Tam sayılı kodlamada her bir genin değeri pozitif ve negatif tam sayılardan oluşmaktadır. Tam sayılı kodlama, birleşim

problemlerinde sıkça kullanılmaktadır. Daha ileri bir kodlama şekli olan ondalıklı kodlamada ise, genler ondalıklı sayılardan oluşmaktadır.

GA'da zaman zaman kullanılan bir diğer gösterim şekli de dallar halinde gösterimdir. Bu gösterim şeklinde, bireyler kromozomlar yerine dallar halinde kodlanır. Diğer gösterim şekillerinden temel farkı, dallarda hem elemanların, hem de elemanlara uygulanacak işlemcilerin gösterilmesidir. Farklı gösterimler Şekil 4.1'de görülmektedir.



Şekil 4.1 Farklı gösterim şekilleri: (a) ikili kodlama, (b) tam sayılı kodlama, (c) ondalıklı kodlama, (d) dallar halinde gösterim.

4.1.2 Başlangıç Nüfusu

Başlangıç nüfusunu belirlenirken ele alınması gereken başlıca konular, nüfus boyutu ve bireylerin oluşturulma yöntemidir. Geçmiş çalışmalarda (literatürde), nüfus boyutunun belirlenmesi için çeşitli öneriler bulunmaktadır. Burada unutulmaması gereken konu, nüfus boyutunun seçiminde, etkinlik ve verimlilik arasında bir ödünleşme bulunduğudur (Reeves, 2003). Nüfus boyutu, çok büyük seçildiğinde, algoritma süresince çok daha fazla sayıda bireyin uygunluğu incelenmiş olur, ancak çözüme ulaşma süresi de o derece uzar. Diğer bir taraftan, nüfus boyutunun gereğinden çok küçük seçilmesi durumunda, erken yakınsama ve yerel en iyilere takılma sorunu neredeyse kaçınılmaz hale gelmektedir.

Başlangıç nüfusundaki bireylerin oluşturulma yöntemi için de farklı yöntemler önerilmiştir. Ancak, uygulamalarda hem basitliği hem de tatmin edici sonuçlar elde edilebildiği için tamamen rasgele oluşturma yöntemi kullanılmaktadır (Reeves, 2003). Burada dikkat edilmesi

gereken nokta, rasgele oluşturma esnasında arama uzayı düzenli olarak temsil edilemeyeceği için, bazı değerlerin algoritma süresince göz ardı edilebileceğidir. Bu sorun başkalaşım işlemi ile çözülebilmesine karşın, bunun bir garantisi olmaması, mümkünse algoritmanın birden fazla sayıda çalıştırılıp, oluşan sonuçların değerlendirilmesi tavsiye edilmektedir.

4.1.3 Uygunluk Değerlendirmesi

GA'da bireylerin ortama ne denli uyum sağladığının, ya da diğer bir deyişle türünü devam ettirebilmesi için ne kadar güçlü olduğunun göstergesi olan bir uyum (amaç) fonksiyonu bulunmaktadır. Bu fonksiyon aynı zamanda en iyilenmesi hedeflenen fonksiyondur. Nüfustaki her bir bireyin uyum değeri, bu fonksiyona uygulanarak hesaplanır. Çözümlerin kalitesinin belirlenmesinde kullanılan teknik, genetik algoritmanın performansında oldukça etkilidir. Amaç fonksiyonundan elde edilen ham sonuçların kullanımı algoritmanın başlangıç aşamalarında yeterli olabilir. Ancak algoritma ilerledikçe ve özellikle uyum değerlerinin birbirine çok yakın değerler aldığı durumlarda *iyi* ile *daha iyi* çözümler veya *kötü* ile *daha kötü* çözümler arasındaki farkı ayırt etmek zorlaşır. Bu da genetik algoritmanın çözümleri geliştirmesini olumsuz etkiler. Bu yüzden uygunluk değerlendirilmesinde ölçekleme işlemine gerek duyulur. GA ölçekleme işlemine karşı oldukça hassastır. Araştırma bir taraftan, iyi çözümlerin bulunduğu tarafa doğru odaklanmaya çalışırken, diğer taraftan, algoritmanın küresel en iyileme işlemini gerçekleştirebilmesi için bireyler arasında yeterli çeşitliliğin sağlanması gerekir. Aksi halde erken yakınsama problemi ortaya çıkacak ve bu da tüm araştırma uzayının tamamen araştırılmasını engelleyecektir. Ölçekleme işleminde yaygın olarak kullanılan teknik ölçekleme penceresi işlemidir (Karaboğa, 2004). Ölçekleme penceresi, uyum fonksiyonundan elde edilen değerleri, seçim fonksiyonunda kullanıma uygun hale getirecek şekilde ölçekleme işlemidir [2].

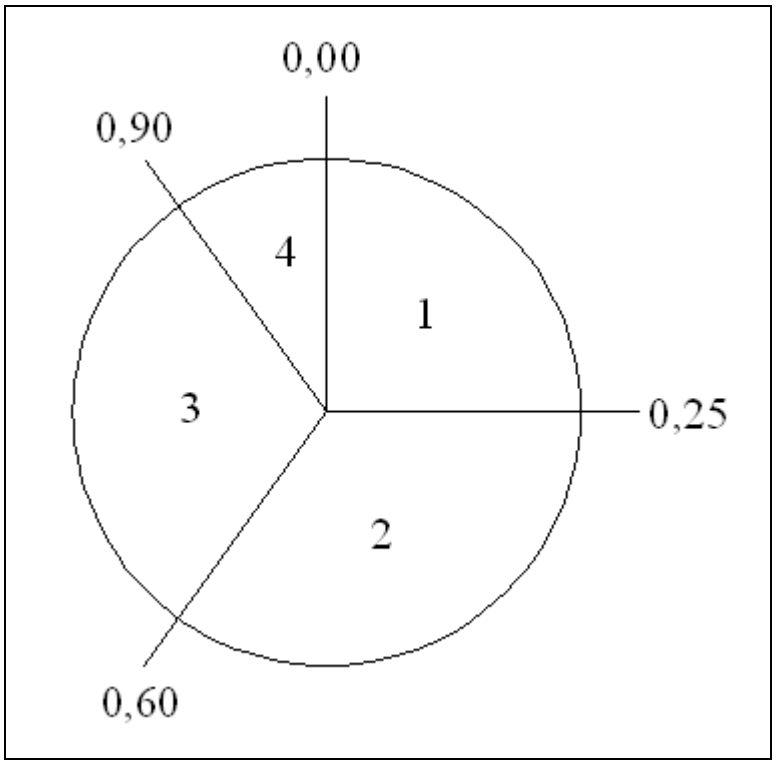
4.1.4 Seçim

Nüfusa dahil olan bireyler, nesillerini devam ettirebilmek, ya da diğer bir deyişle üreme işlemine katılabilmek için bir seçim işlemine tabi tutulurlar. GA'da seçim işlemini gerçekleştirmek için farklı yöntemler geliştirilmiştir. Bunlardan en sık kullanılanları, uyum değerine orantılı seçim, sıralama sistemi ve turnuva sistemidir.

4.1.4.1 Uyum Değerine Orantılı Seçim

Üreme işlemini gerçekleştirebilmek için seçilecek olan bireyler hakkındaki temel görüş, bu seçimin uyum değerine uygun olarak gerçekleştirilmesidir. Bunun için gerçekleştirilen

yöntem, *rulet-tekerleği* olarak bilinir. Şekil 4.2'de basit bir rulet tekerleği görülmektedir. Rulet tekerleği metodunda, her bir birey rulet tekerleğinde uyum değerine orantılı olarak bir yer işgal eder. Üretilen rasgele bir sayının karşılık geldiği birey seçilerek üreme havuzuna dahil edilir (Reeves, 2003). Örneğin, Şekil 4.2 için üretilmiş 0,13 sayısı 1. bireyi, 0,75 sayısı da 3. bireyi seçerek üreme havuzuna dâhil edecektir. Rulet tekerleği, istenilen birey sayısı kadar döndürülüp, bireyler üreme havuzuna dahil edildiğinde, seçim işlemi tamamlanmış olur. Çizelge 4.1'de uyum değerlerine orantılı seçim yöntemi için bir örnek görülmektedir.



Şekil 4.2 Bir rulet tekerleği örneği

Çizelge 4.1 x^3 fonksiyonu için hesaplanan seçilme olasılıkları

Birey	**10'luk Sistemdeki Karşılığı**	**Uyum Değeri** $f(x) = x^3$	**Seçilme Olasılığı**
0010	2	8	0,001
1101	13	2197	0,379
1111	15	3375	0,582
0110	6	216	0,037

4.1.4.2 Sıralama

Kromozomları uyum değerine göre sıralama işlemi, bazı bilgilerin kaybolmasını sağlayabilir. Ancak ölçeklendirmeye gerek duyulmaz ve seçim algoritması daha basit ve daha etkilidir. Uyum değerlerine göre *k*. sırada bulunan bir bireyin seçilme olasılığı *P[k]* olarak kabul edildiğinde, doğrusal sıralama işleminde,

$$P[k] = \alpha + \beta k \tag{4.1}$$

formülü kullanılır. Burada α ve β değerleri sabit sayılardır. *P[k]*'nın bir olasılık dağılım fonksiyonu olabilmesi için gerekli şart:

$$\sum_{k=1}^{M} (\alpha + \beta k) = 1 \tag{4.2}$$

olmasıdır. Burada α ve β değerleri *seçim baskınlığı*nı gösterir. Doğrusal sıralama dışında başka sıralama yöntemleri de kullanılmaktadır, ancak uygulamalarda, basitliği ve yeterliliği açısından doğrusal sıralama tercih edilmektedir (Reeves, 2003). Seçilme olasılıkları belirlendikten sonra, üreme havuzuna eklenecek bireyler, rulet tekerleği işlemindeki gibi rasgele sayı üretilerek seçilir. Çizelge 4.2'de seçilme olasılıklarının belirlenmesine ilişkin bir örnek görülmektedir.

Çizelge 4.2 x^3 fonksiyonu için, $\alpha = 0{,}10$, $\beta = 1{,}0352 \times 10^{-4}$ seçilerek bulunmuş olasılık dağılımları

Birey	**10'luk Sistemdeki Karşılığı**	**Uyum Değeri** $\mathbf{f(x) = x^3}$	**Seçilme Olasılığı** $P[k] = 0{,}1 + 1{,}0352 \times 10^{-4} x^3$
0010	2	8	0,101
1101	13	2197	0,327
1111	15	3375	0,449
0110	6	216	0,122

4.1.4.3 Turnuva seçimi

Katı bir seçim sistemi olan uyum değerine orantılı seçime bir diğer alternatif de τ adet bireyin rasgele seçilip, en iyi uyum değerine sahip olan bireyin üreme havuzuna kopyalandığı turnuva seçimidir. Bu yöntem, $\tau = 2$ değeri için, α değeri 0 alındığı takdirde, sıralama yöntemine oldukça benzemektedir. Turnuva seçiminin en büyük faydası, sadece bireyler arasında sıralama yapmasıdır. Bu sayede matematiksel bir uyum fonksiyonu bulunmayan durumlarda ya da başka bir deyişle, tamamen öznel amaç fonksiyonları söz konusu olduğunda dahi seçim işlemi gerçekleştirilebilmektedir (Reeves, 2003). Turnuva seçiminin bir özelliği, her adımda en düşük uyum değerine sahip bireyin elenmesinin garantilenmesidir. Diğer seçim yöntemlerinde olduğu gibi, istenilen sayıda birey seçim havuzuna kopyalandığında seçim işlemi tamamlanmış olur. Turnuva seçim sistemine ilişkin bir örnek Çizelge 4.3'de görülmektedir.

Çizelge 4.3 x^3 fonksiyonu için, bireylerin turnuva yöntemi ile seçimi

Birey	**10'luk Sistemdeki Karşılığı**	**Uyum Değeri** $f(x) = x^3$	**Rasgele Seçilen Bireyler ($\tau = 2$)**	**Üreme Havuzu**
0010	2	8	1101	1111
			1111	
1101	13	2197	1101	1101
			0110	
1111	15	3375	0010	0110
			0110	
0110	6	216	1111	1111
			0110	

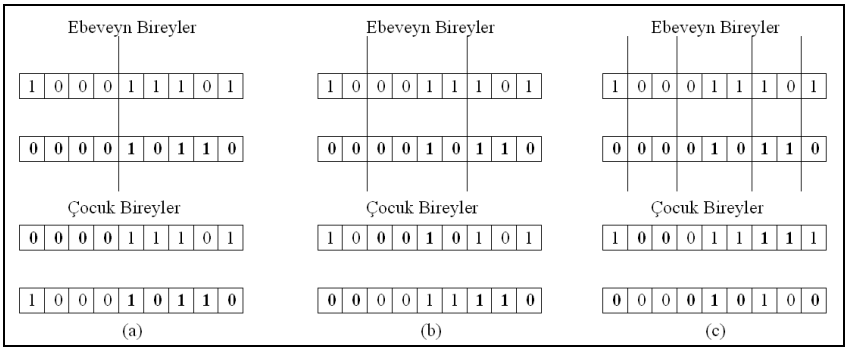
4.1.5 Çaprazlama

Çaprazlama işlemcisi, doğal sistemlerde meydana gelen veya genetik çaprazlama olayıyla ortaya çıkan melez bireylerin üretilmesine eşdeğer bir özelliği GA'ya kazandırır. Üreme havuzunda bulunan bireylerin birer çifti rasgele seçilir ve çaprazlama işlemcisi bu iki bireyden yeni iki birey meydana getirmek için kullanılır (Karaboğa, 2004).

Çaprazlama işlemcisi *P(Ç)*, 0'la 1 arasında bir değer alacak şekilde seçilir. Seçilen iki birey için üretilen rasgele sayı eğer saptanan çaprazlama işlemcisinden küçük ise, bu iki bireye çaprazlama işlemi uygulanarak, yeni iki birey elde edilir. Aksi durumda bu iki birey korunarak bir sonraki nüfusa aynen aktarılır. Çaprazlama işlemcisinin çok küçük seçildiği durumlarda, algoritmanın yakınsaması oldukça uzun sürebilmektedir. Pratikte, çaprazlama işlemcileri 0,7'den büyük olarak seçilmekte, sıklıkla da 1 değerini almaktadır. Çaprazlama işlemcisinin 1 olarak seçilmesi durumunda, her nüfustaki en iyi birey(ler)in kaybolması riski

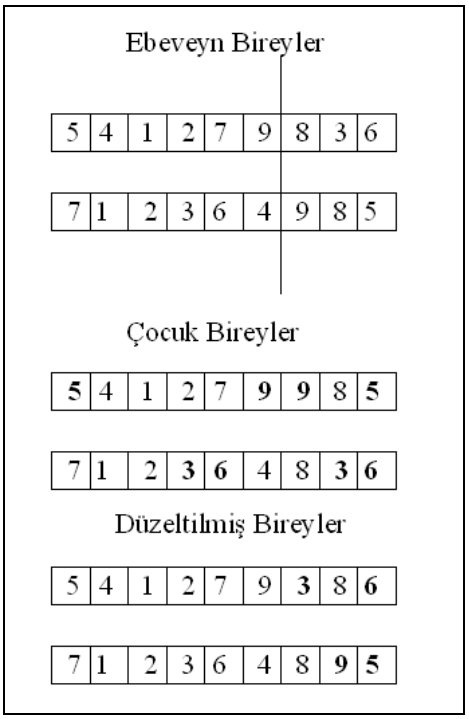
ortaya çıkmaktadır. Bu riski engellemek için, üreme havuzuna dahil edilecek bireylerin sayısı, nüfusta bulunan birey sayısından az tutularak, aradaki fark adedi kadar birey aynen kopyalanarak bir sonraki nüfusa aktarılmaktadır. Uygulamada sıkça görülen bu yönteme *elitizm* adı verilmektedir.

Çaprazlama işleminin farklı uygulamaları bulunmaktadır. Tek noktalı çaprazlamada, kromozomlar rasgele bir noktadan kesilerek, sağında bulunan veriler takas edilmektedir. İki noktalı çaprazlamada, rasgele seçilen iki nokta arasından kesilen kromozomların ortasında bulunan veriler takas edilmektedir. Çok noktalı çaprazlamada da ikiden fazla rasgele noktadan kesilen kromozomların içerdiği verilen kesim noktaları arasında takas edilir. Bu üç çaprazlama yöntemine ilişkin örnekler Şekil 4.3'de görülmektedir.



Şekil 4.3 Çaprazlama örnekleri (a) tek noktalı çaprazlama, (b) iki noktalı çaprazlama, (c) çok noktalı çaprazlama

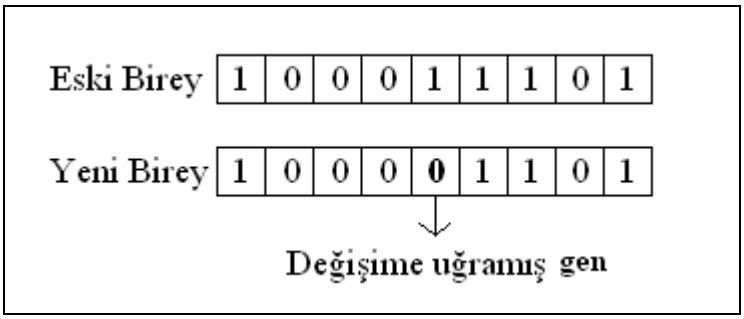
Değişim (permutasyon) problemleri söz konusu olduğunda, çaprazlama sonucunda oluşan yeni bireylerde, bazı elemanların iki defa yer alması ve buna bağlı olarak da bazı elemanların hiç bulunmaması görülebilir. Bu durumda sıkça uygulanan yöntemlerden biri, kromozomda tekrarlanan verilerinden birinin, kromozomda hiç yer bulamamış olan diğer bir veri ile yer değiştirmesidir. Buna ilişkin bir örnek Şekil 4.4'de görülmektedir. Tekrarlamaları önlemek için geliştirilmiş farklı yöntemler de çeşitli kaynaklarda mevcuttur.



Şekil 4.4 Değişim probleminde çaprazlama sonrası yapılan düzeltmeye bir örnek

4.1.6 Başkalaşım

Başkalaşım işlemcisi doğal genetik başkalaşım olayının benzeşimini yapmakta ve genetik algoritmanın başarımında temel rol oynamaktadır. Yeni nüfusta bulunan çözümlere ait her bir gen tek tek kontrol edilir ve *başkalaşım oranına* göre değeri değiştirilir. İkili kodlamada iki tabanlı sayıların değerleri tersine döndürülür, yani sayının değeri 1'se 0'a, 0'sa 1'e dönüştürülür. Şekil 4.5'de bir gen üzerine başkalaşım uygulanması görülmektedir.



Şekil 4.5 Bir kromozomun 5. genine başkalaşım uygulanması

Başkalaşım işlemcisi olmayan bir GA, elemanların seçim ve çaprazlama işlemlerinde değişmediği varsayılırsa en iyi çözümü, ancak, çözüm elemanlarının başlangıç nüfusunda

bulunması halinde bulabilir. Başkalaşım yeni, görülmemiş ve araştırılmamış çözüm elemanlarının bulunmasını sağlar. Başkalaşım işlemcisi kullanmayan GA için iyi bir başarım, ancak nüfusun oldukça büyük tutulmasıyla garanti edilebilir. Aynı zamanda başkalaşım işlemiyle GA'nın yerel en iyi çözümlere takılması engellenir. Çünkü, başkalaşım daha önceden atılmış iyi çözüm elemanlarının tekrar üretilmesini sağlar (Karaboğa, 2004).

GA'da başkalaşım oranının seçilmesi oldukça önemli bir rol oynamaktadır. Başkalaşım oranının büyük seçilmesi, çeşitliliği arttıracak, ancak yakınsama işlemi oldukça uzayacaktır. Gereğinden küçük bir başkalaşım oranı da yerel en iyi noktalarından daha uzun sürede kurtulmayı sağlayacaktır. Başkalaşım işlemcisi genellikle oldukça küçük değerler alır. Başkalaşım işlemcisinin saptanmasına ilişkin kaynaklarda matematiksel çalışmalar bulunmaktadır. Ancak başkalaşım işlemcisinin belirlenmesinde sıklıkla kullanılan basit bir yöntem, *n* kromozom uzunluğunu göstermek üzere;

$$P(M) = 10^{-n} \qquad (4.3)$$

şeklindedir. Başkalaşım oranı, her bir kromozomdaki her bir gene ayrı ayrı uygulanmaktadır. Gen için rasgele üretilen 0'la 1 arasındaki sayı, eğer başkalaşım oranından daha küçük ise, genin değeri değiştirilir. Eğer gen iki tabanlı bir sayıdan oluşuyor ise, genin değeri 1'den çıkarılır. Başka bir gösterim şekli kullanılmakta ise genin değeri rasgele olarak değiştirilir.

4.1.7 Sonlandırma

Yerel en iyi noktalarına ulaşınca sonlanan basit komşuluk arama yöntemlerinin aksine, GA'ın arama yöntemleri teorik olarak sonsuza dek arama yapabilir. Uygulamada bir sonlandırma ölçütü kullanmak gereklidir. Sonlandırmada kullanılan yöntemler, incelenecek nüfus sayısını sınırlama, bilgisayar saatini sınırlama ve nüfus çeşitliliğini inceleyip, belirli bir eşiğin altına düştüğünde sonlandırma şeklindedir (Reeves, 2003).

4.2 Bir Genetik Algoritma Adımı Örneği

Bu bölümde basit bir GA adımı gösterilmektedir. En büyüklenecek fonksiyon olarak, $f(x) = -x^2+20x$ seçilmiştir. Gösterim şekli iki tabanlı sayılar halindedir. Seçim sistemi olarak, basitliğinden dolayı uyum değerine orantılı seçim tercih edilmiştir. Her bir nüfus 4 bireyden oluşmaktadır. Çaprazlama işlemcisi $P(Ç) = 1$, başkalaşım işlemcisi $P(M)$ de $0,001$ değerini almaktadır.

Çizelge 4.4'de görülebileceği üzere, başlangıç nüfusunda birbirinden farklı 4 adet birey rasgele üretilmiştir. Bu bireylerin uyum değerleri, uyum fonksiyonuna göre hesaplanmış ve üreme işlemi için seçilme olasılıkları elde edilmiştir. Daha sonra rasgele üretilen sayılara göre, 2. kromozomdan 2 adet, 3. ve 4. kromozomdan da 1'er adet kopyalanarak, üreme havuzuna dâhil edilmiştir.

Çizelge 4.4 $-x^2+20x$ fonksiyonu için, rasgele üretilen başlangıç nüfusu

Birey	**10'luk Sistemdeki Karşılığı**	**Uyum Değeri** $f(x) = -x^2+20x$	**Seçilme Olasılığı**	**Seçilme Adedi**
0010	2	36	0,126	0
1101	13	91	0,318	2
1111	15	75	0,262	1
0110	6	84	0,294	1
Toplam		286	1	4

Çizelge 4.5'de üreme havuzunda bulunan bireylere çaprazlama işlemi uygulanmıştır. Rasgele seçilen ilk iki bireye, 2. genlerinden, 3. ve 4. bireye de 1. genlerinden çaprazlama işlemi uygulanmış ve çaprazlama sonrası yeni bireyler oluşmuştur. Başkalaşım operatörü çok küçük seçildiğinden, bu bireyler üzerinde herhangi bir başkalaşım işlemi gerçekleştirilmemiştir. Daha sonra oluşan yeni bireylerin uyum değerleri hesaplanmıştır. Dikkat edileceği üzere, daha ilk adımda uyum değerleri toplamı 286'dan, 353'e yükselmiş, en iyi uyum değeri 91'den 96'ya, en kötü uyum değeri de 36'dan 75'e yükselmiştir. Tüm bu veriler, daha ilk adımda algoritmanın yakınsamaya başladığını göstermektedir. Dikkate değer bir diğer nokta da nüfustaki her bireyin 3. geninin 1 değerine sahip olması olmuştur. Bu da algoritmada çeşitliliğin azalmaya başladığını göstermektedir. Algoritma yeteri kadar çalıştırıldığında, bu fonksiyon için en büyük değer olan 100'e (1010 kodlu birey için) yakınsayacaktır. Ancak, ilk adımda 3. genin sahip olması gereken 0 değeri kaybolduğu için, bu değer tekrar ancak başkalaşım işlemcisi herhangi bir bireyin 3. geni üzerinde uygulandığı takdirde gerçekleşebilir. Bu da GA'ın en iyi sonuca yakın sonuçlar sunmasına rağmen her koşulda en

iyi sonucu sunmayı garanti etmediğini göstermektedir.

Çizelge 4.5 Çaprazlama işlemi

Üreme Havuzu	**Rasgele Seçilen Bireyler**	**Çaprazlama Noktası**	**Çaprazlama Sonrası Oluşan Bireyler**	**10'luk Sistemdeki Karşılığı**	**Uyum Değeri $f(x) = -x^2+20x$**
1101	1111	2	1101	13	91
1101	1101		1111	15	75
1111	0110	1	0111	7	91
0110	1101		1100	12	96
Toplam					353

5. GENETİK ALGORİTMALARIN YENİDEN ÇİZELGELEME PROBLEMİNE UYGULANMASI

5.1 Giriş

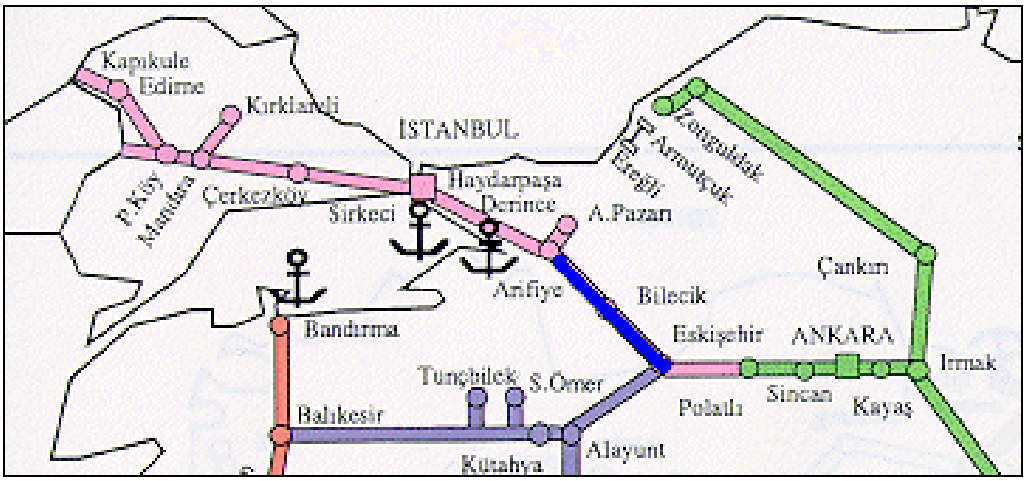
Günümüzde trenlerarası çatışmaların çözümünü deneyimli tren dispeçerleri yapmaktadır. Dispeçerler bu çözümleri yaparken, uzun yıllar boyu kazandıkları bilgi ve deneyimlerinden yararlanmaktadır. Çatışmaların çözümlenmesi oldukça zor ve deneyim gerektiren bir işlemdir çünkü trenler çatışmaya girdiklerinde, çatışmadan dolayı bekleyen tren, yaptığı gecikme nedeniyle, daha sonra karşılaşacağı trenlerle de çatışmaya girebilecek ve gecikme değerleri git gide artacaktır. Benzer şekilde, çatışma çözümü için bir tren bekletildiğinde, arkadan gelen (daha hızlı) bir trenin ona yetişmesi ve önüne geçmesi (öne geçme çatışması) söz konusu olabilmektedir. Böylece ağ bazında gecikmeler geriye doğru yayılmakta ve sürekli artış göstermektedir.

Az sayıda istasyon/yan hat içeren ve kısıtlı sayıda tren çalıştırılan bir şebeke üzerindeki bir çatışmanın çözümünün, ileride gerçekleşecek muhtemel çatışmalara etkisinin ne olacağı ve dolayısıyla sistem bazında gecikme toplamlarının ne kadar olacağı tahmin edilebilir. Ancak, gerçek hayatta, bir şebeke üzerinde çok sayıda istasyon/yan hat bulunmakta ve ulaşım talebini karşılayabilecek sayıda tren işletilmektedir. Bu nedenle, bir çatışmanın çözümünün ağ kesimi üzerindeki diğer trenleri nasıl etkileyeceği ancak oldukça kısıtlı bir şekilde tahmin edilebilmektedir. Trenlerin toplam ya da ağırlıklı toplam gecikmelerini en küçükleyecek çözümün dispeçer tarafından tahmin edilebilmesi neredeyse imkânsızdır. Dispeçerler, trenlerarası çatışmaları çözerken, yılların getirdiği bilgi ve deneyimleri sayesinde bazı kolaylaştırıcı kurallar geliştirmişlerdir. Örneğin, öncelikli bir trenin, önceliği olmayan bir tren karşısında bekletilmemesi ya da hâlihazırda gecikmiş bir treni daha fazla geciktirmemek dispeçerin kullandıkları kolaylaştırıcı kurallardan iki tanesidir. Bu örneklerden de görülebileceği üzere, dispeçerler yaptıkları çözümün iyilik derecesini bilememektedirler.

Dispeçerlerin en iyi sonucu bulması gerçekten de kolay değildir. Bir yönde 5, diğer yönde de 5 tren işletilen tek hatlı bir demiryolu kesimi düşünüldüğünde, $ixj = 5 \times 5 = 25$ adet potansiyel karşılaşma çatışması bulunmaktadır. Her bir çatışmanın 2 farklı çözümü (trenlerden birinin ya da diğerinin bekletilmesi) bulunduğu düşünülürse, 25 adet çatışmanın $2^{25} = 33.554.432$ farklı çözümü bulunmaktadır. Doğal olarak, insan beyninin bu ihtimallerin her birini tek tek hesaplaması ve en iyi çözümü bulması (çözüm için kısıtlı bir süre bulunduğu düşünülürse) olanaksızdır. Günümüzde kullanılan standart bir bilgisayarın bu olasılıkların tümünü

hesaplaması istendiğinde bile, bilgisayar çözüme ulaştığında trenler yolculuklarını çoktan sona erdirmiş olabilmektedirler.

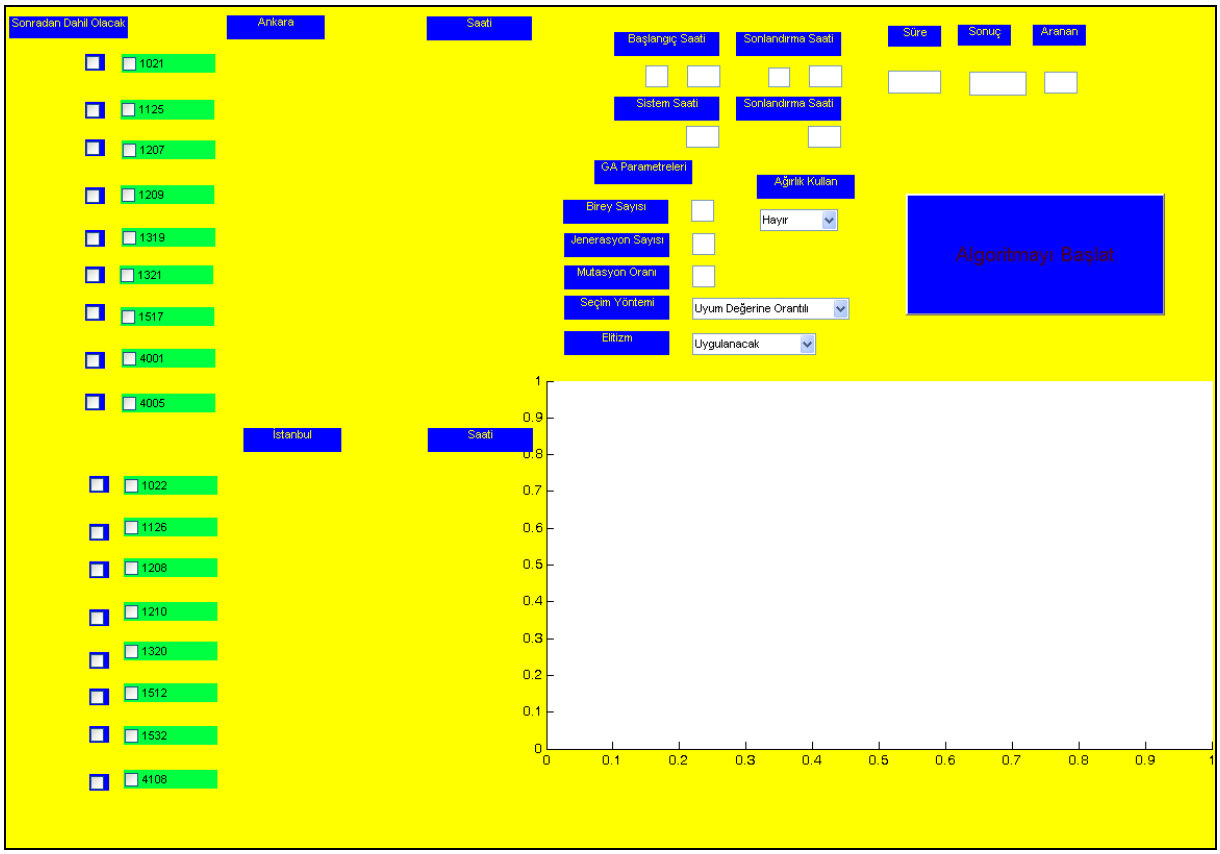
Bu çalışma kapsamında, dispeçerlerin karar alırken faydalanabilecekleri en iyi sonucu hesaplamayı hedeflemekten ziyade, 5 dakika gibi kısa sayılabilecek bir süre içinde uygun sonuçlar elde etmeyi amaçlayan bir program geliştirilmiştir. Program MATLAB paket programı kullanılarak yazılmış; arama uzayının farklı noktalarından başlayan ve hızlı bir şekilde en iyiye yakınsama özelliğine sahip olan genetik algoritmalar kullanılmıştır. Algoritmanın test bölgesi olarak İstanbul-Ankara demiryolu bağlantısının tek hatlı olan Arifiye-Eskişehir kesimi (Şekil 5.1) seçilmiştir. Uzunluğu yaklaşık 163 kilometre olan bu kesimin seçilmesinin önemli bir nedeni vardır. İstanbul ve Ankara terminallerinden kalkan trenler 00:00 – 06:00 zaman aralığında bu bölge içinde buluşmaktadırlar. Bu zaman aralığında Arifiye-Eskişehir kesimini kullanan bir yönde 8, diğer yönde 9 adet tren bulunmaktadır. Bu durum, tren dispeçerleri için, trenlerarası çatışmaların çözümlenmesine yönelik birçok kararın alındığı, oldukça yoğun bir mesai anlamına gelmektedir. Meydana gelebilecek 72 potansiyel karşılaşma çatışmasının 2^{72} = 4.722.366.482.869.645.213.696 farklı alternatif çözümü bulunacaktır. Oldukça üstün özelliklere sahip bir bilgisayar ile saniyede 1.000.000 farklı alternatifin çözülebilmesi durumunda dahi her bir çözümü hesaplayıp, en iyi çözümü saptayabilmek 149.745.258 yılın üzerinde sürmektedir. Oysa ki dispeçerler, dakikalar hatta saniyeler içerisinde çatışma çözümlerini gerçekleştirip, trenleri harekete geçirmek ya da durdurmak zorundadır. Bu nedenle geliştirilen program, 5 dakika gibi kısa bir sürede elde ettiği en iyi çözümü dispeçere sunmakta ve bir nevi karar destek sistemi olarak çalışmaktadır.



Şekil 5.1 İstanbul-Ankara demiryolu bağlantısının tek hatlı Arifiye-Eskişehir kesimi

5.2 Geliştirilen Bilgisayar Programının Özellikleri

Geliştirilen programın kullanımının kolay olabilmesi için, bir arayüz tasarlanmıştır (Şekil 5.2). Arayüzde 00:00 – 06:00 zaman dilimi içerisinde Arifiye-Eskişehir kesimini kullanan Ankara ve İstanbul yönünden gelen trenlerin kodları görülmektedir. İçinde bulunulan zaman itibarıyla sistemde bulunan trenin kodunun solundaki kutucuğa tıklandığında, trenin konumuna ilişkin ilave bilgilerin girilebileceği yeni kutucuklar aktif hale gelmektedir (Şekil 5.3).



Şekil 5.2 Geliştirilen arayüz

Şekil 5.3 1022 kodlu trenin konumuna ilişkin ilave bilgilerin girilebileceği kutucuklar

Burada trenin kodunun (1022) hemen sağında bulunan kutucuktan (listeden), trenin en son bulunduğu istasyon (Alifuatpaşa) seçilmelidir. İstasyon bilgisinin sağındaki kutucuktan ise, bu istasyona giriş ya da çıkış yaptığı bilgisi (çıkışı) girilmelidir. Daha sonra trenin bu istasyonda en son görüldüğü zaman, saat ve dakika olarak (1:58) sağdaki iki kutucuğa yazılır. Sistemde bulunan tüm trenler için bu bilgiler programa tanıtılmalıdır. Eğer bir tren henüz Arifiye-Eskişehir kesimine girmemiş fakat ileriki bir zamanda sisteme dahil olacak ise, tren numarasının hemen solunda bulunan "sonradan dahil olacak" kutucuğu seçili hale getirilmelidir.

Trenlerin konumunun belirlendiği kutucuğun sağında ve yukarıda, algoritmanın başlama ve bitiş saatlerinin programa girileceği 2'şer adet kutucuk bulunmaktadır. Bu kutucuklara da tıpkı tren konumunun belirlenmesi esnasında yapıldığı gibi saat ve dakika değerleri girilir. Bu şekilde sistemin durumu, programa tanıltılmış olmaktadır.

Sistemin başlangıç ve sonlandırma saatlerine ilişkin bölümün hemen altında genetik algoritmanın kullanacağı parametre değerlerinin girildiği bölüm bulunmaktadır. Bu parametreler sırasıyla, bir nesilde bulunan **birey sayısı**, algoritmanın hesaplayacağı **nesil (jenerasyon) sayısı**, **başkalaşım (mutasyon) oranı**, çaprazlamaya girecek bireylerin **seçim yöntemi** ve **elitizm** uygulanıp uygulanmayacağıdır.

5.2.1 Birey Sayısı

Birey sayısı çok küçük seçildiğinde, algoritma oldukça kısa sürede yakınsayacak ancak bulduğu sonuçlar en iyi sonuçtan uzak, yerel en iyi sonuçlar olacaktır. Buna karşın gereğinden büyük bir birey sayısı, algoritmanın çalışma süresini (gereğinden fazla) uzatacaktır. Birey sayısı genellikle deneme-yanılma yöntemi ile tayin edilir. Geliştirilen problemlerin çözümünde öncelikle 10 adet birey seçilmiş, bunun yeterli olmadığı görüldüğünde birey sayısı 2'şer adet arttırılmıştır.

5.2.2 Nesil Sayısı

Nesil sayısı çok küçük seçildiğinde algoritma henüz tamamen yakınsayamadan sonlanmakta, büyük seçildiğinde ise, algoritma gereğinden uzun bir süre çalışmaktadır. Bu yüzden nesil sayısı da birey sayısı gibi deneme-yanılma yöntemi ile saptanır. Bu çalışmadaki tüm örneklerde nesil sayısı 100 alınmıştır.

5.2.3 Başkalaşım Oranı

Başkalaşım genetik algoritma için kullanılması zorunlu olmayan ancak kullanıldığında nesildeki çeşitliliği arttırıp, erken yakınsamaların önüne geçebildiği için kullanılması tavsiye edilen bir yöntemdir. Başkalaşım oranı uygulamada küçük değerler seçilir. Bu çalışma kapsamındaki örneklerde, Başkalaşım oranı 0,001 alınmıştır. Her bir ikili sayı için üretilen rasgele değer 0,001'den küçük ise ikili sayının değeri tersine çevrilir, büyük ise aynen korunur.

5.2.4 Seçim Yöntemi

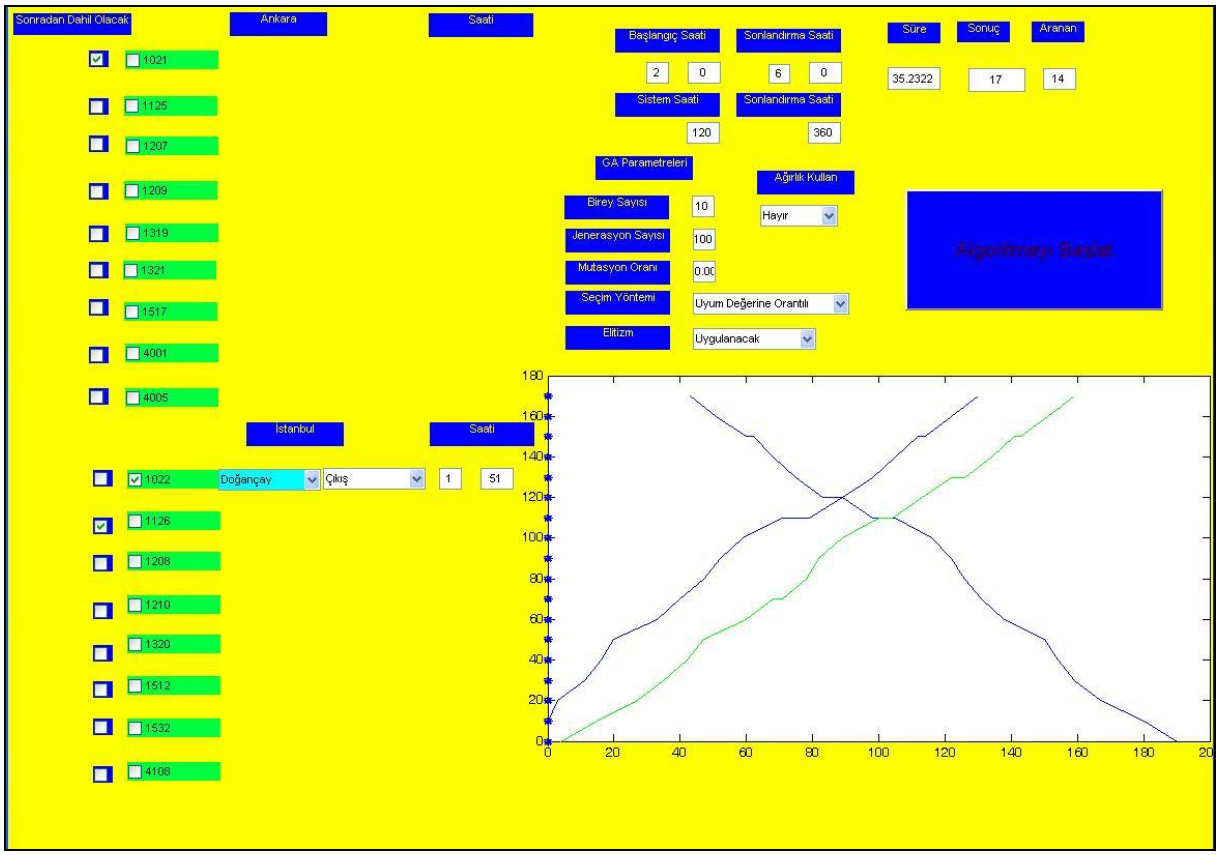
Algoritma için Bölüm 4'de anlatılan 3 adet seçim yöntemi (uyum değerine orantılı seçim, sıralama ve çaprazlama seçimi) geliştirilmiş, ancak örnek problemlerin çözümünde yalnızca "uyum değerine orantılı seçim" kullanılmıştır.

5.2.5 Elitizm

Her bir nesilde elde edilen en iyi bireylerin korunup korunmayacağının tercihi de kullanıcıya bırakılmıştır. Şüphesiz ki en iyi bireylerin korunması kullanıcı lehine olmaktadır. Birey sayısı tek olarak seçildiğinde 1, çift olarak seçildiğinde ise 2 birey aynen korunarak bir sonraki nesile aktarılır.

Şekil 5.2'deki arayüzde, genetik algoritma parametrelerinin hemen sağında problemin gecikme toplamlarını mı yoksa ağırlıklı gecikme toplamlarını mı en küçükleyeceğini belirleyen seçenek bulunur. Problemde ağırlık kullanılmaması halinde, trenlerin gecikme değerleri toplanır ve en küçüklenir. Ağırlık kullanıldığında ise, her bir trene Dündar (2003)'de kullanılan temel öncelik sayılarına orantılı ağırlık değerleri verilerek, öncelikli trenlerin daha az bekletilmesi sağlanır.

Bütün bu bilgiler girildikten sonra, ağırlık kullanma seçeneğinin sağında bulunan "Algoritmayı başlat" düğmesi tıklanarak, program başlatılır. Algoritmanın çalışması sonlandığında ise elde edilen en iyi çözümün sağladığı yeni çizelge hemen alttaki beyaz pencerede çizdirilir. Bu kesimdeki düşey eksen istasyonları (yolu) yatay eksen ise zamanı temsil etmektedir. "Algoritmayı başlat" düğmesinin üst tarafında ise sırasıyla algoritmanın çalışma süresi (saniye cinsinden), elde ettiği (ağırlıklı) toplam gecikme (dakika cinsinden) ve algoritmanın çözüm alternatiflerinden (2^{ixj}) kaç adedini hesapladığı raporlanmaktadır. Algoritmanın sonlanması sonucunda elde edilen bir çözüme ait örnek Şekil 5.4'de görülmektedir.



Şekil 5.4 Bir örnek çözüm

5.3 Geliştirilen Algoritmanın Çalışma Prensibi

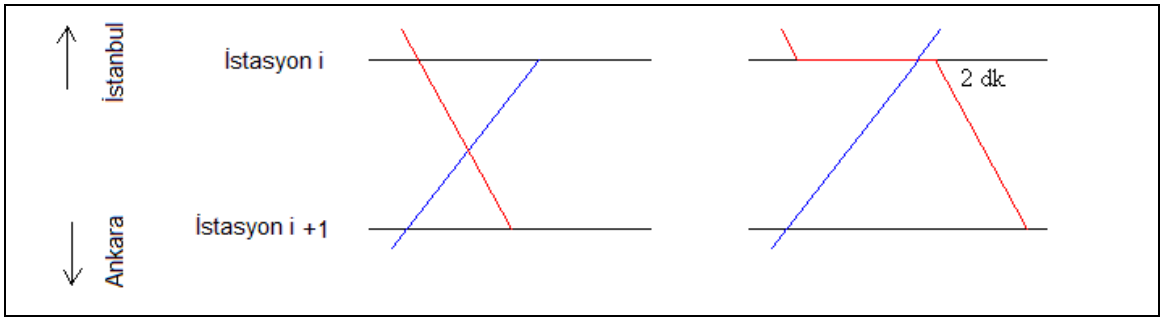
Sistemin durumu arayüz aracılığı ile tanımlanıp, program çalıştırıldığında, sistemde o an itibarıyla herhangi bir çatışma bulunup bulunmadığı araştırılır. Çatışmaların belirlenmesi için komşu istasyonlar arasındaki her kesim bir blok olarak tanımlanmıştır. Eğer herhangi bir blok içinde birden fazla tren bulunursa o blokta bir çatışmanın bulunduğu kabul edilir. Eğer sistemde herhangi bir çatışma bulunmuyor ise, sistem saati bir dakika ileri alınır ve her bir tren tabii (normal) seyir sürelerine uygun olarak ilerletilir. Sistemde ilk çatışma saptanıncaya kadar bu işlem tekrarlanır. İlk çatışma saptandığında ise, programa bütünleşik olan genetik algoritma başlatılır.

Genetik algoritma başlatıldığında, algoritma ilk olarak arayüzde girilen sayı kadar rasgele birey (kromozom) oluşturur. Bu bireyler 0 ve 1 değerlerinden oluşan genler halindedir. Kromozomların uzunluğu (kromozomda bulunan gen sayısı) da sistemde oluşabilecek potansiyel karşılaşma çatışmaları ile öne geçme çatışmalarının toplamına eşittir. Öne geçme çatışmalarının sayısının algoritma tarafından saptanabilmesi çok zor olacağı için, bu değer her örnek problem için sabit bir değer olarak 4 alınmıştır. Bu sayı yeterli gelmediği takdirde,

algoritma hata vermekte ve bu değer büyütülerek, algoritma tekrar başlatılmaktadır. Örneğin, sistemde bir yönde hareket eden 3, diğer yönde hareket eden 2 tren bulunduğunda, kromozom, 3 x 2 + 4 = 10 basamaklı ikili sayı uzunluğundadır. Yani problemin 2^{10} adet farklı çözümünün bulunduğu kabul edilmektedir.

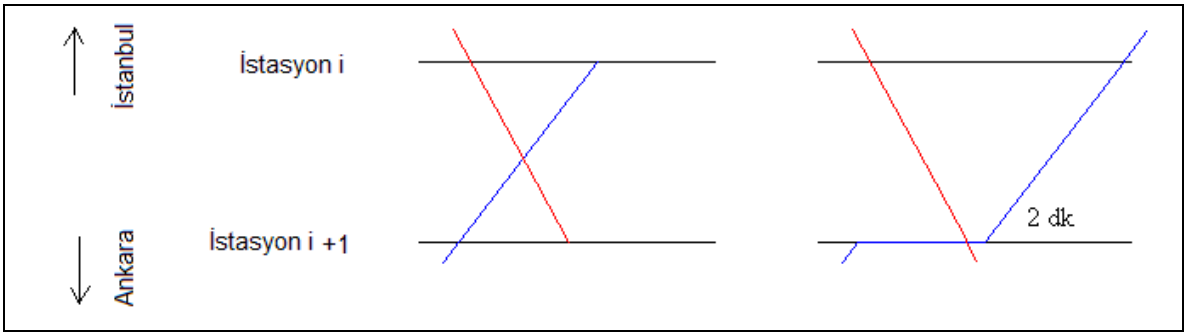
5.3.1 Uyum Değerlerinin Hesaplanması

Başlangıç nesli oluşturulduktan sonra, bu nesilde bulunan her bir bireyin (kromozomun) çözümü yapılır (uyum değerleri hesaplanır). Kromozom çözümleri yapılırken, *i* ve *i+1* istasyonları/yan hatları arasında tespit edilen her karşılaşma çatışması için ilgili kromozomun değerine bakılır. Eğer bu değer 0 ise, İstanbul yönünden gelen tren *i* istasyonuna geri alınarak, Ankara yönünden gelen tren varıncaya değin bekletilir. Ankara yönünden gelen tren *i* istasyonuna vardıktan 2 dakika sonra, İstanbul yönünden gelen trenin hareketine izin verilir (Şekil 5.5).



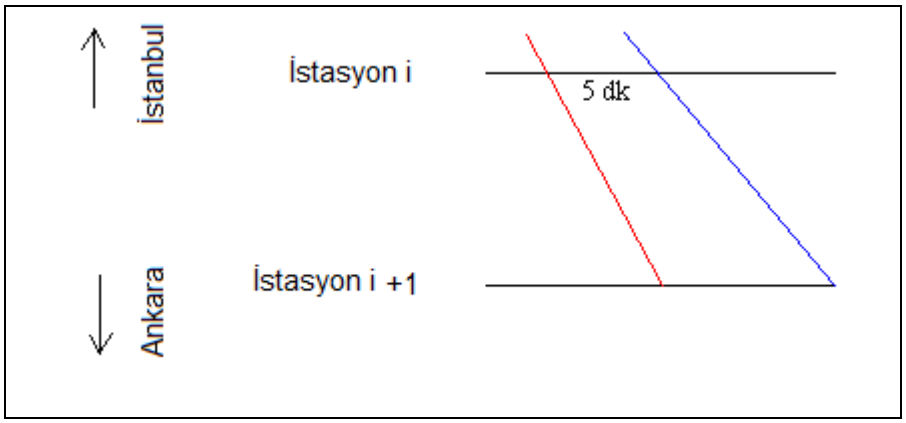
Şekil 5.5 Bir karşılaşma çatışması ve Ankara yönünden gelen tren lehine çözümü

Eğer kromozom 1 değerini taşıyor ise, bu kez Ankara yönünden gelen tren *i+1* istasyonuna geri alınarak, İstanbul yönünden gelen tren varıncaya değin bekletilir. İstanbul yönünden gelen tren *i+1* istasyonuna vardıktan sonra, Ankara yönünden gelen trenin hareketine izin verilir (Şekil 5.6).



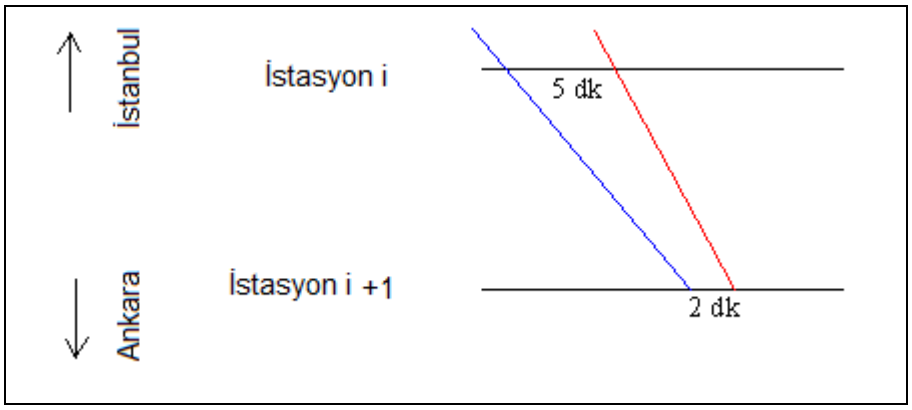
Şekil 5.6 Bir karşılaşma çatışması ve İstanbul yönünden gelen tren lehine çözümü

Çatışma bir öne geçme çatışması ise, farklı bir yöntem uygulanır. Eğer öndeki tren arkadakinden daha hızlı bir tren ise, çatışma yoktur ancak, arkadaki trenin *i* istasyonundan hareketine, güvenliği sağlayabilmek için önceki trenin bu istasyonu terk edişinden 5 dakika sonra izin verilir (Şekil 5.7).



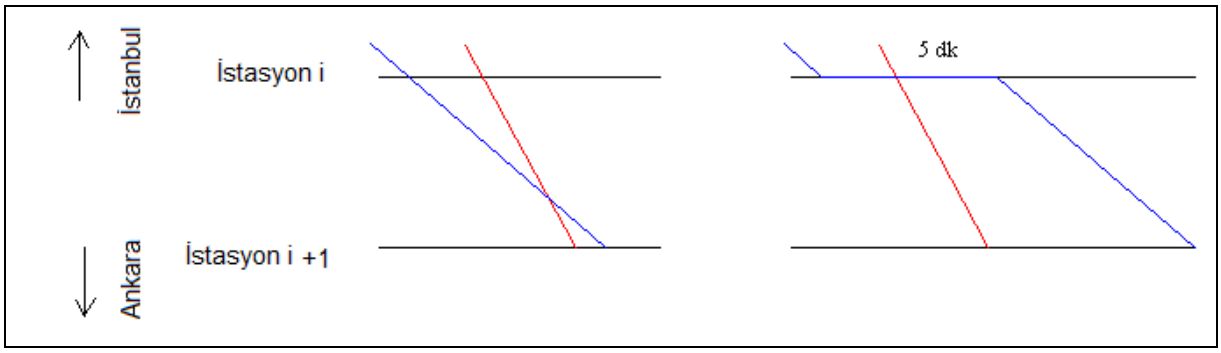
Şekil 5.7 Çatışmasız durum ve 5 dakikalık izleme süresi

Eğer arkadaki tren öndekinden daha hızlı ise, öndeki trenin arkadaki trenden önce bir sonraki istasyona varıp varamayacağına bakılır. Eğer öndeki tren arkadakinden önce istasyona varıyor ise yine çatışma yoktur ancak arkadaki tren öndekinden en az 5 dakika sonra *i* istasyonunu terk etmeli ve yine en az 2 dakika sonra *i+1* istasyonuna varmalıdır (Şekil 5.8).



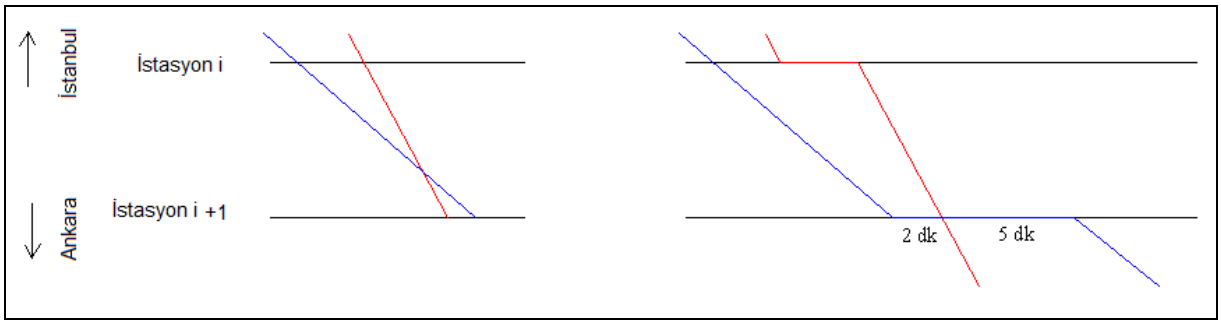
Şekil 5.8 Çatışmasız durum ve 5 dakika kalkış ve 2 dakika varış izleme süreleri

Eğer öndeki tren arkadakinden daha yavaş ise ve arkadaki trenin öndekini geçmesi isteniyorsa çatışma vardır. Çatışmayı çözmek için yine çatışmayı temsil eden kromozomun değerine bakılır. Eğer çatışmayı temsil eden kromozomun değeri 0 ise, öndeki (yavaş) tren bir önceki *i* istasyonuna geri alınır, hızlı trenin geçişine izin verilir ve hızlı tren bu istasyonu terk ettikten 5 dakika sonra yavaş trenin hareketine izin verilir (Şekil 5.9).



Şekil 5.9 Öne geçme çatışması ve hızlı tren lehine çözümü

Eğer çatışmayı temsil eden kromozom 1 değerini taşıyor ise, öndeki trenin hareketi devam eder, arkadaki tren ancak öndeki tren *i+1* istasyonuna vardıktan 2 dakika sonra bu istasyona varacak şekilde *i* istasyonundan çıkabilir. Öndeki tren *i+1* istasyonuna vardıktan sonra, arkadaki trenin varmasını bekler. Hızlı tren *i+1* istasyonundan çıktıktan 5 dakika sonra bu istasyondan çıkış yapabilir (Şekil 5.10).



Şekil 5.10 Öne geçme çatışması ve yavaş tren lehine çözümü

Bir kromozom için oluşan tüm çatışmalar zaman sırasıyla çözüldükten sonra, uygulanabilir bir çizelge elde edilmiş olur. Bu çizelge uygulandığı takdirde her bir trenin gecikme değeri hesaplanır ve bunlar (ağırlıklı olarak) toplanarak kromozomun uyum değeri hesaplanır. Bu bireylerin daha sonraki nesillerde tekrar bulunması halinde tekrar hesaplanmaması için bireylerin onluk sistemdeki karşılığı hesaplanır ve bu karşılıkların uyum değerleri hafızada saklanır. Bir nesildeki bütün bireyler (kromozomlar) uyum değerleri hesaplandığında, seçim işlemine geçilir.

5.3.2 Seçim İşlemi

Bir nesilde bulunan tüm kromozomlar için uyum değerleri hesaplandığında, bu bireylerin hangilerinin bir sonraki nesli oluşturmak için seçileceğinin saptanması gerekmektedir. Bu

çalışma kapsamındaki örneklerde yalnızca uyum değerine orantılı seçim yöntemi kullanılmıştır. Bu yöntemde, en küçükleme uygulanacağı için öncelikle her bir kromozomun uyum değeri, nesildeki en büyük uyum değerinin 1 fazlasından çıkartılır. Böylece, daha az gecikmenin, daha iyi bir uyum değerine sahip olması sağlanır. Daha sonra yine her bir bireyin yeniden hesaplanan uyum değeri önceki bireylerin uyum değerleri ile toplanarak yığışımlı uyum değerleri elde edilir. Daha sonra bu değerler, uyum değerleri toplamına bölünerek, 0 ve 1 arasında değerler alması sağlanır. 0'la 1 arasında üretilen rasgele sayının temsil ettiği kromozom seçim havuzuna kopyalanır. Eğer elitizm uygulanmıyor ise, bu işlem birey sayısı kadar tekrarlanır. Elitizm uygulandığı durumda da, elitizm sayesinde korunacak birey sayısı kadar daha az birey seçim havuzuna kopyalanır.

5.3.3 Çaprazlama İşlemi

Seçim havuzu tamamlandıktan sonra, havuzda bulunan bireyler ikişerli olarak (rasgele veya sistematik olarak) eşleştirilir. Eşleştirilen bireylere tek noktalı çaprazlama işlemi uygulanır. Tek noktalı çaprazlamada, eşleşen iki birey, aynı rasgele noktadan kesilerek, noktanın sağında bulunan veriler birbirleriyle değiştirilir. Seçim havuzundaki her bireye çaprazlama uygulandığında bu işlem sona erer.

5.3.4 Başkalaşım İşlemi

Çaprazlama işlemi tamamlandıktan sonra oluşan yeni bireylerdeki her bir ikili sayı için 0 ve 1 arasında rasgele değerler üretilir. Eğer üretilen rasgele sayı, arayüzde girilen başkalaşım oranından küçük ise, ikili sayının değeri tersine çevrilir (0'sa 1 değerini, 1'se 0 değerini alır). Eğer üretilen rasgele sayı başkalaşım oranından büyük ise, ikili sayı aynen kalır.

5.3.5 Elitizm

Başkalaşım işlemi tamamlandıktan sonra, nesildeki en iyi bireyler (birey sayısı tek ise 1, çift ise 2 adet en iyi birey) seçim havuzuna kopyalanır. Bu bireylere herhangi bir çaprazlama ve/veya başkalaşım işlemi uygulanmaz

5.3.6 Yeni Neslin Oluşturulması

Çaprazlama, başkalaşım ve elitizm tamamlandıktan sonra, seçim havuzunda bulunan tüm bireyler, önceki nesli oluşturan bireylerin yerine konur ve seçim havuzu boşaltılır.

5.3.7 Algoritmanın Sonlandırılması

Bu işlemler arayüzde girilen nesil sayısı kadar tekrarlandığında algoritmanın çalışması sonlandırılır. Algoritmanın uygun bir aşamada sonlandırılması için, nesil sayısının yanında iki adet daha sonlandırma ölçütü program içinde tanımlanmıştır. Bu sonlandırma ölçütlerinin ilkinde, programın çalışması 300 saniyeyi geçiyor ise, mevcut neslin hesaplanması tamamlandığında algoritma sonlandırılmaktadır. Diğer sonlandırma ölçütünde ise her bir nesildeki en iyi uyum değeri ve en kötü uyum değeri arasındaki fark hesap edilerek, bu fark en iyi uyum değerine bölünür. Bulunan değer %5'den daha küçük ise, algoritmanın yeterince yakınsadığı kabul edilerek, sonlandırılır.

Algoritma sonlandığında, elde edilen en iyi uyum değerine sahip kromozom için trenlerin hareketleri farklı renkler ile çizdirilerek, algoritmanın elde ettiği en iyi çizelge arayüz üzerinde kullanıcıya görüntülenir.

5.4 Algoritmanın Sınandığı Problemler

Algoritmanın sınanması için farklı örnek problemler geliştirilmiş ve bu problemler algoritmaya çözdürülmüştür. Problemlerin çözümünde, 1 Gb belleğe sahip bir Pentium IV 2.0 GHz işlemcili kişisel bilgisayar kullanılmıştır. Bir yönde 2, diğer yönde 1 trenden başlamak üzere, sistemde çalışan en fazla tren sayısı olan, bir yönde 8, diğer yönde 9 trenin çalıştığı değişik problemler oluşturulmuştur. Çatışmaların bulunduğu bir çizelge oluşturmak için bazı trenlerin gecikmiş olduğu kabul edilmiştir. Çizelgelerdeki zaman aralığı 02:00 ile 06:00'dır. Dolayısıyla, sistemde çalışan trenlerin saat 02:00 itibarıyla konumları arayüz üzerinde tanımlanmış ve algoritma başlatılmıştır. Bir yönde 2 tren çalışan her bir problem için ağırlıklı ve ağırlıksız gecikmeler ayrı ayrı hesaplanmış, daha büyük boyutlu problemler için ise yalnızca ağırlıklı gecikmeler en küçüklenmiştir. Genetik algoritmanın çalışmasının sınanması için her bir problem 10'ar kez çalıştırılmıştır. Her bir problem için elde edilen sonuçlar Çizelge 5.1'de görülmektedir. Çizelge 5.2'de ise, örnek problemler için yapılan 10 çalıştırma için çözüm süresi, uyum fonksiyonu değeri ve aranan çözüm adedinin ortalaması, standart sapması ve değişkenlik katsayısı görülmektedir. Çizelge 5.1 ve 5.2'de de görüleceği üzere, küçük boyutlu problemlerde elde edilen sonuçlar arasında çok fazla bir fark bulunmamaktadır (küçük standart sapmalı uyum değerleri). Problem boyutu büyüdükçe algoritmanın çözümleri birbirinden farklılık gösterebilmektedir (büyük standart sapmalı uyum değerleri). En iyi ve en kötü sonuçlar arasındaki fark %10'u aştığında, bir nesilde bulunan birey sayısının yeterli gelmediği saptanıp, bu değer arttırılarak, problem tekrar çözdürülmüştür.

Çizelge 5.1 Örnek problemler için elde edilen sonuçlar

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonk. değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Değişkenlik Katsayısı* Çözüm Süresi	Değişkenlik Katsayısı* Sonuç (uyum fonk. değeri)	Değişkenlik Katsayısı* Aranan Çözüm Adedi	En Küçük Sonuç (uyum fonk. değeri) (dak)	En Büyük Sonuç (uyum fonk. değeri) (dak)	Fark Oranı (%) (13-12)/ (12)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
1	2x1	Hayır	10	32,85	9,00	13,00	20,31	0,19	0,00	0,18	9,00	9,00	0,00
2	2x1	Evet	10	29,95	4,50	12,60	19,69	0,22	0,00	0,20	4,50	4,50	0,00
3	2x2	Hayır	10	39,38	9,00	16,20	6,32	0,19	0,00	0,22	9,00	9,00	0,00
4	2x2	Evet	10	36,17	4,50	15,10	5,89	0,13	0,00	0,12	4,50	4,50	0,00
5	2x3	Hayır	10	39,41	9,00	17,10	1,67	0,20	0,00	0,20	9,00	9,00	0,00
6	2x3	Evet	10	41,03	4,50	17,60	1,71	0,17	0,00	0,17	4,50	4,50	0,00
7	2x4	Hayır	10	81,61	9,00	24,00	$5,85x10^{-1}$	0,08	0,00	0,13	9,00	9,00	0,00
8	2x4	Evet	10	43,60	4,50	19,00	$4,64x10^{-1}$	0,21	0,00	0,24	4,50	4,50	0,00
9	2x5	Hayır	10	108,74	21,20	23,90	$1,46x10^{-1}$	0,38	0,05	0,26	20,00	22,00	10,00

Çizelge 5.1 Örnek problemler için elde edilen sonuçlar (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonk. değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonk. değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi	En Küçük Sonuç (uyum fonk. değeri) (dak)	En Büyük Sonuç (uyum fonk. değeri) (dak)	Fark Oranı (%) (13-12)/ (12)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
10	2x5	Evet	10	149,42	6,36	23,50	$1,43x10^{-1}$	0,18	0,00	0,15	6,36	6,36	0,00
11	2x6	Hayır	12	211,07	24,40	36,00	$5,49x10^{-2}$	0,32	0,05	0,27	22,00	25,00	13,64
12	2x6	Evet	12	231,50	6,79	36,20	$5,52x10^{-2}$	0,39	0,01	0,26	6,75	6,84	1,35
13	2x7	Hayır	12	163,80	109,40	35,10	$1,34x10^{-2}$	0,33	0,05	0,30	104,00	116,00	11,54
14	2x7	Evet	12	142,47	18,52	41,20	$1,57x10^{-2}$	0,23	0,00	0,27	18,47	18,56	0,46
15	2x8	Hayır	12	120,63	111,40	37,10	$3,54x10^{-3}$	0,24	0,05	0,20	104,00	118,00	13,46
16	2x8	Evet	14	176,24	18,51	50,50	$4,82x10^{-3}$	0,25	0,00	0,25	18,47	18,56	0,46
17	2x9	Hayır	14	179,00	140,20	48,60	$1,16x10^{-3}$	0,25	0,02	0,13	136,00	147,00	8,09
18	2x9	Evet	16	260,10	23,08	60,20	$1,44x10^{-3}$	0,24	0,04	0,13	22,47	24,50	9,03

Çizelge 5.1 Örnek problemler için elde edilen sonuçlar (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonk. değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonk. değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi	En Küçük Sonuç (uyum fonk. değeri) (dak)	En Büyük Sonuç (uyum fonk. değeri) (dak)	Fark Oranı (%) (13-12)/ (12)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
19	3x1	Evet	10	75,01	5,10	15,20	11,88	0,21	0,00	0,21	5,10	5,10	0,00
20	3x3	Evet	10	93,03	5,10	19,70	$2,40x10^{-1}$	0,25	0,00	0,18	5,10	5,10	0,00
21	3x4	Evet	10	62,22	5,10	19,30	$2,94x10^{-2}$	0,28	0,00	0,26	5,10	5,10	0,00
22	3x5	Evet	10	97,64	6,97	25,80	$4,92x10^{-3}$	0,34	0,01	0,29	6,96	7,07	1,64
23	3x6	Evet	12	194,36	7,42	33,80	$8,06x10^{-4}$	0,35	0,01	0,21	7,36	7,44	1,17
24	3x7	Evet	14	269,50	19,49	49,10	$1,46x10^{-4}$	0,22	0,04	0,17	19,07	21,10	10,64
25	3x8	Evet	14	245,72	19,93	49,80	$1,86x10^{-5}$	0,28	0,05	0,05	19,07	21,10	10,64
26	3x9	Evet	14	214,99	23,51	63,90	$2,98x10^{-6}$	0,30	0,03	0,29	23,07	25,20	9,23
27	4x1	Evet	10	49,55	5,10	15,60	6,09	0,43	0,00	0,19	5,10	5,10	0,00

Çizelge 5.1 Örnek problemler için elde edilen sonuçlar (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonk. değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonk. değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi	En Küçük Sonuç (uyum fonk. değeri) (dak)	En Büyük Sonuç (uyum fonk. değeri) (dak)	Fark Oranı (%) (13-12)/ (12)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
28	4x4	Evet	10	61,74	5,10	20,90	$1,99x10^{-3}$	0,19	0,00	0,16	5,10	5,10	0,00
29	4x5	Evet	10	94,36	6,96	30,00	$1,79x10^{-4}$	0,21	0,00	0,17	6,96	6,96	0,00
30	4x6	Evet	10	77,30	9,21	27,70	$1,03x10^{-5}$	0,28	0,00	0,27	9,16	9,24	0,94
31	4x7	Evet	14	185,55	20,92	60,50	$2,25x10^{-5}$	0,25	0,00	0,25	20,87	20,96	0,41
32	4x8	Evet	14	189,00	21,31	61,00	$8,88x10^{-8}$	0,32	0,04	0,30	20,87	22,90	9,72
33	4x9	Evet	14	279,79	26,40	58,70	$8,54x10^{-8}$	0,20	0,08	0,17	24,96	30,47	22,10
34	5x1	Evet	10	48,16	36,24	17,80	3,48	0,24	0,00	0,24	36,24	36,24	0,00
35	5x5	Evet	12	166,78	41,89	47,10	$8,77x10^{-6}$	0,29	0,05	0,31	40,53	44,90	10,79
36	5x6	Evet	14	164,88	44,54	51,00	$2,97x10^{-7}$	0,32	0,04	0,27	43,33	48,20	11,24

Çizelge 5.1 Örnek problemler için elde edilen sonuçlar (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonk. değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonk. değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi	En Küçük Sonuç (uyum fonk. değeri) (dak)	En Büyük Sonuç (uyum fonk. değeri) (dak)	Fark Oranı (%) (13-12)/ (12)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
37	5x7	Evet	12	147,72	57,91	50,20	$9{,}13x10^{-9}$	0,11	0,04	0,13	55,04	62,04	12,72
38	5x8	Evet	12	191,22	57,53	50,80	$2{,}89x10^{-10}$	0,36	0,04	0,32	55,04	61,01	10,85
39	5x9	Evet	12	235,64	70,72	39,90	$7{,}09x10^{-12}$	0,33	0,04	0,18	67,97	75,56	11,16
40	6x1	Evet	10	89,70	35,96	22,90	2,24	0,28	0,00	0,22	35,96	35,96	0,00
41	6x6	Evet	16	290,00	49,45	80,20	$7{,}29x10^{-9}$	0,07	0,08	0,07	43,33	54,94	26,81
42	6x7	Evet	16	301,94	62,14	71,70	$1{,}02x10^{-10}$	0,08	0,06	0,13	55,04	66,27	20,40
43	6x8	Evet	16	288,86	64,55	64,20	$1{,}43x10^{-12}$	0,19	0,08	0,26	57,19	71,91	25,76
44	6x9	Evet	16	326,33	71,13	59,80	$2{,}07x10^{-14}$	0,08	0,03	0,33	68,37	75,76	10,80
45	7x1	Evet	10	133,20	36,16	24,70	1,21	0,24	0,02	0,19	35,96	37,96	5,56

Çizelge 5.1 Örnek problemler için elde edilen sonuçlar (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonk. değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonk. değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi	En Küçük Sonuç (uyum fonk. değeri) (dak)	En Büyük Sonuç (uyum fonk. değeri) (dak)	Fark Oranı (%) (13-12)/ (12)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
46	7x7	Evet	16	309,70	60,44	65,70	$7,2 \times 10^{-13}$	0,11	0,09	0,18	55,59	71,49	28,60
47	7x8	Evet	16	304,95	64,67	67,50	$5,85 \times 10^{-15}$	0,10	0,09	0,16	55,59	71,86	29,27
48	7x9	Evet	16	331,58	69,86	51,90	$3,52 \times 10^{-17}$	0,07	0,08	0,19	61,84	77,27	24,95
49	8x1	Evet	10	129,04	36,76	23,30	$5,69 \times 10^{-1}$	0,37	0,03	0,23	35,96	37,96	5,56
50	8x8	Evet	16	320,19	70,83	56,70	$1,92 \times 10^{-17}$	0,07	0,05	0,20	63,69	74,66	17,23
51	8x9	Evet	16	286,84	71,83	61,50	$8,14 \times 10^{-20}$	0,16	0,05	0,22	67,33	76,47	13,58

*Değişkenlik katsayısı: Standart sapma/Ortalama değer. Ortalamaya göre normalize edilmiş standart sapmayı gösteren bu büyüklük, örnek sonuçlarındaki değişkenliğin karşılaştırılmasında kullanılır. Değişkenlik katsayısının büyük olması, ortalamaya göre standart sapmanın (saçılmanın) büyük olduğunun işaretidir.

Çizelge 5.2 Örnek problemler için bazı istatistiksel veriler

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonksiyonu değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Standart Sapma Çözüm Süresi (sn)	Standart Sapma Sonuç (uyum fonksiyonu değeri) (dak)	Standart Sapma Aranan Çözüm Adedi	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonksiyonu değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
1	2x1	Hayır	10	32,85	9,00	13,00	20,31	6,13	0,00	2,31	0,19	0,00	0,18
2	2x1	Evet	10	29,95	4,50	12,60	19,69	6,52	0,00	2,55	0,22	0,00	0,20
3	2x2	Hayır	10	39,38	9,00	16,20	6,32	7,42	0,00	3,52	0,19	0,00	0,22
4	2x2	Evet	10	36,17	4,50	15,10	5,89	4,64	0,00	1,79	0,13	0,00	0,12
5	2x3	Hayır	10	39,41	9,00	17,10	1,67	7,78	0,00	3,38	0,20	0,00	0,20
6	2x3	Evet	10	41,03	4,50	17,60	1,71	6,87	0,00	3,03	0,17	0,00	0,17
7	2x4	Hayır	10	81,61	9,00	24,00	$5{,}85 \times 10^{-1}$	6,63	0,00	3,10	0,08	0,00	0,13
8	2x4	Evet	10	43,60	4,50	19,00	$4{,}64 \times 10^{-1}$	9,23	0,00	4,50	0,21	0,00	0,24
9	2x5	Hayır	10	108,74	21,20	23,90	$1{,}46 \times 10^{-1}$	41,35	1,03	6,28	0,38	0,05	0,26

Çizelge 5.2 Örnek problemler için bazı istatistiksel veriler (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonksiyonu değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Standart Sapma Çözüm Süresi (sn)	Standart Sapma Sonuç (uyum fonksiyonu değeri) (dak)	Standart Sapma Aranan Çözüm Adedi	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonksiyonu değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
10	2x5	Evet	10	149,42	6,36	23,50	1,43x10-1	26,83	0,00	3,50	0,18	0,00	0,15
11	2x6	Hayır	10	211,07	24,40	36,00	$5,49x10^{-2}$	68,24	1,26	9,60	0,32	0,05	0,27
12	2x6	Evet	12	231,50	6,79	36,20	$5,52x10^{-2}$	89,40	0,04	9,53	0,39	0,01	0,26
13	2x7	Hayır	12	163,80	109,40	35,10	$1,34x10^{-2}$	54,02	5,80	10,48	0,33	0,05	0,30
14	2x7	Evet	12	142,47	18,52	41,20	$1,57x10^{-2}$	32,26	0,04	11,22	0,23	0,00	0,27
15	2x8	Hayır	12	120,63	111,40	37,10	$3,54x10^{-3}$	28,87	5,46	7,34	0,24	0,05	0,20
16	2x8	Evet	14	119,43	18,51	50,50	$4,82x10^{-3}$	43,46	0,04	12,63	0,25	0,00	0,25
17	2x9	Hayır	14	179,00	140,20	48,60	$1,16x10^{-3}$	45,00	3,46	6,40	0,25	0,02	0,13
18	2x9	Evet	16	260,10	23,08	60,20	$1,44x10^{-3}$	61,36	0,82	7,67	0,24	0,04	0,13

Çizelge 5.2 Örnek problemler için bazı istatistiksel veriler (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonksiyonu değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Standart Sapma Çözüm Süresi (sn)	Standart Sapma Sonuç (uyum fonksiyonu değeri) (dak)	Standart Sapma Aranan Çözüm Adedi	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonksiyonu değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
19	3x1	Evet	10	75,01	5,10	15,20	11,88	15,70	0,00	3,12	0,21	0,00	0,21
20	3x3	Evet	10	93,03	5,10	19,70	2,40x10-1	23,51	0,00	3,47	0,25	0,00	0,18
21	3x4	Evet	10	62,22	5,10	19,30	$2,94x10^{-2}$	17,16	0,00	4,99	0,28	0,00	0,26
22	3x5	Evet	10	97,64	6,97	25,80	$4,92x10^{-3}$	32,73	0,04	7,41	0,34	0,01	0,29
23	3x6	Evet	12	194,36	7,42	33,80	$8,06x10^{-4}$	67,33	0,04	7,11	0,35	0,01	0,21
24	3x7	Evet	14	269,50	19,49	49,10	$1,46x10^{-4}$	59,98	0,85	8,43	0,22	0,04	0,17
25	3x8	Evet	14	245,72	19,93	49,80	$1,86x10^{-5}$	69,16	1,01	2,57	0,28	0,05	0,05
26	3x9	Evet	14	214,99	23,51	63,90	$2,98x10^{-6}$	64,34	0,68	18,64	0,30	0,03	0,29
27	4x1	Evet	10	49,55	5,10	15,60	6,09	21,30	0,00	2,99	0,43	0,00	0,19

Çizelge 5.2 Örnek problemler için bazı istatistiksel veriler (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonksiyonu değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Standart Sapma Çözüm Süresi (sn)	Standart Sapma Sonuç (uyum fonksiyonu değeri) (dak)	Standart Sapma Aranan Çözüm Adedi	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonksiyonu değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
28	4x4	Evet	10	61,74	5,10	20,90	$1,99x10^{-3}$	11,71	0,00	3,41	0,19	0,00	0,16
29	4x5	Evet	10	94,36	6,96	30,00	$1,79x10^{-4}$	19,44	0,00	5,10	0,21	0,00	0,17
30	4x6	Evet	10	77,30	9,21	27,70	$1,03x10^{-5}$	21,35	0,04	7,53	0,28	0,00	0,27
31	4x7	Evet	14	185,55	20,92	60,50	$2,25x10^{-5}$	46,45	0,04	15,42	0,25	0,00	0,25
32	4x8	Evet	14	189,00	21,31	61,00	$8,88x10^{-8}$	60,41	0,84	18,15	0,32	0,04	0,30
33	4x9	Evet	14	279,79	26,40	58,70	$8,54x10^{-8}$	57,22	2,01	10,15	0,20	0,08	0,17
34	5x1	Evet	10	48,16	36,24	17,80	3,48	11,56	0,00	4,18	0,24	0,00	0,24
35	5x5	Evet	12	166,78	41,89	47,10	$8,77x10^{-6}$	48,80	1,89	14,60	0,29	0,04	0,31
36	5x6	Evet	14	164,88	44,54	51,00	$2,97x10^{-7}$	53,58	1,59	13,87	0,32	0,04	0,27

Çizelge 5.2 Örnek problemler için bazı istatistiksel veriler (devam)

Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonksiyonu değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Standart Sapma Çözüm Süresi (sn)	Standart Sapma Sonuç (uyum fonksiyonu değeri) (dak)	Standart Sapma Aranan Çözüm Adedi	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonksiyonu değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
37	5x7	Evet	12	147,72	57,91	50,20	$9{,}13 \times 10^{-9}$	16,11	2,12	6,56	0,11	0,04	0,13
38	5x8	Evet	12	191,22	57,53	50,80	$2{,}89 \times 10^{-10}$	67,90	2,19	16,06	0,36	0,04	0,32
39	5x9	Evet	12	235,64	70,72	39,90	$7{,}09 \times 10^{-12}$	78,36	2,62	7,31	0,33	0,04	0,18
40	6x1	Evet	10	89,70	35,96	22,90	2,24	25,03	0,00	5,11	0,28	0,00	0,22
41	6x6	Evet	16	270,10	49,45	80,20	$7{,}29 \times 10^{-9}$	893,39	4,11	5,96	0,07	0,08	0,07
42	6x7	Evet	16	301,94	62,14	71,70	$1{,}02 \times 10^{-10}$	25,14	3,47	8,99	0,08	0,06	0,13
43	6x8	Evet	16	288,86	64,55	64,20	$1{,}43 \times 10^{-12}$	55,49	4,90	16,51	0,19	0,08	0,26
44	6x9	Evet	16	326,33	71,13	59,80	$2{,}07 \times 10^{-14}$	24,99	2,19	19,75	0,08	0,03	0,33
45	7x1	Evet	10	133,20	36,16	24,70	1,21	31,31	0,63	4,60	0,24	0,02	0,19

Çizelge 5.2 Örnek problemler için bazı istatistiksel veriler (devam)

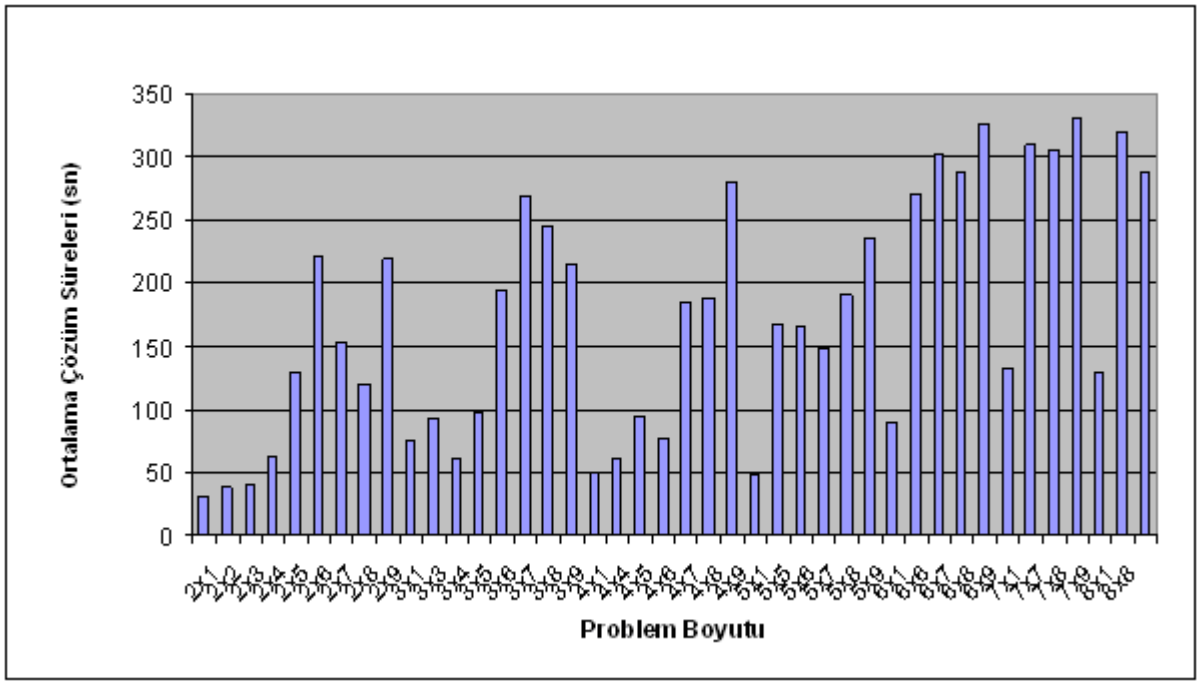
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Sonuç (uyum fonksiyonu değeri) (dak)	Ortalama Aranan Çözüm Adedi	Problem Uzayının Arama %si	Standart Sapma Çözüm Süresi (sn)	Standart Sapma Sonuç (uyum fonksiyonu değeri) (dak)	Standart Sapma Aranan Çözüm Adedi	Değişkenlik Katsayısı Çözüm Süresi	Değişkenlik Katsayısı Sonuç (uyum fonksiyonu değeri)	Değişkenlik Katsayısı Aranan Çözüm Adedi
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)
46	7x7	Evet	16	309,70	60,44	65,70	$7,2 \times 10^{-13}$	33,99	5,46	11,90	0,11	0,09	0,18
47	7x8	Evet	16	304,95	64,67	67,50	$5,85 \times 10^{-15}$	29,53	5,51	10,52	0,10	0,09	0,16
48	7x9	Evet	16	331,58	69,86	51,90	$3,52 \times 10^{-17}$	23,30	5,51	9,86	0,07	0,08	0,19
49	8x1	Evet	10	129,04	36,76	23,30	$5,69 \times 10^{-1}$	48,20	1,03	5,29	0,37	0,03	0,23
50	8x8	Evet	16	320,19	70,83	56,70	$1,92 \times 10^{-17}$	20,80	3,45	11,32	0,06	0,05	0,20
51	8x9	Evet	16	286,84	71,83	61,50	$8,14 \times 10^{-20}$	45,14	3,44	13,61	0,16	0,05	0,22

Çizelge 5.1'de de görülebileceği üzere, algoritmanın sınanması için 51 adet örnek problem geliştirilmiştir. Bir yönde hareket eden 2, diğer yönde hareket eden 1 trenin bulunduğu basit problemlerden başlanmış, bir yönde hareket eden 8, diğer yönde hareket eden 9 trenin bulunduğu nispeten zor problemlere kadar çeşitli problemler çözülmüştür. Genetik Algoritmaların rastlantısal özelliğinden ya da bir başka deyişle, her çalıştırıldığında ayrı bir sonuç elde edilebildiğinden ötürü her bir problem 10'ar defa çalıştırılmıştır. Bir yönde hareket eden 2 tren bulunan tüm problemler, hem gecikme toplamlarını en küçükleme, hem de ağırlıklı gecikme toplamlarını en küçükleme amaçlarıyla ayrı ayrı çözülmüş, diğer tüm problemlerde, amaç, ağırlıklı gecikme toplamlarını en küçüklemek olmuştur.

Problemlere kısa sürede çözüm elde edebilmek için, genetik algoritmaların değişkenleri mümkün olduğunca küçük tutulmuş, problem boyutu artıkça, problemin ihtiyacına bağlı olarak değişkenlerin boyutları da arttırılmıştır. Bu değişkenlerden bir tanesi olan her bir nesilde bulunan birey sayısı her problemde öncelikle 10 olarak alınmıştır. Algoritmanın 10 defa çalıştırılması sonucunda elde edilen en büyük ve en küçük değer arasındaki farkın, en küçük değere olan oranı belirli bir eşik değerini (%10'u) geçtiğinde, her nesilde bulunan birey sayısı 2'şer adet arttırılmıştır. Ancak 8x9 (bir yönde hareket eden 8, diğer yönde hareket eden 9 trenin bulunduğu) gibi büyük boyutlu problemlerde, her bir nesilde 16 birey kullanılmasına karşın, problem 5 dakikada sonlandırıldığı için, algoritmanın elde ettiği en küçük değer ile en büyük değer arasındaki fark %30'lara kadar ulaşmıştır.

Çizelge 5.2'de ayrıca, her bir problemin 10'ar defa çalıştırılması sonucunda, her bir çalıştırmanın ortalama süresi, ortalama uyum fonksiyonu değeri ve ortalama incelenen çözüm sayısının yanı sıra, bu değerlerin standart sapmaları ile ortalaması arasındaki oranı temsil eden değişkenlik katsayısının yanı sıra algoritmanın 10 defa çalışması sonucu elde edilen en büyük değer ile en küçük değer de görülmektedir. Ayrıca incelenen ortalama çözüm sayısına bağlı olarak, problem uzayının ne kadarlık bir kesiminin incelenerek çözümün elde edildiği ile en küçük ve en büyük çözüm arasındaki farkın en küçük çözüme oranı da görülmektedir.

Çizelge 5.1'den de görülebileceği üzere, özellikle küçük boyutlu problemlerde, algoritma 2 ila 3 dakika arasında çözüm elde edebilmektedir. Elde edilen çözümlerin ortalamaları, standart sapmaları ve fark oranları incelendiğinde, genellikle 10 çalışmanın tutarlı sonuçlar verdiğinden bahsedilebilir. Problem boyutu büyüdükçe, algoritmanın çalışma süresi de git gide uzamaktadır. Şekil 5.11'de problemin boyutu ile çözüm süresi arasındaki ilişki görülmektedir.



Şekil 5.11 Problem boyutu ile ortalama çözüm süresi arasındaki ilişki

Algoritmanın arama uzayının inceleyebildiği kesimi problem boyutu büyüdükçe azalmaktadır. Bunun nedeni, 5 dakika içinde incelenebilen çözüm sayısı sınırlı iken (bu çalışma kapsamında algoritmanın bir kez çalıştırılmasında en fazla 75 çözüm incelenebilmiştir) problem boyutu ile muhtemel çözümlerin sayısının üstel olarak artmasıdır.

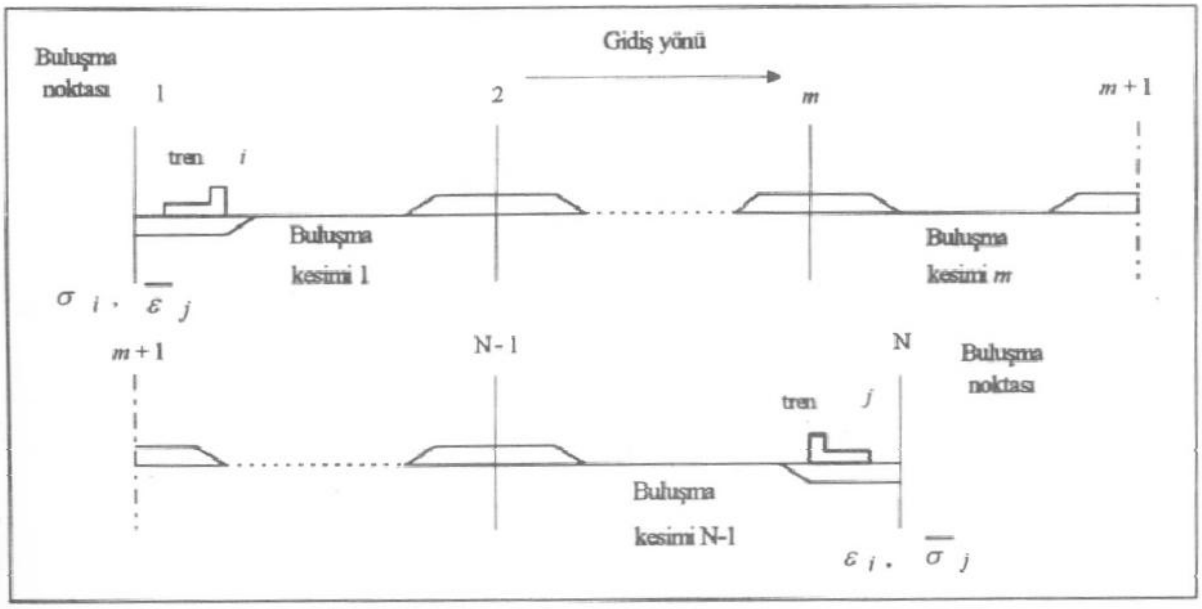
İzleyen bölümlerde ele alınan problem örnekleri için "en iyileme/optimizasyon" ve "yapay sinir ağı" yöntemleriyle çözümler elde edilmiştir. Böylece, genetik algoritma çözümlerinin ne kadar iyi olduğuna karar verilebilmektedir.

5.5 En İyileme Modeli

Geliştirilen genetik algoritmanın başarımını ölçmek için, öncelikle problem için (tren gecikmelerinin ya da ağırlıklı gecikmelerin en küçüklendiği) en uygun çözüm elde edilip, bu çözüm ile genetik algoritma çözümünün karşılaştırılması hedeflenmiştir. En iyi çözümün elde edilebilmesi için Şahin'in (1996 ve 1999) kullandığı modelden yararlanılmıştır.

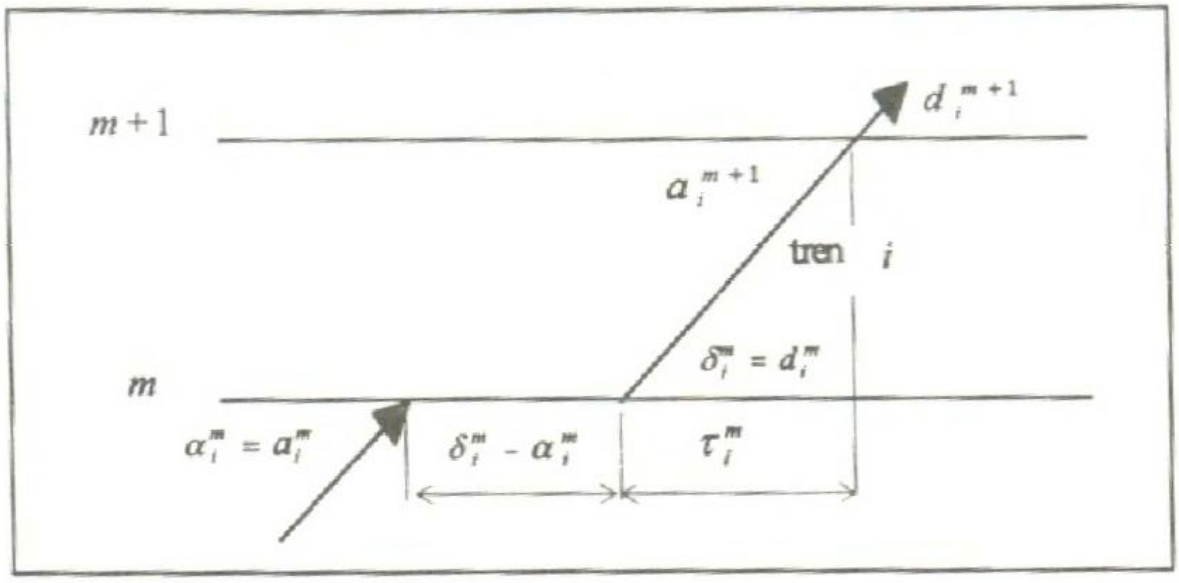
Bu modelde, hattın uç istasyonlarıyla, hat boyunca bulunan ara istasyon ve yan hatlar buluşma noktası olarak tanımlanmıştır. Buluşma noktaları gidiş yönünde sırasıyla ve "m" indeksiyle numaralanmış; ilk istasyona 1 ve son istasyona N indeks sayıları verilmiştir (Şekil 5.12). Her buluşma noktası m'de, her birinin uzunluğu μ^m ile belirli, iki hat (anahat ve yan

hat) bulunmaktadır. Bir tren yan hatta, uygun şekilde düzenlenmiş makas(lar) üzerinden geçer. Bir buluşma noktasında durdurulacak trenin uzunluğu, buluşma noktasının uzunluğundan küçük ya da eşit olmak zorundadır. Hat boyunca hız kısıtlaması uygulanan kesimler de bilinmektedir.



Şekil 5.12 Tek hatlı demiryolu, buluşma noktaları ve trenler

Giden trenlerin kümesi *I* ve dönen trenlerin kümesi *J* ile gösterilmektedir. Giden tren *i*'ye ait bazı özellikler aşağıdaki gibidir. Trenin ilk kalkış buluşma noktasının indeksi σ_i ve son varış buluşma noktasının ε_i olmakta; tren *i*'nin hareket yönünde sıralanmak üzere, belirli bir zamanda varması planlanmış ve/veya durması planlanmış buluşma noktalarının kümesi S_i, (σ_i, $\varepsilon_i \in S_i$); planlanmış varış zamanı α_i^m ve planlanmış kalkış zamanı δ_i^m (buna göre, planlı duruşun yapıldığı buluşma noktasındaki en küçük duruş süresi δ_i^m - α_i^m olmaktadır). $\forall m \in S_i$, trenin uzunluğu λ_i, ve tabii seyir süreleri $\tau_i = (\tau_i^{\sigma i},..., \tau_i^m, \tau_i^{m+1},..., \tau_i^{\varepsilon i-1})$ olmakta, burada τ_i^m buluşma noktaları *m* ve *m+1* arasındaki (ya da buluşma kesimi *m*) tabii seyir süresini göstermekte (Şekil 5.13); ve trenin, sistemdeki diğer trenler üzerindeki göreli önemi, temel öncelik sayısı indeksi, p_i ile gösterilmiştir. Temel öncelik sayısı, genellikle trenin hızına ve hareket yönüne göre işletme tarafından atanır; sayının küçük olması trenin önceliği olduğu anlamına gelmektedir.



Şekil 5.13 Tren *i*'nin tabii seyir ve en küçük duruş süresi

Trafik kontrol sistemine ilişkin olarak kullanılan indeksler de aşağıdaki gibidir: Giden tren *k*'nin giden tren *i*'yi buluşma noktaları *m* ve *m+1* arasında, hız kısıtı yapmaksızın izleyebilmesi için gerekli en küçük izleme süresi η_{tk}^m olmakta, (*i, k*$\in$ *I*) (Şekil 5.14); ve dönen trenler *j* ve *r* arasındaki en küçük izleme süresi $\bar{\eta}_{tk}^m$ olmaktadır (*j, r*$\in$ *J*). Ayrıca, giden tren *i*'nin *m* buluşma noktasına varış zamanıyla dönen tren *j*'nin *m*'den kalkış zamanı arasındaki fark en küçük güvenlik (karşılaşma) süresi ρ_{ij}^m (Şekil 5.15); ve dönen tren *j* ile giden tren *i* arasındaki en küçük karşılaşma süresi ρ_{ji}^m olmaktadır. Bir tren için yol tanzim etmek ve/veya sinyal açmak üzere harcanan süre trafik kontrol sisteminin özellikleriyle ilgilidir ve bu süre en küçük izleme ve karşılaşma süreleri içinde tanımlanmıştır.

Modelde ikili karar değişkenleri olan b_{ik}^m, $\bar{b}_{jr}^m$, c_{ij}^m için açıklama aşağıda verilmiştir:

b_{ik}^m = 1 eğer gidiş yönündeki tren *i*, buluşma noktası *m*'den, dönüş yönündeki tren *k*'den önce hareket ederse; aksi taktirde

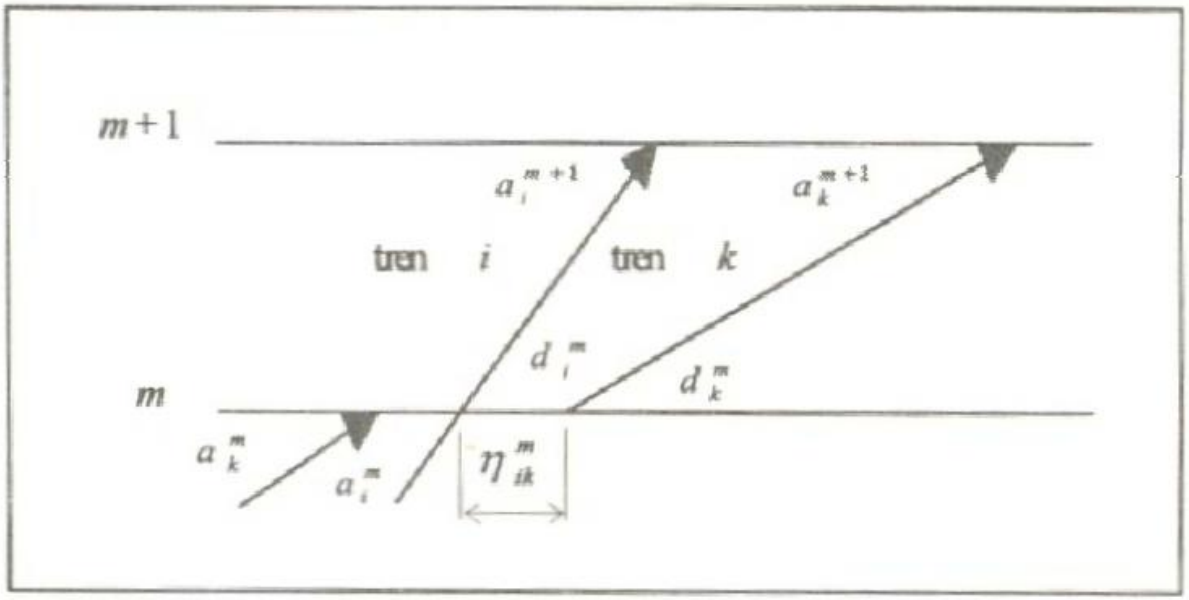
= 0 yani, tren *k* tren *i*'den önce hareket ederse

$\bar{b}_{jr}^m$ = 1 eğer dönüş yönündeki tren *j*, buluşma noktası *m*'den, gidiş yönündeki tren *r*'den önce hareket ederse; aksi taktirde

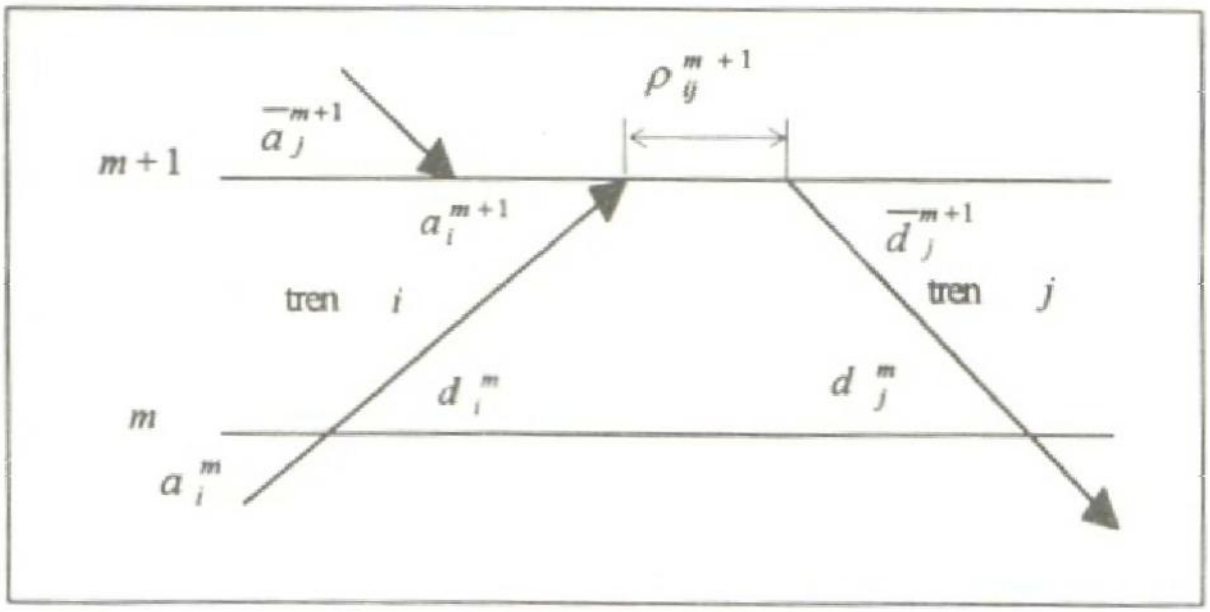
= 0 yani, tren *r* tren *j*'den önce hareket ederse

c_{ij}^{m} = 1 eğer gidiş yönündeki tren *i*, *m* ve *m+1* noktaları arasındaki kesimi (buluşma kesimi *m*) dönüş yönündeki tren *j*'den önce kat ederse; aksi taktirde

= 0 yani, tren *j* tren *i*'den önce kat ederse.



Şekil 5.14 Bir izleme çatışmasının çözümü ve öne geçme durumu (b_{ik}^{m} = 1)



Şekil 5.15 Bir karşılaşma çatışmasının çözümü (c_{ij}^{m} = 1)

İkili karar değişkenkleri, trenlerin (buluşma noktaları arasındaki) hat kesimlerini kullanma sırasını belirlemektedir. Bu değişkenler 0 ya da 1 değerini almaktadır. Model içinde bu değişkenlere sabit bir değer atanmadığı taktirde, model, bu ikili değişkenlerin alacağı değerleri, amaç fonksiyonunun en küçüklenmesi durumu için belirleyecektir. Bir başka deyimle, serbest bırakılan ikili değişkenler, tüm trenlerin aynı ve sabit temel öncelik sayısına sahip olduğu durum için değerlendirilirler. Böylece, çatışma halindeki iki trenden her birinin, hareket planı içinde kalan buluşma kesimlerini hemen kullanma ya da bekleme olasılığı %50 olmaktadır. Ancak, model, belirli kesimlerde belirli trenlere öncelik verilmesine olanak tanımaktadır. Örneğin, tren *i*'nin, tren *j* ile buluşma kesimi *m*'de karşılaşma çatışmasına girmesi durumunda, tren *i*'nin bu çatışmadan etkilenmemesi için $c_{ij}^{m} = 1$ olması gerekmektedir. Modele bir kısıt olarak eklenecek bu eşitlik yardımıyla, istenen durum gerçekleştirilmiş olur. Benzer işlem, aynı yönde hareket eden tren çiftleri için de yapılabilir. Fakat, bu işlemin modeldeki kısıt sayısını arttıracağı unutulmamalıdır.

Yardımcı sürekli değişkenler a_i^m ve d_i^m, sırasıyla, tren *i*'nin buluşma noktası *m*'deki gerçekleşen varış ve kalkış zamanlarını göstermektedir. Benzer şekilde $\bar{a}_j^m$ ve $\bar{d}_j^m$ tren *j*'nin buluşma noktası *m*'deki gerçekleşen varış ve kalkış zamanlarını göstermektedir. Gidiş yönündeki tren *i*'nin, hareket planındaki sapma sebebiyle, buluşma noktası *m*'ye gecikmeli olarak varması, her $m \in S_i$ için pozitif değişken e_i^m ile gösterilir; ve tren *j*'nin gecikmesi $\bar{e}_j^m$ ile gösterilir. ψ, model gereği, ikili karar değişkenlerini içeren kısıtların aktif olup olmamasını etkileyen, yeteri kadar büyük bir pozitif sayı (örneğin, 48 saat, Jovanovic, 1989) olarak kabul edilmiştir. İzleyen bölümde problemin matematik modeli verilmiştir.

5.5.1 Problemin Matematiksel Modeli

Amaç fonksiyonu:

Gecikme toplamlarının en küçüklenmesi için:

$$En\,küçük\, z = \sum_{i \in I} \sum_{m \in S_i} e_i^m + \sum_{j \in J} \sum_{m \in \bar{S}_i} \bar{e}_j^m \tag{5.1}$$

Ağırlıklı gecikme toplamlarının en küçüklenmesi için:

$$En\,küçük\, z = \sum_{i \in I} \sum_{m \in S_i} w_i e_i^m + \sum_{j \in J} \sum_{m \in \bar{S}_i} \overline{w_j e}_j^m \tag{5.2}$$

Kalkış ve varış zamanı kısıtları:

$$d_i^m \geq \delta_i^m \qquad \forall i \in I, m \in S_i \tag{5.3}$$

$$a_i^m \leq \alpha_i^m + e_i^m \quad \forall i \in I, m \in S_i \tag{5.4}$$

$$\bar{d}_j^m \geq \bar{\delta}_j^m \qquad \forall j \in J, m \in \bar{S}_j \tag{5.5}$$

$$\bar{a}_j^m \leq \bar{\alpha}_j^m + \bar{e}_j^m \quad \forall j \in J, m \in \bar{S}_j \tag{5.6}$$

Seyir süresi kısıtları:

$$a_i^{m+1} \geq d_i^m + \tau_i^m \qquad \forall i \in I, m = \sigma_i, ..., \varepsilon_i - 1 \tag{5.7}$$

$$a_j^m \geq \bar{d}_j^{m+1} + \bar{\tau}_j^m \qquad \forall j \in J, m = \bar{\varepsilon}_j, ..., \bar{\sigma}_j - 1 \tag{5.8}$$

En küçük duruş süresi kısıtları:

$$d_i^m \geq a_i^m + \delta_i^m - \alpha_i^m \quad \forall i \in I, m \in S_i \tag{5.9}$$

$$\bar{d}_j^m \geq \bar{a}_j^m + \bar{\delta}_j^m - \bar{\alpha}_j^m \quad \forall_j \in J, m \in \bar{S}_j \tag{5.10}$$

Gidiş yönündeki trenler için izleme ve öne geçme kısıtları:

$$d_i^m - d_k^m \geq \eta_{ki}^m - \psi b_{ik}^m \qquad \forall i,k \in I, i \neq k, m \in \left((\sigma_i, ..., \varepsilon_i - 1) \cap (\sigma_k, ..., \varepsilon_k - 1)\right) \tag{5.11}$$

$$d_k^m - d_i^m \geq \eta_{ki}^m - \psi(1 - b_{ik}^m) \quad \forall i,k \in I, i \neq k, m \in \left((\sigma_i, ..., \varepsilon_i - 1) \cap (\sigma_k, ..., \varepsilon_k - 1)\right) \tag{5.12}$$

$$a_i^{m+1} - a_k^{m+1} \geq \xi_{ki}^m - \psi b_{ik}^m \qquad \forall i,k \in I, i \neq k, m \in \left((\sigma_i, ..., \varepsilon_i - 1) \cap (\sigma_k, ..., \varepsilon_k - 1)\right) \tag{5.13}$$

$$a_k^{m+1} - a_i^{m+1} \geq \xi_{ki}^m - \psi(1 - b_{ik}^m) \quad \forall i,k \in I, i \neq k, m \in \left((\sigma_i, ..., \varepsilon_i - 1) \cap (\sigma_k, ..., \varepsilon_k - 1)\right) \tag{5.14}$$

Dönüş yönündeki trenler için izleme ve öne geçme kısıtları:

$$\bar{d}_j^{m+1} - \bar{d}_r^{m+1} \geq \bar{\eta}_{ik}^m - \psi \bar{b}_{ik}^m \qquad \forall j,r \in J,\ j \neq r, m \in \left(\left(\bar{\varepsilon}_j + 1, ..., \bar{\sigma}_j\right) \cap \left(\bar{\varepsilon}_r + 1, ..., \bar{\sigma}_r\right)\right) \tag{5.15}$$

$$\bar{d}_r^{m+1} - \bar{d}_j^{m+1} \geq \bar{\eta}_{jr}^m - \psi(1 - \bar{b}_{jr}^m) \qquad \forall j,r \in J,\ j \neq r, m \in \left(\left(\bar{\varepsilon}_j + 1, ..., \bar{\sigma}_j\right) \cap \left(\bar{\varepsilon}_r + 1, ..., \bar{\sigma}_r\right)\right) \tag{5.16}$$

$$\bar{a}_j^m - \bar{a}_r^m \geq \bar{\xi}_{ik}^m - \psi \bar{b}_{ik}^m \qquad \forall j,r \in J,\ j \neq r, m \in \left(\left(\bar{\varepsilon}_j + 1, ..., \bar{\sigma}_j\right) \cap \left(\bar{\varepsilon}_r + 1, ..., \bar{\sigma}_r\right)\right) \tag{5.17}$$

$$\bar{a}_r^m - \bar{a}_j^m \geq \bar{\xi}_{jr}^m - \psi(1-\bar{b}_{jr}^m) \qquad \forall j,r \in J,\ j \neq r, m \in \left(\left(\bar{\varepsilon}_j + 1,..., \bar{\sigma}_j\right) \cap \left(\bar{\varepsilon}_r + 1,..., \bar{\sigma}_r\right)\right) \qquad (5.18)$$

Karşılaşma kısıtları:

$$d_i^m - \bar{a}_j^m \geq \rho_{ji}^m - \psi c_{ij}^m \qquad \forall i \in I,\ j \in J, m \in \left(\left(\sigma_i,..., \varepsilon_i - 1\right) \cap \left(\bar{\varepsilon}_j,..., \bar{\sigma}_j - 1\right)\right) \qquad (5.19)$$

$$\bar{d}_j^{m+1} - a_i^{m+1} \geq \rho_{ji}^{m+1} - \psi(1-c_{ij}^m) \qquad \forall i \in I,\ j \in J, m \in \left(\left(\sigma_i,..., \varepsilon_i - 1\right) \cap \left(\bar{\varepsilon}_j,..., \bar{\sigma}_j - 1\right)\right) \qquad (5.20)$$

Ayrıca,

$$e_i^m \geq 0 \qquad \forall i \in I, m \in S_i \qquad (5.21)$$

$$\bar{e}_j^m \geq 0 \qquad \forall j \in J, m \in \bar{S}_j \qquad (5.22)$$

$$b_{ik}^m, \bar{b}_{jr}^m, c_{ij}^m = 1, \quad 0 \qquad \forall i,k,l \in I,\ j,p,r \in J$$

5.5.2 Modelin Açıklanması

Amaç fonksiyonu:

Amaç fonksiyonu (5.1), hatta işletilen tüm trenlerin belirli buluşma noktalarında gerçekleşen varış zamanlarının, planlanmış varış zamanlarından sapmalarının $\left(e_i^m, \bar{e}_j^m\right)$ toplamının en küçüklenmesidir.

Amaç fonksiyonu (5.2) ise, hatta işletilen tüm trenlerin belirli buluşma noktalarında gerçekleşen varış zamanlarının, planlanmış varış zamanlarından sapmalarının $\left(e_i^m, \bar{e}_j^m\right)$ ağırlıklı toplamının en küçüklenmesidir. Ağırlık değerleri, her trenin temel öncelik katsayısı esas alınarak belirlenmiştir.

Kalkış ve varış zamanı kısıtları:

Kısıtlar (5.3)-(5.6), trenlerin belirli noktalarda planlanmış zamandan daha erken kalkmasını önlediği gibi, geç varma durumlarında sapma değişkenleri $\left(e_i^m, \bar{e}_j^m\right)$ aracılığıyla, trenlerin belirli noktalardaki gecikmelerini göz önünde bulundurur.

Seyir süresi kısıtları:

Kısıtlar (5.7) ve (5.8), trenlerin performansına bağlı olarak, buluşma noktaları arasındaki tabii seyir sürelerini belirler (Şekil 5.13).

En küçük duruş süresi kısıtları:

Kısıtlar (5.9) ve (5.10), trenlerin belirli noktalarda, süresi önceden saptanmış beklemeleri yapmasını zorunlu kılar; bu süre yolcu veya yük hizmetleri için tahsis edilmiştir (Şekil 5.13).

Gidiş yönünde hareket eden trenler için izleme ve öne geçme kısıtları:

Kısıtlar (5.11) ve (5.12), bir buluşma noktasından kalkan ve aynı yönde hareket eden iki trenin hareket zamanları arasında en küçük izleme süresi bulunmasını zorunlu kılar. Eğer tren *i* buluşma noktası *m*'den tren *k*'den önce hareket ederse, $b_{ik}^{m} = 1$ olur, dolayısıyla kısıt (5.12) aktiftir. $b_{ik}^{m} = 0$ olması durumunda kısıt (5.11) aktiftir (Şekil 5.14).

Kısıtlar (5.13) ve (5.14), bir buluşma noktasına varan ve aynı yönde hareket eden iki trenin varış zamanları arasında en küçük izleme süresi bulunmasını zorunlu kılar. Eğer tren *i* buluşma noktası *m*'den tren *k*'den önce varırsa, $b_{ik}^{m} = 1$ olur, dolayısıyla kısıt (5.14) aktiftir. $b_{ik}^{m} = 0$ olması durumunda kısıt (5.13) aktiftir.

Dönüş yönündeki trenler için izleme ve öne geçme kısıtlarının açıklaması, gidiş yönündeki trenlerinki ile aynıdır.

Karşılaşma kısıtları:

Kısıtlar (5.19) ve (5.20), trenlerin tek hat üzerinde karşılaşmalarını önler. Karşılaşmaya sadece buluşma noktalarında izin verilir. Eğer tren *i* buluşma noktaları *m* ve *m+1* arasındaki kesimi tren *j*'den önce kat ederse, $c_{ij}^{m} = 1$ olur, dolayısıyla kısıt (5.22) aktiftir. Tersi gerçekleşirse $c_{ij}^{m} = 0$ olur ve kısıt (5.21) aktiftir (Şekil 5.15).

5.5.3 Modelin Sonuçları

LINDO paket programı ile yapılan çözümler sonucunda 2x1 boyutlu problemlere 1 saniyeden daha kısa bir süre içerisinde çözüm elde edilmiştir. Ancak, problem boyutunun büyümesi, problemin içerdiği değişken sayısını üstel olarak arttırmış, bu da programın çalışma süresini aşırı denilebilecek bir biçimde (2x6 boyutlu problemlere 24 saat geçmesine karşın çözüm elde edilememiştir) arttırmıştır. Bu nedenle, genetik algoritma sonuçlarını başka bir yöntem ile de karşılaştırma gereği doğmuştur. Genetik algoritma ile tamamen aynı sonuçları elde eden en iyileme modelinin sonuçları Çizelge 5.3'de görülmektedir.

Çizelge 5.3 Örnek problemler için genetik algoritma ve en iyileme modeli çözümlerinin karşılaştırması

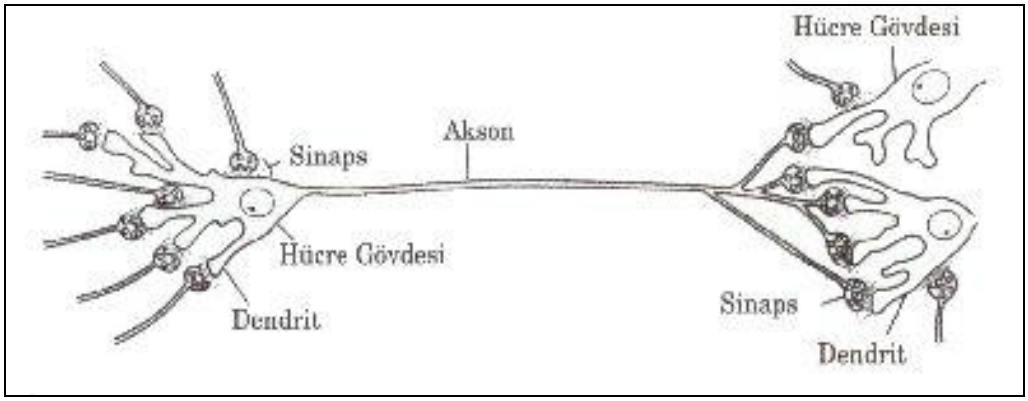
			Genetik Algoritma					En İyileme	
Prob. No	Boyut (iki yön. tren sayıları)	Ağırlık	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme	En Büyük Toplam Gecikme	En İyileme Sonuç (dak)	En İyileme Çözüm Süresi (sn)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
1	2x1	Hayır	32,85	9,00	13,00	9,00	9,00	9,00	<1
2	2x1	Evet	29,95	4,50	12,60	4,50	4,50	4,50	<1
3	2x2	Hayır	39,38	9,00	16,20	9,00	9,00	9,00	4,00
4	2x2	Evet	36,17	4,50	15,10	4,50	4,50	4,50	2,00
5	2x3	Hayır	39,41	9,00	17,10	9,00	9,00	9,00	66,00
6	2x3	Evet	41,03	4,50	17,60	4,50	4,50	4,50	65,00
7	2x4	Hayır	81,61	9,00	24,00	9,00	9,00	9,00	172,00
8	2x4	Evet	43,60	4,50	19,00	4,50	4,50	4,50	320,00
9	2x5	Hayır	108,74	21,20	23,90	20,00	22,00	-	> 86400
10	2x5	Evet	149,42	6,36	23,50	6,36	6,36	-	> 86400

5.6 Yapay Sinir Ağı Yöntemi

En iyileme modeli ile yalnızca çok küçük boyutlu problemlerde sonuç elde edilebildiği için, genetik algoritma sonuçlarını karşılaştırabilmek için başka bir yönteme ihtiyaç duyulmaktadır. Bu nedenle tren dispeçerlerinin karar davranışını taklit eden bir *yapay sinir ağı* modeli oluşturulmuştur. Oluşturulan modelin sonuçları, genetik algoritma sonuçları ile karşılaştırılmış, bu sayede genetik algoritmanın tren dispeçerlerine göre ne kadar iyi sonuçlar verdiği belirlenmeye çalışılmıştır.

Yapay sinir ağları (YSA), insan beyninin temel çalışma sistemi taklit edilerek geliştirilmiş bir yapay zekâ sistemidir. İnsan beyni milyonlarca adet nöron adı verilen sinir hücrelerinden oluşur (Şekil 5.16). Nöronlar, dendritlerden aldıkları elektrik sinyallerini hücre gövdesinde

işleyerek, sinapsları vasıtasıyla aksona ve aksondan da diğer hücrelere iletirler. Beş duyu tarafından tüm algılamalar beynin farklı bölgelerinde bu şekilde yapılır. Bu nedenle sinir hücrelerinin birbirlerinden farklı görevleri bulunmaktadır.

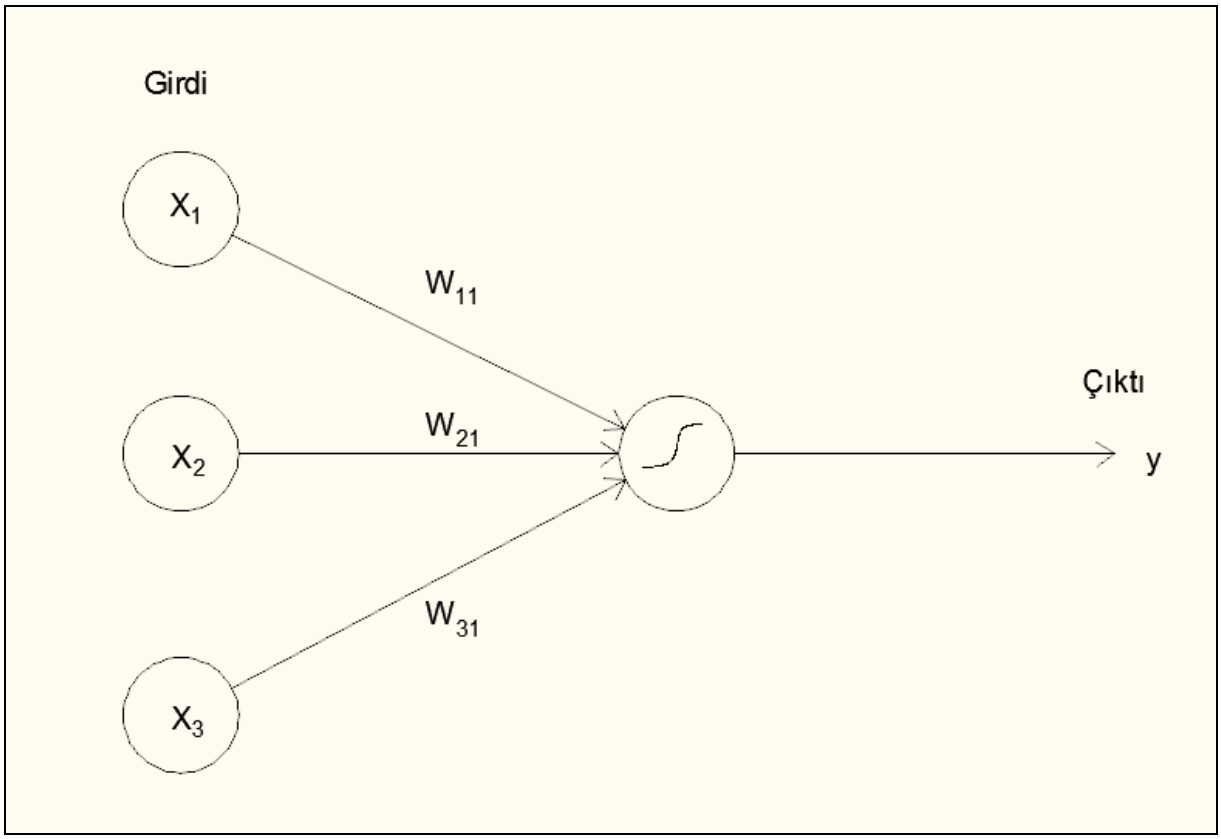


Şekil 5.16 Bir sinir hücresi (nöron) modeli

Biyolojik sistemlerde öğrenme, nöronlar arasındaki *sinaptik* bağlantıların ayarlanması ile olur. Yani, insanlar doğumlarından itibaren bir *yaşayarak öğrenme* süreci içerisine girerler. Bu süreç içinde beyin sürekli bir gelişme göstermektedir. Yaşayıp denedikçe sinaptik bağlantılar ayarlanır ve hatta yeni bağlantılar oluşur. Bu sayede öğrenme gerçekleşir.

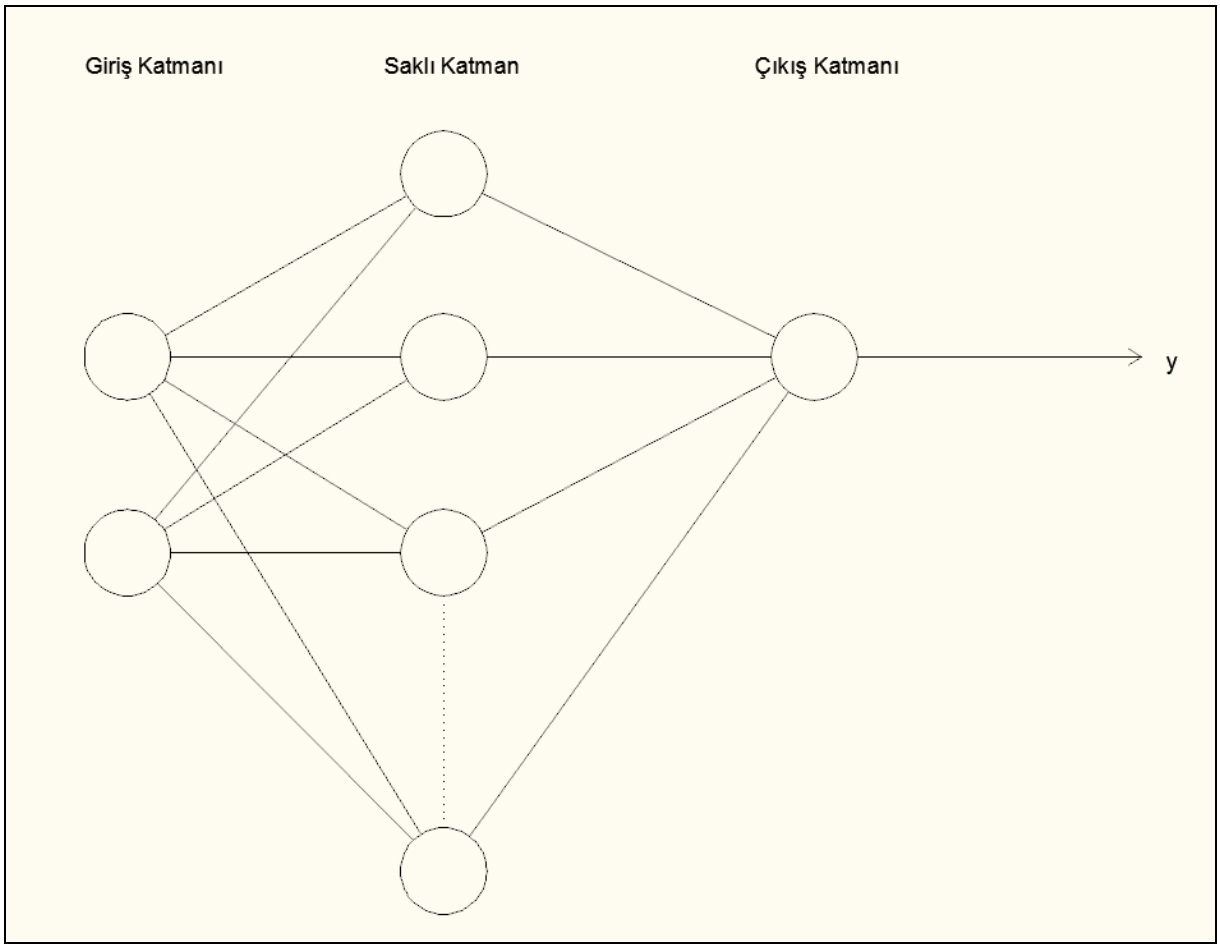
YSA, insan beyninin temel çalışma sistemi taklit edilerek geliştirilmiş bir yöntem olduğu için, tıpkı insan beyni gibi öğrenme, hatırlama, ezberleme gibi işlemleri yapabilme yeteneğine sahiptir. YSA tıpkı genetik algoritmalar gibi bir *kara kutu modeli*dir. Diğer bir deyişle, mevcut verilerden istenilen sonuçların elde edilmesi hedeflenen bir modeldir. Bu nedenle verilerden, sonuç elde edilmesine kadar geçen süreç içerisinde ne yapıldığından ziyade , elde edilen sonucun doğruluk değeri önem taşımaktadır.

YSA da tıpkı insan beyni gibi, yapay nöronların birbiriyle (kısmen ya da tamamen) bağlanması ile oluşur. Nöronlar dışarıdan gelen bilgiyi uygun bir biçimde işleyerek, diğer nöronlara ya da çıktı olarak kullanıcıya iletirler. Basit bir yapay sinir hücresi modeli Şekil 5.17'de görülebilmektedir.



Şekil 5.17 Bir yapay sinir hücresi (nöron) modeli

YSA bir ya da daha fazla nöron içeren katmanlar halinde oluşturulmaktadır (Şekil 5.18). Her bir nöron kendisinden önceki ve sonraki katmanlardaki nöronlar ile bağlı durumdadır. YSA'nda öğrenme de tıpkı beyin hücrelerinde olduğu gibi, nöronlar arasındaki sinaptik bağlantıların (ağırlıkların) değerlerinin ayarlanması ile gerçekleştirilir. Modelin elde ettiği sonuç ile hedeflenen sonuç arasındaki fark (hata) ağ boyunca geriye yayılarak, ağırlık değerleri güncellenir ve bu sayede de öğrenme gerçekleştirilir.



Şekil 5.18 Bir Yapay Sinir Ağı Modeli

YSA'nda sıklıkla kullanılan öğrenme yöntemi olan ileri beslemeli, geri yayılım'da (feed forward-back propagation) öncelikle giriş katmanındaki her bir nöronun giriş değerlerinin ağırlıklı toplamları alınır. Bu toplamlar bir aktivasyon fonksiyonuna (en sık kullanılanları doğrusal fonksiyon, sigmoid fonksiyonu ve tanjant sigmoid fonksiyonudur) uygulanır. Aktivasyon fonksiyonunun ürettiği değer, aynı zamanda nöronun da çıkış değeri, takip eden katmanlardaki nöronların da giriş değeridir. Her katmandaki nöronlar için bu işlem tekrarlandığında, en son katmandan elde edilen çıkış değer(ler)i, aynı zamanda modelin de çıktısını oluşturmaktadır. Elde edilen çıkış değeri ile hedeflenen çıkış değeri arasındaki fark, son katmandaki ağırlık değerlerinin güncellenmesinden başlayarak, geriye doğru yayılarak, her bir katmandaki ağırlık değerleri güncellenir. Eğitme kümesindeki her bir veri kümesi için bu işlem yapıldığında bir eğitme adımı gerçekleştirilmiş olur.

YSA'nda eğitme, modelin başarımını etkileyen en önemli aşamadır. Modelin genelleme yapabilmesi, yani öğretilmemiş değerler için de iyi çözümler sunabilmesi YSA'nın en önemli özelliklerindendir. Bunu sağlayabilmek için, mevcut veriler eğitme ve sınama olmak üzere iki

kümeye ayrılır. Ağırlık değerlerinin güncellenmesi esnasında yalnızca eğitme kümesindeki veriler kullanılırken, test kümesindeki veriler ezberlemenin (yalnızca ağa öğretilen veriler için iyi sonuçlar elde etme) önüne geçebilmek için bir sonlandırma ölçütü olarak ve ağın başarımını ölçmekte kullanılır.

5.6.1 Kullanılan Yapay Sinir Ağı Modeli

Bu çalışma kapsamında, Dündar (2003) tarafından geliştirilen yapay sinir ağı modeli kullanılmıştır. YSA, 8 giriş nöronu, saklı katmanda 20 nöron ve 1 çıkış nöronundan oluşan bir çok katmanlı ağ modelidir. Ağ, çatışmaya giren trenleri ikili olarak karşılaştırmakta ve hangi trenin geçeceğini veya bekleyeceğini saptamaktadır. Bu kararı verirken, çatışmaya giren trenlere ait 4'er adet veriyi değerlendirmektedir. Şahin (1996 ve 1999) ile Dündar (2003)'de kullanılan bu veriler;

- Temel Öncelik Sayısı
- Kritik Oran
- Miyopik Çözümdeki Gecikme
- Mevcut Çatışma Çözüldükten Sonra Trenin Gireceği Muhtemel Çatışma Sayısı

olarak belirlenmiştir. Her bir tren için 4'er adet olan bu veriler, YSA'nın giriş değerlerini oluşturmaktadır.

Temel öncelik sayısı, her trenin hızına bağlı olarak atanan sabit bir sayıdır. Genel kural, en hızlı trene en küçük sayının atanması şeklindedir. Kritik oran, trenin durumunu, planlanmış son varış zamanına kalan sürenin, kalan en küçük seyir süresine oranı olarak tanımlayan dinamik bir kuraldır. Küçük bir kritik oran, bir acil durum göstergesi olarak, trenin çizelgeden daha fazla geri kalmasını önlemek için, kısıtlamasız ilerlemesi gerektiği anlamına gelir.

Trenin girdiği çatışmanın kendi aleyhine çözümlenmesi durumunda, bu trenin, buluşma noktasında diğer trenin yolu boşaltıncaya kadar bekletilme süresi, miyopik çözümdeki gecikmesi olmaktadır. Trenin aktif çatışmadan sonra, son varış noktasına kadar, girebileceği muhtemel çatışmaların sayısı da, uygulamada göz önüne alınan nitelikler arasında yer almaktadır.

Geliştirilen YSA'nın çıktı değeri ise, trenlerden hangisinin bekletilip, hangisinin yoluna devam edeceği bilgisi olmaktadır. Çıkış olarak elde edilen 1 değeri, modeldeki ilk trenin

yoluna devam ettiğini, ikinci trenin bekletildiğini, 0 değeri ise bunun tam tersini temsil etmektedir.

YSA'nın geliştirilmesi için 10 adet test günü içerisinde çözümü dispeçerler tarafından yapılmış 331 adet çatışma saptanmıştır. Bu çatışmaların 173 tanesi eğitme kümesini, kalan 158 tanesi de sınama kümesini oluşturmuştur. Geliştirilen ağ, 331 çatışma çözümünün 327 tanesinde dispeçer kararlarını taklit etmeyi başarmıştır. %99'luk bir başarım oranı, benzer çalışmalarda rastlanan başarımların bir hayli üzerinde bir değer olarak göze çarpmaktadır.

5.6.2 Yapay Sinir Ağı Modeli Çözümü

Dündar (2003) tarafından geliştirilen YSA bu çalışmada da aynen kullanılmıştır. Oluşturulan 51 adet örnek problemde meydana gelen çatışmalar, geliştirilen YSA'na çözdürülmüştür. Rastlanılan her bir çatışmada, çatışmaya giren iki trenin 4'er adet özelliği ağa girilmiş, ağın çıktısı olan trenlerin bekletilme ve geçme kararları da el ile uygulanıp, trenlerin normal hareketlerine devam ettikleri varsayılmıştır. Çatışma çözümleri sırasında, 5.3.1'de bahsedilen izleme/güvenlik süreleri de kullanılmıştır. Bu nedenle, YSA'nın çözümü ile GA çözümü aynı olduğu takdirde, elde edilecek toplam (ağırlıklı) gecikme değerleri de aynı olmaktadır.

Uzman dispeçerlerin kararlarını taklit eden YSA'nın çözümleri, Çizelge 5.4'de görülebileceği üzere, küçük boyutlu problemler için GA çözümü ile büyük oranda benzerlik ya da yakınlık göstermektedir. Ancak problemin boyutu büyüdükçe, YSA'nın çözümlerinin, gecikmeleri en aza indirici çözüm olmaktan git gide uzaklaşmakta olduğu göze çarpmaktadır. Dikkat edilmesi gereken bir diğer nokta da, bir yönde 2 tren bulunan problemlerde 2 farklı amaç (gecikme toplamlarının en küçüklenmesi veya ağırlıklı gecikme toplamlarının en küçüklenmesi) bulunurken, YSA çözümü her iki problem için de aynıdır.

Oluşturulan 51 problemden, 14 tanesinde, YSA kararı, dolayısıyla da dispeçerin kararı en iyi GA'nın çözümü ile aynı olmuştur. Geri kalan 37 problemde ise, GA, YSA'dan daha iyi sonuç sağlamıştır. GA'nın çözüm ortalamaları incelendiğinde, 9 ve 17 no'lu problemlerde, YSA'nın başarımının altında kaldığı görülmektedir. Bunun nedeni de, GA'nın 10 kez çalıştırılmasında en az bir kere en iyi sonuçtan farklı bir sonuç sağlamış olmasıdır.

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
1	2x1	Hayır	10	32,85	9,00	13,00	9,00	9,00	11,00	22,22	22,22
2	2x1	Evet	10	29,95	4,50	12,60	4,50	4,50	5,50	0,00	0,00
3	2x2	Hayır	10	39,38	9,00	16,20	9,00	9,00	9,00	0,00	0,00
4	2x2	Evet	10	36,17	4,50	15,10	4,50	4,50	4,50	0,00	0,00
5	2x3	Hayır	10	39,4106	9,00	17,10	9,00	9,00	9,00	0,00	0,00
6	2x3	Evet	10	41,03	4,50	17,60	4,50	4,50	4,50	0,00	0,00
7	2x4	Hayır	10	81,61	9,00	24,00	9,00	9,00	9,00	0,00	0,00
8	2x4	Evet	10	43,60	4,50	19,00	4,50	4,50	4,50	0,00	0,00

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
9	2x5	Hayır	10	108,74	21,20	23,90	20,00	22,00	20,00	-5,66	0,00
10	2x5	Evet	10	149,42	6,36	23,50	6,36	6,36	6,47	1,80	1,80
11	2x6	Hayır	10	211,07	24,40	36,00	22,00	25,00	54,00	121,31	145,45
12	2x6	Evet	12	231,50	6,79	36,20	6,75	6,84	15,23	124,25	125,57
13	2x7	Hayır	12	163,80	109,40	35,10	104,00	116,00	126,00	15,17	21,15
14	2x7	Evet	12	142,47	18,52	41,20	18,47	18,56	26,19	41,37	26,19
15	2x8	Hayır	12	120,63	111,40	37,10	104,00	118,00	126,00	13,11	21,15
16	2x8	Evet	14	119,43	18,51	50,50	18,47	18,56	26,19	41,50	41,76

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
17	2x9	Hayır	14	179,00	140,20	48,60	136,00	147,00	136,00	-3,00	0,00
18	2x9	Evet	16	260,10	23,08	60,20	22,47	24,50	33,86	46,71	50,67
19	3x1	Evet	10	75,01	5,10	15,20	5,10	5,10	10,60	107,84	107,84
20	3x3	Evet	10	93,03	5,10	19,70	5,10	5,10	5,10	0,00	0,00
21	3x4	Evet	10	62,22	5,10	19,30	5,10	5,10	5,10	0,00	0,00
22	3x5	Evet	10	97,64	6,97	25,80	6,96	7,07	13,97	100,49	100,82
23	3x6	Evet	12	194,36	7,42	33,80	7,36	7,44	15,83	113,41	115,15
24	3x7	Evet	14	269,50	19,49	49,10	19,07	21,10	26,79	37,40	40,45

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
25	3x8	Evet	14	245,72	19,93	49,80	19,07	21,10	26,79	34,43	40,45
26	3x9	Evet	14	214,99	23,51	63,90	23,07	25,20	39,11	66,37	69,54
27	4x1	Evet	10	49,55	5,10	15,60	5,10	5,10	5,10	0,00	0,00
28	4x4	Evet	10	61,74	5,10	20,90	5,10	5,10	5,10	0,00	0,00
29	4x5	Evet	10	94,36	6,96	30,00	6,96	6,96	13,97	100,82	100,82
30	4x6	Evet	10	77,30	9,21	27,70	9,16	9,24	18,20	97,64	98,75
31	4x7	Evet	14	185,55	20,93	60,50	20,87	20,96	28,96	38,40	38,74
32	4x8	Evet	14	189,00	21,31	61,00	20,87	22,90	28,96	35,88	38,74

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
33	4x9	Evet	14	279,79	26,40	58,70	24,96	30,47	39,68	50,32	59,02
34	5x1	Evet	10	48,16	36,24	17,80	36,24	36,24	36,24	0,00	0,00
35	5x5	Evet	12	166,78	41,89	47,10	40,53	44,90	52,30	24,85	29,04
36	5x6	Evet	14	164,88	44,54	51,00	43,33	48,20	57,76	29,68	33,30
37	5x7	Evet	12	147,72	57,91	50,20	55,04	62,04	62,01	7,09	12,67
38	5x8	Evet	12	191,22	57,53	50,80	55,04	61,01	76,17	32,41	38,39
39	5x9	Evet	12	235,64	70,72	39,90	67,97	75,56	85,34	20,68	25,56
40	6x1	Evet	10	89,70	35,96	22,90	35,96	35,96	53,17	47,88	47,87

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
41	6x6	Evet	16	270,10	49,45	80,20	43,33	54,94	68,56	38,63	58,23
42	6x7	Evet	16	301,94	62,14	71,70	55,04	66,27	78,37	26,11	42,38
43	6x8	Evet	16	288,86	64,55	64,20	57,19	71,91	96,07	48,82	68,00
44	6x9	Evet	16	326,33	71,13	59,80	68,37	75,76	92,66	30,27	35,52
45	7x1	Evet	10	133,20	36,16	24,70	35,96	37,96	64,99	79,73	80,73
46	7x7	Evet	16	309,70	60,44	65,70	55,59	71,49	119,44	97,63	114,88
47	7x8	Evet	16	304,95	64,67	67,50	55,59	71,86	100,46	55,33	80,72
48	7x9	Evet	16	331,58	69,86	51,90	61,84	77,27	111,59	59,74	80,43

Çizelge 5.4 Örnek problemler için genetik algoritma ve yapay sinir ağı çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Yapay Sinir Ağı		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Yapay Sinir Ağı Çözümündeki Gecikme	YSA ile GA Ortalama Farkı (%) (10-6)/(6)	YSA ile GA En Küçük Farkı (%) (10-8)/(8)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
49	8x1	Evet	10	129,04	36,76	23,30	35,96	37,96	64,99	76,80	80,73
50	8x8	Evet	16	320,19	70,83	56,70	63,69	74,66	120,57	70,23	89,32
51	8x9	Evet	16	286,84	71,83	61,50	67,33	76,47	120,37	67,58	78,78

5.7 Melez Model

Bu çalışma kapsamında geliştirilen GA'nın başarımının arttırılabilmesi için, YSA ile birlikte bir melez model tasarlanmıştır. Model öncelikle, YSA dispeçer kararlarını taklit ederek, bir başlangıç çözüm üretmektedir. Üretilen başlangıç çözümü, 2 ayrı kromozom (birey) halinde kodlanarak GA'nın başlangıç nesline girilmektedir. Bu bireylerden ilkinde, algoritmanın çözdüğü çatışmalar dışındaki çatışmaları temsil eden kromozomlar 0, ikincisinde de 1 değerini almaktadır. Bu sayede, çatışmaların farklı çözülmesi durumunda ortaya çıkabilecek diğer çatışmalar için tüm olasılıklar da algoritmanın başlangıç neslinde bulunmuş olmaktadır. Daha sonra GA, tıpkı bölüm 5.3'de anlatıldığı gibi çalıştırılmaktadır.

Geliştirilen melez model, yalnızca birkaç örnek problem üzerinde denenmiştir. Denemelerde, tıpkı GA'da olduğu gibi model 10 kez çalıştırılmış ve her bir çalışmanın ortalaması alınmıştır. Küçük boyutlu problemlerde melez model GA ile aynı sonucu vermekte, ancak çalışma süresi çoz az da olsa düşmektedir. Problem boyutu büyüdükçe, orta ve büyük boyutlu problemler söz konusu olduğunda modelin başarımı kayda değer bir artış göstermemektedir. Bunun nedeni de, YSA çözümlerinin (ağırlıklı) gecikme toplamlarını en küçükleyen çözümden oldukça farklı olmasıdır. Çizelge 5.5'de melez modelin çalıştırılan örnek problemlerdeki başarımı görülmektedir.

Çizelge 5.5 Örnek problemler için genetik algoritma ve melez modelin çözümlerinin karşılaştırması

			Genetik Algoritma						Melez Model		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Melez Model En İyi Çözümü (gecikme)	Melez Model Ortalama Toplam Gecikme (dak)	Melez Model Ortalama Çözüm Süresi (sn)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
1	2x2	Hayır	10	39,38	9,00	16,20	9,00	9,00	9,00	9,00	31,23
2	2x2	Evet	10	36,17	4,50	15,10	4,50	4,50	4,50	4,50	32,96
3	3x1	Evet	10	75,01	5,10	15,20	5,10	5,10	5,10	5,10	70,54
4	3x3	Evet	10	93,03	5,10	19,70	5,10	5,10	5,10	5,10	88,24
5	4x5	Evet	10	94,36	6,96	30,00	6,96	6,96	6,96	6,96	96,27
6	4x6	Evet	10	77,30	9,21	27,70	9,16	9,24	9,16	9,23	76,77
7	4x7	Evet	14	185,55	20,92	60,50	20,87	20,96	20,87	20,87	166,44
8	6x7	Evet	16	301,94	62,14	71,70	55,04	66,27	57,04	61,53	299,46

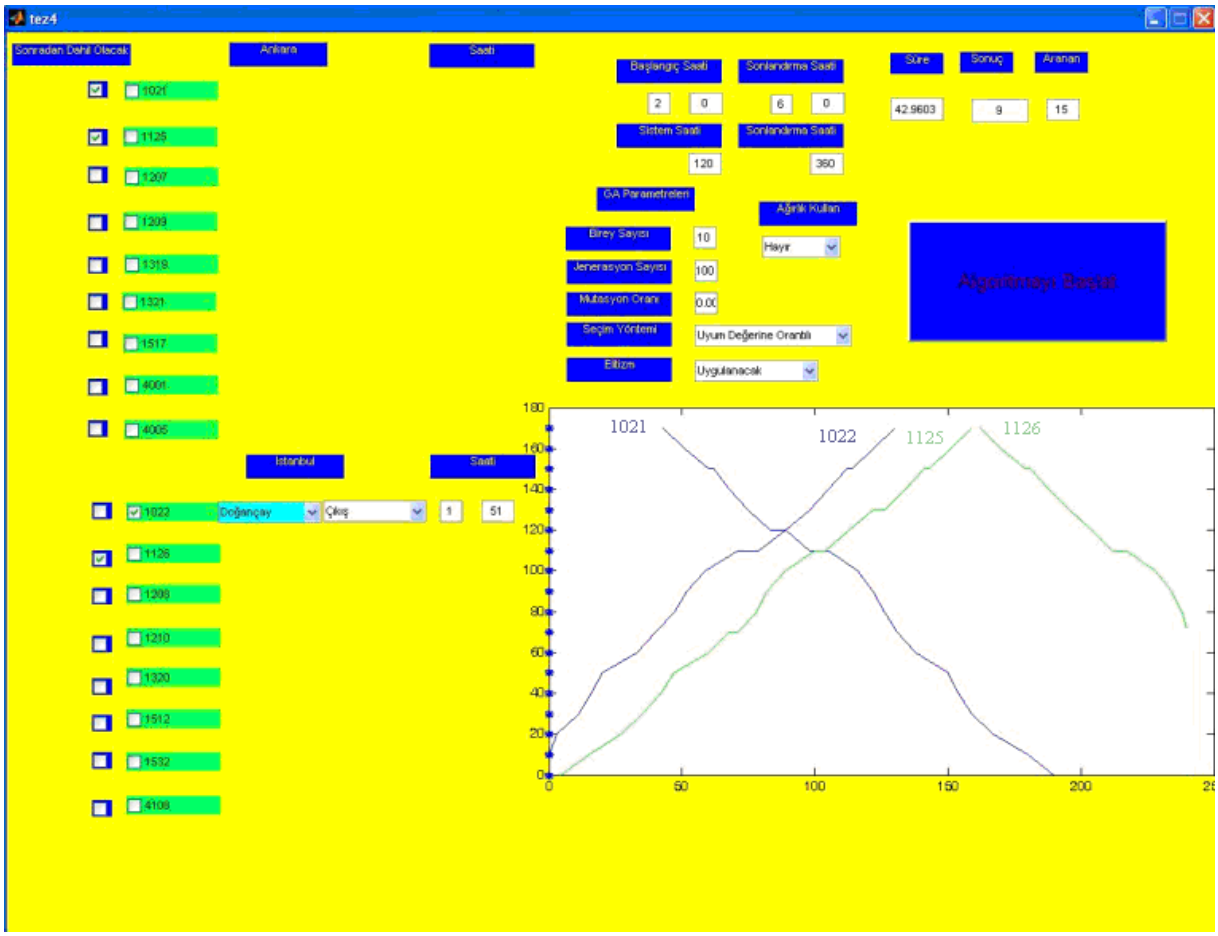
Çizelge 5.5 Örnek problemler için genetik algoritma ve melez modelin çözümlerinin karşılaştırması (devam)

			Genetik Algoritma						Melez Model		
Problem No	Boyut (iki yöndeki tren sayıları)	Ağırlık	Nesildeki Birey Sayısı	Ortalama Çözüm Süresi (sn)	Ortalama Toplam Gecikme (dak)	Ortalama Aranan Çözüm Adedi	En Küçük Toplam Gecikme (dak)	En Büyük Toplam Gecikme (dak)	Melez Model En İyi Çözümü (gecikme)	Melez Model Ortalama Toplam Gecikme (dak)	Melez Model Ortalama Çözüm Süresi (sn)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
9	6x8	Evet	16	288,86	64,55	64,20	57,19	71,91	57,19	66,33	295,65
10	6x9	Evet	16	326,33	71,13	59,80	68,37	75,76	69,44	71,04	311,13
11	8x8	Evet	16	320,19	70,83	56,70	63,69	74,66	64,63	70,24	318,57
12	8x9	Evet	16	286,84	71,83	61,50	67,33	76,47	67,33	74,56	298,44

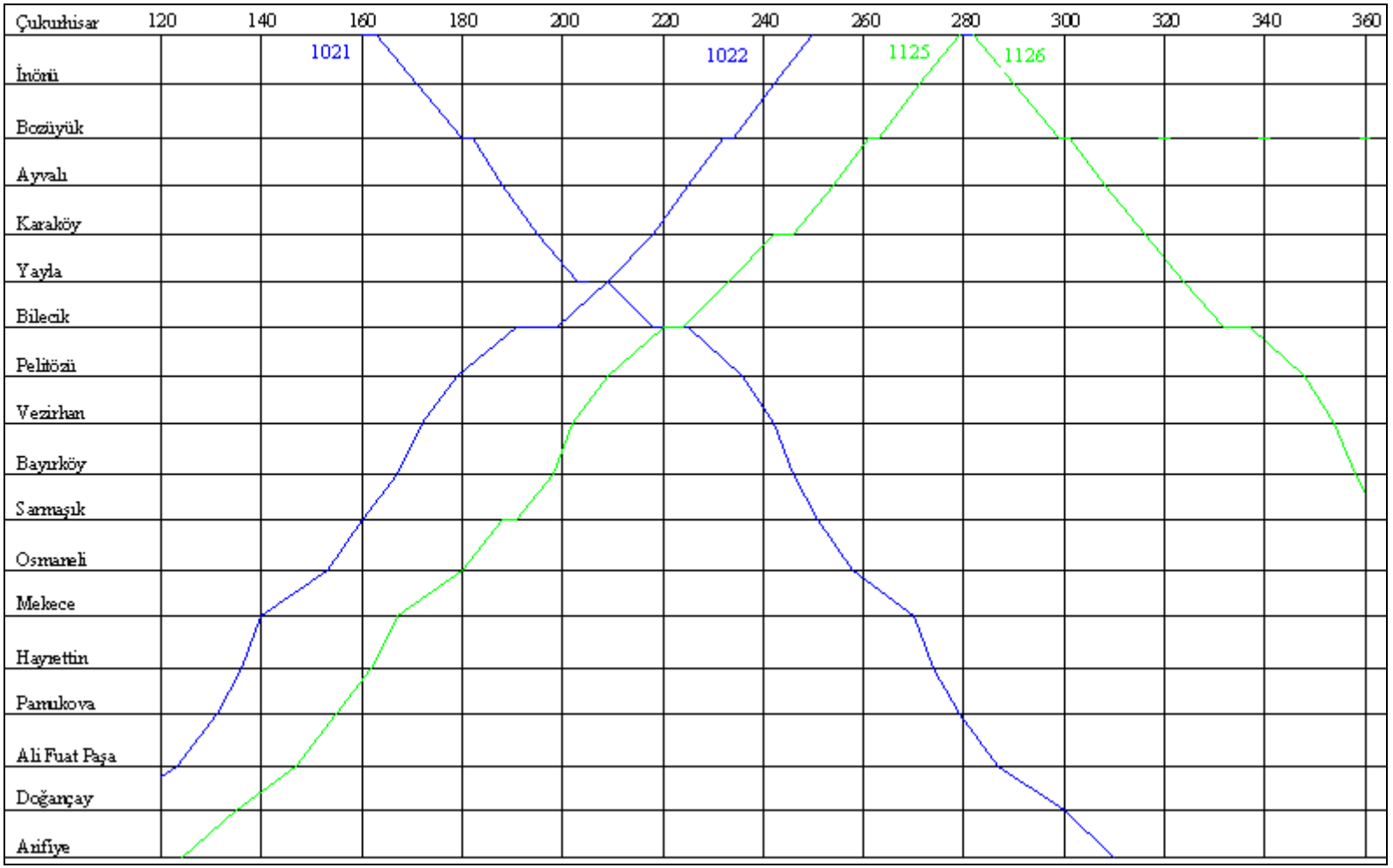
5.8 Bazı Örnek Problemlerin İncelenmesi

5.8.1 Örnek Problem 4 (2x2 Tren)

Örnek problem 4'de Ankara yönünden gelen 1021 ve 1125 kodlu trenler ile İstanbul yönünden gelen 1022 ve 1126 kodlu trenler bulunmaktadır. Bu trenlerin 02:00 itibarıyla konum bilgileri ve genetik algoritmanın elde ettiği en iyi çözüm Şekil 5.19'da görülmektedir. Burada, 1021 kodlu trenin, 1022 kodlu trenle girdiği karşılaşma çatışmasında, 1021 beklemiş ve 1022 yoluna devam etmiştir. Bu çatışma çözümü sonucunda trenlerin toplam gecikmesi 9 dakika olmaktadır. GA bu çözüme ortalama 36,17 saniyede ulaşmış ve 10 çalışmanın tamamında aynı sonuç elde edilmiştir. YSA çözümü de GA algoritma ile tamamen aynı sonucu sağlamaktadır. YSA'nın el ile çizdirilen çözümü Şekil 5.20'de görülmektedir.



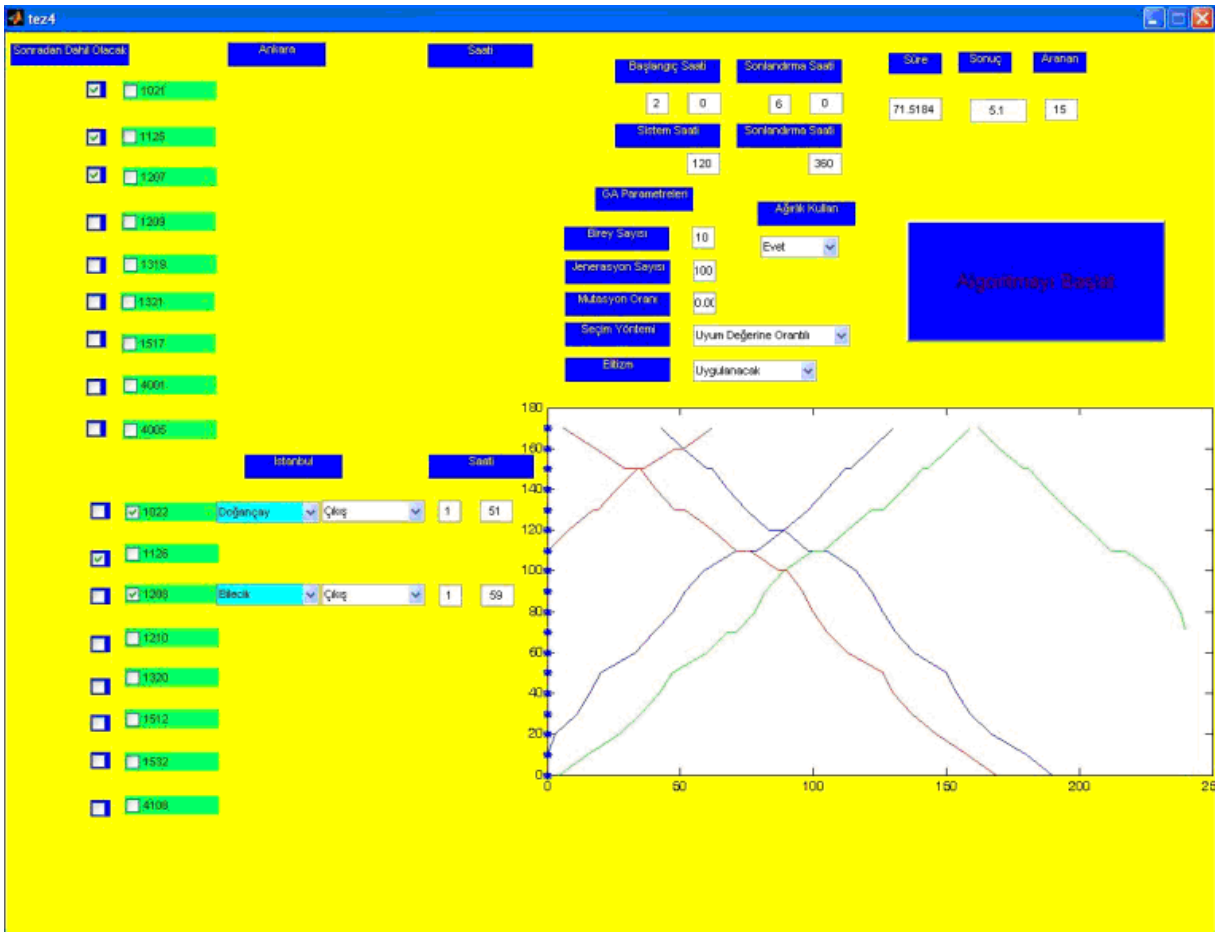
Şekil 5.19 Problem 4'ün genetik algoritma çözümü



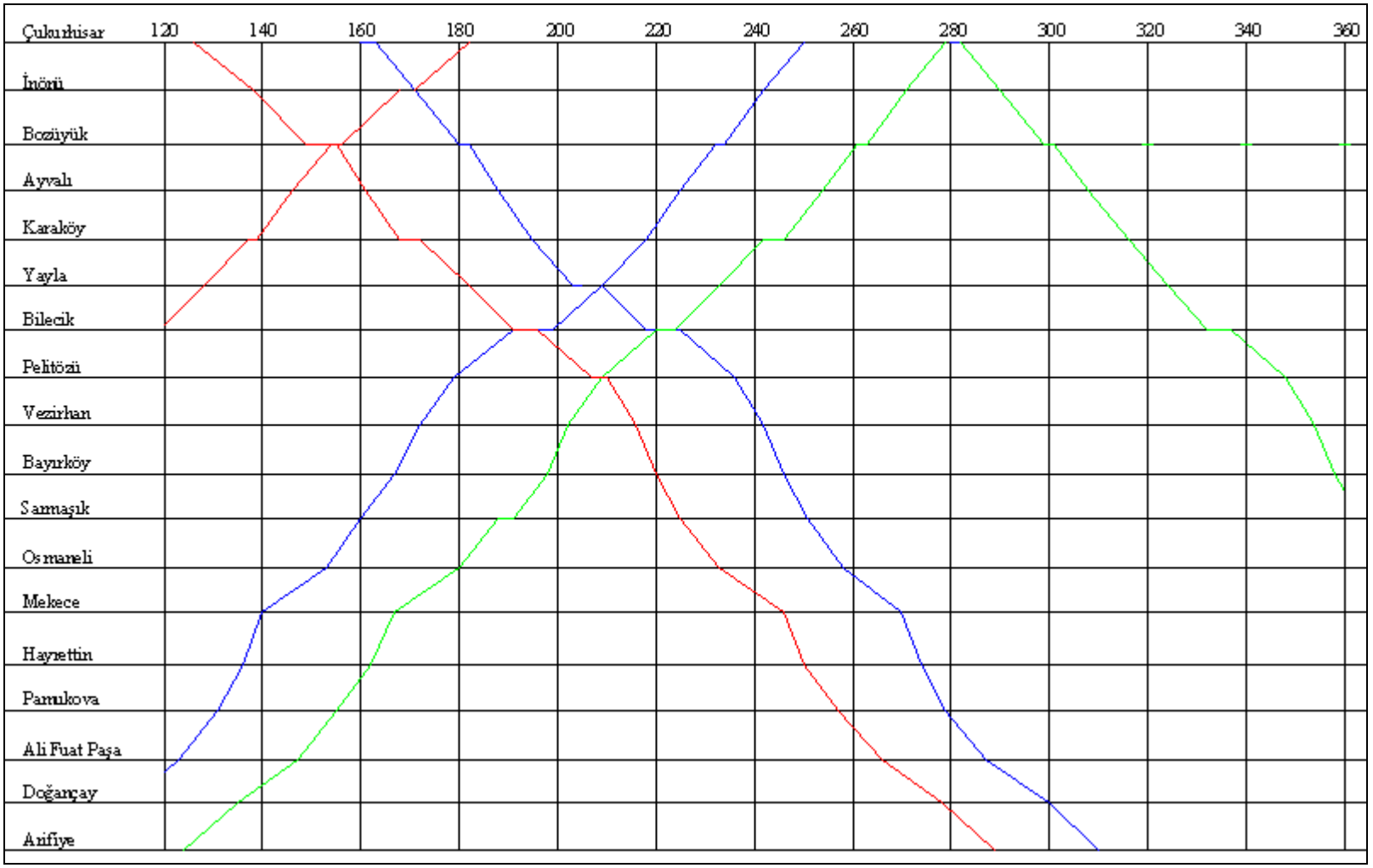
Şekil 5.20 Problem 4'ün yapay sinir ağı çözümü

5.8.2 Örnek Problem 20 (3x3 Tren)

Bu problemde, her bir yönde hareket eden 3'er adet tren bulunmaktadır. Bazı trenler, 02:00 itibarıyla sistemde hareketlerini sürdürürken, diğerleri sisteme ilk istasyondan hareket saatleri geldiğinde dahil olmaktadırlar. GA'nın 10 kere çalışması sonucu elde edilen en iyi çözüm Şekil 5.21'de görülmektedir. Bu çatışma çözümü sonucunda, ağırlıklı gecikme toplamları 5,1 dakika olmuştur. Algoritmanın her bir çalışması ortalama 93,03 saniye sürmektedir. YSA tarafından yapılan çözüm de GA algoritma ile tamamen aynı sonucu sağlamaktadır. YSA'nın el ile çizdirilen çözümü Şekil 5.22'de görülmektedir.



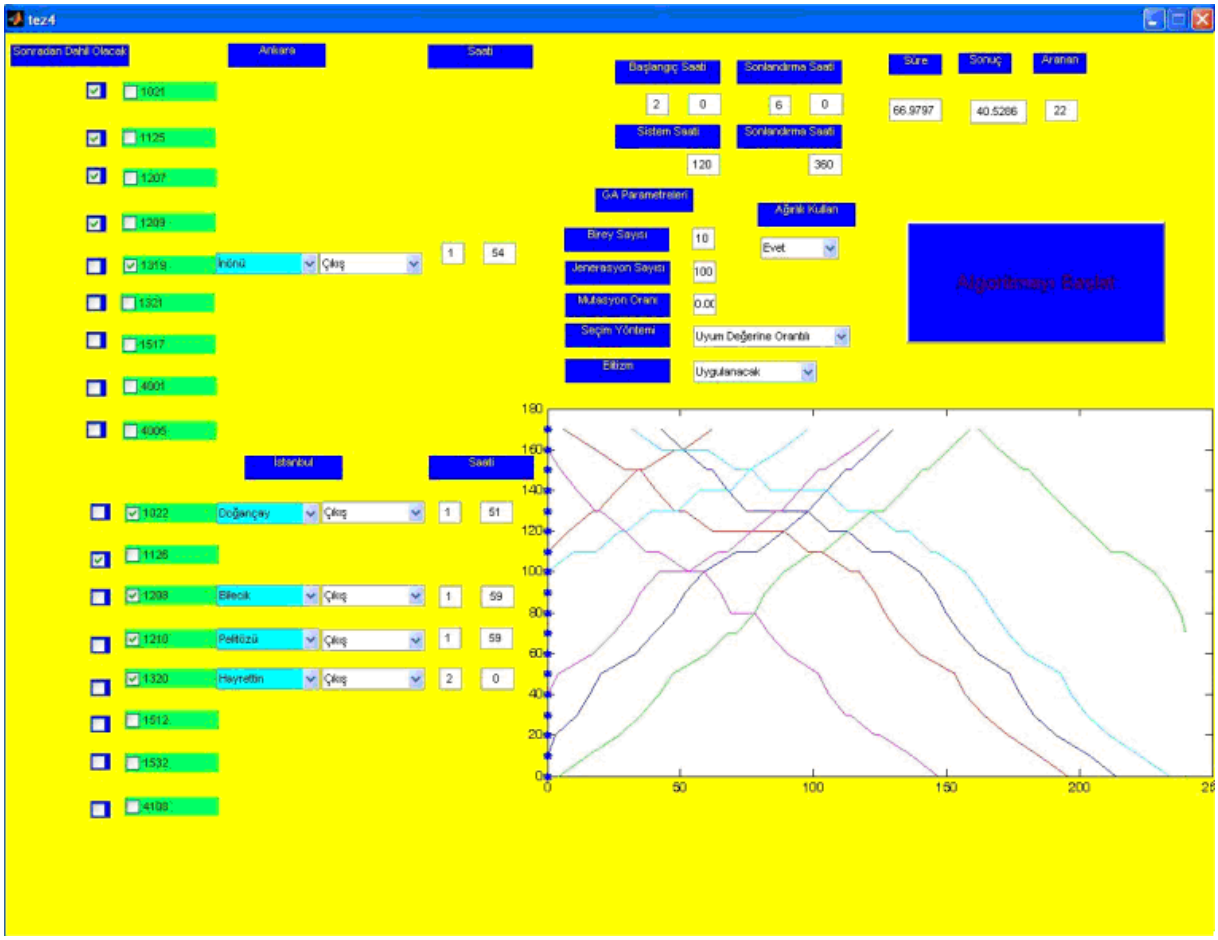
Şekil 5.21 Problem 20'nin genetik algoritma çözümü



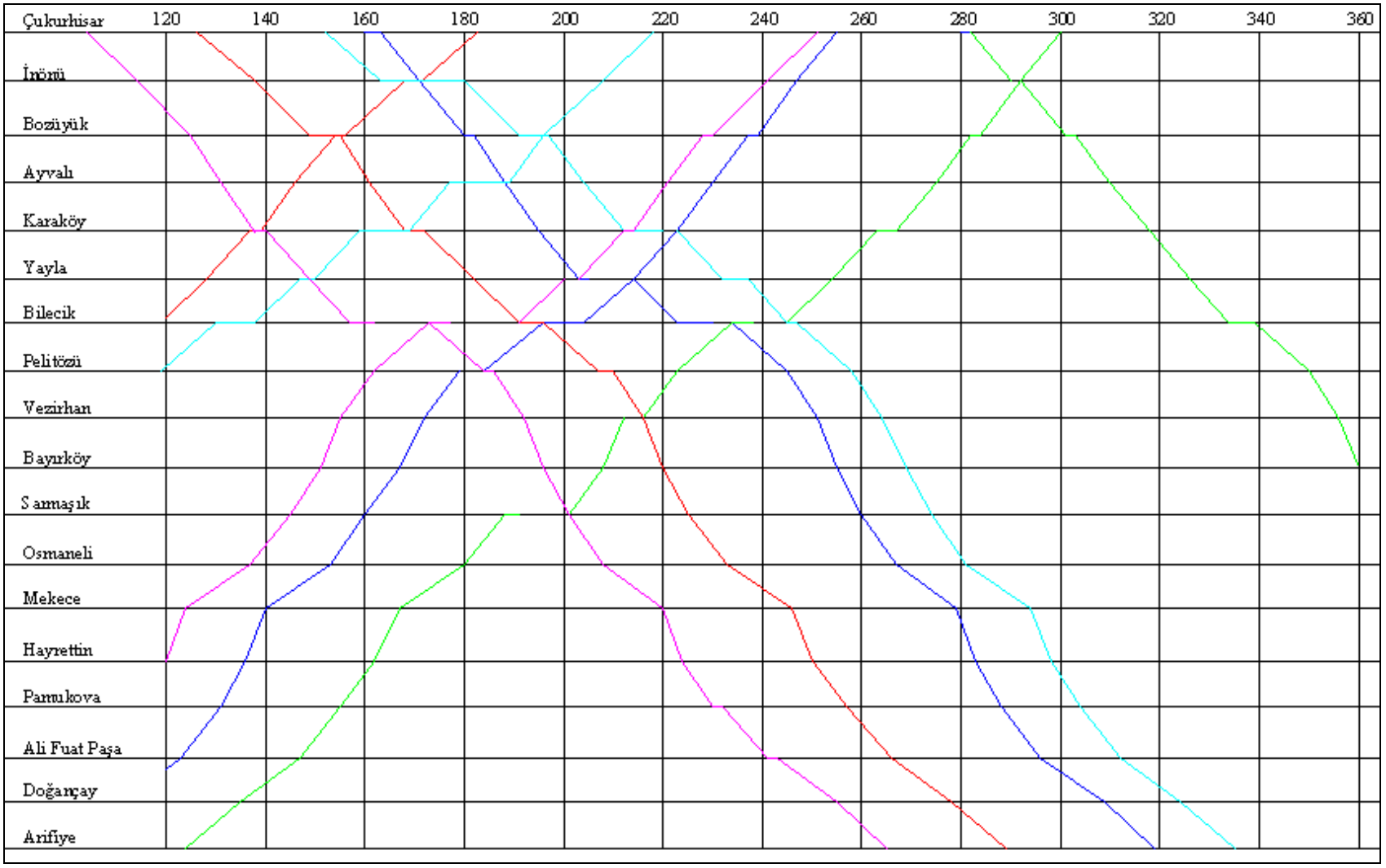
Şekil 5.22 Problem 20'nin yapay sinir ağı çözümü

5.8.3 Örnek Problem 36 (5x5 Tren)

Bu problemde, her bir yönde hareket eden 5'er adet tren bulunmaktadır. Bazı trenler, 02:00 itibarıyla sistemde hareketlerini sürdürürken, diğerleri sisteme ilk istasyondan hareket saatleri geldiğinde dahil olmaktadırlar. GA'nın 10 kere çalışması sonucu elde edilen en iyi çözüm Şekil 5.23'de görülmektedir. Bu çatışma çözümü sonucunda, ağırlıklı gecikme toplamları 40,53 dakika olmuştur. Algoritmanın her bir çalışması ortalama 166,78 saniye sürmektedir. YSA tarafından yapılan çözümde ise elde edilen ağırlıklı gecikme toplamı 52,3 dakikadır. Bu da GA çözümünden %24,85 daha kötü bir değerdir. YSA'nın el ile çizdirilen çözümü Şekil 5.24'de görülmektedir.



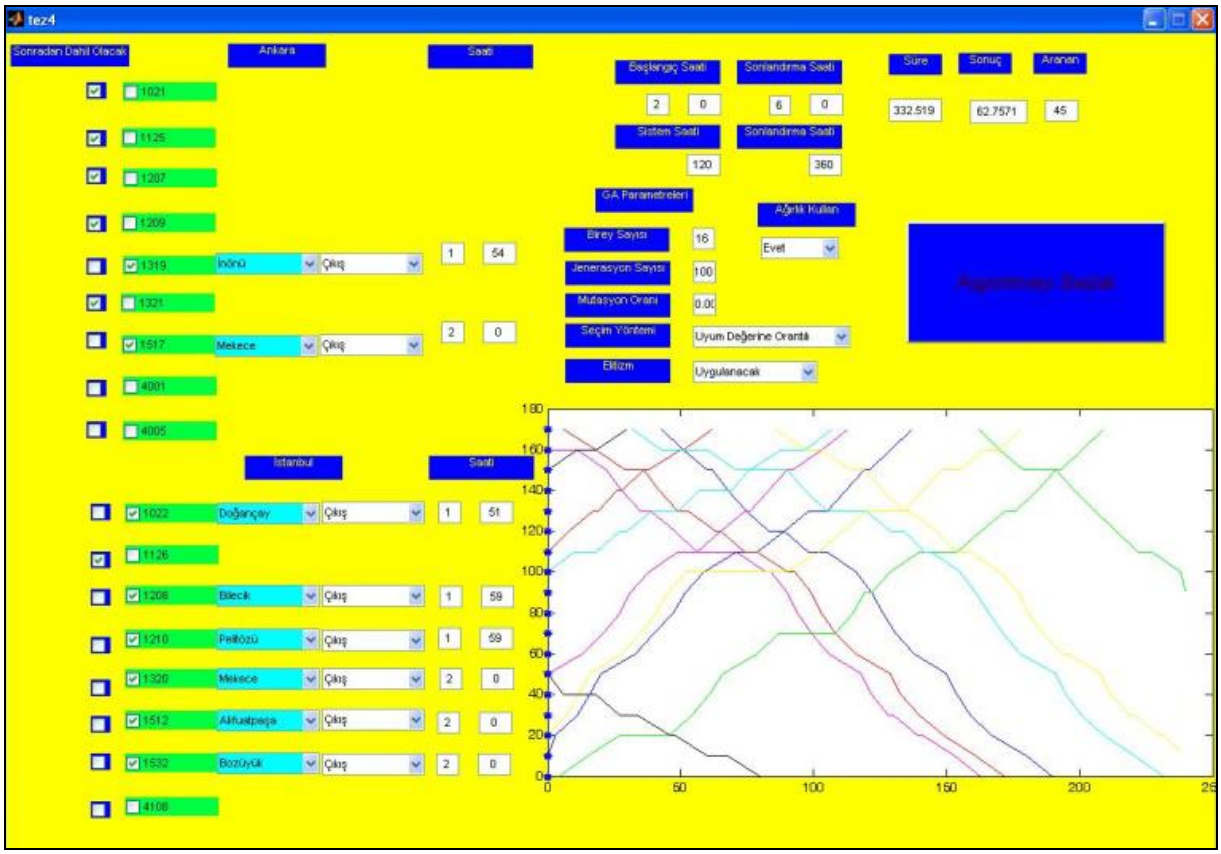
Şekil 5.23 Problem 36'nın genetik algoritma çözümü



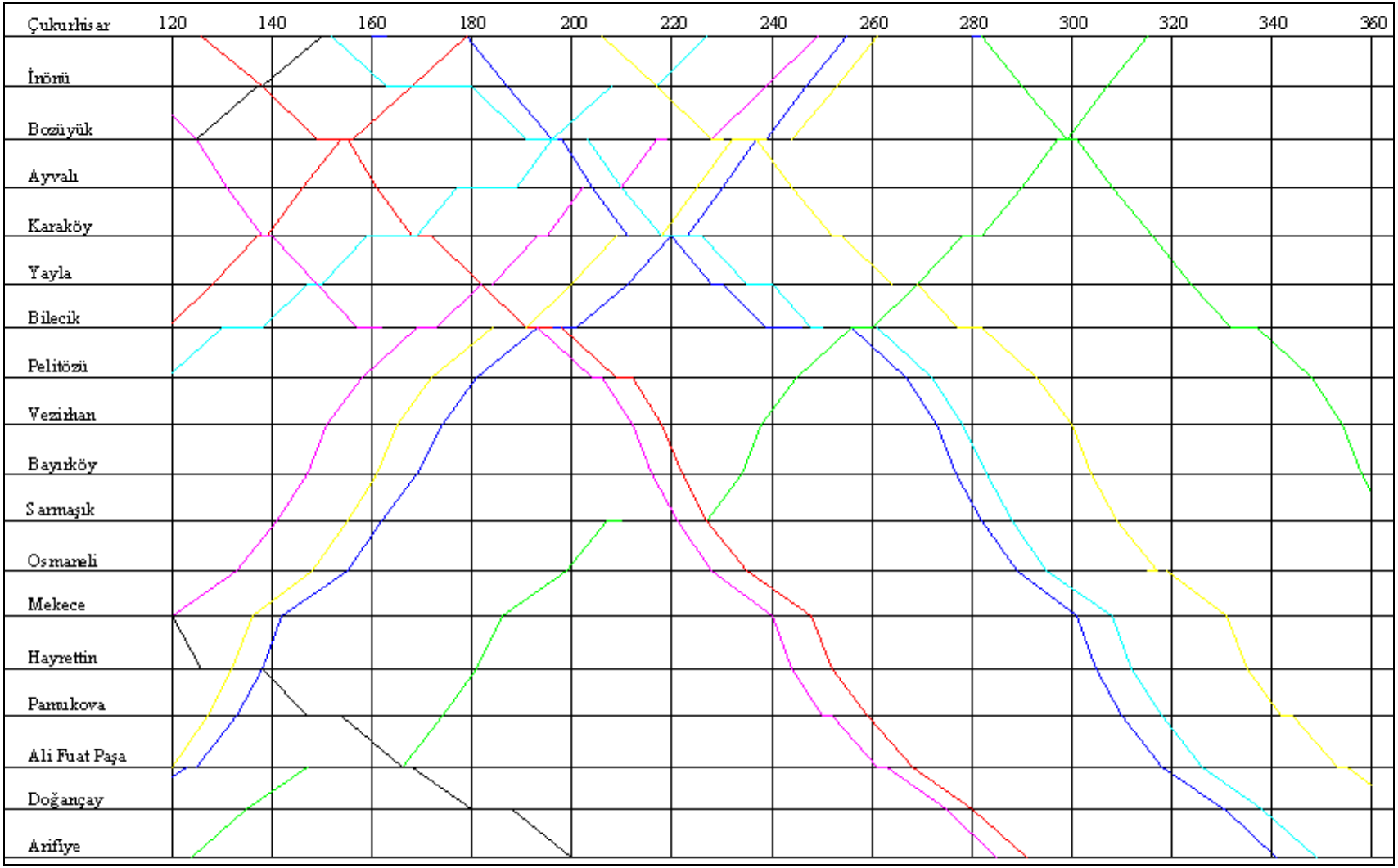
Şekil 5.24 Problem 36'nın yapay sinir ağı çözümü

5.8.4 Örnek Problem 46 (7x7 Tren)

Bu problemde, her bir yönde hareket eden 7'şer adet tren bulunmaktadır. Bazı trenler, 02:00 itibarıyla sistemde hareketlerini sürdürürken, diğerleri sisteme ilk istasyondan hareket saatleri geldiğinde dahil olmaktadırlar. GA'nın 10 kere çalışması sonucu elde edilen en iyi çözüm Şekil 5.25'de görülmektedir. Bu çatışma çözümü sonucunda, ağırlıklı gecikme toplamları 55,59 dakika olmuştur. Algoritmanın her bir çalışması ortalama 309,70 saniye sürmektedir. YSA tarafından yapılan çözümde ise elde edilen ağırlıklı gecikme toplamı 119,44 dakikadır. Bu da GA çözümünün neredeyse iki katına eşit bir ağırlıklı gecikme toplamıdır. YSA'nın el ile çizdirilen çözümü Şekil 5.28'de görülmektedir. Şekil 5.25 ve 5.26'dan görüleceği üzere, daha ilk çatışmada GA kararı ile YSA kararı farklılık göstermektedir. Bu da tamamen farklı bir çizelge elde edilmesine yol açmakta, ağırlıklı gecikme toplamlarını farklılaştırmaktadır.



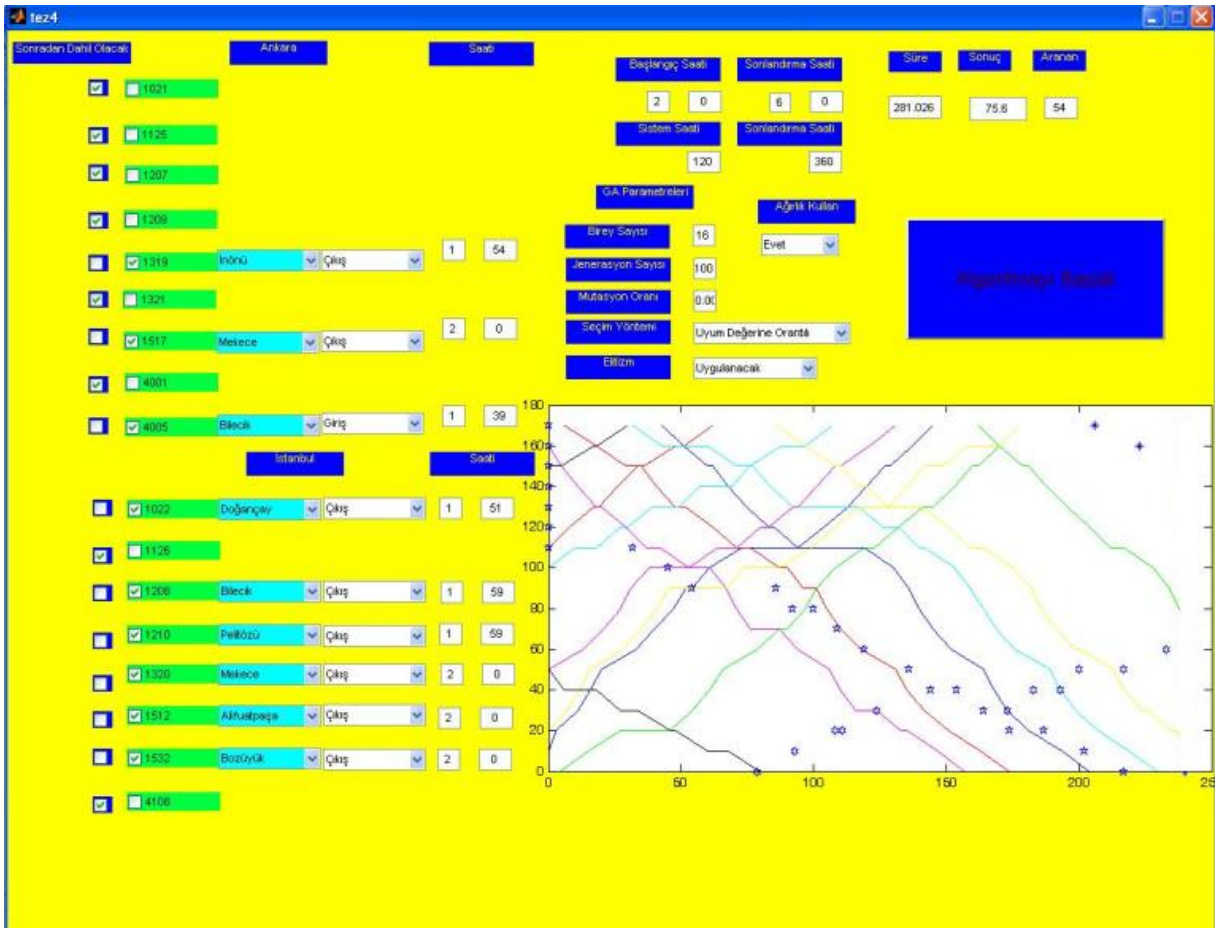
Şekil 5.25 Problem 46'nın genetik algoritma çözümü



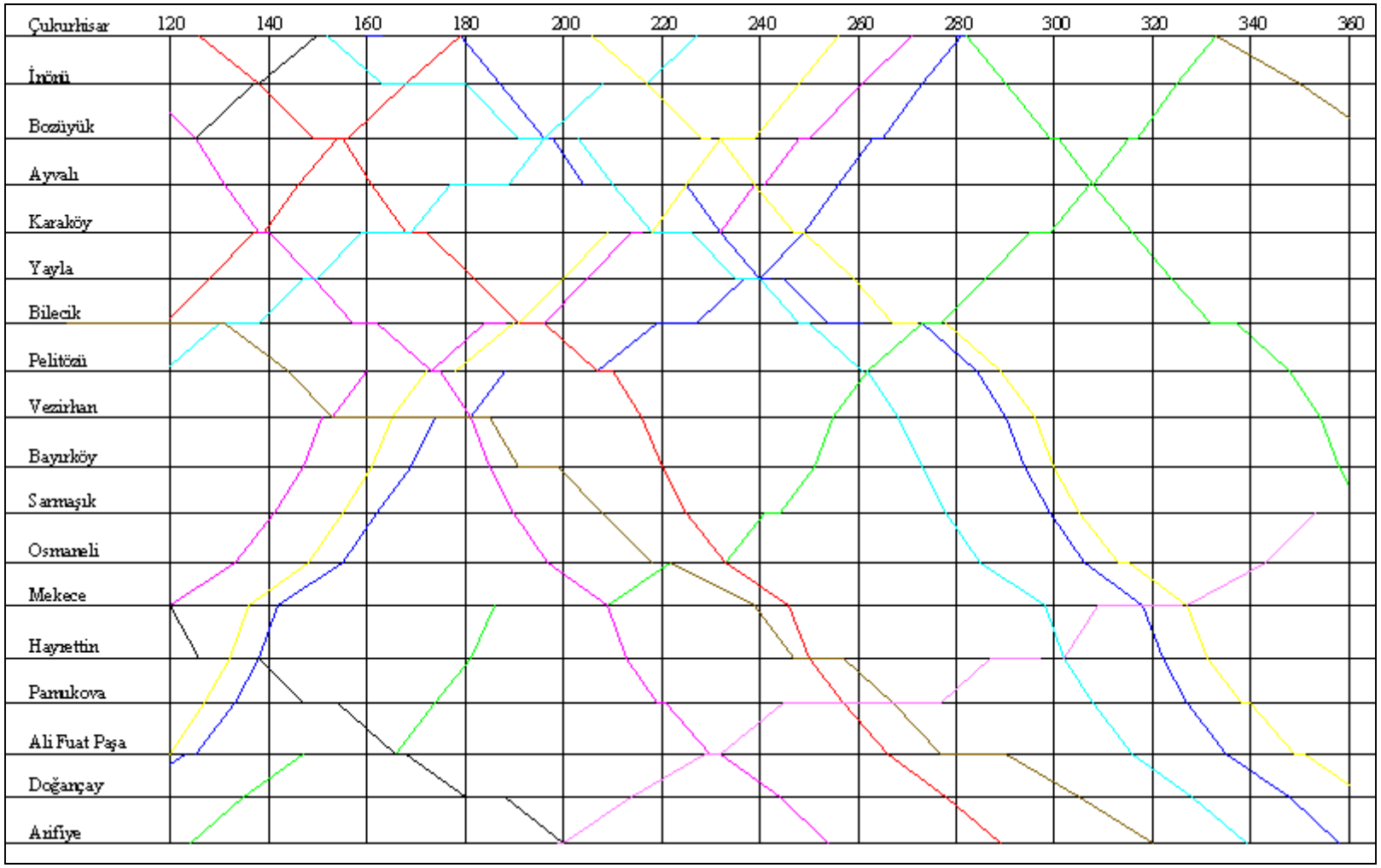
Şekil 5.26 Problem 46'nın yapay sinir ağı çözümü

5.8.5 Örnek Problem 51 (8x9 Tren)

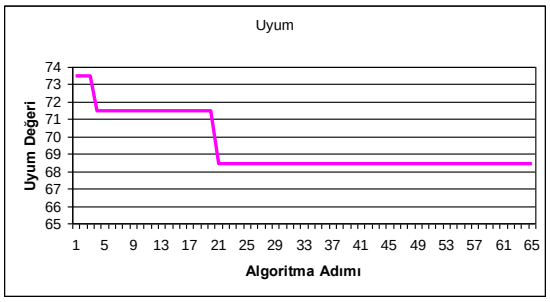
Bu problemde, Ankara yönünden İstanbul yönüne giden 9, İstanbul yönünden Ankara yönüne giden de 8 adet tren bulunmaktadır. Algoritmanın 10 kere çalışması sonucu elde edilen en iyi çözüm Şekil 5.27'de görülmektedir. Bu çatışma çözümü sonucunda, ağırlıklı gecikme toplamları 67,33 dakika olmuştur. Algoritmanın her bir çalışması ortalama 286,84 saniye sürmektedir. YSA tarafından yapılan çözümde ise elde edilen ağırlıklı gecikme toplamı 120,37 dakikadır. Bu da GA çözümünün neredeyse iki katına eşit bir ağırlıklı gecikme toplamıdır. YSA'nın el ile çizdirilen çözümü Şekil 5.28'de görülmektedir. Şekil 5.27 ve 5.28'de de görülebileceği üzere, problemin en iyi sonucunu bulmak artık neredeyse imkansız haldedir. Şekil 5.29'da ise, örnek problem 51'e ait 10 çalıştırmaya ait yakınsama grafikleri sunulmaktadır. Grafiklerin yatay ekseni, algoritma adımını, düşey ekseni ise her adımdaki en küçük uyum değerini göstermektedir.

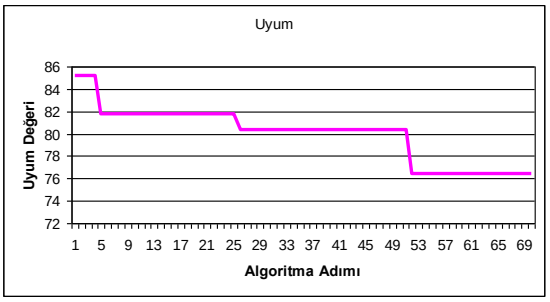


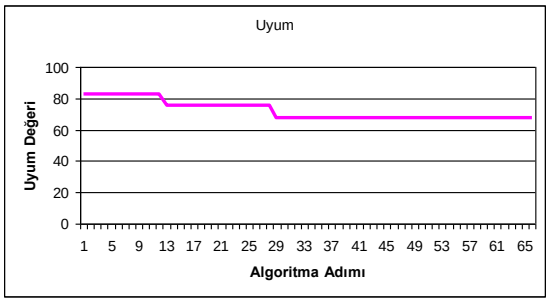
Şekil 5.27 Problem 51'in genetik algoritma çözümü

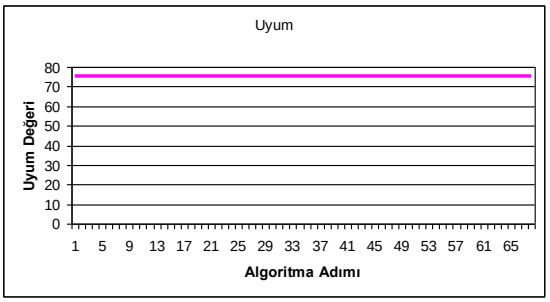


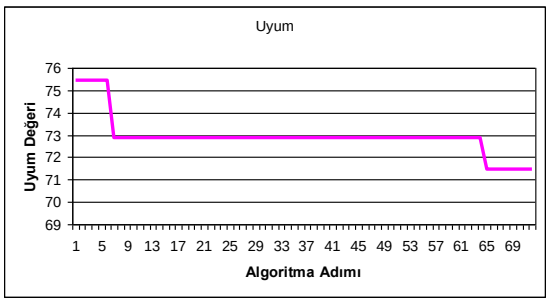
Şekil 5.28 Problem 51'in yapay sinir ağı çözümü

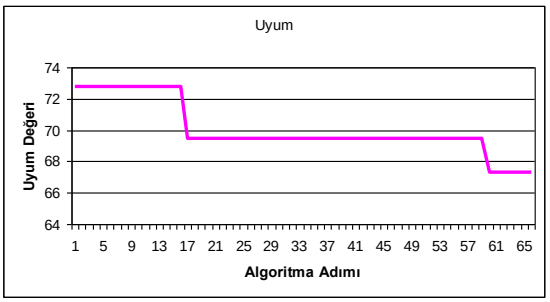


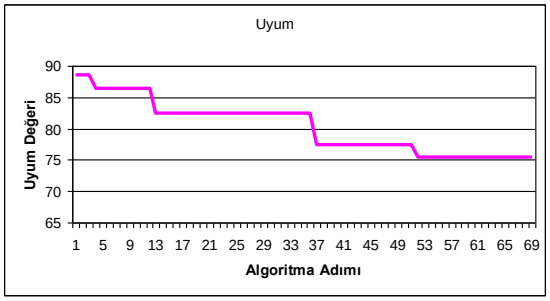


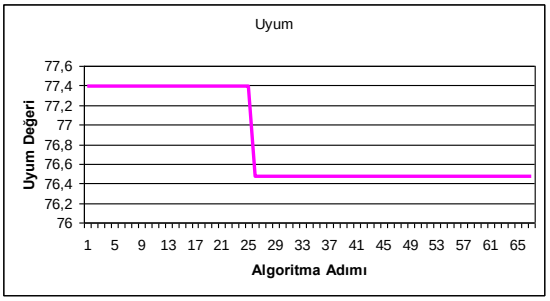


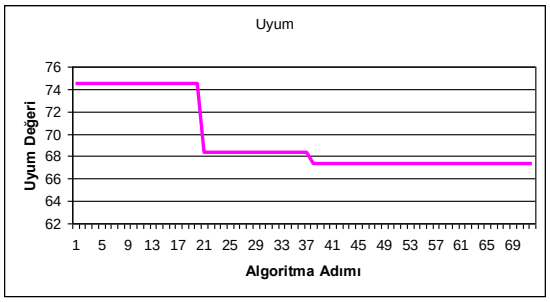


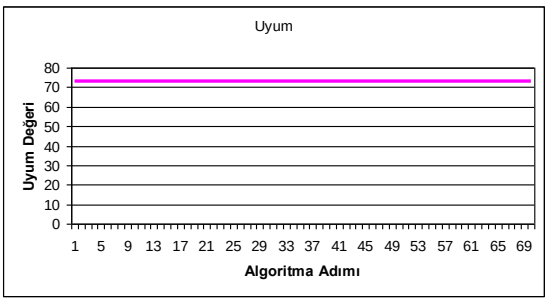












Şekil 5.29 Problem 51'e ait yakınsama grafikleri

6. SONUÇ VE ÖNERİLER

6.1 Sonuçlar

Demiryolu trafik kontrolü ve yeniden çizelgeleme oldukça büyük boyutlu ve çözümü zor bir problemdir. Trenlerin önceden hazırlanmış çizelgedeki hareketlerinden sapmaları sonucunda trenlerarası çatışmalar ortaya çıkabilmektedir. Trenlerarası çatışmalar deneyimli tren dispeçerleri tarafından çatışmaya giren trenlerin bir takım özellikleri göz önünde bulundurularak çözülmektedir. Çatışma çözümleri sonucu oluşan gecikmelerin hat/ağ kesimi boyunca yayılarak, diğer trenleri de etkilemesi sonucunda sistemde oluşan gecikmeler de git gide artar.

Ağ kesimi boyunca en küçük (ağırlıklı) gecikme toplamını sağlayacak çatışma çözümleri ancak az sayıda tren işletilen kesimlerde elde edilebilir. İşletilen tren sayısı arttıkça, en iyi (optimum) çözümü elde etmek için gerekli süre de üstel olarak artmaktadır. Bu nedenle dispeçerler, çatışma çözümlerini yaparken, ağ kesimi boyunca oluşacak (ağırlıklı) gecikme toplamlarını en küçüklemek yerine, çatışmaya giren trenleri ikili olarak değerlendirip, bu trenlerin bir takım özelliklerini göz önünde bulundurmaktadır. Çatışmaya giren iki trenin özellikleri göz önüne alındığında, gerçekleştirilen çözümün en uygun alternatif olduğu görünebilir. Ancak, bu çözüm ağ kesimi boyunca en küçük (ağırlıklı) gecikme toplamını sağlayacak çözümden oldukça farklı olabilir. Bu da oldukça iyi gözüken bir çözümün aslında esas amaçtan uzaklaştırabileceğini göstermektedir.

Dispeçerlere çatışma çözümlerinde yardımcı olmak, verecekleri kararların gecikme toplamlarına etkisini göstermek için bir karar destek sistemi olarak da kullanılabilecek bir genetik algoritma geliştirilmiştir. Geliştirilen genetik algoritma, 5 dakika gibi kısa bir süre içerisinde çatışma çözümlerini yapmakta ve elde ettiği en uygun çizelgeyi görsel olarak kullanıcıya sunmaktadır.

Geliştirilen örnek problemlere algoritmanın sunduğu çözümler incelendiğinde, küçük boyutlu problemlerde oldukça hızlı yakınsadığı ve kısa bir süre içerisinde en küçük gecikmeli çizelgeyi oluşturduğu görülmektedir. Problem boyutu büyüdükçe, algoritmanın yakınsaması için geçen süre de artmakta, buna karşın algoritmanın her bir çalışması sonucu elde edilen en iyi uyum değerleri (toplam gecikmeler) arasındaki farklar gitgide artmaktadır. Bunun nedeni, 300 saniye ile sınırlandırılmış bulunan algoritmanın çalışma süresinin yetersiz kalmasıdır. Şüphesiz gelişen teknoloji sayesinde daha güçlü bilgisayarlar ile 300 saniye içerisinde çok

daha fazla seçeneğin değerlendirilmesi ve dolayısıyla daha iyi sonuçlar elde edilmesi mümkün olabilecektir.

Genetik algoritmanın başarımının ölçülmesi için elde ettiği çözümler kesin çözüm ile karşılaştırıldığında, küçük boyutlu problemlerde aynı sonuçları verdikleri görülmektedir. Ancak kesin çözüm elde edilmesi için bilgisayarın uzun süre çalışması gerekmektedir. Örneğin 2x6 boyutlu bir problem için, bilgisayarın bir gün boyunca çalışması dahi yeterli gelmemiş, sonuç elde edilememiştir.

Genetik algoritmanın başarımının ölçülmesi için, dispeçer kararlarını %99 başarı ile taklit eden bir yapay sinir ağı geliştirilmiştir. Geliştirilen yapay sinir ağı çatışmaya giren trenleri ikili olarak karşılaştırmakta ve trenlere ait bazı verileri değerlendirerek çatışmaları çözmektedir. Yapay sinir ağı, küçük boyutlu problemler için genetik algoritma ile aynı sonuçları elde edebilmesine karşın, problem boyutu büyüdükçe, genetik algoritmaya nazaran %146'ya varan oranda daha fazla gecikme değerleri veren çizelgeler oluşturmaktadır. Üretilen 51 örnek problemin 14'ünde genetik algoritma ile yapay sinir ağının çözümü aynı olurken, geri kalan 37 problemin tamamında genetik algoritma daha iyi sonuçlar vermektedir.

Genetik algoritmanın başarımının arttırılması için, yapay sinir ağı çözümünü uygun bir başlangıç çözüm kabul eden bir melez model oluşturulmuştur. Ancak oluşturulan melez modelin başarıma ciddi bir katkısı olmadığı görülmüştür.

Genetik algoritmanın dikkat edilmesi gereken bir diğer özelliği de, arama uzayındaki çözümleri (potansiyel çözümler kümesi) araştırma oranıdır. Küçük boyutlu problemlerde arama uzayının %10 civarındaki bir kesimi incelenip, en iyi çözüm bulunmuşken, problem boyutu büyüdüğünde arama uzayının çok küçük bir kesimi incelenebilmektedir. Bu da algoritmanın en iyi sonuca yakınsamayı garantilemesi için daha fazla olasılığı değerlendirmesi gerektiğini göstermektedir. Bu da ancak daha güçlü bilgisayarlar ile yapılabilir. Algoritmanın büyük boyutlu problemler için sunduğu çözümlerin muhtemel bir çözüm olmaktan öte gidememesi açıktır. Bu da dispeçere çatışma çözümlerine ait ancak bir fikir verebilmektedir. Buna karşın, sistemin sonlandırılacağı saat geri alınarak, zaman penceresi kısıtlanabilir (örneğin algoritma saat 06:00 yerine 04:00'de sonlandırılabilir) ve algoritmanın daha fazla çözümü göz önüne alması sağlanabilir.

Geliştirilen algoritma, deneyimli dispeçerler tarafından kullanıldığında, kısa bir çalışma süresi içerisinde dispeçere olası en iyi çözümü ya da en iyiye yakın çözümleri sunabilmektedir. Buna karşın böyle bir programın dispeçerin yerini alması, diğer bir deyişle dispeçer yerine

kararlar alıp, çatışmaları çözümlemesi hedefler arasında değildir. Şüphesiz ki, bilgisayarların hesaplama kapasitesi insan beynine kıyasla çok daha yüksektir. Fakat bilgisayarlar sadece kendilerine programlananı yapabilmekte, insanlar gibi yeni koşullara adapte olup, inisiyatif almamaktadır. Bu nedenle geliştirilen algoritmanın mutlaka bir dispeçerin kontrolü altında kullanılması gerekmektedir.

6.2 Öneriler

Geliştirilen algoritma, kaynak araştırmasında elde edilen çalışmalara kıyasla, kısa süreler içerisinde sonuçlar vermektedir. Elde edilen sonuçlar, farklı yöntemlerle kıyaslandığında başarımının da oldukça iyi olduğu görülmektedir. Ancak, algoritmanın kodunda yapılacak eklemeler ya da düzenlemeler ile algoritmanın başarımı ve/veya daha uygun sonuçlar elde etmesi sağlanabilir. Bu amacı gerçekleştirecek bazı öneriler şöyledir:

- Algoritma iki ya da daha fazla hat içeren demiryolu kesimlerinde (yalnızca öne geçme çatışmaları söz konusudur) meydana gelebilecek çatışmalara çözüm getirecek şekilde genişletilebilir.
- İşletme tarafından sisteme ilave edilebilecek trenlerin meydana getireceği çatışmaların da göz önüne alınabilmesi için, programa yeni trenlerin dahil edilebileceği bir modül geliştirilebilir.
- Hat üzerinde meydana gelebilecek değişikliklere algoritmanın uyum sağlayabilmesi için hat bilgilerinin el ile tanımlanabileceği bir ortam geliştirilebilir.
- Tren uzunlukları ile istasyon ve yan hat kapasiteleri de (uzunlukları ve hat sayıları) sisteme bir ölçüt olarak girilebilir ve daha gerçekçi çözümler elde edilebilir.
- Yazılan program kodu üzerinde yapılacak iyileştirmeler ile, algoritmanın daha hızlı çalışması sağlanabilir.
- Kromozom boyutları yeniden düzenlenerek algoritmanın daha hızlı çalışması sağlanabilir.
- Algoritmaya bir durdurma seçeneği eklenerek, acil karar alınması gereken durumlarda elde edilen en iyi sonucun görüntülenmesi sağlanabilir.

KAYNAKLAR

Adenso-Diaz, B., Oliva Gonzales, M. ve Gonzales-Torre, P., (1999), "On-line Timetable Re-scheduling in Regional Train Services", Transportation Research Part B, 33, 387-398.

Adenso-Diaz, B., Tuya, J., Suarez-Cabal, M.J. ve Goitia-Fuertez, M., (2003), "DSS for Rescheduling of Railway Services Under Unplanned Events", Decision Making Support Systems - Achievements and Challenges for the New Decade, 72-86.

Amit, I. ve Goldfarb, D., (1971), "The Timetable Problem for Railways", Developments in Operations Research, Vol. 2, Gordon and Breach, New York, 379-387.

Andersson, A.W., Sandblad, B. ve Nilsson, A., (1998), "Improving Interface Usability For Train Dispatchers in Future Traffic Control Systems", Proceedings of COMPRAIL 98, Lisbon, September 2-4, 1998.

Andersson, A.W., Frej, I., Gideon, A., Hellström, P. ve Sandblad, B., (1997), "A System Analysis Approach to Modelling Train Traffic Control" Proceedings of WCRR 1997, vol C, Florence, 1997, 673–679.

Araya, S., Abe, K. ve Fukumori, K, (1983), "An Optimal Scheduling for Online Train Traffic Control in Disturbed Situations", Proceeding of the 22nd IEEE Conference on Decision and Control, 489-494.

Araya, S. ve Fukumori, K., (1985), "ESTRAC-II: An Expert system for Train Traffic Control in Disturbed Situations", Advances in Artificial Intelligence, T. O'Shea (Ed.), Elsevier Science Publishers B.V. (North-Holland), 23-32.

Assad, A.A., (1980), "Models for Rail Transportation", Transportation Research, Vol. 14A, 205-220.

Belshaw, P.N., (1976), "Railroad Transportation Planning Models and Their Success", Control in Transportation Systems. Proc. Of the IFAC/IFIP/IFORS Third International Symposium August 9-13, 1976.

Belshaw, P.N. ve Hallonquist, E.M., (1978), "Scheduling of Trains and Engineering Work Blocks Through Modelling", Presented at American Accosiation of Railroads Data Systems Division, San Francisco, October, 1978.

Benyahia, I. ve Potvin, J.Y., (1998), "Decision Support for Vehicle Dispatching Using Genetic Programming", IEEE Transactions on Systems, Man, and Cybernetics- Part A: Systems and Humans, Vol. 28, No. 3, May 1998, 306-314.

Bronzini, M.S. ve Clarke, D.B., (1985), "Estimating Rail Line Capacity and Delay by Computer Simulation", Tribune des Transports, Vol. 2, No. 1, 1985, 5-11

Bussieck, M.R., Kreuzer, P. ve Zimmermann, U.T., (1996), "Optimal Lines for Railway Systems", European Journal of Operational Research, Vol. 96, 1996, 54-63.

Cai, X. ve Goh, J., (1994), "A Fast Heuristic for the train Scheduling Problem", Computers and Operations Research, Vol.21, No.5, 1994, 499-510.

Carey, M., (1994a), "Extending Train Pathing Model from One-Way to Two-Way Track", Transportation Research B Vol. 28B No.5, 1994, 395-400.

Carey, M., (1994b), "A Model and Strategy for Train Pathing with Choice of Lines, Platforms, and Routes", Transportation Research B Vol. 28B No.5, 1994, 333-353.

Carey, M., (1999), "Ex Ante Heuristic Measure of Schedule Reliability", Transportation Research B Vol. 28B No.5, 1994, 333-353.

Carey, M. ve Carville, (2003), "Scheduling and Platforming Trains at Busy Complex Stations", Transportation Research Part A, Vol. 37, 2003, 195-224.

Carey, M. ve Kwiecinski, A., (1994), "Stochastic Approximation to the Effects of Headways on Knock-on Delays of Trains", Transportation Research B Vol. 28B No.4, 1994, 251-267.

Chang, S.C. ve Thia, B.S., (1996), "Online Rescheduling of Mass Rapid Transit Systems: Fuzzy Expert System Approach", IEE Proceedings-Electric Power Applications, Vol. 143, No. 4 July 1996, 307-316.

Chang, S.C. ve Chung, Y.C., (2005), "From Timetabling to Train Regulation – a New Train Operation Model", Information and Software Technology, 47, 575-585.

Chen, A., Yang, H., Lo, H.K. ve Tang, W.H., (2003), "A Capacity Related Reliability for Transportation Networks", Journal of Advanced Transportation, Vol. 33, No.2, 2003, 183-200.

Cheryavskii, A.L., (1971a), "Heuristic Program for Railway Timetable Composition I", Outomatike i Telemekhanica 1, 90-96.

Cheryavskii, A.L., (1971b), "Heuristic Program for Railway Timetable Composition II", Outomatike i Telemekhanica 1, 69-75.

Cheryavskii, A.L., (1972), "A Program for Timetable Compilation by a Look-Ahead Method", Artificial Intelligence, 3, 61-76.

Chiang, T.W., Hau, H.Y., Chiang, H.M., Ko, S.Y. ve Hsieh, C.H., (1998), "Knowledge-Based System for Railway Scheduling", Data&Knowledge Engineering, Vol. 27, 1998, 289-312.

Chiu, C.K., Chou, C.M., Lee, J.H.M., Leung, H.M. ve Leung, Y.W., (1996), "A Constraint-Based Interactive Train Rescheduling Tool", Lecture Notes in Computer Science, Principles and Practice of Constraint Programming — CP96.

Cordeau, J.F., Toth, P. ve Vigo, D., (1998), "A Survey of Optimization Models for Routing and Scheduling", Transportation Science, Vol. 32, No. 4, November 1998, 380-404.

Crainic, T.G. ve Querin, L., (1987), "Approximation of Over-the-Line Delay for Single Line - Single Speed Freight Rail Transportation", Universite de Montreal Centre de Recherche sur les Transports - Publication #519.

D'Ariano, D., Pranzo, M. ve Hansen, I.A., (2002), "Conflict Resolution and Train Speed Co-ordination for Solving Real-time Timetable Perturbations", Journal of Latex Class Files, Vol. 1, No. 11, November 2002, 208-222.

Dalal, M.A. ve Jensen, L.P., (2001), "Simulation Modelling at Union Pacific Railroad", Proceedings of the 2001 Winter Simulation Conference., Arlington, VA, USA, 1048-1055.

Dessouky, M.M., Lu, Q., Zhao, J. ve Leachman, R.C., (2006), "An Exact Solution Procedure for Determining the Optimal Dispatching Times for Complex Rail Networks" IIE Transactions 38, 2006 141-152.

Dorfman, M.J. ve Medanic, J., (2004), "Scheduling Trains on a Railway Network Using a Discrete Event Model of Railway Traffic", Transportation Research Part B, Vol. 38, 2004, 81-98.

Dube, I., Belshaw, P.N., (1977), "The Route Scheduler: A Macro Planning Model for the Analysis of Train Interference and Scheduling", Canadian Operational Research Society's Annual conference May 17, 1977.

Dündar, S., (2003), "Trenlerarası Çatışmaların Çözümünde Dispeçer Kararlarının Yapay Sinir Ağı Modeli", Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, Yıldız Teknik Üniversitesi.

Dündar, S., (2003), "Analysis of Train Dispatching Process with Artificial Neural Networks", Euro/ INFORMS Joint International Meeting, July 06-10 2003, Istanbul 188-196.

Elbrond, J., (1978), "A Method for the Calculation of the Capacity of a Single Track Railroad System", Proceedings of Heavy Haul Railways Conference, Perth, Western Australia 18-22 September 1978.

Fay, A., (2000), "A Fuzzy Knowledge-Based System for Railway Traffic Control", Engineering Applications of Artificial Intelligence 13, 2000, 719-729.

Feng, C.W., Cheng, T.M. ve Wu, H.T., (2004), "Optimizing the Schedule of Dispatching RMC Trucks Through Genetic Algorithms", Automation in Construction, Vol 13, 2004, 327-430.

Ferreira, L., (1997), "Planning Australian Freight Rail Operations: An Overview", Transportation Research A, Vol. 31, No. 4, 1997, 335-348.

Ferreira, L. ve Higgins, A., (1996), "Modelling Reliability of Train Arrival Times", Journal of Transportation Engineering, Vol. 122, No. 6, 1996, 414–420.

Frank, O., (1996), "Two-Way Traffic on a Single Line of Railway", Operations Research, Vol.14, 1996, 801-811.

Fransoo, J.C. ve Bertrand, J.W.M., "An Aggregate Capacity Estimation Model for the Evaluation of Railroad Passing Constructions", Transportation Research Part A, Vol. 34, 2000, 35-49.

Ghoseiri, K., Szidarovszky, F. ve Asgharpour, M.J., (2004), "A Multi-Objective Train Scheduling Model and Solution", Transportation Research Part B, 38, 927-952.

Goldberg, D.E., (1989), Genetic Algorithms in Search, Optimization & Machine Learning, Addison Wesley Longman, ABD.

Greenberg, B.S., Leachman, R.C. ve Wolff, R.W., (1988), "Predicting Dispatching Delays on a Low Speed, Single Track Railroad", Transportation Science Vol. 22, No. 1, February 1988, 31-38.

Hallowell, S.F. ve Harker, P.T., (1996), "Predicting On-Time Haul Performance in Scheduled Railroad Operations", Transportation Science, Vol. 30, No. 4, November 1996, 364-378.

Hallowell, S.F. ve Harker, P.T., (1996), "Predicting On-Time Performance in Scheduled Railroad Operations: Methodology and Application to Train Scheduling", Transportation Research A, Vol. 32 No.4, 1996, 279-295.

Hansen, I.A., (2001), "Improving Railway Punctuality by Automatic Piloting", 2001 IEEE Intelligent Transportation Systems Conference Proceedings - Oakland (CA) USA - August 25-29 2001, 792-797.

Harker, P.T. ve Hong, S., (1990), "Two Moments Estimation of the Delay on Single-Track Rail Lines with Scheduled Traffic", Journal of the Transportation Research Forum, VolXXXI, No. 1, 38-49.

Hasegawa, Y., Konya, H., ve Shinohara, S., (1981), "A Macro-model on Propagation-disappearance Process of Train Delays", Quarterly Reports Vol. 22 No. 2, 78-82.

He, S., Song, R. ve Chaudhry, S.S., (2000), "Fuzzy Dispatching Model and Genetic Algorithms for Railyards Operations", European Journal of Operations Research, 124, 307-331.

Higgins A., Kozan, E. ve Ferreira, L., (1996), "Optimal Scheduling of Trains on a Single Line Track", Transportation Research B, Vol. 30, No.2, 1996, 147-161.

Higgins A., Kozan, E. ve Ferreira, L., (1997), "Heuristic Techniques for Single Line Train Scheduling", Journal of Heuristics, No. 3, 1997, 43-62.

Ho, T.K., Norton, J.P. ve Goodman, C.J., (1997), "Optimal Traffic Control at Railway Junctions", IEE Proceedings, Vol. 144, Part. B. March 1997, 140-148.

Hooghiemstra, J.S. ve Teunisse, M.J.G., (1998), "The Use of Simulation in the Planning of the Dutch Railway Services", Simulation Conference Proceedings, 1998. Winter, Vol. 2, Dec. 13-16, 1998.

Howlett, P., (1996), "Optimal Strategies for the Control of a Train", Automatica, Vol. 32, No. 4, 1996, 519-532.

Huisman, T. Ve Boucherie, R.J., (2001), "Running Times on Railway Sections with Heterogeneous Train traffic", Transportation Research, Part B, Vol.35, 2001, 271-292.

Isaai, M.T. ve Singh, M.G., (2000), "An Object-Oriented, Constraint-Based Heuristic for a Class of Passender-Train Scheduling Problems", IEEE Transactions on on Systems, Man, and Cybernetics- Part C: Applications and Reviews, Vol. 30, No. 1, February 2000, 12-21.

Jelaska, M., (1976), "Computer Elaboration of Time-Table for Single Railway Line", Springer Berlin / Heidelberg, Volume 40/1976, 657-675.

Jiaxin, C. ve Howlett, P., (1992), "Critical Velocities for the minimisation of Fuel Consumption in the Control of Trains", Automatic 28, Vol. 1, 1992, 165–169.

Jovanovic, D. ve Harker, P.T., (1989), "SCAN: A Decision Support System for Railroad Scheduling", IEEE/ASME Joint Railroad Conference, Philadelphia, PA, 97-105.

Jovanovic, D. ve Harker, P.T., (1990), "A Decision Support System For Train Dispatching: An Optimization Methodology", Journal of the Transportation Research Forum, VolXXXI, No. 1, 25-37.

Jovanovic, D. ve Harker, P.T., (1991), "Tactical Scheduling of Rail Operations: The SCAN I System", Transportation Science Vol. 25, No. 1, February 1991, 46-64.

Karaboğa, D., (2006), "Yapay Zeka Optimizasyon Algoritmaları", Atlas Yayın Dağıtım, İstanbul.

Kraay, D., Harker, P.T. ve Chen, B., (1991), "Optimal Pacing of Trains in Freight Railroads: Model Formulation and Solution", Operations Research, Vol. 39, No. 1, January-February, 82-99.

Kraay, D. ve Harker, P.T., (1995), "Real-Time Scheduling of Freight Railroads", Transportation Research B, Vol. 29B, No.3, 1995, 213-229.

Kimura, Y. ve Koga, S., (1985), "A Train Running Curve of the Minimum Energy", Quarterly Reports Vol. 26 No. 1, 30-32.

Komaya, K. ve Fukuda, T., (1989), "ESTRAC-III: An Expert system for Train Traffic Control in Disturbed Situations", IFAC Control, Computers, Communications in Transportation, Paris, France, 147-153.

Kraft, E.R., (1987), "A Branch and Bound Procedure for Optimal Train Dispatching", Journal of the Transportation Research Forum, Vol. XXVIII, No. 1, 262-276.

Leclerc, F. ve Potvin, J.Y., (1997), "Genetic Algorithms for Vehicle Dispatching", International Transportation Operations Research, Vol. 4, No. 5/6, 1997, 391-400.

Leilich, R.H., (1998), "Application of Simulation Models in Capacity Constrained Rail Corridors", Simulation Conference Proceedings, 1998. Winter, Vol. 2, Dec. 13-16, 1998, 1125-1133.

Lenoir, D., Janssen, W., Neerincx, M. ve Schreibers, K., (2006), "Human-factors Engineering for Smart Transport: Decision Support for Car Drivers and Train Traffic Controllers", Applied Ergonomics 37 479-490.

Lu, Q., Dessouky, M. ve Leachman, R.C., (2004), "Modelling Train Movements Throught Complex Rail Networks", ACM Transactions and Modelling on Computer Simulation, Vol. 14, No. 1, January 2004, 48-75.

Lusicic, B., (1983), "Decomposition of Optimal Control in Energy Minimisation in Railway Traffic", Proceedings of the 11th IFIP Conference Copenhagen, Denmark, July 25-29.

Mattson, L.G., (2004), "Train Service Reliability a Survey of Methods for Deriving Relationships for Train Delays", Research Report, Unit for Transport and Location Analysis, Royal Institute of Technology, Stockholm, Sweden.

Mazzarello, M. ve Ottaviani, E., (2005), "A Traffic Management System for Real-Time Traffic Optimisation in Railways" 1st International Seminar on Railway Operations Modelling and Analysis, 8-10 June, 2005, Delft, Netherlands.

Mess, A.I., (1991), "Railway Scheduling by Network Optimization", Mathematical and Computer Modelling Vol.15, No. 1, 1991, pp. 33-42

Middelkoop, D. ve Bouwman, M., (2001), "SIMONE: Large Scale Train Network Simulations", Proceedings of the 2001 Winter Simulation Conference., Arlington, VA, USA, 1042-1047.

Mills, R.G.J. ve Perkins, S.E., (1989), "Nonlinear Programming Applied to the Dynamic Rescheduling of Trains",Internal Report, Department of Mathematics, South Australia Institute of Technology, Adelaide, Australia.

Mills, R.G.J., Perkins, S.E. ve Pudney, P.J., (1991), " Dynamic Rescheduling of Long Haul Trains for Improved Timekeeping and Energy", Asia-Pacific Journal Operational Research 8, 1991, 146–165.

Nachtigall, K. ve Voget, S., (1996), "A Genetic Algorithm Approach to Periodic Railway Synchronization", Computers Operations Research, Vol. 23, No. 5, 1996, 453-463.

Ohtsu, A., (1979), "Study on Shinkansen Train Traffic Control -To Maintain Steady Train Operation-", Quarterly Reports Vol. 20 No. 2, 49-56.

Özekici, S. ve Şengör, S., (1994), "On a Rail Transportation Model with Scheduled Services", Transportatıon Scıence, Vol. 28, No. 3, August 1994, 246-255.

Pachl, I.J. ve White, T., (2003), "Efficiency Through Integrated Planning and Operation", in: Implementation of Heavy Haul Technology for Network Efficiency, International Heavy Haul Association, Proceedings, Virginia Beach 2003, 6.69-6.77.

Peat, Marwick, Mitchell & Co., (1975), "Train Dispatching Simulation Model: Capabilities and Description" USA. Report No. DOT-FR-4-5014-1, Federal Railroad Administration, Department of Transportation, Washington, DC.

Petersen, E.R., (1974), "Over-the-road Transit Time for a Single Track Railway", Transportation Science, Vol. 8, No. 1, 65-74.

Petersen, E.R., (1975), "Interference Delays on a Partially Double-Tracked Railway with Intermediate Signalling", Trans. Res. Forum, Toronto, 1975.

Petersen, E.R. ve Fullerton H.V., (1975), "A Network Model of the Mainline Operation of a Railway", 1975 Annual Conference of CORS, University of Waterloo, Waterloo, Ontario, May 25-28 1975.

Petersen, E.R., (1977), "Capacity of a Single Track Rail Line", Working Paper No. 77-38, Queen's University, Kingston.

Petersen, E.R. ve Taylor, A.J., (1982), "A Structured Model for Rail Line Simulation and Optimization", Transportation Science, Vol. 16, No. 2, 192-206.

Petersen, E.R. ve Taylor, A.J., (1987), "Design of a Single-Track Rail Line For High-Speed Trains", Transportation Research A, Vol 21A, No.1, 47-57.

Petersen, E.R. ve Taylor, A.J., (1988), "An Optimal Scheduling System for the Welland Canal", Transportation Science, Vol 22, No.3, August, 1988, 173-185.

Petersen, E.R. ve Merchant, (1992), "Scheduling of Trains in a Linear Railway System", INFOR, Vol. 19, no.3, 1992, 246-248.

Ping L., Limin, J., ve Xiong, K., (2001), "Study on GA Based Train Dispatching", WCRR Köln 26-28 November 2001. 949-953.

Pyrgidis, C., (1995), "Analysis of Railway Network Capacity", Rail Engineering International Edition, Number 4, 1995, 12-16.

Purdon, R.L, Elbond, J. ve Clark, J.M., (1978), "A Comparison of Theoretical and Actual Traffic Schedules on the Mt. Newman Railroad", Proceedings of Heavy Haul Railways Conference, Perth, Western Australia 18-22 September 1978.

Quinchon, C., (1984), "Timetable Planning", French Railway Review Vol.2 No. 2, 607-616.

Reeves C.R., (1996), "Modern Heuristic Techniques, Published in Modern Heuristic Search Methods", Edited by. Rayward-Smith, V.J., Osman, I.H., Reeves, C.R. ve Smith, C.D., John Wiley & Sons Ltd. West Sussex.

Reeves C.R., (2003), "Genetic Algorithms", Published in Handbook of Metaheuristics, Edited by. Glover, F. ve Kochenberger, G.A., Kluwer's International Series, Massachusetts.

Rodriguez, J., (2005), "A Constraint Programming Model For Real-Time Trains Scheduling at Junctions" 1st International Seminar on Railway Operations Modelling and Analysis, 8-10 June, 2005, Delft, Netherlands.

Roth, E.M., Malsch, N., Multer, J., Coplen, M. ve Rhoads, N.K., (1998), "Analyzing Railroad Dispatchers' Strategies: A Cognitive Task Analysis of A Distributed Team Planning Task",. 1998 IEEE International Conference on Systems, Man, and Cybernetics, 1998, vol. 3, Oct. 11-14, 1998, 2539-2544.

Rudd, D.A. ve Storry, A.J., (1976), "Single Track Railway Simulation New Models and Old", Rail International, June 1976, 335-342.

Salim, V. ve Cai, X., (1995), "Scheduling Cargo Trains Using Genetic Algorithms", IEEE International Conference on Evolutionary Computation, 1995, Perth, WA, Australia.

Salim, V. ve Cai, X., (1997), "A Genetic Algorithm for Railway Scheduling with Environmental Considerations", Environmental Modelling & Software, Vol. 12, No. 4, 1997, 301-309.

Shen, Y., Potvin, J.Y., Rousseau, J.M. ve Roy, S., (1995), "A Computer Assistant for Vehicle Dispatching with Learning Capabilities", Annals of Operations Research, Vol. 61, 1995, 189-211.

Steiner, T., (1978), "Computer Aided Dispatching", Bulletin 334, Union Switch & Signal Division, Westinghouse Air Brake Company.

Szpigel, B., (1973), "Optimal Train Scheduling on a Single Track Railway" OR'72, North-Holland, 343-352.

Şahin, G., Ahuja, R.K. ve Cunha, C.B., (2005), "New Approaches for the Train Dispatching Problem", Submitted to Transportation Research B, 2005.

Şahin, İ., (1996), "Trenlerarası Çatışma Yönetimine Dayalı Trafik Kontrolü İçin Bir Karar Destekleyici Sistem", Doktora Tezi, Fen Bilimleri Enstitüsü, Yıldız Teknik Üniversitesi.

Şahin, İ., (1999), "Railway Traffic Control and Train Scheduling Based on Inter-Train Conflict Management", Transportation Research Part B 33, 511-534.

Şen, Z., (2004), "Genetik Algoritmalar ve En İyileme Yöntemleri", Su Vakfı Yayınları, İstanbul.

Takeuchi, Y. ve Tomii, N., (2005), "Robustness Indices for Train Scheduling" 1st International Seminar on Railway Operations Modelling and Analysis, 8-10 June, 2005, Delft, Netherlands.

Thomas, L.A., (1974), "FRISCO's Train Meet Calculator", AAR/UIC/IRCA Fourth International Symposium on Railroad Cybernetics Washington, 21-26 April 1974.

Van Egmond, R.J., (1999), "Railway Capacity Assessment, an Algebraic Approach", TRAIL Studies in Transportation Science, S99/2, Delf University Press, 1999.

Vromans, M.J.C.M., Dekker, R. ve Kron, L.G., (2006), "Reliability and Heterogenity of Railway Services", European Journal of Operations Research 172, 2006, 647-665.

Vukadinovic, K., Teodorovic, D. ve Pavkovic, G., (1999), "An Application of Neurofuzzy Modelling: The Vehicle Assignment Problem", European Journal of Operations Research, Vol. 114, 1999, 474-488.

Wegele, S. ve Schneider, E., (2005), "Dispatching of Train Operaions Using Genetic Algorithms" 1st International Seminar on Railway Operations Modelling and Analysis, 8-10 June, 2005, Delft, Netherlands.

Welch, N. ve Gussow, J., (1986), "Expansion of Canadian National Railway's Line Capacity", INTERFACES 16: 1 January-February 1986, 51-64.

White, T., (2005), "Alternatives for Railroad Traffic Simulation Analysis", Transportation Research Record: Journal of the Transportation Research Board, Vol. 1916, 2005, 34-41.

Wong, O. ve Rosser, M., (1978), "Improving System Performance for a Single Line Railway with Passing Loops", New Zealand Operational Research Vol. 6, No. 2, July 1978, 137-155.

Yokota, H., (1979), "Performance Analyses of Passing Track and Planning Principles for Its Layout on Double-Track Lines", Quarterly Reports Vol. 20 No. 1, 9-14.

Yokota, H., (1980), "Performance Analyses of Passing Track and Planning Principles for Its Layout on Single-Track Lines", Quarterly Reports Vol. 21 No. 3, 105-114.

İNTERNET KAYNAKLARI

[1]http://tr.wikipedia.org/wiki/Genetik_Algoritmalar **(Son erişim tarihi: 16.12.2009)**

[2]http://www.mathworks.com/access/helpdesk_r13/help/toolbox/gads/ga_tut22.html **(Son erişim tarihi: 16.12.2009)**

ÖZGEÇMİŞ

Doğum tarihi	17.05.1979	
Doğum yeri	İstanbul	
Lise	1989-1996	Özel Kültür Lisesi
Lisans	1996-2000	Yıldız Teknik Üniversitesi İnşaat Fakültesi İnşaat Mühendisliği Bölümü
Yüksek Lisans	2001-2003	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İnşaat Müh. Anabilim Dalı, Ulaştırma Programı
Doktora	2003-2009	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İnşaat Müh. Anabilim Dalı, Ulaştırma Programı

Çalıştığı kurum(lar)

2003-2004	Ünar Yapı ve Malz. Tic. Ltd Şti.
2004-2009	YTÜ İnşaat Fakültesi Araştırma Görevlisi
2009-Devam Ediyor	Bahçeşehir Üniversitesi Ulaştırma Uygulama Araştırma Merkezi