ANKARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARI KULLANARAK TRAFİK AKIŞ PROBLEMİNİN ÇÖZÜMÜ

Adnan Şahin KARACA

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ANKARA
2020

TEZ ONAYI

Adnan Şahin KARACA tarafından hazırlanan **"Çok Amaçlı Optimizasyon Algoritmaları Kullanarak Trafik Akış Probleminin Çözümü"** adlı tez çalışması 31/01/2020 tarihinde aşağıdaki jüri tarafından oy birliği ile Ankara Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Danışman : Dr. Öğr. Üyesi Ökkeş Tolga ALTINÖZ
Ankara Üniversitesi / Elektrik Elektronik Mühendisliği Anabilim Dalı

Jüri Üyeleri:

Başkan: Prof. Dr. Hamit ERDEM
Başkent Üniversitesi / Elektrik Elektronik Mühendisliği Anabilim Dalı

Üye: Doç. Dr. İsa NAVRUZ
Ankara Üniversitesi / Elektrik Elektronik Mühendisliği Anabilim Dalı

Üye: Dr. Öğr. Üyesi Ökkeş Tolga ALTINÖZ
Ankara Üniversitesi / Elektrik Elektronik Mühendisliği Anabilim Dalı

Yukarıdaki sonucu onaylarım.

Prof. Dr. Özlem YILDIRIM
Enstitü Müdürü

ETİK

Ankara Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez içindeki bütün bilgilerin doğru ve tam olduğunu, bilgilerin üretilmesi aşamasında bilimsel etiğe uygun davrandığımı, yararlandığım bütün kaynakları atıf yaparak belirttiğimi beyan ederim.

14.01.2020

Adnan Şahin Karaca

ÖZET

Yüksek Lisans Tezi

ÇOK AMAÇLI OPTİMİZASYON ALGORİTMALARI KULLANARAK TRAFİK AKIŞ PROBLEMİNİN ÇÖZÜMÜ

Adnan Şahin KARACA

Ankara Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üy. Ö.Tolga Altınöz

Trafikteki araç sayısının günden güne artmasıyla birlikte trafik sıkışıklığı, kazalar ve doğaya salınan sera gazı emisyonları artmaktadır. Trafik sıkışıklığının nedenleri arasında kazalar ve hava koşulları gibi ön görülemeyen olaylar ve tatil, bayram gibi özel günler gösterilebilir. Bu durumlarda belirli bir zaman dilimi içindeki araç sayısı beklenenin üzerine çıkmaktadır. Bu değişken durumlara adapte olamaması sebebiyle sabit zamanlı trafik ışığı kullanan kavşaklar trafik sıkışıklığında artışa sebebiyet verirler. Bu sorunu çözmek için istatistiksel veriler kullanılarak günün belirli saatleri için ışık sürelerini değiştiren kapalı çevrim sistemler geliştirilmiştir. Ancak bu çözüm araç sayısı ve artan şehirleşme oranı ile değişen trafik dinamiklerine adapte olmakta yetersiz kalmıştır. Bu sebeple trafik simülatörü kullanılarak kavşakların ışık sürelerini belirlenebildiği bir karar alma mekanizması geliştirilmesi gerekmektedir. Bu tezde oluşan trafik sıkışıklığını araçların bekleme sürelerini ve oluşan sera gazı salınımını en aza indirmek amaçlanmıştır. Trafik simülatörü olarak Şehiriçi Hareketlilik Simülasyonu (SUMO) yazılımı kullanılmıştır. İlk olarak trafikteki hareketliliğin trafik sıkışıklığı ve sera gazı emisyonlarına etkisini açıklamak amacıyla tek kavşaklı model incelenecektir. Sonrasında gerçek hayat uygulaması için çok kavşaklı model incelenecek ve trafik ışık süreleri optimize edilecektir. Optimizasyon işlemi için tek amaçlı ve çok amaçlı algoritmalar kullanılacaktır. Tek amaçlı optimizasyon algoritmaları olarak genetik algoritma ve parçacık sürüsü optimizasyonu kullanılacaktır. Çok amaçlı optimizasyon algoritmaları olarak Baskılanmayanları Sıralayan Genetik Algoritma II (NSGA-II) ile Ayrıştırmalı Çok Amaçlı Evrimsel Algoritma (MOEA/D) seçilmiştir.

Mart 2020, 69 Sayfa

Anahtar Kelimeler: Çok Amaçlı Optimizasyon, Genetik Algoritma (GA), Parçacık Sürü Optimizasyonu (PSO), Baskılanmayanları Sıralayan Genetik Algoritma-II (NSGA-II), Ayrıştırmalı Çok Amaçlı Evrimsel Algoritma (MOEA/d), trafik akış problemi, Şehiriçi Hareketlilik Simülasyonu (Sumo)

ABSTRACT

Master Thesis

TRAFFIC FLOW PROBLEM SOLUTION USING MULTI-OBJECTIVE OPTIMIZATION ALGORITHMS

Adnan Şahin KARACA

Ankara University
Graduate School of Natural and Applied Sciences
Department of Electrical and Electronics Engineering

Supervisor: Asst. Prof. Dr. Ö. Tolga ALTINÖZ

Traffic jam, accidents and greenhouse gas emissions are increasing due to vehicle numbers going up day by day on the road. Unexpected incidents such as traffic accidents, weather condition and road vehicle number increases at a given time such as holidays and religious days can be shown for the causes of traffic jam. In these cases vehicle numbers increase exceeds expectations in a specific time frame. Intersections using fixed time signals causes increase on jamming due to not being able to adapt to this dynamic states. To solve this problem, closed cycle systems that change signal durations according to the time of the day based on statistical information have improved. However this solution is not suffice to adapt on fast changing traffic dynamics such as number of vehicles and increasing urbanization. For this reason developing a decision making mechanism to determine signal durations on intersecions using traffic simulator is required. In this thesis it is aimed to minimise jamming, vehicle waiting times and greenhouse emission. For traffic simulator Simulation of Urban Mobility (Sumo) software is used. Aimin to explain simulation environment dynamics firstly one intersection network will be reviewed. Then for the real life application adjecent junctions will be reviewed and signal times will be optimized. Single and Multi objective algorithm will be used for optimization. For single objective algorithms genetic algorithm and particle swarm optimization will be used. For Multi Objective Genetic Algorithms it is selected Non-dominated Sorting Genetic Algorithm II and Multi Objective Evolutionary Algorithm with decomposition.

March 2020, 69 pages

Key Words: Multiobjective Optimization, Genetic Algorithm, GA, Particle Swarm Optimization, PSO, Non-dominated Sorting Genetic Algorithm-II, NSGA-II, MOEA/D, traffic flow problem, Sumo, Simulation of Urban Mobility

ÖNSÖZ

Bana destek olan ailem başta olmak üzere değerli fikirleriyle katkıda bulunan tez danışmanım Ökkeş Tolga ALTINÖZ'e (Ankara Üniversitesi Elektrik Elektronik Mühendisliği Anabilim Dalı Öğretim Üyesi), beraber ders aldığımız arkadaşlarıma, izin konusunda yardımcı olan değerli Türk Standartları Enstitüsü yöneticilerine ve bana destek olan çalışma arkadaşlarıma içtenlikle teşekkür ederim.

Adnan Şahin KARACA
Ankara, Ocak 2020

İÇİNDEKİLER

KISALTMALAR DİZİNİ

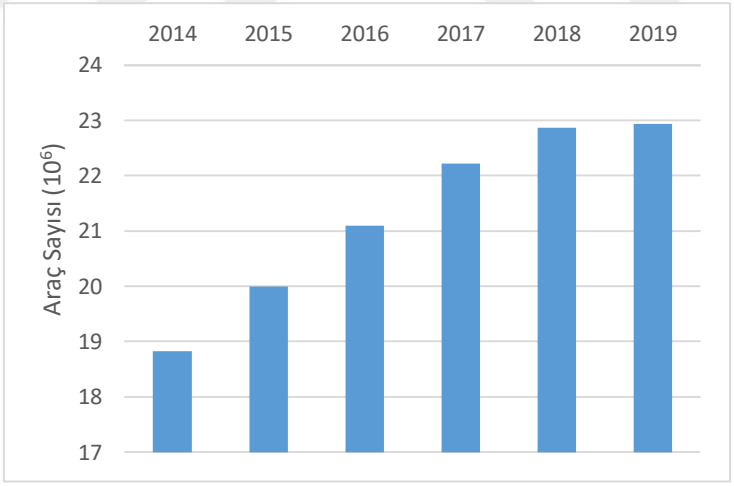
DLR	Almanya Havacılık ve Uzay Merkezi (Deutschen Zentrums für Luft)
GA	Genetik Algoritma
GUI	Grafiksel Kullanıcı Arayüzü
MOEA/D	Ayrıştırma kullanan Çok amaçlı Evrimsel Algoritma (Multi Objective Evolutionary Algorithm with Decomposition)
MOPSO	Çok Amaçlı Parçacık Sürü Optimizasyonu
NSGA-II	Baskılanmayanları Sıralayan Genetik Algoritması 2. Versiyon
PSO	Parçacık Sürü Optimizasyonu
SUMO	Şehiriçi Hareketlilik Simülasyonu (Simulation of Urban Mobility)
XML	Genişletilebilir İşaretleme Dili
TraCI	Trafik Kontrol Arayüzü (Traffic Control Interface)
TÜİK	Türkiye İstatistik Kurumu

ÇİZELGELER DİZİNİ

ŞEKİLLER DİZİNİ

1. GİRİŞ

Trafiğe çıkan araç sayısında meydana gelen artış sebebiyle mevcut yolların kullanım yoğunluğu artmakta ve buna bağlı trafikte sıkışmalar meydana gelmektedir. Şekil 1.1.'de yıllara göre araç sayılarındaki artış gösterilmektedir. Görüldüğü gibi her yıl araç sayısı artmakta ve bu durumda trafik sıkışıklığına neden olmaktadır. Bu sıkışmalar sebebiyle trafikte yaşanan gecikmeler zaman ve para kaybına sebep olmaktadır. Ayrıca araçların trafikte daha çok zaman harcamasıyla oluşan egzoz gazı ve gürültü yaşam kalitesini düşürmektedir. Trafiğe çıkan araç sayılarındaki artış trafik sıkışıklığına sebeb olan nedenlerden biri olmakla birlikte trafik sinyalizasyon sistemlerinin mevcut duruma adapte olamamasıde bu durumu kötüleştirmektedir. Bir başka ifade ile trafikte meydana gelen sıkışıklığın nedenlerinden biri de kavşaklardaki trafik ışık sürelerinin sabit olmasıdır.



Şekil 1.1 Yıllara Göre Taşıt Sayıları Değişimi (TÜİK, 2019)

1.1 Sinyalize Kavşakların Kontrol Sistemleri

Trafik ağlarında kullanılan sinyalize kavşaklar için farklı kontrol metotları önerilmiştir. Bu metotlar geleneksel yaklaşım ve modern yaklaşım olarak iki grupta incelenebilir. Geleneksel Yaklaşımda Sabit Zamanlı Kontrol ve Yarı Uyarımlı Kontrol adı altında iki yöntem bulunmaktadır (Bodur, 2013). Modern yaklaşım olarak ise adaptif kontrol sistemi kullanılmaktadır.

Sabit Zamanlı Kontrol Sistemi: Bilinen en eski sinyalize kavşak kontrol sistemidir. Kavşaklar önceden belirlenen sabit sinyalizasyon süreleri ile çalıştırılır. Kavşaklar gözlemlenerek trafik ile ilgili sayısal veriler toplanır ve bu veriler sonucunda yoğun saatlere göre kavşak kollarının yeşil süreleri ve faz sayıları belirlenerek kavşaklardaki sinyalizasyon sistemi programlanır. Gün/Hafta/Ay/Yıl içerisinde trafik hareketlerinin canlı olarak değişim göstermesinden dolayı kavşak kollarında meydana gelen gereksiz beklemeler bu metodun dezavantajıdır.

Yarı Uyarımlı Kontrol Sistemi: Bu sistemlerde, sabit zamanlı kavşak kontrol sisteminin dezavantajı olan gereksiz beklemele durumunu ortadan kaldırmak için çözümler getirilmiş, talebin az olduğu kavşak kollarına sensörler yerleştirilerek, sensörlerin bağlı bulundukları kavşak kollarının yeşil ışık yanma süreleri talebe göre ayarlanabilir hale getirilmiştir. Genelde sensör kullanılan yerler, askeriye çıkışları, feribot çıkışları, okul önleri, vb. tali yol trafik talebinin günün yalnızca belli zamanlarında var olduğu kavşak tipleridir. Bu kontrol sisteminde araçlar ve yayalar için talep sensörleri mevcuttur. Örneğin feribot çıkışlarında araçlar için sensörler kullanılırken okul önlerinde öğrenci giriş çıkış saatlerinde hizmet vermesi amacıyla yaya butonu denilen yaya sensörleri kullanılmaktadır.

Adaptif Kontrol Sistemi: Adaptif Kontrol sistemi, Sabit Zamanlı ve Yarı Uyarımlı kontrol sistemlerinin tüm özelliklerini bünyesinde barındırmaktadır. Ayrıca sahadaki tüm trafik senaryolarına gerçek zamanlı olarak cevap verebilir ve sinyalizasyon programlarını

ayarlayabilir. Adaptif Kontrol Sistemi dünya üzerinde kullanılan en gelişmiş trafik kontrol sistemidir. Kavşak kollarındaki ve bağlantılı kavşaklardaki trafik yüklerinin gerçek zamanlı olarak ölçülmesi ve ölçülen değerlere göre en uygun sinyal planlarının otomatik olarak oluşturulması esasına göre çalışır. Bu sistemleri diğer sistemlerden ayıran en temel özellik gerçek zamanlı verileri trafik ölçüm sistemlerinden alarak önceden tanımlanmış sinyal zaman planları ile eşleştirmek yerine, çevrimiçi bilgisayar yardımı ile en iyi sinyal zaman planını otomatik olarak oluşturmalarıdır. Bu sistemde sinyalize kavşaklar izole olarak çalışmak yerine birbirleri ile koordineli şekilde çalışabilmektedir (Bodur, 2013).

Simülasyonun detay seviyesine göre 4 çeşit trafik akış modeli vardır. Makroskopik şehirler arası yolları model alır. Mezoskopik cadde bazında trafik akış modeli ile şehirler arası model arasında yer almaktadır. Araç hareketi kuyruk (queue) yaklaşımları kullanılarak simüle edilir. Bu kuyruklar arasında araçlar münferit olarak davranış gösterebilmektedir. Mikroskobik modelde caddede bulunan araçlar model alınmıştır. Şehir içi simülasyon buna örnektir. Aracın davranışının hem taşıtın fiziksel hareket kabiliyetine hem de sürücünün kontrol davranışına bağlı olduğunu varsayar. Alt Mikroskobik Model ise Mikroskobik gibi münferit araçları model alır ancak bunları, aracın hızına veya sürücünün tercih edilen vites değiştirme işlemlerine göre motorun dönüş hızını tanımlayan diğer altyapılara ayırır. Bu, basit mikroskopik simülasyonlara kıyasla daha detaylı hesaplamalar sağlar. Bununla birlikte, alt mikroskobik modeller daha uzun hesaplama süreleri gerektirir. Bu, simüle edilecek ağların boyutunu kısıtlar (Sumo, 2019). Şehir içi simülatörlere SIMVAS++ (Ringel, 1995) TRANSYT (Robertson, 1969) SATURN (Van Vliet, 1982) VISSIM (Fellendorf, 1994) SYNCHRO (Trafficware, 2001) PARAMICS (Cameron, Wylie ve Mcarthur, 1994) örnek verilebilir.

Trafiğin araç sayısıyla doğru orantılı dinamik bir problem olması, bu alanda kurulacak sistemlerin maliyetli olması ve kurulduktan sonra uzun süre değişiklik yapılamaması sebebiyle modellenmesi ve test edilmesi gerekmektedir (Webster, 1958). Değişken trafik ışık zamanlarının ve değişken mesaj işaretlerinin (variable message signs) farklı trafik yoğunluklarında tepkisini ölçmek ve belirli algoritmalar vasıtasıyla bu trafik gereçlerinin

kontrolünün test edilmesi simülasyon programlarıyla mümkündür. Bu alanda Alman Havacılık Merkezinin (DLR) geliştirdiği Şehir Hareketlilik Simülasyonu (SUMO) yazılımının ücretsiz, dış kaynaklı yazılımlarla uyumlu olması ve Matlab ile programlanabilmesi sebebiyle tercih edilmiştir. SUMO mikroskobik bir trafik simülasyonu programıdır. OpenGL kütüphanesinden yararlanabilmekte ve araç tipi tanımlamalarına izin vermektedir. XML dosyasındaki verileri okuyarak parametre alabilir. SUMO alt programı NetEditor sayesinde giriş çıkış verilerinin düzenlenmesine olanak sağlamaktadır. Trafik için kullanılacak harita ise elle veya dış kaynaklı yazılımlarla eklenebilir. SUMO'ya dışardan erişim için TraCI arayüzü kullanılır ve bu arayüz Matlab programıyla bağlantıyı sağlar (D. Krajzewicz, 2011). Bu çalışmada Matlab ortamında geliştirilecek çok amaçlı evrimsel algoritmaları uygulanacaktır. Trafik Kontrol Arayüzü (TraCI) Alman Havacılık ve Uzay Merkezi (DLR) tarafından geliştirilmiştir. Simülasyon dış kaynaklı uygulamadan başlatılır ve her çalıştırmada dış kaynaklı uygulamadan çağırılır. TraCI, taşıtların, trafik ışıklarının, yol altyapısının ve diğer simülasyon nesnelerinin dış kaynaklı yazılımlarla etkileşimine izin verir. Bir trafik ışığının fazı, süresi ve programı bu arayüz kullanılarak değiştirilebilir. TraCI ayrıca bir aracın maksimum hızını değiştirmeye, frenlemeye veya şerit değiştirmeye, yeni bir varış yeri vermeye veya bir aracın rotasının yeniden hesaplanmasına izin verir (Acosta, 2019). SUMO tarafından elde edilen verilerle hesaplanan amaç fonksiyonları ile trafik ışık süreleri değiştirilerek sistemde öncelikle tek amaçlı optimizasyon sağlanacaktır. Tek amaçlı optimizasyonda tek bir kriter üzerinden işlem yapılır. Sonucu skaler bir sayıdır. Sonrasında amaçlar ikili olarak seçilerek çok amaçlı optimizasyona geçilecektir. Çok amaçlı optimizasyon birden fazla ve birbiriyle çelişen kriterleri eş zamanlı olarak optimize etme işlemidir (Deb, 1998). Çok amaçlı optimizasyon problemlerinde her amacı optimize eden tek bir çözüm bulunmamaktadır. Bu durumda amaç fonksiyonları değerlerinden bazıları iyileşirken diğerleri kötüleşme eğilimi gösterir. Bu durum gözlemlendiğinde Pareto Optimal olarak adlandırılan çözüm seti bulunur.

Bu çalışmada tek ve çok kavşaklı trafik ağları için yoğun trafik senaryosunda inceleme gerçekleştirilecektir. Yeşil ışık süreleri değiştirilerek amaç değerleri olan emisyon gazları, bekleme ve seyahat süreleri minimize edilecektir. Tez içerisinde ikinci bölümde

literatürde yapılan çalışmalarda sezgisel algoritmaların kullanıldığı tespit edilmiştir. Bu nedenle optimizasyon algoritmaları sezgisel algoritmalar tercih edilmiştir. Üçüncü bölümde seçilen optimizasyon algoritmaları tek amaç için çalışan GA ve PSO, çok amaç için çalışan NSGA-II ve MOEA/d hakkında bilgiler verilmektedir. Ayrıca çok amaçlı problemi tek amaçlı çözüm üretebilen algoritmalarla çözebilmeye olanak sağlayan skalerizasyon tekniği hakkında bilgiler verilecektir. Sonuçların değerlendirilebilmesi için kullanılan Boşluk ve Hiper hacim metriklerinden bahsedilmiştir. Devamında simülasyon yazılımı ve oluşturulan kavşak modelleri hakkında bilgiler mevcuttur. Dördüncü bölümde tek kavşaklı ve çok kavşaklı uygulamalar için gerçekleştirilen çalışmalardan bahsedilmiştir. Amaç fonksiyonları sabit ışık sürelerinde değerlendirilmiştir. Sonrasında farklı sabit ışık sürelerinde amaç değerlerinin değişimi incelenmiştir. Çıkarılan sonuçlara göre tek kavşaklı ağ için algoritmalar çalıştırılmış ve sonuçlar elde edilmiştir. Devamında çok kavşaklı trafik ağı için parametreler değerlendirilmiştir. Belirlenen parametrelerden sonra Monte Carlo tekrarları ile simülasyon sonuçları elde edilmiştir. Bu sonuçlar üçüncü bölümde bahsedilen metriklere göre değerlendirilmiştir. Beşinci bölümde elde edilen sonuçlar yorumlanacaktır.

2. KURAMSAL TEMELLER

Traik akış problemini ile ilgili yapılan çalışmalar ilk olarak gerçek hayatta kullanılan trafik parametrelerinin matematik modele aktarılması ile başlanmıştır (Webster, 1958). Bu çalışmada kavşaklardaki ışık sürelerinin belirlenmesi için öncelikle trafikte yaşanan gecikmeler incelenmiştir. Gecikme belirli bir zaman diliminde kuyrukta bekleyen araç sayısı olarak tanımlanmıştır. Bütün zaman dilimlerinin toplanması ile de toplam trafik gecikmesi bulunabilir. Eğer kuyrukta n araç varsa ve kuyruktaki araçlar her u saniyede bir sisteme giriyorsa toplam gecikme $n * u$ araç-saniye olarak tanımlanır. Laboratuvar ortamında gerçekleştirilen simülasyonlarda gecikme miktarları yukarıda açıklandığı gibi modellenmiş ve farklı yeşil ışık süreleri göz önüne alınarak optimizasyon yapılmıştır. Farazi olarak kurgulanan farklı kavşak yoğunluklarında ışık sürelerinin toplam gecikme üzerine etkisi incelenmiştir. Yeşil ışık yanması ile birlikte araçlar giderek hızlanan bir şekilde kavşağı terk ederler. Araçlar hızlanmaya başladıktan sonra belirli bir zaman sonra hızlanma durur ve kavşağı terk eden araç sayısı sabit kalır. Buna trafik doygunluk oranı denir. Trafik doygunluk oranı hesaplarken tek şeritli yollardan oluşan tek kavşak göz önüne alınmıştır. Her kolda oluşan trafik akış oranı eşit alınmıştır. Trafik akış oranı bir zaman diliminde kavşaktan geçen araç sayısıdır. Trafik akış oranına göre ışık sürelerinin değiştirilmesinin ortalama gecikmeye etki ettiği tespit edilmiştir. Çalışmada yapılan tespitlere göre trafik akışı incelenirken günün en yoğun saatlerinin uygulamaya dahil edilmesi gerekmektedir. Bu uygulamaya sağa dönüş işlemi ve ticari araçların toplam gecikme ve trafik akışına etki etmesi sebebiyle dahil edilmesi gerektiği saptanmıştır. Trafik gecikme etkilerine göre farklı araç sınıflarının binek araç karşılığı: 1 Ağır veya orta sınıf ticari araç 1 ¾ binek araca, 1 otobüs 2 ¼ binek araca, 1 tramvay 2 ½ binek araca, 1 hafif yük taşıtı 1 binek araca, 1 motorsiklet 2/3 binek araca, 1 bisiklet 1/3 binek araca tekabül etmektedir. Dönüşlerde ise sağ dönen bir araç 1 ¾ düz giden araca tekabül etmektedir. Yalnız çoğu kavşakta sağa dönüşlerde kontrollü geçişler serbest olduğu için kavşaktan kavşağa trafik etkisi değişebilmektedir. Park etmiş bir aracın ise gecikmeye etkisi, yol hacminin bir şerit daralmasına eşittir. Webster tarafından geliştirilen

simülasyon modelinde araçların kavşağa rastgele vardıkları kabul edilmiştir. Kuyruk oluşturan araçların sabit bir oranla (doygun akış) yeşil ışıkta kuyruktan ayrılmaktadır. Kavşaktan doğan gecikme, kavşak olarak tanımlanan yolun mesafesini aracın alacağı süre ve kavşak nedeniyle kaybettiği zaman arasındaki farktır. Başka bir deyişle kavşakta yavaşlama, durma ve hızlanma nedeniyle oluşan zaman kaybıdır. Trafik sistemlerinin karmaşık olması sebebiyle simülasyon yapılmasının gerekliliğinden bahsedilmiştir. Literatürdeki diğer çalışmalarda dış kaynaklı simülasyon yazılımları kullanılmıştır.

2.1 Literatürdeki Çalışmalar

Trafikteki ışık sürelerinin ayarlanması ve özellikle uyarlamalı/dinamik ışık sürelerinin belirlenmesi için literatürde yapılan çalışmalarda optimizasyon algoritmalarından faydalanılmıştır. Yagar ve Han (1994) gerçek zamanlı sinyal planları için kural tabanlı bir prosedür önerilmiştir. Bu prosedürde toplu taşıma araçlarının yolcu indirip bindirmelerinden kaynaklanan gecikmeler göz önüne alınmıştır. Genel olarak önerilen yöntemde belirli sayıda karar kuralına istinaden taşıt öncelikleri belirlenmiş ve buna göre kısa vadeli alternatif gerçek zamanlı faz döngüleri (belirli bir yol için tekrar eden ışık düzeni) üretilmiştir. Sonrasında bu sinyal döngülerini değerlendirmeye tutarak bütün trafiğe en az maliyeti getiren seçenek tercih edilmektedir. Toronto'nun Queen Caddesi koridoru üzerindeki gerçek veriler kullanılarak simülasyonu gerçekleştirilmiştir. Kırmızı ışık zamanları ile yolcu indirme zamanları kesiştirilerek gecikme süreleri büyük miktarda azaltılmıştır. Sonuç olarak toplu taşıt önceliği sağlayan adaptif kontrol için bir modelleme tekniği oluşturulmuştur. Anderson Vd. farklı bir yaklaşım olarak bulanık mantık ile karar alan bir kavşak mekanizmasında birden fazla amaç denemiş ve eş zamanlı olarak seyahat zamanı ve emisyonu minimize edilemediğini tespit etmiştir. Bu nedenle çalışmada çok amaçlı genetik algoritmaların kullanılması gerektiği belirtilmiştir (Anderson, Sayers ve Bell, 1998). Kalganova, Russell ve Cummings yaptıkları çalışmada (1999) genetik algoritma ile bir ağ içerisindeki bütün kavşakları kontrol etmeye çalışarak trafik optimizasyonu gerçekleştirmiştir. Ağdaki toplam gecikme amaç fonksiyonu olarak belirlenmiştir. Sabit zamanlı trafik sistemi ile karşılaştırılmış ve net bir sonuç elde

edememişlerdir. Parametrelerin genetik algoritma sonucuna etki ettiğini tespit etmişlerdir.

Vogel vd. tarafından gerçekleştirilen SIMVAS++ (Ringel, 1995) isimli bir yazılım kullanan başka bir çalışmada evrimsel algoritmalar karar alma mekanizması olarak seçilmiştir. Farklı sinyal planları ve ağlar ile denemeler yapılmıştır. Parametre değişimi incelenmiştir. Gecikme süreleri kullanılarak optimizasyon gerçekleştirilmiştir (Vogel, Goerick ve Seelen, 2000). Ceylan ve Bell (2004) TRANSYT (Robertson, 1969) yazılımı kullanılan bir çalışmada karar alıcı olarak genetik algoritma (GA) kullanmışlardır. Araçların güzergah değiştirmelerini modele eklemişlerdir. Modelin 33. nesilden sonra yakınsadığı görülmüştür. Optimizasyon sonucunun sabit ışık düzenine göre %34 performans iyileştirmesi sağladığı tespit edilmiştir.

Amerika Teksas Federal Otoyollar Başkanlığınca düzenlenen bir raporda kapalı çevrim trafik sistemleri incelenmiştir. Burada istatistiğe dayalı sabit ışık sürelerinin yanında adaptif sistemler de konu olmaktadır. Genetik algoritmaların yanında NSGA-II gibi çok amaçlı optimizasyon teknikleri de kullanılmıştır. Çalışmada 3 ile 5 arası değişen seri kavşaklar trafik ağı olarak kullanılmaktadır. Amaç olarak gecikme, durma sayısı ve komşu kavşaklardaki araçların ayrılma sayısı kullanılmıştır. Aldıkları sonuç itibariyle 500 iterasyon incelenmiş ve 50 iterasyondan sonra çok küçük farklarla yakınsama sağlandığı görülmüştür (Abbas, Chaudhary, Pesti ve Sharma, 2005). Teklu ve Sumalee (2007) sürücülerin güzergâh değişimini kullanan bir çalışmada GA kullanmışlar ve toplam seyahat süresi amaç fonksiyonu olarak belirlemişlerdir. SATURN (Van Vliet, 1982) yazılımını simülasyon için kullanmışlardır. İngilterenin Chester şehri uygulama için seçilmiştir. Optimizasyon algoritması 70 nesil için çalıştırılmış ve 3 farklı trafik yoğunluğu için 10'ar defa sürdürülmüştür. GA parametrelerinin sonuç alma süresine etki ettiği ve üzerinde ayrı bir çalışma yapılmasının gerektiği ortaya konulmuştur. Güzergâh değişiminin hesaba katılmasının optimizasyon sağladığı belirtilmiştir.

Başka bir çalışmada çok amaçlı evrimsel bir algoritma olan Baskılanmayanları Sıralama Genetik Algoritması-II (NSGA-II) optimizasyon algoritmasını trafik optimizasyonu için denenmiş ve gelenksel yöntemlere kıyasla daha iyi sonuçlar verdiği gözlenmiştir (Branke, Goldate ve Prothmann, 2007). Simülasyon için VISSIM (Fellendorf, 1994) yazılımı kullanmışlardır. Çalışmada Haid-und-Neu-Str. ve Ostringin Karlsruhe, Almanya caddeleri model alınmıştır. Sonuçlar TRANSYT-7F (Robertson, 1969) ve SYNCHRO (Trafficware, 2001) algoritmalarıyla kıyaslanmıştır. NSGA-II ile daha iyi sonuçlar elde edilmiştir. Sing, Tripathi ve Arora (2009) genetik algoritma kullanarak gerçek zamanlı ışık süresi kontrolü sağlayan bir çalışma ile geniş bir trafik ağını kontrol etmişlerdir. Yazdıkları simülasyon programını kullanmışlardır. Sabit zamanlı trafik ışık sistemine göre gelişme kaydedildiği belirtilmiştir. Luo, Ma ve Huang (2009) tarafından gerçekleştirilen bir çalışmada toplam gecikmenin amaç olarak seçildiği bir çalışmada GA karar alıcı olarak kullanılmış ve buna ek olarak yeni bir algoritma önerilmiştir. Çok amaçlı bağışıklık algoritması ismini vermişlerdir. Sabit zamanlı trafik ışığı sistemiyle kıyaslandığında iyi sonuçlar verdiği belirtilmektedir. Önerilen algoritma ile kavşaklar arasındaki faz farkı kontrol edilmiştir. Çalışmada 3 kavşaklı bir ağ kullanılmıştır. Zeng, He ve Chen (2010) yaptıkları çalışmada tekli bir kavşağı modellemiş, Karınca Koloni tabanlı çok amaçlı evrimsel optimizasyon algoritması ile optimize bir sinyal planı oluşturmuşlardır. Bu işlemler için TRANSYT (Robertson, 1969) yazılımı kullanılmıştır. Yapılan çalışmada sadece trafikteki araçlar değil, yayalar için bekleme süresi, trafik kapasitesi ve sinyalizasyondaki meydana gelen kuyruk uzunluğu da göz önüne almıştır. Dağüstü (2010) Webster yöntemi kullanarak ışık sürelerini değiştiren bir algoritma geliştirmiş, model olarak İstanbul ilindeki bazı kavşakların zamanlama planlarını almıştır. Simülasyon programı olarak VISSIM (Fellendorf, 1994) kullanılmış ve geliştirilen sürelerin uygulanmasının sabit zamanlı sinyalizasyona göre daha iyi sonuçlar verdiği gözlemlenmiştir. Performans kriteri olarak taşıt başına ortalama gecikme süresi, ortalama duruş sayısı, ortalama durma süresi, toplam CO ve NOx emisyonu, yakıt tüketimi, gözlenen bir saatlik süre içinde kavşağı terk eden taşıt sayısı ve toplam seyahat süresi alınmıştır.

Chin, Yong, Bolong, Yang Ve Teo (2011) tarafından önerilen TSTM yazılımının trafik simülatörü olarak kullanıldığı bir çalışmada GA, optimizasyon algoritması olarak belirlenmiş ve iki bitişik kavşağın kontrolü sağlanmıştır. Minimum gecikme amaç değeri olarak seçilmiştir. Optimizasyon algoritmasının amaç değerlerinde iyileşme sağladığı gözlenmiştir. Chen ve Yuan (2011) tarafından gerçekleştirilen bir çalışmada toplam yolculuk gecikmesinin ikiye bölünmüştür. Mevcut gecikme değeri besleme ve besleme olmayan gecikme olarak adlandırılmıştır. Bu isimlendirmeler trafik akışlarına göre verilmiştir. Çok amaçlı algoritmalardan uyarlanmış kontrol algoritması (modified compatible control algorithm) kullanmışlardır. Çalışmada toplamda onbir kavşaklı bir trafik modeli oluşturmuşlardır. Bu işlemler için TRANSYT-7F (Robertson, 1969) yazılımı kullanmışlardır. Robles (2012) tarafından gerçekleştirilen VISSIM (Fellendorf, 1994) simülasyon yazılımının kullanıldığı bir tez çalışmasında çok amaçlı optimizasyon algoritmaları mikroskobik trafik ağını optimize etmek için kullanılmıştır. Optimizasyon algoritması olarak NSGA-II seçilmiştir. İlk olarak Wuhan bölgesindeki bir tek kavşak üzerinden sonrasında Stockholm'de bulunan iki kavşak üzerinden simülasyon gerçekleştirilmiştir. Gecikme, ortalama durma sayısı ve ortalama yakıt sarfiyatı amaç olarak seçilmiştir. Simülasyon süresi 60 dakika olan bir sistem kullanılmıştır. Optimizasyon sisteminin sabit süreli sisteme göre daha iyi sonuç verdiği gözlemlenmiştir. Çalışmada 3 amaçlı ve 2 amaçlı sistemler incelenmiş ve düşük amaç sayısı ile algoritmanın daha iyi sonuçlar verdiği tespit edilmiştir. Tekli kavşakta sistem uzun döngü süreleri bulmuşken ikili kavşakta kısa döngü sürelerine yönelmiştir. Simülasyon programı ile çalışmanın zaman tüketimini arttırdığından bahsedilmiştir. NSGA-II'nin sonuç çeşitliliği bakımından yetersiz olduğunu ve farklı algoritmalarla kıyaslanması gerektiğini belirtmişlerdir. Otobüs geçiş üstünlüğünün tasarıma eklendiği başka bir çalışmada amaç fonksiyonu olarak seyahat süresi, gecikme, yeşil dalga etkisi ve diğer faktörler hesaba katılmıştır (Che, Zhang, Lin, Luo ve Wu, 2012). Çalışmada PARAMICS (Cameron, Wylie ve Mcarthur, 1994) yazılımı kullanılmıştır. Trafik akışı önerilen yöntemle zamansal bazda arttırılmıştır. Algoritma olarak Kuantum Genetik Algoritması kullanılmıştır. Ma (2012) çalışmasında anlık emisyon modeli kullanılarak çok amaçlı evrimsel algoritmalarıyla simülasyon ortamında entegre edilen bir model ile kavşak sinyal zamanlarının optimizasyonunu önermiştir. Seyahat süresinin yanında enerji ve

çevresel sonuçlarını da göz önünde bulundurmuştur. Çok amaçlı algoritma olarak NSGA-II seçilmiştir. VISSIM (Fellendorf, 1994) simülasyon yazılımında gerçekleştirmesi yapılmıştır. İzole tekli bir kavşak model olarak bu çalışmada kullanılmış ve önerilen algoritmanın daha iyi sonuçlar verdiği gözlemlenmiştir. Yapılan çalışmada algoritma hesaplama bakımından çok maliyetli olduğu vurgulanmıştır. Li vd. yaptıkları çalışmada izole bir kavşak içerisinde çok amaçlı optimizasyon algoritması NSGA-II ile çözüm sağlanmıştır (Li, Yu, Tao ve Chen, 2013). Maksimum geçiş ve ortalama kuyruk oranının minimum olması amaç olarak seçilmiştir. Yoğun trafikteki bir kavşak uygulama için seçilmiştir. VISSIM SCAPI (Fellendorf, 1994) simülasyon yazılımı sonuçların karşılaştırılması için kullanılmıştır. Simülasyon sonuçları SYNCHRO (Trafficware, 2001) benzetim ortamına kıyasla önerilen algoritmanın daha verimli olduğunu göstermiştir. Önerilen algoritma ayrıca farklı trafik koşulları ve yoğunluklarında test edilmiştir. Rahmat (2013) tarafından gerçekleştirilen bir çalışmada Malezyanın Bangi şehrinde beş kavşaklı bir sistemde optimizasyon gerçekleştirilmiştir. Bunun için GA algoritması kullanılmıştır. Gecikme ve trafik akış oranı amaç olarak seçilmiştir. Görüntü işleme teknolojisiyle araç sayıları gerçek hayat verileri üzerinden alınmıştır. Trafik akış oranı, kuyruk uzunluğu ve akış hızı gibi veriler kameradan alınmıştır. Sakin zamanda %56 yoğun zamanda %34 iyileşme gözlemlenmiştir. Yeşil ışık süresini belirlemek için GA parametreleri 60 popülasyon, 200 nesil ve değişkenlerin alacağı değer aralığı 120 saniye sabit ışık süresinin yüzde 10'u ile 60'ı arasındadır. Zhou ve Cai (2014) değişken sinyal zamanlarında PARAMIKS mikroskobik trafik simülasyon yazılımıyla çok amaçlı genetik algoritmaların kullanıldığı bir çalışmada uygulama alanı olarak Çin'de JiangNanXi-BaoGang yolu 4 kollu kavşağını seçmişlerdir. Rulet tekerinin model olarak kullanıldığı Genetik Algoritmanın daha iyi performans verdiği belirtilmektedir. Webster metodu sabit zamana göre daha başarılı olduğu sonucuna varılmıştır. Chen (2014) yaptığı çalışmada trafik kontrol problemlerinde, trafik akışının ortalamasını kullanmaları sebebiyle geleneksel yöntemlerin gerçek koşullardan ayrıldıkları belirtilmiştir. Çalışmada bağışık tabanlı çok amaçlı uyumlu optimizasyon kontrol algoritması (RMOOC) önerilmiş ve aralık veya bölge kontrolü amaçlarıyla min-maks bağışıklık modeliyle trafik ışığı kontrol problemi çözülmeye çalışılmıştır. Simülasyon sonuçlarına göre önerilen yöntemin ağ kontrolünde etkili olduğu belirtilmiştir. Damay (2015) çalışmasında SUMO

yazılımını simülasyon için kullanmıştır. Bu çalışamda genetik algoritma ışık sürelerini belirlemede kullanılmıştır. Gerçek hayat uygulaması için Rouen, Fransa seçilmiştir. Seçilen bölge orta ölçekli bir trafik ağına sahiptir. Bu şehirde 11 kavşak 168 trafik ışığı ve 40 muhtemel dönüş bulunmaktadır ve 20 sensör vasıtasıyla bu ağ izlenmektedir. Bu çalışmada ayrıca çok amaçlı evrimsel algoritmalar kullanılmıştır. Ana algoritma olarak NSGA-II seçilmiştir. SPEA-II algoritması ise karşılaştırma için kullanılmıştır. SUMO yazılımının sağlayabileceği amaç fonksiyonları arasındaki ilinti durumu ikişerli olarak incelenmiş ve ilintisi en az olan amaç ikilisi seçilmiştir. Grafik çizdirildiğinde aralarında ters orantı bulunan bekleme süresi ve PMx gazı emisyonu amaç olarak ilk aşamada belirlenmiş, sonrasında bekleme süresi ve CO_2 seçilmiştir. Amaç değerleri önce tek amaçlı GA ile sonrasında çok amaçlı algoritmalar ile optimize edilmiştir. Karadeniz (2016) yaptığı çalışmada gerçek hayat uygulaması olarak Karabük-Safranbolu arasındaki kavşakları model olarak alan ve sinyal zamanlarını simülasyon ortamına aktaran sabit zamanlı, yeşil dalga ve gerçek zamanlı trafik ışık sistemlerini karşılaştırmıştır. Çalışma SUMO ve diğer simülasyon ortamları kullanılarak gerçekleştirilmiştir. SUMO yazılımı kullanılmıştır. Kurulan sistemde ortalama hız, ortalama bekleme süresi ve ortalama seyahat süreleri hesaplanmıştır. Altı farklı trafik yoğunluğu için üç kavşak modeli karşılaştırılmış ve düşük yoğunluklarda trafik ışık modelinin amaç değerlerinin optimizasyonu üzerine kayda değer bir etkisi bulunmadığı sonucuna varmıştır. Yoğunluğun fazla olduğu senaryolar incelenirse en başarılı olan sistem gerçek zamanlı ve sonrasında yeşil dalga trafik ışık sistemi olduğu gösterilmiştir. Bu sistemlerin sabit zamanlı sinyalizasyona göre daha verimli olduğu gözlemlenmiştir. Gerçek zamanlı trafik kontrolü için çok amaçlı parçacık sürü optimizasyon (MOPSO) algoritması denenen ve başarılı sonuçlar elde edilen bir çalışmada zamanla değişen trafik akışına göre ortalama gecikme süresi minimum ortalama durma süresi ve maksimum etkin trafik kapasitesi gibi birbirinden bağımsız üç amaç belirlenmiş ve çözülmüştür (Jiao, Li ve Li, 2016). Benzer bir çalışmada çok amaçlı parçacık sürü optimizasyonu (MOPSO) algoritması kullanılmıştır (Hatri ve Boumhidi, 2017). Trafik sıkışıklığını önlemek adına güzergâh belirleme ve trafik ışıklarının kontrolü için bir yöntem geliştirilmiştir. Çalışmada amaç fonksiyonları maksimum trafik akışı ve minimum ortalama kavşak bekleme süresi seçilmiştir. MOPSO algoritmasının performans ve hassasiyetini arttırmak için Q-

Learning algoritması kullanılmıştır. Bu sürüdeki her öğenin uygun MOPSO parametresi seçebilme ve problem yapısına adapte olabilme yeteneklerini kazandırmıştır. Aynı şekilde trafiğin yoğunlaşması durumunda güzergâh belirlemede karar alarak yeni gelen araçların farklı rotalardan gitmelerini sağlamıştır. Simülasyon için SUMO yazılımı kullanılmıştır.

Literatürdeki çalışmalar göz önüne alındığında genel eğilim stokastik yapıya sahip sezgizel algoritmalar ile ışık süresini kontrol etmektir. Çoğunlukla optimizasyon simülasyon yazılımları üzerinden gerçekleştirilmiştir. Gerçek hayat verileri üzerinden eş zamanlı olarak gerçekleştirilen optimizasyon çalışma sayısı azdır. Gerçek hayat uygulamalarında kentlerinden bir bölümün simülasyon yazılımına aktarılması ile ışık sürelerini kontrol ederek amaç değerlerini minimize etmeye çalıştıkları gözlemlenmektedir. Uygulamalarda tek ve birbirine seri bağlı kavşaklar kullanılmıştır. Seri bağlı kavşakların ışık düzenleri beraber kontrol edilmiştir. Literatürde GA ve NSGA-II algoritmalarının sıklıkla kullanıldığı görülmektedir. Bu çalışmada da genel eğilime uyulacak ve kontrol mekanizması için tek amaçlı GA ve PSO algoritmaları, çok amaçlı NSGA-II ve MOEA/D algoritmaları kullanılacaktır. Çevrimiçi kavşak kontrolüne kıyasla benzetim ortamında algoritmaların test edilmesi daha az maliyetli olması sebebiyle bu yöntem tercih edilmiştir. SUMO yazılımı benzetim için kullanılacaktır. Gerçek hayat uygulaması için Ankara'nın Keçiören ve Çankaya ilçelerini birbirine bağlayan bir güzergâh seçilmiştir. Büyük bölümünü Atatürk Bulvarı oluşturmakta ve Çankırı Caddesiyle Cinnah Caddesini ihtiva etmektedir. Buraya ait olan 5 adet seri bağlı kavşak noktasındaki ışık süreleri kontrol edilerek amaç olarak seçtiğimiz CO_2, Bekleme ve Seyahat süreleri minimize edilmeye çalışılacaktır. Çalışmada literatürden farklı olarak tek ve çoklu kavşak aynı anda değerlendirilmiştir. Ayrıca tek amaçlı problemleri çözebilen algoritmalar skalerizasyon yöntemiyle çok amaçlı problemde uygulanmış ve çok amaçlı algoritmalar ile beraber kullanılmıştır. Bilgi dahilinde MOEA/D algoritması ile yapılan ışık sürelerini kontrol etme amaçlı bir çalışma yapılmadığı görülmüştür. Literatürde Çok amaçlı PSO algoritması MOPSO kullanılmış veya PSO hibrit olarak kullanılmıştır ancak doğrudan kullanılmamıştır. Bu sebeple PSO da çalışmaya dahil edilmiştir.

3. MATERYAL ve YÖNTEM

Bu çalışmada adaptif kavşak modeli kullanılacaktır. Adaptif kavşak modelinde ışık sürelerinin belirlenmesinde tek ve çok amaçlı genetik algoritmalar kullanılacaktır. Algoritmalar SUMO ortamında oluşturulan trafik senaryosunu TraCI ara yüzü ile kontrol edecektir. TraCI komutları kullanılarak SUMO'dan CO_2 emisyonu ve araç sayıları belirli bir simülasyon çalışma süresince için okunacaktır. Toplam seyahat ve bekleme süreleri araç sayılarına göre hesaplanacaktır. Bu değerleri minimize etmek için kavşak ışık süreleri değişken olarak belirlenmiştir. Tek ve çok amaçlı optimizasyon algoritmaları kullanılarak çözüm uzayı içerisinden en uygun çözüm trafik kavşak sinyalizasyonu için seçilecektir. Literatürde trafik problemlerinde yaygın olarak kullanılan GA ve NSGA-II'nin yanında trafik problemlinde daha az kullanılan PSO ve MOEA/D algoritmaları ile sabit zamanlı trafik ışığı modeli karşılaştırılacaktır.

3.1 Tek Amaçlı Optimizasyon

Tanımlı bir aralık içerisinde istenen amaç değerlerinin minimize veya maksimize etme eylemine optimizasyon denilir.

$f : R^n \rightarrow R$'ye giden bir vektör amaç fonksiyonu olmak üzere

$$\min y = f(\overline{x})$$
$$g_j(\bar{x}) \leq 0, \quad j = 1, 2, \dots, m$$
$$h_k(\bar{x}) = 0, \quad k = 1, 2, \dots, e$$
$$\bar{x}_i^l \leq \bar{x}_i \leq \bar{x}_i^u, i = 1,2, \dots, n \qquad (3.1)$$

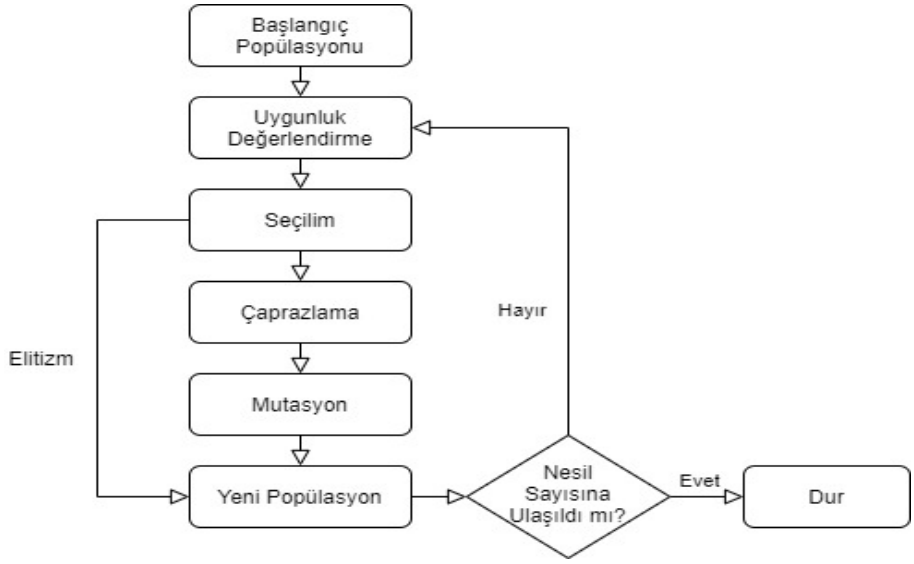
Denklem (3.1) ele alındığında: y amaç değerini, $\overline{x}$ karar değişkenini, f ise maliyet veya amaç fonksiyonunu ifade etmektedir. g eşitsizlik h ise eşitlik sınırlama fonksiyonlarıdır. $\bar{x}^l$ ve $\bar{x}^u$ ise karar değişkeninin alabileceği alt ve üst sınırları ifade etmektedir. Bu tez çalışmasında 9 ve 40 saniye olarak belirlenmişlerdir. Ana yollarda gerçek hayat

uygulamalarında yeşil ışık yandığında ilk araç yaklaşık 3 saniye sonraki araçlar da yaklaşık 2 saniye bir gecikme ile hareket ederler (Karadeniz, 2016). Otobüs güzergahındaki bir anayol için yeşil ışık süresinin alt sınırının 9 saniyenin altında değerler olması karşılaşılmış bir uygulama değildir. Rahmat (2013) yaptığı çalışmada 12 saniye alt limit olarak belirlemiştir. Simülasyon yazılımında alt değer 0 saniye seçildiğinde optimum süreyi 5 saniye bulmakta ve 1 saniyelik gecikme ile araçlar harekete geçmektedir. Bu veriler göz önünde bulundurulduğunda sarı ışık süreleri de düşünülerek alt değer 9 saniye alınmıştır. Tez çalışması kapsamında yapılan çalışmada üst değer olarak 40 saniye üzerindeki değerlerin bulunmadığı tespit edilmiştir. Karar değişkenimiz ışık süresidir. Amaç değerlerimiz bekleme ve seyahat süreleri ile CO_2 emisyon miktarlarıdır. Program optimizasyon algoritmasının ürettiği ışık değerlerini SUMO yazılımına gönderecek ve elde edilen amaç değerlerine göre optimizasyonu gerçekleştirecektir.

Tek amaçlı optimizasyon problemlerinde amaç değeri tek bir skaler sayıdır. Trafik problemleri karmaşıklıkları sebebiyle klasik yöntemlerle optimizasyon yapılamamaktadır. Bu nedenle sezgisel algoritmalara ihtiyaç vardır. Stokastik yapıları ve literatürdeki çalışmalarda yüksek başarı sağlamaları sebebiyle bu çalışmada genetik algoritma ve parçacık sürü optimizasyonu algoritmaları tek amaçlı optimizasyon için tercih edilmiştir. Amaç değerleri olarak sırayla seyahat süresi, CO_2 emisyon değeri ve gecikme süresi ayrı ayrı dikkate alınacaktır.

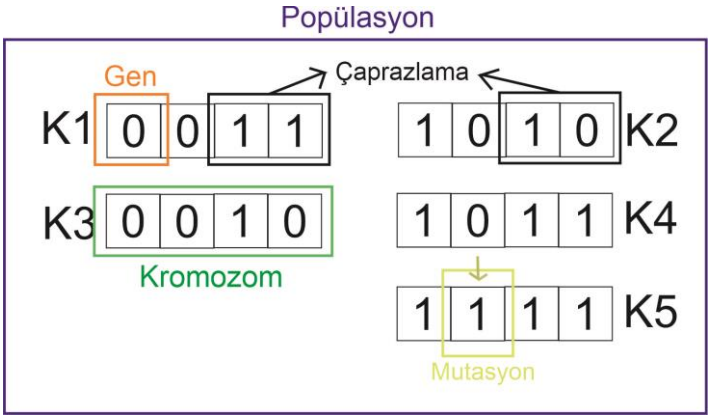
3.1.1 Genetik algoritma

Evrim teorisi model alınarak geliştirilmiş bir arama algoritmasıdır. 1975 yılında John Holland tarafından temelleri ortaya konulmuştur. Çoğalma için en uygun bireylerin seçildiği doğal seçilim sayesinde sonraki neslin bireyleri elde edilir.



Şekil 3.1 Genetik Algoritma Akış Şeması

Seçilim: Rastgele oluşturulmuş başlangıç popülasyonundan en uygun bireylerin ikili turnuva yöntemiyle seçilmesi işlemidir. Arama problemlerine bu ilke uygulandığında bir dizi çözüm üretildikten sonra en iyileri seçilir. Bu çözümler üretilirken karar değişkenlerini oluşturan sayılar belirli yöntemlerle kodlanır. İkili kodlama sayesinde Şekil 3.2'de verildiği gibi kromozom adını verdiğimiz sayı dizilerine dönüştürülür.



Şekil 3.2 Genetik Algoritma İkili Kodlama Gösterimi

Rastgele sayılar karar değişkenleri sınırlarına göre düzgün dağılımla üretilerek başlangıç popülasyonu oluşturulur. Popülasyonda kaç adet kromozom olacağı kullanıcı tarafından belirlenir. Bu çalışmada 20 alınmıştır. Popülasyon içindeki her bir kromozom bir çözümü ifade ettiğinden verilen sınırlar içerisinde kalmalıdır. Sonrasında amaç fonksiyonu adı verilen her bir çözümün ne kadar uygun olduğunu gösteren fonksiyondan geçirilerek bireylere puan verilir. Bu çalışmada amaç fonksiyonu için trafik simülatöründen yararlanılacaktır. Karar değişkeni olarak üretilen ışık süreleri değerleri programa verilir ve CO_2, Bekleme Süresi ve Seyahat süreleri minimize edilir. Bu değerleri minimize etmek hedeflendiğinden en küçük değerleri veren çözümler daha yüksek puan alarak sonraki nesillerde çoğalma ihtimallerini arttırırlar.

Başlangıç popülasyonunun bütün bireyleri incelendikten sonra yeni neslin oluşturulma işlemi başlar. Amaç puanı sonraki nesillere genini aktarabilir. Belirli ihtimaller dahilinde üç yöntemle yeni nesil oluşturulur. Birincisi herhangi bir işlem olmadan olduğu gibi sonraki nesile geçmeleri, ikincisi çaprazlama yaşadıktan sonra iki farklı birey olarak sonraki nesile katılmaları, üçüncüsü ise mutasyon yaşamalarıdır.

Çaprazlama: Kromozomlar genlerden meydana gelmektedir. Ebeveyn olarak seçilen K1 ve K2 kromozomlarının içerisinde belirli bir grup genin karşılıklı yer değiştirmesi işlemine çaprazlama denir. Değişimden sonra oluşan K3 ve K4 kromozomlarına ise çocuk nesil adı verilir. Genler içinde çaprazlama noktası veya noktaları rastgele belirlenir. Örneğin 0011 ve 1010 kromozoma sahip iki ebeveyn bireyi ele alalım. Çaprazlama sayısı iki seçildiğinde 00/11 ve 10/10 olarak ayrılırlar. Yeni nesiller ise 0010 ve 1011 olur. Bu nesiller popülasyona eklenir. Seçilen kromozomlar kullanıcı tarafından belirlenen bir olasılık oranında çaprazlamaya maruz kalırlar. Bu orana çaprazlama parametresi denir. Bu olasılık 0 ile 1 arası bir sayı olmalıdır.

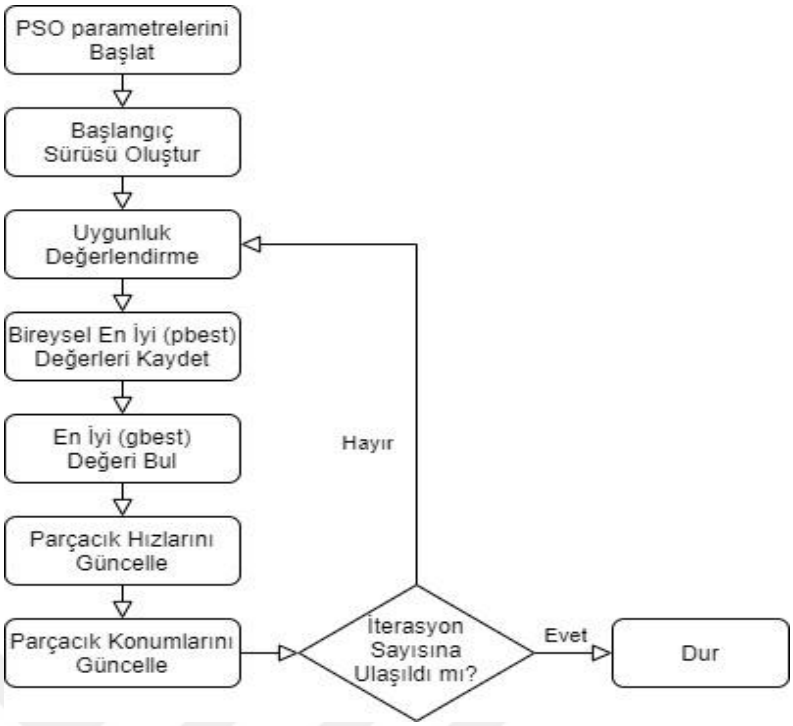
Mutasyon: Çaprazlama işlemi kromozomlar üzerinden yapılırken mutasyon işlemi genler üzerinden yapılmaktadır. Seçilen bir genin değerinin değiştirilmesi işlemine mutasyon adı verilir. Yeni oluşturulan çocuk nesillerde genler mutasyona maruz kalabilir. Bitlerden

bir veya birkaçı 1'ken 0 veya 0'ken 1 olur. Popülasyondaki çeşitliliği arttırmak ve erken yakınsamayı önlemek için gereklidir. Parça değişime benzer olarak bir mutasyon parametresine sahiptir. Şekilde K4 kromozomunun 2. Biti işaret değiştirerek mutasyona uğramıştır. Aradaki farkın görülebilmesi için K5 kromozomu olarak ifade edilmiştir. Mutasyon birey sayısını arttırmamaktadır. Mutasyon parametresi yaygın kullanımda çaprazlama parametresinin aksine düşük olasılığa sahiptir.

Hafıza sınırlamaları nedeniyle popülasyon sayısı sabittir. Yeni bireyler oluşturulan nesile eklendikçe sabit sayıda popülasyon elde edebilmek için bazılarının elenmesi gerekmektedir. Bu da kötü bireylerin doğal seçilim yoluyla elenmesi ile sağlanır. En yüksek amaç puanı olan bireyler seçilerek sonraki nesillere aktarılır. Düşük puanlı kromozomlar popülasyondan çıkartılır. Yeni nesil önceki nesilden daha iyi sonuç veremediği zaman algoritma yakınsamaya gider (Goldberg, 1989). Simülasyon ortamında kavşak programı tam sayı değerlerle çalışmaktadır. Bu çalışma için yakınsama kriteri olarak en iyi sonucun tekrar etmesi durumunda algoritmanın durdurulması kriteri kaldırılmıştır. Sabit olarak belirlenen 20 populasyon değeri örnek çalışmalar incelenerek seçilmiştir. Nesil değeri problemden probleme değişiklik gösterdiği için farklı değerler denenmiştir. Öncelikle nesil değeri 100 seçilmiştir. Ancak 40 nesilden önce algoritmaların yakınsamaya gittiği görülmüştür. Amaç fonksiyonu için karar değişkenleri 9 ile 40 saniye arası seçilmiştir.

3.1.2 Parçacık sürü optimizasyonu

Sürü halinde hareket eden canlıların davranışlarından esinlenerek bulunmuş bir algoritmadır (Kennedy ve Eberhart, 1995). Buradaki popülasyona sürü adı verilmektedir. Bireylere ise parçacık denmektedir.



Şekil 3.3 Parçacık Sürü Optimizasyonu Akış Şeması

Sürü belirli sayıda parçacığın çok boyutlu arama uzayında farklı yönlere dağılması ile çözüme ulaşmaya çalışır. Parçacıkların sonuca ne kadar yakınına ulaşabildiğini anlamak için amaç fonksiyonu burada da kullanılmaktadır. Sadece bir parçacık için kendisinin çözüme en çok yakınlaştığı duruma *pbest* (personel best) bütün sürüde en çok çözüme yakınlaşıldığı duruma ise *gbest* (global best) denilmektedir. Parçacıklar bireysel ve diğer komşu bireylerin başarılarına göre sürü içerisinde konum değiştirirler. Bu pozisyon değişimi her iterasyonda mevcut pozisyon, mevcut hız, *pbest* değeri, mevcut pozisyon ile *gbest* değeri arasındaki mesafe kullanılarak gerçekleştirilir. Burada çözüme yakınlaşılmasından kasıt problemin tanımına göre amaç fonksiyonunda en küçük veya en büyük değerin bulunmasıdır. Kullanıcı tarafından belirlenen sürü büyüklüğü, iterasyon sayısı gibi parametreler girildikten sonra amaç fonksiyonu kullanılarak parçacıkların mesafe değerleri ölçülür. Bu hesaplamaya göre *pbest* ve *gbest* değerleri güncellenir. Sonrasında değişim hızı fonksiyonu ile parçacıkların yapacağı hareketler hesaplanır ve konumları belirlenir. Bu süreç bir döngü içerisinde durdurma kriteri sağlanıncaya kadar devam eder. Parçacıkların değişim hızları:

$$V_{i+1} = \omega * V_i + c_1 * rand1 * (pbest - x_i) + c_1 * rand2 * (gbest - x_i) \quad (3.2)$$

Denklem (3.2)'de parçacık konumu *x*, parçacık hızı v, Bilişsel İvme Katsayısı c_1, Sosyal İvme Katsayısı c_2, Atalet Ağırlık Katsayısı ω olarak tanımlanmıştır. *rand1* ve *rand2* rastgele düzgün dağılımla üretilen değerlerdir. Hız güncellendikten sonra (3.3) denklemiyle konum güncellenir.

$$x_{i+1} = x_i + V_{i+1} \quad (3.3)$$

Genetik algoritma ile karşılaştırma yapılabilmesi için sürü boyutu 20 iterasyon sayısı 100 ve 40 seçilmiştir.

3.2 Çok Amaçlı Optimizasyon

Çoğu gerçek hayat probleminde birden fazla amaç değerinin eş zamanlı olarak optimize edilmesi gerekir. Çok amaçlı optimizasyon, birden fazla amaç bulunduran problemlerin optimizasyonu işlemidir.

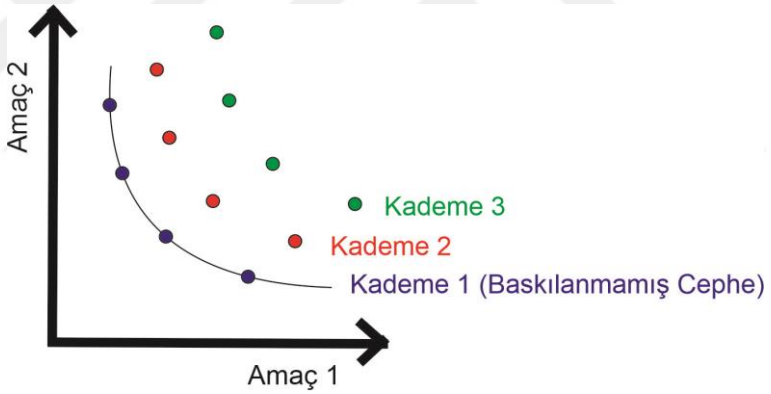
$f : R^n \rightarrow R^k$'ya giden bir vektör amaç fonksiyonu olsun,

$$minf(\bar{x}) = \left(f_1(\bar{x}), f_2(\bar{x}), \dots, f_k(\bar{x})\right)$$
$$g_j(\bar{x}) \leq 0, \quad j = 1, 2, \dots, m$$
$$h_m(\bar{x}) = 0, \quad m = 1, 2, \dots, e$$
$$\bar{x}_i^l \leq \bar{x}_i \leq \bar{x}_i^u, i = 1,2, \dots, n \quad (3.4)$$

(3.4) denkleminde *g* ve *h* eşitsizlik ve eşitlik kısıtlama fonksiyonlarıdır. $\bar{x}^l$ ve $\bar{x}^u$ ise karar değişkeninin alabileceği alt ve üst sınırları ifade etmektedir. *k* amaç fonksiyonu sayısı, *m* eşitsizlik sınırlama fonksiyon sayısını, *l* eşitlik sınırlama fonksiyonu sayısını verir. $\bar{x} \in$

R^n karar değişkenleri vektörüdür. Burada n, $\bar{x}_i$'nin karar değişken sayısıdır. Tez çalışmasında beş kavşak için yeşil ışık sürelerini kontrol edeceğimiz için n=5, iki amaç fonksiyonunu k=2 seçilmiştir. Bu amaç değerleri de sırayla Seyahat süresi ile CO_2 emisyonu, sonrasında ise CO_2 emisyonu ile Gecikme süresi alınacaktır. Çok amaçlı optimizasyon problemlerinde çözüm tek bir skaler sayı yerine bir çözüm kümesidir. Bu kümeye pareto optimal küme adı verilir. Pareto optimal için bir A noktası $\bar{x}^* \in X$ pareto optimaldir eğer en az bir fonksiyon için $\bar{x} \in X$ çözüm kümeleri içerisinde $f(\bar{x}) \leq f(\bar{x}^*)$ ve $f_i(\bar{x}) \leq f_i(\bar{x}^*)$ koşulunu sağlayan bir nokta yoksa. Bir vektör amaç fonksiyonu $f_i(\bar{x}^*) \in Z$ baskılanmamıştır eğer; $f(\bar{x}) \in Z$ amaç uzayında en az bir $f_i(\bar{x}) \leq f_i(\bar{x}^*)$ olan $f(\bar{x}) \leq f(\bar{x}^*)$ vektörü yoksa. Aksi halde $f(\bar{x})$ baskılanmıştır (Marler ve Arora, 2004).

Çok amaçlı optimizasyonda amaç değerleri vektör olacağından skaler kavramı ortadan kalkar ve pareto kümesi oluştururlar.



Şekil 3.4 Çok amaçlı optimizasyon problemi için çözüm kümesi

Çok amaçlı optimizasyon problemlerinde optimizasyon gerçekleştirirken farklı kademe çözümlerin birbirlerini baskıladığı Şekil 3.4'te görülebilmektedir. İki değeri eş zamanlı olarak minimize etmeye çalışırken amaç değerinin birinin azalması durumunda diğer amaç değeri artıyorsa bu çözüm bir önceki çözüme baskılayamıyor demektir. Şekilde Kademe 1 olarak belirtilen eğri kendini baskılayan başka bir çözüm bulunmaması sebebiyle bir sınır çizdiğinden pareto cephesi olarak ifade edilmektedir. Bu eğri üzerinde bulunan çözüm kümesi pareto kümesidir. Gerçek hayat uygulamalarında pareto cehpesine

yakınsama sağlanması arzu edilir. Şekilden anlaşılacağı üzere çok amaçlı optimizasyon problemlerinde amaç değerleri seçilirken amaç değerlerinden birinin artması durumunda diğerlerinin azalması gerekmektedir.

Çok amaçlı problemlerde her amacı eş zamanlı olarak çözen algoritmalar geliştirilmiştir. Ancak ilk olarak tek amaçlı optimizasyon algoritmaları çok amaçlı problemleri çözecek şekilde uyarlanmıştır. Bu çalışmada skalerizasyon adı verilen bir yöntemle iki amaç teke indirilecek ve tek amaçlı optimizasyon algoritmaları olan GA ve PSO ile çözülecektir. Sonrasında literatürde sıklıkla kullanılan olan NSGA-II ve MOEA/D çok amaçlı optimizasyon algoritmaları ile çözümler üretilecektir. Devamında çok amaçlı optimizasyon algoritmalarının sonuçlarını karşılaştırabilmek için metriklerden bahsedilecektir. Bu tez kapsamında boşluk metriği ve hiper hacim metriği kullanılmıştır.

3.2.1 Skalerizasyon

Çok amaçlı problemlerde amaç sayısını teke indirgeme işlemidir. Çok amaçlı problemlerin çözüm kümeleri ifade bakımından vektör oluşturmaktadır. Bu vektör içerisindeki değerler belirli yöntemlerle tek amaca indirgenerek skaler bir sayı elde edilir. Bu tez kapsamında ağırlıklı toplam yöntemi kullanılacaktır.

$$\min_{x \in X} \sum_{i=1}^{k} \omega i f i(x),\ \omega i > 0, \sum_{i=1}^{k} \omega i = 1 \tag{3.7}$$

Ağırlıklar hakkında ön bilgiye ihtiyaç duyulması sorun teşkil etmektedir. Ağırlıklar rastgele verildiği zaman veya yanlış seçilmesi durumunda optimizasyon sonuçları amaçlardan biri doğrultusunda yanlılık oluşturacaktır. (Pardalos, Zilinkas ve Zilinkas, 2017). Her amaç değeri ilk aşamada belirli bir ağırlık değeriyle çarpılarak toplanır. Tek kavşak uygulamasında ağırlıklar doğrudan 0,5 alınmış ve eşit ağırlıktaki sonuçları gözlenmiştir. Yalnız amaç değerlerinden CO_2 sayısal olarak büyük değerleri çıkarması nedeniyle eşit ağırlıkta baskın çıktığı ve algoritmanın bu sebeple yanlılık gösterdiği

görülmektedir. Ağırlıklar değiştirilerek yeniden hesaplama yoluna gidilir. Yalnız çok kavşaklı modele geçildiğinde her seferinde yeni bir ağırlıkla çarpmak problematik hale gelmiştir. Bu nedenle $\frac{(Amaç\ Değeri - Minimum\ Değer)}{(Maksimum\ Değer - Minimum\ Değer)}$ denklemine göre ifade normalize edilerek yeni ağırlıklar hesaplanır.

3.2.2 Baskılanmayanları sıralayan genetik algoritma-II

NSGA-II algoritması önceki algoritmalardan farklı olarak hızlı baskılanmayanları sıralama prosedürü, hızlı kalabalık mesafe kestirim prosedürü ve sade bir kalabalık karşılaştırma prosedürüne sahiptir (Deb, 2002). Başlangıçta geleneksel genetik operatörler ile ebeveyn popülasyondan çocuk nesil üretilir. Sonrasında iki popülasyon birleştirilir. Pareto cephesine yakınlıklarına göre çözümler kademelere ayrılırlar. Bu kademelere ayrılan noktaların sırasıyla kalabalık mesafeleri ölçülür. Sonrasında kalabalık karşılaştırma işlemi kullanılarak bireyler seçilime maruz kalırlar. Noktalar arasından kademesi yüksek olan veya eşit kademede kalabalık mesafesi düşük olan birey diğer bireyi baskılamış kabul edilir. Baskılanmamış kabul edilen bireyler sonraki iterasyonda ebeveyn olarak kullanılacaktır. Bitirme kriterine ulaşılıncaya kadar bu süreç devam eder. Algoritmadaki operatörler incelenirse:

Hızlı Baskılanmayanları Sıralama: N popülasyon boyutu M amaç sayısı için hızlı sıralama algoritması şu şekildedir:

Her çözüm için iki değer hesaplanır. Bunlar baskılanma sayısı n_p (p çözümünü baskılayan çözüm sayısı), diğeri S_p, p çözümünün baskıladığı çözüm sayısı.

İlk bulunan baskılanmamış cephedeki bütün çözümlerin baskılanma sayısı n_p =0 olacaktır. Sonrasında her n_p =0 değerine sahip p çözümü için S_p kümesindeki q adet üye ziyaret edilir ve baskılanması bir azaltılır. Buradan q'nun herhangi bir üyesi içinde baskılanma sayısı 0 olursa ayrı bir liste olan Q içine kaydedilir. Bu üyeler ikinci

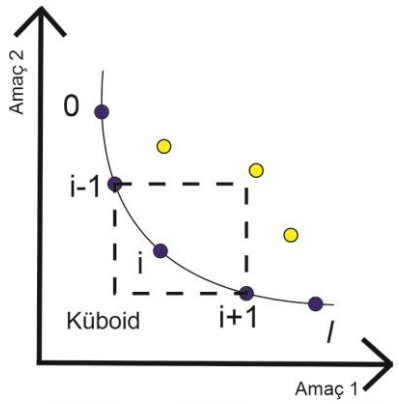
baskılanmayan cepheye aittir. Yukardaki prosedür yeniden uygulanır ve üçüncü cephe elde edilir. Bu süreç bütün cepheler belirlenene kadar devam eder. Algoritmanın sözde kodu:

Hızlı Baskılanmayanları Sıralama (P)

Her $p \in P$ için:
  $S_p = \emptyset, n_p = 0$
  Her $q \in P$ için:
    Eğer (p<q) ise — Eğer p q'yu baskılıyorsa
    $S_p = S_p \cup \{q\}$ — q'yu, p tarafından baskılanan edilen çözümlere ekle
    Değilse eğer (q<p)
    $n_p = n_p + 1$ — p'nin baskılanma sayacını bir arttır
  Eğer $n_p = 0$ ise — p ilk cepheye aittir
    $p_{rank} = 1, \quad F_1 = F_1 \cup \{q\}$
i=1 — cephe sayacını başlat
$F_i \neq \emptyset$ olduğu sürece
  $Q = \emptyset$ — sonraki cephenin üyelerini saklamak için kullanılır
  Her $p \in F_i$ için
    Her $q \in S_p$ için
      $n_q = n_q - 1$
      Eğer $n_q = 0$ ise — q sonraki cepheye aittir
        $q_{rank} = i + 1$
        $Q = Q \cup \{q\}$
$i = i + 1$
$F_i = Q$

Kalabalık Mesafe Kestirim Prosedürü: Pareto optimal kümeye yakınsamasının yanında çözümlerin iyi bir dağılıma sahip olması istenir. NSGA-II algoritmasında bu işi kalabalık karşılaştırma yaklaşımı yapmaktadır. Bu yaklaşımda yoğunluk kestirim metriği kullanılmaktadır. Bu metrikte popülasyon içerisindeki belli bir çözümü çevreleyen çözümlerin kestirimini elde etmek için bu çözüm noktasının iki tarafındaki noktalar her amaç doğrultusunca uzanan ortalama mesafesi hesaplanır. i_{mesafe} niceliği, aynı cephedeki en yakın komşu noktalar köşe olarak kullanıldığında oluşacak dikdörtgenler prizmasının/küboidin (iki amaç kullandığımız için bu çalışmada dikdörtgenin) çevresinin

kestirimi olarak kullanılacaktır. Bu dikdörtgene kalabalıklaşma mesafesi denir. Şekil 3.4'te *i*. çözüm için kesikli çizgilerle çevrelenen dikdörtgen içinde kalan çözümdür.



Şekil 3.5 Kalabalık Mesafe Hesabı

Kalabalık mesafe hesabı için popülasyondaki elemanların amaç değerlerine göre artan şekilde sıralanması gereklidir. Şekil 3.5'te bunun bir örneği görülebilir. Sonrasında her amaç fonksiyonu için sınır çözümlere (en büyük ve en küçük fonksiyon değerine sahip olan çözümler) sonsuz mesafe değeri atanır. Aradaki diğer çözümlere iki komşu çözümün fonksiyon değerlerinin mutlak normalize farkının sayısal değeri atanır. Bu hesaplama her amaç değeri için gerçekleştirilir. Bir nokta için toplam kalabalık mesafe değeri her amaç için bulunan değerlerin toplanması ile elde edilir. Her amaç değeri kalabalık mesafesi hesaplanmadan önce normalize edilir. Baskılanmayan *I* seti içerisindeki bütün çözümlerin kalabalık mesafe hesabı aşağıdaki algoritmada verilmiştir.

Kalabalık Mesafe Atanması(*I*)

$l = |I|$ — *I'* daki çözüm sayısı

Her *i* için, $I[i]_{mesafe}$ ata — mesafe başlat

Her *m* amacı için

$I = sırala(I, m)$ — amaç değerlerini kullanarak sırala

$I[i]_{mesafe}$=$I[l]_{mesafe} = \infty$ — bu sayede sınır noktaları daima seçilidir

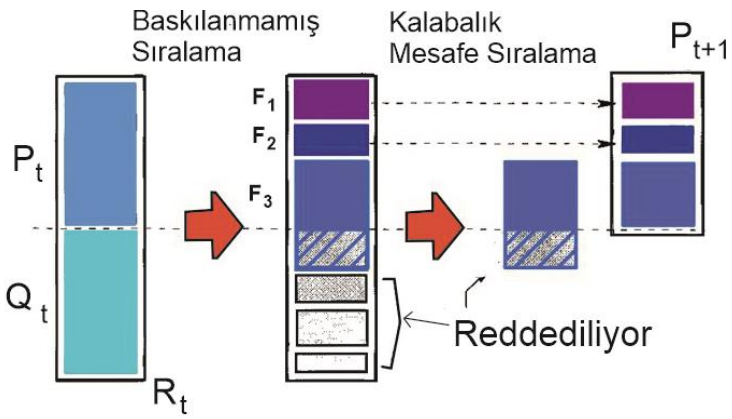
$i = 2'den(l-1)$'e kadar değerler için — kalan bütün noktalar

$I[i]_{mesafe} = I[i]_{mesafe} + (I[i+1].m - I[i-1].m)/(f_m^{maks} - f_m^{min})$

Burada $I[i].m$ değeri I kümesinin içindeki i bireyinin m. amaç değerini verir. f_m^{maks} ve f_m^{min} parametreleri de m. amaç fonksiyonunun maksimum ve minimum değerleridir. I kümesindeki bütün popülasyon üyelerine mesafe metriği atandıktan sonra iki çözümü yakınlık derecesine göre diğer çözümlerle karşılaştırabilir. Daha küçük mesafe değeri olan çözümün etrafı diğer çözümlerin yakınlığı sebebiyle daha kalabalıktır. Kalabalık karşılaştırma operatörü bu karşılaştırma işini yapmaktadır.

Kalabalık karşılaştırma operatörü **($<_n$)** algoritmanın çeşitli aşamalarında görev alarak pareto optimal cephe içerisindeki çözümlerin düzgün dağılımla dağılmasını sağlar. Popülasyon içindeki her i bireyinin baskılanmama kademesi (i_{kademe}) ve kalabalık mesafesi (i_{mesafe}) değeri bulunmaktadır. Burada:

Eğer ($i_{kademe} < j_{kademe}$) veya (($i_{kademe} = j_{kademe}$) ve ($i_{mesafe} > j_{mesafe}$) şartlarından biri sağlanıyorsa $i <_n j$ şeklinde iki birey sıralanır. Burada baskılanmama kademesi düşük olan birey veya aynı kademede yer alan bireylerden daha düşük kalabalık değerine sahip çözümler tercih edilir.



Şekil 3.6 NSGA-II Prosedürü (Deb, 2002)

Ana döngü içerisinde başlangıçta rastgele popülasyon P_0 üretilir. Baskılanma değerlerine göre popülasyon sıralanır. Her çözümün baskılanma seviyesine ve amaç değerine göre kademe değeri atanır (1 en iyi seviye, 2 sonraki en iyi seviye vb.). Bu sayede amaç

değerleri kademelere göre sıralanmıştır ve en küçük kademe en iyi çözüm kümesini vermiş olur. Algoritma başlangıç aşamasından sonra *N* boyutunda Q_0 nesil popülasyonun üretilmesi için geleneksel ikili turnuva, kombinasyon ve mutasyon operatörleri kullanılır. Algoritmanın performanısını arttırmak ve en iyi bireyleri bir sonraki nesilde hayatta tutmak için mevcut popülasyonla önceki en iyi baskılanmayan çözümleri karşılaştırarak elitist operatör çalıştırılır. İlk kombine popülasyon $R_t = P_t \cup Q_t$ oluşturulur. R_t popülasyonu 2*N* boyutundadır. Bütün önceki ve mevcut popülasyon üyeleri R_t içerisine dahil edilir. En iyi baskılanmayan küme olan F_1'e ait olan çözümler birleşik popülasyon içerisindeki en iyi çözümlerdir ve birleşik popülasyonda daha çok ön plana çıkarılır. Eğer F_1 boyutu *N*'den küçükse P_{t+1} yeni popülasyonu için F_1 kümesinin üyelerinin tamamı seçilmelidir. P_{t+1} popülasyonunun kalan üyeleri baskılanmayan bireylerin tutulduğu küme içerisinden kademelerine göre seçilmelidir. Sonrasında F_2 kümesindeki çözümler ve bunu takiben F_3 kümesindeki çözümler seçilirler. Bu prosedür daha fazla küme yerleştirilemeyene kadar devam eder. Genelde F_1'den F_t'ye kadar olan çözüm sayısı popülasyon boyutundan daha büyük olur. *N* popülasyon üyesini tam olarak seçmek için son F_t kümesindeki çözümleri kalabalık karşılaştırma operatörü kullanarak azalan biçimde sıralarız ve bütün popülasyon yuvalarını dolduracak gerekli çözümler seçilir ve yeni nesil olarak hayatta kalır.

$R_t = P_t \cup Q_t$	ebeveyn ve çocuk popülasyonların birleşimi
$F = hızlı\ baskılanmayanları\ sıralama(R_t)$	$F=(F_1,F_2,\ldots)$ R_t'nin bütün baskılanmayan cepheleri
$P_{t+1} = \emptyset$ ve i=1	
$\|P_{t+1}\| + \|F_i\| \leq N$'e kadar	ebeveyn popülasyonu doldurulana kadar
Kalabalık mesafe atanması (F_i)	F_i'deki kabalık mesafe hesapla
$P_{t+1} = P_{t+1} \cup F_i$	Ebeveyn popülasyonda i. baskılanmayan cepheyi ekle
$i = i + 1$	eklemek için sonraki cepheyi kontrol et
$Sırala(Fi, <_n)$	$<_n$ kullanarak azalan biçimde sırala
$P_{t+1} = P_{t+1} \cup F_i[1:(N - \|P_{t+1}\|)]$	F_i öğelerinden ilk $(N - \|P_{t+1}\|)$ seç
Q_{t+1} =yeni-pop-yap(P_{t+1})	seçilim, çaprazlama ve mutasyon kullanarak yeni pop. Üret
$t=t+1$	nesil sayacını arttır

N büyüklüğündeki P_{t+1} yeni popülasyonu seçilim, çaprazlama ve mutasyon için kullanılır ve N büyüklüğündeki Q_{t+1} yeni popülasyon oluşturulur. İkili turnuva, kalabalık karşılaştırma operatörü kullanılarak gerçekleştirilir. Bu operatör kademe ve kalabalık mesafesine her çözüm için ihtiyaç duyduğundan P_{t+1} oluşturulurken bu nicelikler hesaplanır. Ayrıca P_{t+1} için N üye bulunması ile sıralama prosedürü sonlandırılır. Bu sayede $2N$ büyüklüğündeki bütün popülasyonun değerlendirilmesine gerek kalmaz.

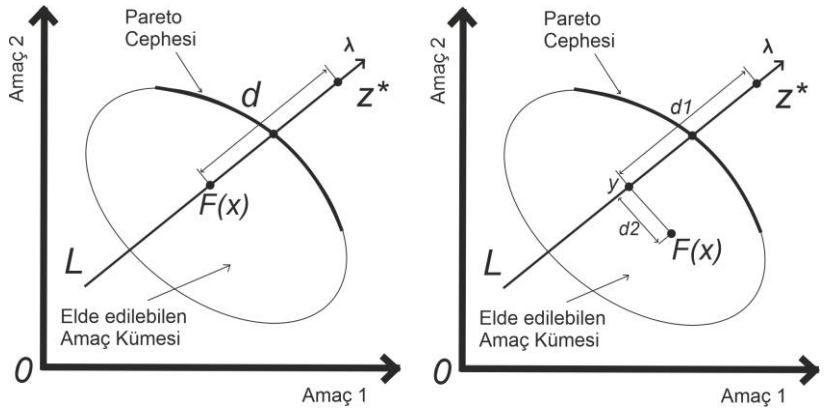
3.2.3 Ayrıştırmalı çok amaçlı evrimsel algoritma

Ayrıştırmalı Çok Amaçlı Evrimsel Algoritma adı verilen bu yöntemle ayrıştırma (decomposition) sayesinde problem küçük alt problemlere bölünür ve yöntem bu problemleri eş zamanlı olarak çözer (Q. Zhang, H. Li 2007). Ayrıştırma işlemi ağırlıklı toplam ve Tchebycheff metodu ile sağlanabilir. Program girdi olarak; Çok Amaçlı Problem, Durdurma Kriteri, N alt problem sayısı, düzgün dağılmış N ağırlık vektörleri $\lambda^1,\ldots,\lambda^N$ ve her ağırlık vektörü komşuluğundaki T ağırlık vektörleri sayısı değerlerini alır. Çıktı olarak EP (harici popülasyon) adı verilen algoritmanın çalıştırılması ile bulunan baskılanmayan çözümleri verir.

Alt problemlere bölme işlemi 3 yöntemle yapılabilir. Bunlar Ağırlıklı toplam, Tchebycheff yöntemi ve Sınır Kesişim Yaklaşımıdır. Tezde kullandığımız yöntem cezalı sınır kesişimi yöntemi (PBI)'dır. Matematiksel olarak skaler optimizasyon alt problemi:

$$\begin{aligned} &\min g^{bi}(x|\lambda, z^*) = d \\ &z^* - F(x) = d\lambda\,, x \in \Omega \text{ olduğunda} \end{aligned} \tag{3.8}$$

(3.8) denkleminde λ ağırlık vektörü ve z^* referans noktasıdır.



Şekil 3.7 Sınır Kesişim Yaklaşımı ve Ceza Tabanlı Versiyonu

Şekil 3.7'de $z^* - F(x) = d\lambda$ sınırlaması $F(x)$'in daima L (λ doğrultusunda z^*üzerinden geçen çizgi) içinde olmasını sağlar. Amaç $F(x)$'i elde edilebilir amaç kümesi sınırına ulaşıncaya kadar dışbükey pareto cephesi için yükseğe itmektir. Bu tez çalışmasında ceza (penalty) yöntemi kullanılmıştır. Yeni ifadede:

$$d_1 = \frac{\|(z^* - F(x))^T \lambda\|}{\|\lambda\|} \text{ ve } d_2 = \|F(x) - (z^* - d_1\lambda)\| \tag{3.9}$$

$$\text{Min } g^{bip}(x|\lambda, z^*) = d_1 + \theta d_2, x \in \Omega \tag{3.10}$$

Denklem (3.9) olduğunda (3.10) ifadesi geçerlidir. $\theta > 0$ mevcut ceza parametresindeyken *y*, *L* hattına $F(x)$'in iz düşümüdür. Şekil 3.5'de görüldüğü üzere d_1; z^* ve *y* arasındaki mesafedir. d_1; *L* ve $F(x)$ arasındaki mesafedir. Algoritmada ayrıştırma sağlandıktan sonra adımları şu şekildedir:

Çıktı olarak kullanacağımız *EP* harici popülasyon kümesi boş küme atanır. Ağırlık vektörlerinin ikili olarak öklit mesafeleri ve *T* komşulukları hesaplanır. En yakın komşuların indisleri *B* hücresine atanır. *N* adet *x* değeri başlangıç popülasyonu olarak rastgele atanır. $FV^i=F(x^i)$ fonksiyon değeri atanır yani başlangıç popülasyonu amaç fonksiyonuna göre değerlendirilmeye tabi tutulurarak değerleri kaydedilir. *z* referans

noktası önceki bölümde bahsedilen metodlardan biri ile ideal veya ütopik bölge olarak tabir edilen alan içerisinde başlatılır.

Sonrasında bütün alt problemler için oluşturulan *B* içerisinden rastgele iki çözüm seçilir ve genetik operatörler kullanılarak yeni çözüm oluşturulur. Karar uzayına denk gelmeyen kısımlar onarma operatörü kullanılarak çıkarılır ve yerine karar uzayı içerisinden bir nokta atanır. Oluşan çözüm kümesi *y* içerisinde *z* değerlerini baskılayan ifade varsa bunun yerine atanır. Oluşturulan kümedeki iyi çözümler böylelikle seçilmiş olur. İterasyon bittiği zaman baskılanmamış çözümler *EP* içerisine kaydedilir.

Adım 1 ***Başlangıç*****:**

- $EP = \emptyset$ atanır.
- Her ikili ağırlık vektörleri arasında öklit mesafesi hesaplanır ve her vektöre en yakın *T* sayıda ağırlık vektörü çıkartılır. Her $i = 1, \dots, N$ kümesi için $\lambda^{i_1}, \dots, \lambda^{i_T}$'de T λ^i'nin en yakın ağırlıkları olduğunda $B(i) = \{i_1, \dots, i_t\}$'dir.
- Başlangıç popülasyonu $x^1, \dots, x^N$ rastgele veya belirlenmiş bir yöntemle atanır. $FV^i = F(x^i)$ atanır.
- Probleme özgü metodla $z=(z_1, \dots, z_m)^T$ başlatılır.

Adım 2 ***Güncelleme*****:**

i=1'den N'e kadar döngüde

- Çoğalma: *B(i)* içinden rastgele *k* ve *l* üst indislerini seç ve x^k ve x^l'den genetik operatörleri kullanarak yeni çözüm üret
- Geliştirme: y' elde etmek için problem özelinde onarma/geliştirme sezgisel yöntemini *y* 'ye uygula
- *z*'nin güncellenmesi: *j*=1'den *m*'ye her $z_j < f_j(y')$ ise $z_j = f_j(y')$ atanır
- Komşu çözümlerin güncellenmesi: her indis $j \in B(i)$ için eğer $g^{te}(y'|\lambda^j,z) \leq g^{te}(x^j|\lambda^j,z)$ ise; $x^j = y'$ ata ve $FV^j = F(y')$ ata.
- *EP*'nin güncellenmesi: *F(y')* tarafından baskılanan bütün vektörleri *EP*'den çıkar. *EP* içinde herhangi bir vektör *F(y')*'yi baskılamıyorsa *F(y')* ekle

Adım 3 ***Durdurma Kriteri*****:** Eğer durdurma kriteri sağlanırsa dur, sağlanmazsa Adım 2'ye git

3.2.4 Boşluk metriği

Boşluk (S) metriği pareto cephesi olarak bilinen komşu vektörlerin mesafe varyanslarını ölçebilmek için önerilmiştir.

$$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\overline{d} - d_i)^2}, \tag{3.11}$$

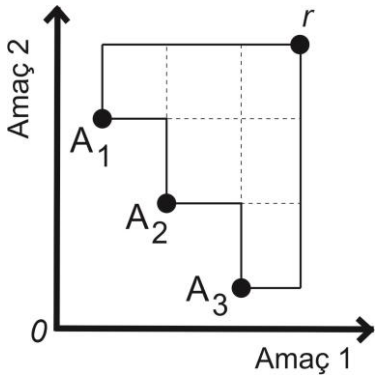
Boşluk (3.12) denklemindeki biçimde tanımlanmıştır. Burada:

$$d_i = min_j\left(\left|f_1^i(\vec{x}) - f_1^j(\vec{x})\right| + \left|f_2^i(\vec{x}) - f_2^j(\vec{x})\right|\right), i, j = 1, \dots, n \tag{3.12}$$

İse $\overline{d}$ tüm d_i mesafelerinin ortalamasını, n ise algoritma tarafından pareto cephesinde bulunan vektör sayısını ifade etmektedir. Bu metrikte S değerinin sıfır olması bütün baskılanmamış çözümlerin eşit aralıklarla dağıldığını gösterir (Coello Coello ve Cruz, 2005).

3.2.5 Hiper hacim metriği

Çok amaçlı bir optimizasyon algoritmasının sağladığı sonuçların pareto cephesine ne kadar yakınsadığının bilinmesi gerekir. Bunun için kullanılan metriklerden biri hiper hacimdir. Bir *S* pareto kümesinin hiper hacimi, çözüm kümesi tarafından baskılanan alan içerisinde seçilen bir referans noktası ile pareto kümesinin arasında kalan alandır. Bu referans noktası genelde uzaydaki en kötü noktadır. (Eğer uzaydaki en kötü nokta bilinmiyorsa her amaç değerlendirilerek en kötü değerleri referans noktası olarak seçilir). Eğer bir *S* kümesi diğer bir S' kümesinden daha büyük hiper hacime sahipse pareto cephesine S, S' kümesinden daha yakındır demektir. (While, Bradstreet ve Barone, 2012).

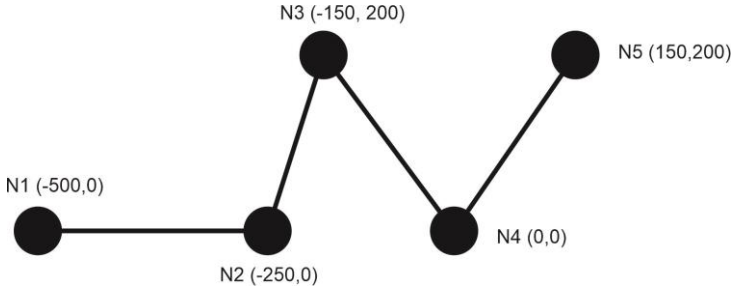


Şekil 3.8 Hiper Hacim Hesabı

Bu çalışmada 2 boyutlu olarak kullanılmıştır. Şekil 3.8'de hesap ile ilgili görsel mevcuttur. *r* referans noktası ile A_k noktalarını arasında kalan alanın toplamıdır.

3.3 Şehiriçi Hareketlilik Simülasyonu (SUMO)

SUMO yazılımı içerisinde oluşturulan trafik ağında; yol, şerit sayısı ve kavşak tanımlamaları yapabilmektedir. Bunun yanında hususi ve ticari araç tiplerinin yanında acil durum araç tipleri de tanımlanabilmektedir. Trafik akışı elle veya rastgele olarak oluşturulabilir. Trafik ışıklarının faz süreleri ve faz düzenleri değiştirilebilmektedir. SUMO, Matlab ve Phyton gibi yazılımlarla uyumlu çalışabilmekte ve farklı algoritmaları çalıştırılabilmektedir (D. Krajzewicz, 2011). SUMO Phyton yazılımı ile çalışmaktadır. Yolun uzunluk ve bağlanma biçimi gibi geometrik özellikleri, kavşaklar, yoldaki hız sınırları ve hangi tipte araçların kullanacağı trafik ağı çerisinde tanımlanmaktadır. Sonrasında rota belirlemeleri yapılır. Oluşturulan rota planlarına göre simülasyon çalışmaya hazır hale gelir. SUMO içerisinde yol ağı 3 şekilde üretilebilir. Bunlar El ile, Harici kaynaktan indirme ile ve NetGenerate komutu ile mümkündür (SUMO, 2019).



Şekil 3.9 Düzlemde El ile Trafik Ağı oluşturulması

Şekil 3.9'a göre N1~N5 kavşak noktaları düğüm (node) olarak tanımlanır. İki düğüm arasındaki yol ise kenar (edge) olarak tanımlanır. El ile ağ oluşturlurken öncelikle kavşakları temsil edecek olan düğüm noktaları oluşturulur. Kavşaklarda öncelik ve trafik ışığı tanımlaması yapılabilir. Her bir düğüm noktası için farklı ID (kimlik) tanımlamaları yapılmakdadır. Sonrasında simülasyonda hangi koordinatlara bu düğümlerin geleceği tek tek tanımlanır. SUMO'da birimsiz olarak kullanılan mesafeler metreyi ifade etmektedir. Sonrasında ise tip (type) tanımlamaları yapılır. Tip tanımlamalarında yol öncelikleri (en çok tercih edilen yollar), şerit sayıları, hız sınırları ve izin verilen araç tipleri yer alır. Örnek olarak 3L45 etiketi 3 şeritli ve maksimum hızı 45 m/s olan yola aittir. Kenar tanımları kavşaklara bağlı olarak oluşturulur. Gerçek hayat uygulamalarının Km/Sa olarak tanımlandığı unutulmamalı ve gerekli dönüşümler yapılmalıdır. "From … to…" kalıbı yolun nereden başlayıp nerede biteceğini tanımlamaktadır. İki yönlü akan trafik koşullarında aşağıdaki gibi uygulama yapılmalıdır.

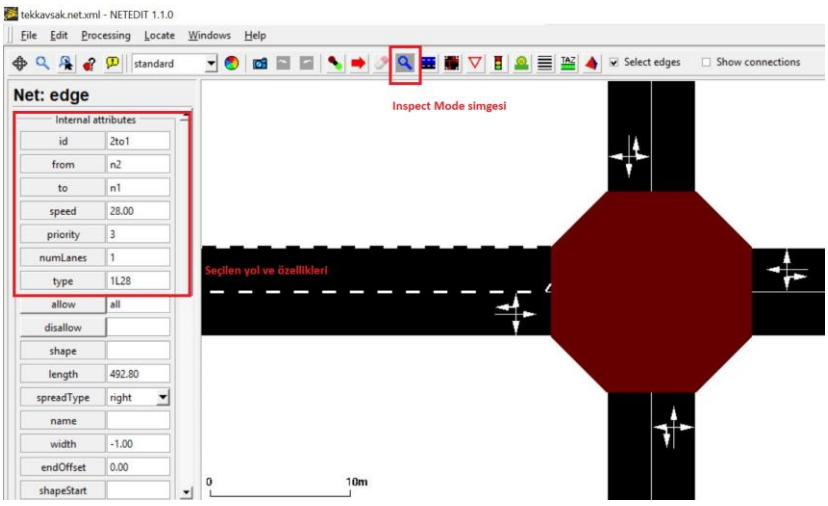
<edge from="n1" to="n2" id="1to2" type="3L45"/>
<edge from="n2" to="n1" id="2to1" type="3L45"/>

Oluşturulan düğüm, kenar ve tip dosyaları aynı klasöre atıldıktan sonra adres çubuğunda CMD komutuyla Command Prompt çalıştırılır. "netconvert --node-files my_nodes.nod.xml -- edge-files my_edge.edg.xml –t my_type.type.xml -o my_net.net.xml" koduyla trafik ağı dosyası elde edilir. NETCONVERT komutu oluşturulan 3 dosya olan "my_nodes.nod.xml", "my_edge.edg.xml" ve "my_type.type.xml" dosyalarını girdi olarak almakta ve "-o" output ifadesinden sonra ismini verdiğimiz "my_net.net.xml" dosyasına dönüştürmektedir. Bu işlemle beraber

simülasyonun statik kısımları tamamlanmış ve fiziksel yapı oluşturulur. Sonraki aşamada rota tanımlama dosyası oluşturulur. Rota tanımlamaları simülasyonun dinamik kısmını oluşturur. Dinamik ile statik yapı birleştirilerek konfigürasyon dosyası ".sumocfg" oluşturulur. Rota tanımlamasında aracın hızlanma ve frenleme ivmeleri, tip ID'leri, uzunlukları ve maksimum hızları girilir. Ayrıca rota SUMO/tools içerisinde bulunan python build-in kodu randomTrips.py kullanılarak eklenir: "python PATH\randomTrips.py -n test.net.xml –r test.rou.xml -e 50 –l" veya "py PATH\randomTrips.py -n test.net.xml -r test.rou.xml -e 50 –l" burada net.xml olarak isimlendirilen dosya statik yapıyı, rou.xml olarak isimlendirilen dosya da oluşturulan rota tanımlama dosyasını göstermekte ve dinamik yapıyı oluşturmaktadır. Örnek için araç sayısı 50 seçilmiştir. SUMO konfigürasyon dosyasında başlangıç ve bitiş zamanları saniye cinsinden belirlenebilir.

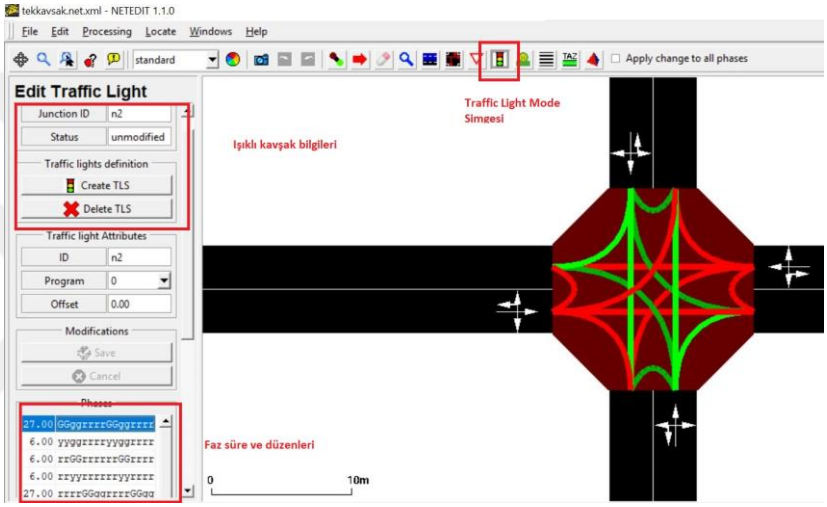
3.3.1 NETEDIT uygulamasının kullanılması

SUMO yazılımında bulunan NETEDIT veya NETEDITOR programı sayesinde mevcut trafik ağı içerisinde düzenleme yapabiliyorken yabancı kaynaklardan gelen trafik ağları üzerinde de bilgi okuma ve değişiklik yapılabilmektedir.



Şekil 3.10 Netedit ekranı yol öğesi özellikleri

Şekil 3.10'da File -> Open Network komutu ile karşımıza getirilen tek kavşak uygulaması görüntülenmektedir. Inspect mode seçildiği zaman ağ içerisindeki herhangi bir öğe seçildiğinde ID, güzergah ve maksimum hız gibi özellikleri incelenebilmektedir. Bu özellik yabancı kaynaklardan uyarlanan trafik ağları için kullanışlıdır. Bunun yanında harici uygulama ara yüzü TraCI içerisinde kullanılacak yol ve kavşakların ID'leri sıklıkla lazım olmaktadır. Inspect mode içerisinde trafik ışığı seçildiği zaman ID olarak görülen n2 düğümü üzerinde tanımladığımız trafik ışığı tip kısmında çıkmaktadır. Ayrıca harita üzerindeki konumu da görülebilmektedir.



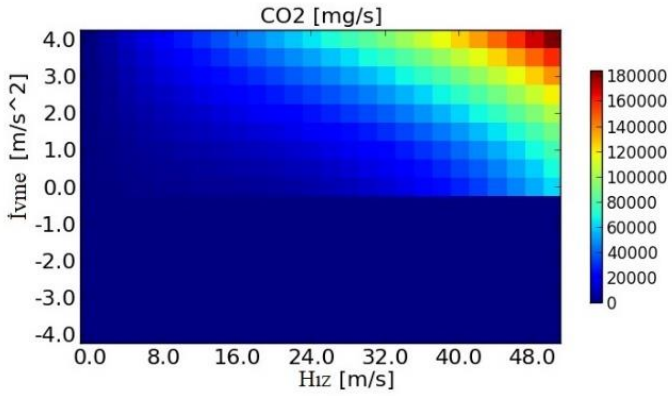
Şekil 3.11 Netedit ekranı Trafik Işık Modu

Şekil 3.11'de trafik ışıkları için tanımlanmış faz süresi ve bu fazların hangi sırayla çalışacağını gösteren faz düzenleri için traffic light mode kullanılmaktadır. Edit sekmesi altındaki Traffic Light Mode yanında ekrandaki trafik ışık sembolü tıklanarak da bu özelliğe girilebilir. Tanımlanmamış bölgeler için TLS olarak belirtilen trafik ışık sinyalizasyonu (Traffic Light Signalling) ekleme veya silme yapılabilir. Phases bölümünden kaç adet ışık fazı tanımlı ise düzeni ve süresi okunup düzenlenebilmektedir. Simülasyonun karar verdiği faz düzenleri programlar içerisinde uyumsuzluk yaratmaması adına olduğu gibi alınmıştır. Bu çalışmada kullanılan tek kavşaklı ağda faz düzeni: "27 Saniye GGggsrrrGGggsrrr; 6 saniye yyggsrrryyggsrrr; 6 saniye srGGsrrrsrGGsrrr; 6

saniye sryysrrrsryysrrr; 27 saniye srrrGGggsrrrGGgg; 6 saniye srrryyggsrrryygg; 6 saniye srrrsrGGsrrrsrGG; 6 saniye srrrsryysrrrsryy" şeklinde belirlenmiştir.

3.3.2 Araçların emisyon sınıflandırılması

SUMO yazılımı Euro 4 emisyon modeli kullanmaktadır (SUMO, 2019). Ülkemizde 2015 yılı itibariyle Euro5 ve Euro6 motorlar zorunlu hale getirilmiştir. Yalnız mevcut araçlar trafikten çekilmediğinden daha kötü emisyon değerlerine sahip araçların hala trafikte olduğu bilinmektedir. Euro emisyon standartları müsaade edilen minimum emisyon oranlarını belirtmektedir. Trafikteki her araç farklı miktarda model yılı ve motor verimine göre emisyona sahip olacağından değişiklilik arz etmektedir. Bu sebeple ortalama bir değer kullanmak makul olacaktır. Simülasyonda kullanılan emisyon modelinde ivme ve hızın emisyona etkisi Şekil 3.12'de verilmiştir.



Şekil 3.12 Hız ve ivmelenmeye göre birim zamandaki emisyon miktarı (SUMO, 2019)

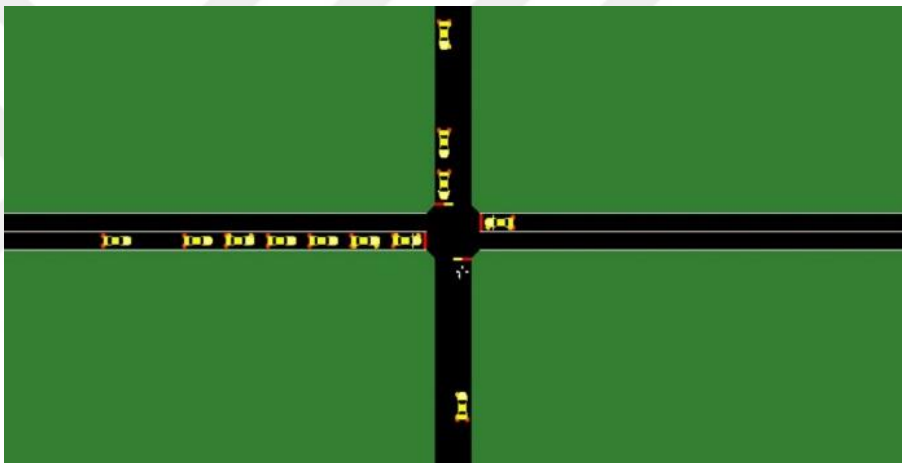
3.3.3 TraCI arayüzü

Matlab içerisinden SUMO yazılımına ulaşmak için çalışan bir trafik ağı gereklidir (Acosta,2019). TraCI 13 adet SUMO nesnesine erişebilmekte ve GET/SET komutları ile bunları okuyup değiştirebilmektedir. Bir döngü içerisinde, belirli bir döngü adımı (saniye)

veya belirli bir araç sayısına ulaşıncaya kadar simülasyon çalışır. Bu çalışmada önce 60 dakika karşılığı olan 3600 saniye için sonrasında 30 dakika karşılığı olan 1800 saniye için döngüler ayarlanmıştır. Amaç fonksiyonu olarak kişi başına düşen ortalama gecikme süresi, kişi başı ortalama CO_2 emisyonu ve kişi başı ortalama seyahat süresi alınmıştır. Trafik modeli yoğun trafik akışında çalıştırılacaktır. Buradaki yoğunluk ışıklarda bekleme ve süre bittiğinde ağ içerisinde bütün araçların çıkış yapamamasıdır. Tek ve çok amaçlı optimizasyon algoritmaları ışık sürelerini kontrol etmek için kullanılacaktır. Bahsedilen ortalama amaç değerlerini hesaplamak için sisteme yüklenen gerçek araç sayısı izlenmelidir. Ayrıca trafik ağı yoğunluğu nedeniyle yüklenemeyen araç sayıları da göz önünde bulundurulabilir. Yüklenen toplam araç sayıları için traci.simulation.getDepartedNumber(); kodu bir simülasyon adımında belirli bir noktadan ayrılan araç sayısını vermektedir. Döngü içerisinde toplanarak simülasyonun çalışma süresinde toplamda ne kadar aracın ağı kullandığı öğrenilmektedir. Toplam CO_2, seyahat süresi ve bekleme süresinin araç sayısına bölünmesiyle yeni parametreler elde edilecektir. Döngü adımlarında her bir yolda mevcut araç sayıları alınmış ve toplanmıştır. Bu toplamın birimi $araç * saniye$'dir. Bu değer trafik ağına yüklenen araç sayısına bölündüğünde araç başına düşen ortalama seyahat süresini bulmuş oluruz. İlk aşamada fonksiyonun çıktısı olarak bu değer seçilir. traci.edge.getCO2Emission('1to2'): her bir döngü adımındaki CO_2 emisyonları bu kod vasıtasıyla okunabilir. 1to2 olarak tanımlanan parantez içindeki ifade belirli bir yolun ID'sini belirtmektedir. Sistemde tanımladığımız her yol ayrı ayrı düzenlendikten sonra döngü adımlarındaki emisyonlar toplanarak toplam CO_2 emisyonu elde edilir. traci.edge.getLastStepHaltingNumber('1to2'): Sistemde belirli bir yolda bir döngü adımında kaç aracın durduğu bilgisini verir. Gecikme hesabında bu kod kullanılır. traci.edge.getLastStepVehicleNumber('1to2'): Sistemde ID'si verilen yol için bir döngü adımında kaç aracın mevcut olduğu bilgisini verir. Toplam değerler yüklenen araca bölündüğünde amaç değerinin karşılığı fonksiyonun çıktısı olarak okunur. traci.trafficlights.setRedYellowGreenState('n2', 'rrrrrGGGrrrrrrrGGGrr'): n2 ID'ye sahip bir yol için ışık süresini belirlemeye yarar. Optimizasyon algoritmalarının rastgele oluşturduğu süre bilgileri bu kod sayesinde kavşaklarda tanımlanır ve amaç değerleri hesaplanır.

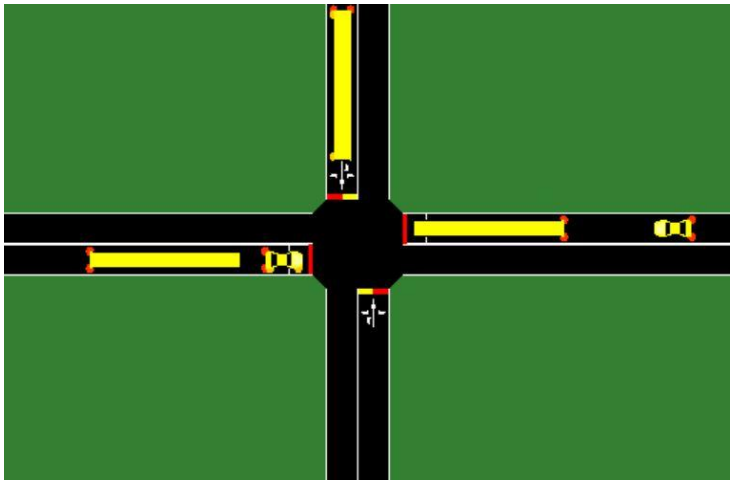
3.3.4 Tek kavşaklı trafik ağ bilgileri

Amaç olarak seçilen emisyon, bekleme ve seyahat süresi fonksiyonlarının araç tiplerine göre değişimini gözlemlemek için Webster'in (1958) çalışmalarında model aldığı 1 şerit geliş ve gidişe sahip 4 yolun birleşiminden oluşan tek bir kavşak incelenmiştir. Bütün yönlere düzgün dağılımlı olarak rastgele akan trafik ile bir trafik ağı oluşturulmuştur. Başlangıçta sadece otomobil sınıfı araçlar benzetim ortamında tanımlanmıştır. Yol uzunlukları kol başına 500m seçildi. Araçlar karşı sağ ve sola olacak şekilde 3 yola eşit ihtimalle gidebilmektedir. 3960 adet araç sisteme düzgün dağılımlı olarak rastgele girmektedir.



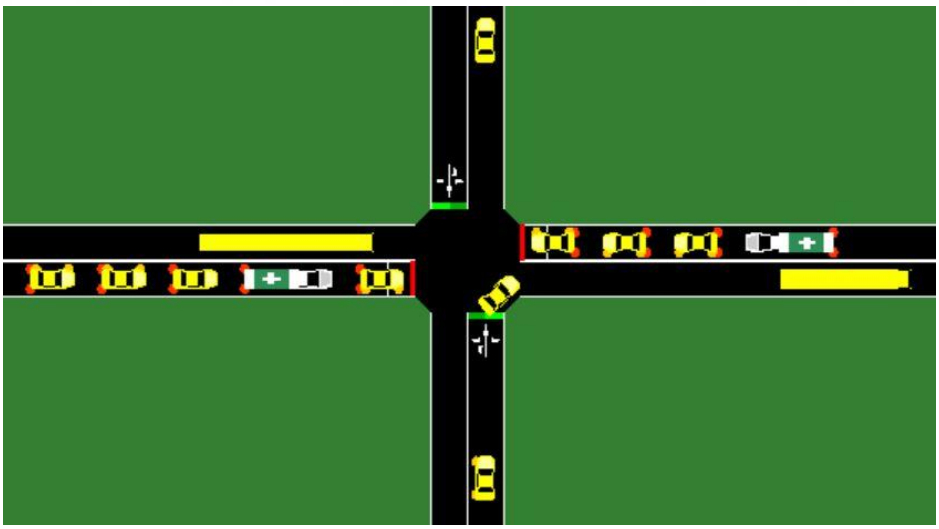
Şekil 3.13 Tek Kavşaklı Ağ için SUMO ekran görüntüsü

Şekil 3.13'te çalışma görüntüsü verilen ağda 1000 m uzunluğundaki iki yolun kesişiminden oluşan kavşak vardır. Otomobil sınıfı için yasal şehir içi hız sınırı olan 50 km/sa belirlenmiştir. İvmelenme olarak saniye karede 3 metre değer belirlenmiş ve araç uzunluğu 4 metre seçilmiştir.



Şekil 3.14 Tek Kavşaklı Ağ için otobüs eklendiğinde SUMO ekran görüntüsü

Rastgele rota oluşturacak şekilde trafik ağına giren araçların yarısı otobüs olacak şekilde yeniden düzenlendi. Şekil 3.14'te görüldüğü üzere otobüs uzunluğu için de görsel olarak fark edilebilmesi açısından 16 m uzunluk alınmıştır. Hız sınırı şehir içi trafiğini yansıttığı için aynı seçilmiştir. İvmelenmeleri daha az olduğu için 2 m/s^2 seçilmiştir. Araçların toplamı yine 3960 seçilmiştir.



Şekil 3.15 Tek Kavşaklı Ağ için farklı araç tipleri ve ambulans dahil edildiğinde SUMO ekran görüntüsü

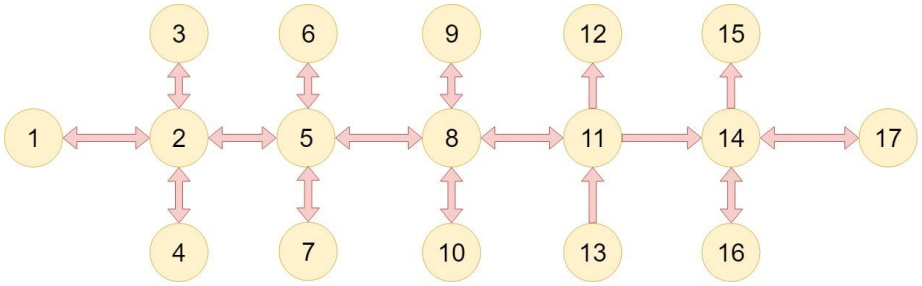
Şekil 3.15'te trafik ağı içerisinde kamyon ve acil durum (ambulans) tanımlandığında alınan ekran görüntüsü yukarıdaki gibidir. Burada her on araçtan biri acil durum aracı, diğer araçlar da %30 olacak şekilde trafik akış rejimleri ayarlanmıştır. Ambulans için uzunluk 8m, ivmelenme 3 m/s, maksimum hızı da 100 km/sa olarak belirlenmiştir. Kamyon tanımlaması için 12m uzunluk, 1m/s ivmelenme ve 30 km/sa hız sınırı belirlenmiştir. Toplam araç sayısı 3960'tır.

Trafik ağı içerisinde otobüslerin trafiği yavaşlattığını buna karşın da çok yolcu taşıyarak toplam araç sayısını azaltmak suretiyle trafiği rahatlattığı söylenebilir. Ancak sağlıklı veriler elde edebilmek için gerçek trafik koşullarına yakın bir trafik ağı uygulamak gerekmektedir. Türkiye'deki toplam araçlar Otomobil %54,3; Kamyon ve kamyonet %20.1; Minibüs ve otobüs %3,2 oranındadır (TÜİK, 2019). Şehir içi trafikte traktör olmadığı ve trafik ışıklarından motorsiklet tipi araçlar etkilenmediği için uzayın dışına alınmıştır. Yeni duruma göre sayıları oranlarsak: Otomobil %70; Kamyon %26; Otobüs %4'tür. Kurduğumuz trafik ağı içerisinde bu oranlar eşit ve %30'dur ve bunlardan hariç %10'luk bir dilimde acil durum araçları yer almaktadır. Çalışmada 3960 araç için rota tanımlanmıştır. Seçilen yolda yapılan gözlemler bu miktarda kamyon bulunmadığını göstermektedir. Ayrıca otobüs miktarı da %4'ten seçilen yolun otobüs güzergahı olması sebebiyle fazladır. Trafik ışıklarının artan trafik talebine adapte olabilmesi için kontrol edilmesi gerekmektedir. Webster (1958) düzeltme katsayılarına göre kullanılarak trafik yoğunluğu yeniden düzenlenecektir. Şehir içine kamyon girmemekle beraber %19,5'lik bir otomobil artışına, otobüsler ise %5'lik bir artışa denk gelmektedir. Sonraki çalışmalarda toplamda trafik yoğunluğu en az %25 arttırılmalıdır.

3.3.5 Çok kavşaklı trafik ağ bilgileri

Türkiye'nin en büyük iki ilçesi olması sebebiyle Keçiören ve Çankaya arasında bir güzergâh seçilmiştir. Seçilen güzergahın büyük bölümünü Atatürk Bulvarı oluşturmaktadır. Ayrıca güzergaha eski İstanbul Yolu, Eskişehir Yolu ve Konya Yolunun

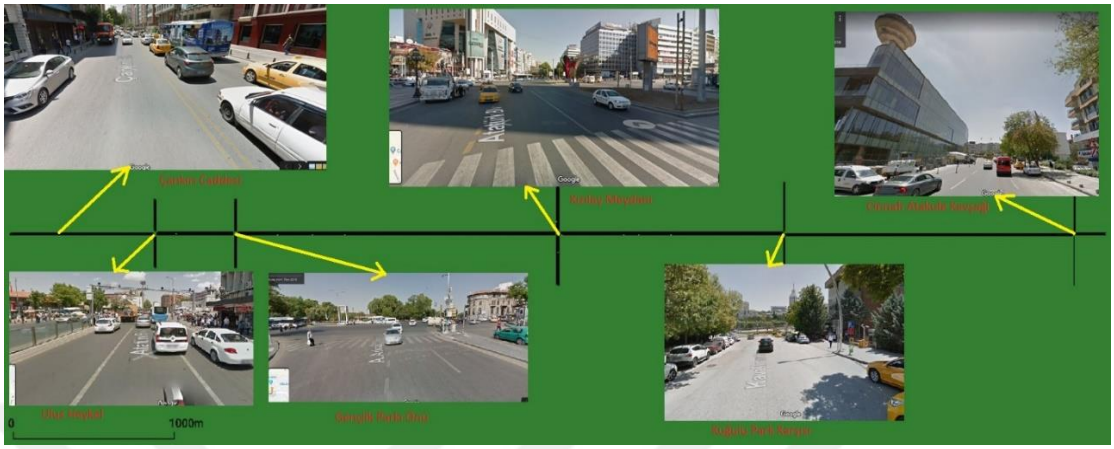
bağlandığı görülmekte ve Ankara'nın ulaşım merkezi olan Kızılay semtini de içine almaktadır. Yol boyutları için Google maps uygulamasında ölçülen değerler alınmıştır.



Şekil 3.16 Çok kavşak uygulaması için kavşak numaraları ve yol kuralları

Şekil 3.16 ve 3.17'de temsil edilen ağda 1 nolu düğümde temsil edilen nokta Ulus semtinde yer alan Çankırı caddesi üzerinde bir noktadır. İki şerit geliş ve gidişe sahiptir. Şeridin bir kısmının araçların park etmesi sebebiyle şerit meşgul edildiği Şekil 3.17'de görülebilmektedir. Yol yer yer 3 şeride çıksa da sürekli araçlar durakladığı için 2 şeritli kabul edilecektir. Sistemde 2 nolu düğüm Ulus Heykeli önündeki kavşağı temsil etmektedir. Atatürk bulvarı hariç kolları 2 şerit alınmıştır. Sistemde 5 Nolu düğüm gençlik parkı önündeki kavşağı temsil etmektedir ve 3 şeritli 4 yolun birleşmesinden meydana gelir. Sistemde 8 Nolu kavşak Kızılay AVM önündeki kavşaktır ve 3 şeritli 4 yolun birleşmesi şeklinde temsil edilmiştir. Sistemde 11 Nolu düğüm kuğulu park önündeki kavşağı temsil etmektedir. Bu kavşakta köprülü altgeçit olması sebebiyle Atatürk Bulvarı güzergahından gelen araçlar için karşıya geçiş tanımlanmamış ve sadece sola dönüş verilmiştir. Cinnah Caddesi haricindeki yollar 3 şeritlidir. Sistemde 14 Nolu düğüm Atakule önündeki kavşağı temsil etmektedir. Burada 4 şeritli Cinnah caddesinden tek yönlü olarak gelen araçlar 3 yönlü dönüş yapabilmektedir. Şekildeki gibi yol boyutları ve kuralları tespit edildikten sonra ağ oluşturulur. Hız sınırları ve araç özellikleri tek

kavşaklı çalışmada anlatılanlarla aynı değerdedir. Şekil 3.17'de Google Street map görüntüleri incelenebilir.



Şekil 3.17 Çok Kavşaklı Gerçek Hayat Uygulaması

4. ARAŞTIRMA BULGULARI

Başlangıçta tek kavşak ve tek şerit yollardan oluşan bir trafik modeli oluşturulmuş ve düzgün dağılımlı rotalar kullanılmıştır. Simülasyon yazılımının sağladığı farklı amaç fonksiyonları incelenmiştir. Simülasyon süresi 3600 saniye için bütün yollar için amaç değerleri her saniye için toplanarak matris oluşturulmuştur. Bu değerler şekle aktarılmıştır. Webster (1958) çalışmasında trafik ağına yüklenen taşıt tiplerinin farklı etkilere sahip olduğundan bahsetmektedir. Bu çalışmada taşıt tiplerinin etkilerinin incelenmesi adına otomobil; otomobil ve otobüs; otomobil, otobüs, kamyon ve acil durum taşıtlarını ihtiva edecek şekilde 3 farklı trafik düzenlemesi yapılmıştır. Bu trafik düzenlemelerinin amaç fonksiyonlarına etkisi Çizelgeler üzerinden incelenmiştir. Webster çalışmasının devamında trafik ışıklarının süreleri değiştirilerek belirli amaç değerlerinin optimizasyonu üzerine çalışmıştır. Bu çalışmada faz süresinin amaç fonksiyonlarına etkisinin incelenmesi için SUMO yazılımının belirlediği süre olan 27 saniye ile yakın bir değer olan 30 saniye ve ağın alabileceği maksimum süre olarak kabul edilen 100 saniye ışık süreleri karşılaştırılmıştır. Literatürdeki çalışmalar incelendiğinde simülasyon süresi boyunca toplanan amaç değerleri üzerinden veya bu toplam değerin araç sayısına bölünmesi ile elde edilen ortalama değerler üzerinden optimizasyon gerçekleştirdikleri görülmüştür. Bu nedenle amaç değerleri hesaplama tekniği değiştirilmiştir. Yeni yöntemle oluşturulan amaç fonksiyonları kullanılarak GA ve PSO algoritmaları önce tek amaçlı olarak çalıştırılmış, daha iyi sonuç verdiği tespit edilen PSO algoritması skalerizasyon yöntemiyle sonuç almıştır. Tek kavşaklı trafik ağında bütün çalışmalar mutlak minimum değerini bulmuştur. Bu nedenle monte carlo tekrarları yaptırılmasına gerek olmadan çok kavşaklı daha kompleks bir ağa geçilmiştir.

Bu noktadan sonra kavşak modeli değiştirilerek beş seri bağlı kavşak eş zamanlı olarak optimize edilmiştir. Ankara ilinden bir güzergah seçilmiştir. Simülasyon süresi, yakınsama grafikleri ve amaç fonksiyonu metrikleri incelenerek parametrelerde değişime gidilmiştir. Simülasyon süresi 1800 saniyeye düşürülmüş ve iterasyon sayısı 100 iken

40’a düşürülmüştür. Algoritmaların karşılaştırılabilmesi adına yakınsama kriteri ortaya koyulmamıştır. Fonksiyon değerlendirmesi bittiği zaman (40 iterasyon $*$ 20 popülasyon) algoritmalar sonuç vermektedir. Algoritmaların nihai bulduğu değerleri kaçıncı iterasyondan itibaren bulduğu algoritmalar için bir performans verilmiştir. Tez içerisinde yakınsama olarak bahsedilmiştir. Parametre çalışmasının devamında amaç fonksiyonu metriğinde ise bütün yollardan ölçüm almak yerine sisteme kavşaklara giriş yapan yollardan ölçüm alınmış ve bu değerler toplanarak araç sayısına bölünmüştür. Bu güncellemelerden sonra başlangıçta skalerizasyon yöntemi önceki çalışmada iyi sonuç veren PSO yöntemi ile çalıştırılmıştır. Skalerizasyonda farklı ağırlıklar seçilerek bu yöntem 9 defa tekrarlanmıştır. Sonrasında çok amaçlı optimizasyon problemlerini çözen NSGA-II ve MOEA/d algoritmaları çalıştırılmıştır. Üç yöntem de bekleme süresi ve CO_2 emisyonu amaçları için 10’ar defa çalıştırılmıştır. Boşluk ve Hiper Hacim metrikleriyle algoritmaların performansları ölçülmüştür. Çok amaçlı 3 yöntemle elde edilen 10 sonuç sabit ışık süresi sonuçlarıyla kıyaslanmıştır.

4.1 Tek Kavşaklı Trafik Ağı Sonuçları

Bu bölümde algoritmaların parametrelerine dair bilgiler verilmiştir. Farklı amaç fonksiyonları bu bölümde incelenmiştir. Sonrasında gerçekleştirilen optimizasyon işlemlerinin sonuçları verilmiştir.

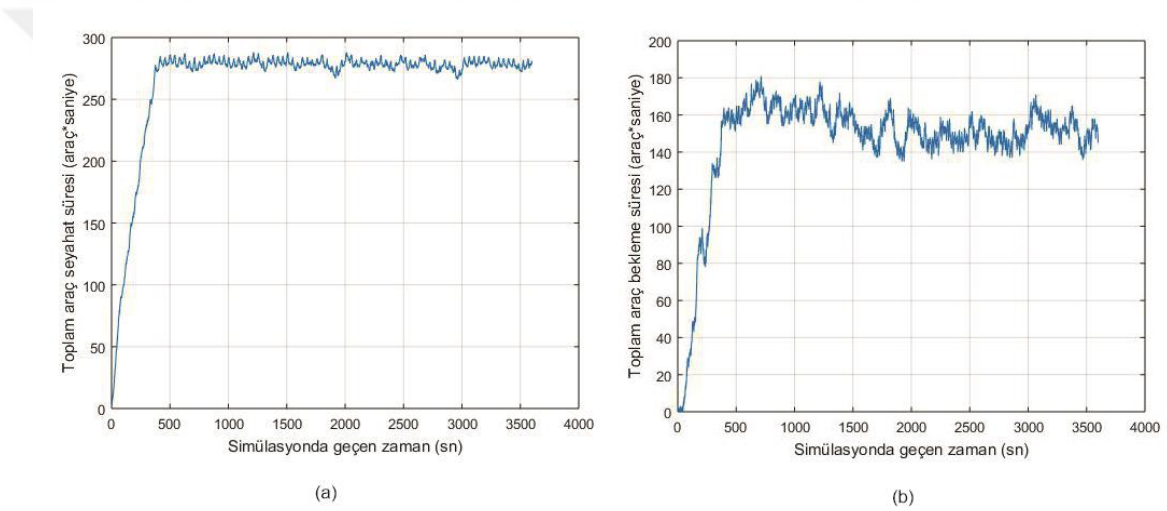
4.1.1 Amaç değerlerinin hesaplanması

İlk olarak sistemde 3600 saniye için anlık değerler incelenmiştir. Anlık değerlere göre amaç fonksiyonu *f*, 4 yön için amaç fonksiyonlarının toplanmasıyla hesaplanır.

$$f_a(x_a, t) = \sum_{i=1}^{8} \quad x_a\,(i, t) \tag{4.1}$$

(4.1) denkleminde a değeri bu çalışmada amaç fonksiyonu tipidir ve emisyon gazları, seyahat veya bekleme süresi olabilmektedir. x her yol için değer veren amaç fonksiyonudur. i değeri de 4 adet yön için 8 adet gidiş ve geliş yolun toplamıdır. Her döngü adımında 8 yoldaki değerler ölçülerek kaydedilir.

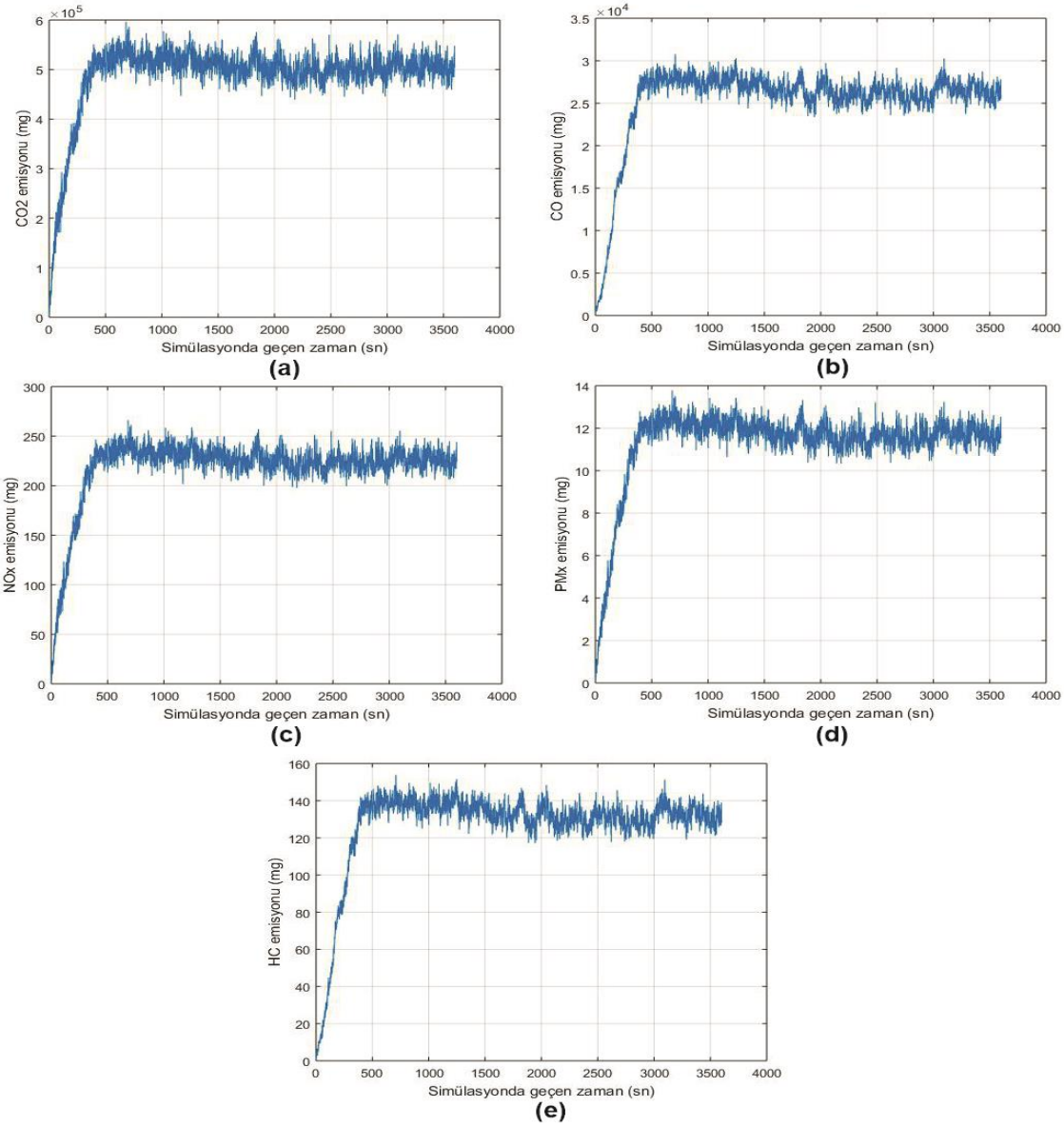
Sadece otomobillerden oluşan bir trafik incelendiğinde 3600 saniye simülasyon süresi için ortalama değerler HC için 126,8 mg; PMx için 11,3 mg; NOx için 218,1 mg; CO için 25332 mg; CO_2 için 488160 mg; bekleme süresi için 145,8 saniye; seyahat süresi için 265,6 saniye çıkmaktadır. Çizelgeler incelenirse:



Şekil 4.1 Anlık Seyahat (a) ve Bekleme (b) süresi değişimi

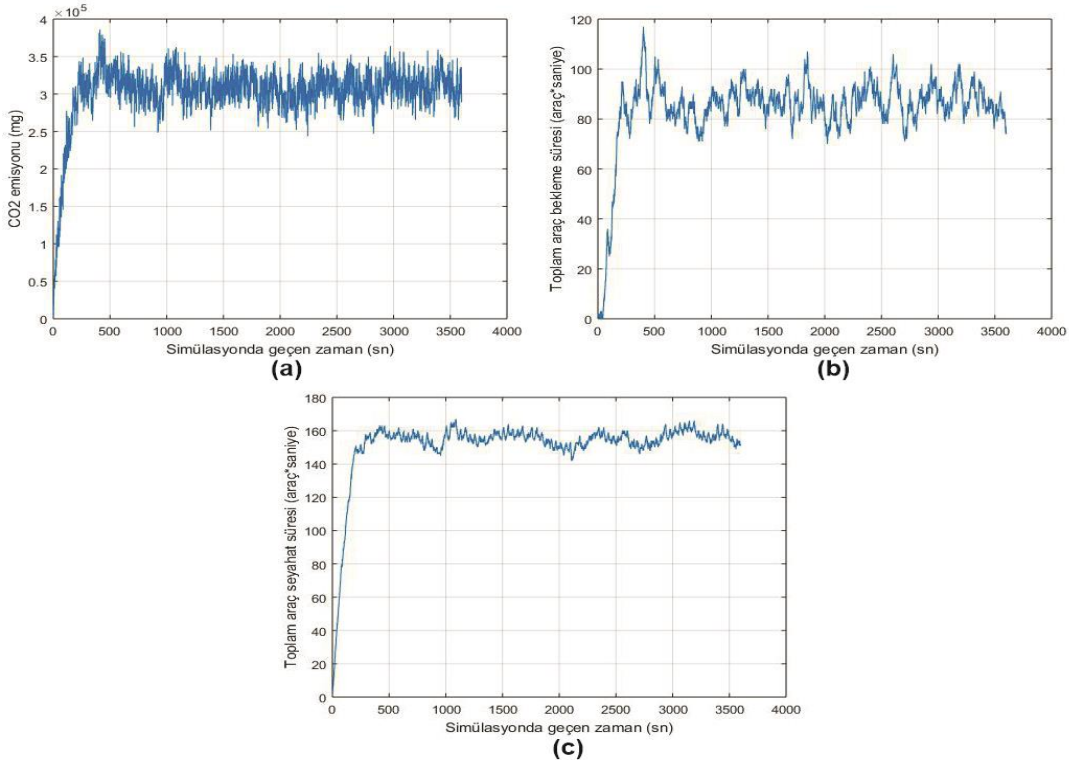
Araçlar sisteme düzgün dağılımlı olarak rastgele biçimde girmektedir. Sistemdeki bir saniyede okunan araç sayısı toplamda trafik ağında geçirilen süreyi vermektedir. Şekil 4.1'de sistem başlangıçta 0 araçla ölçüm almaya başladığı için 0 seyahat süresi ile başlamaktadır. Araç sayısı arttıkça seyahat ve bekleme süreleri doyuma ulaşana kadar artmaktadır. Başlangıç aşamasında araçların seyahatleri bitmediği için seyahat ve bekleme süreleri artmaktadır. Normal şartlarda bir aracın ilk aracın ışığa varma süresi 500 metrelik bir yol için 36 saniyedir. Bu nedenle grafiğin ilk noktalarında keskin bir sıçrama görülmektedir. Trafik doyuma ulaştıktan sonra yeni araçların giriş yaptıkça aynı miktarda aracın çıkış yapmaktadır. Şekilde görüldüğü üzere 265,6 $araç * saniye$ seyahat süresi

etrafında dalgalanma yaşanmaktadır. Bu da bir saniyede ortalam 265,6 aracın sistemimizde olduğunu gösterir. Saniyedeki bekleme süresi bir simülasyon adımında, durur vaziyetteki araçların sayısını toplar. Şekilde görüldüğü gibi doyum noktasına ulaşılana kadar artış göstermektedir. 145,8 saniye ortalama bekleme süresi vermektedir. Başka bir deyişle bir saniye için sistemde saniyede ortalama 145,8 araç beklemektedir. Toplam yol alanı sabit olduğu için 1 şeritten oluşan yollarda amaç değerlerinin doyuma gitmesi beklenir.



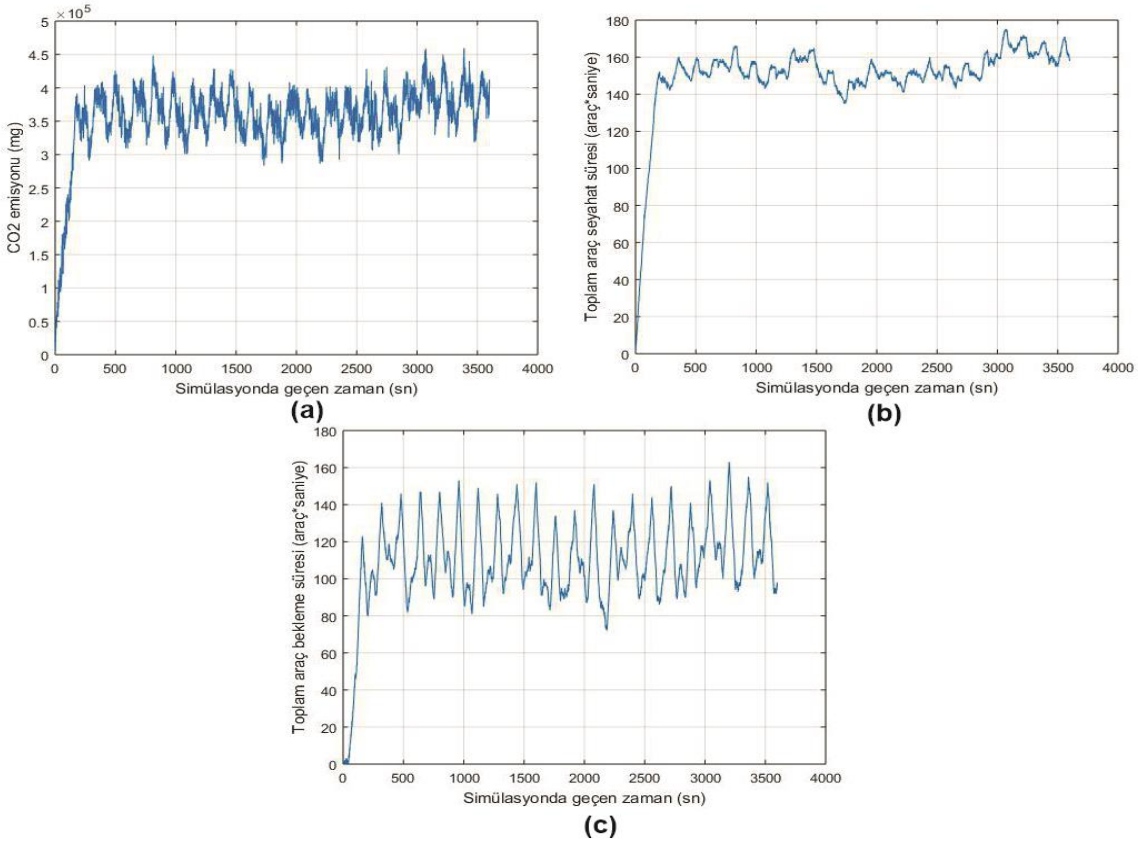
Şekil 4.2 CO_2 (a) CO (b) NOx (c) PMx (d) ve HC (e) emisyonları değişimi

Şekil 4.2'de göre sistemden ilk araç çıkışı olana kadar (yaklaşık 72 saniye) emisyon grafiklerinde hızlı bir artış meydana gelmiştir. Bu andan sonra dalgalanmalar yaşanmaktadır. Yakıt sarfiyatı ve emisyon değerleri aracın hızına ve ivmesine bağlı olarak değişmektedir. Aynı ivme ve maksimum hız değerine sahip iki araç çok yakın değerde emisyon üretmektedir. Ayrıca yanma olayı kimyasal bir olay olduğundan yakıt sarfiyatı, CO_2 ve diğer gazlar birbiriyle benzer karakteristikte grafikler vermesi beklenen bir sonuçtur. Şekil 4.2 incelendiğinde grafiklerin benzer olduğu görülebilmektedir. Bu durum da beraber amaç fonksiyonu olarak kullanılmaları tercih edilmemektedir. Literatürde sıklıkla kullanılması ve en büyük sera gazı kaynağı olması sebebiyle CO_2 sonraki çalışmalarda amaç fonksiyonu olarak seçilmiştir. Çalışmanın bundan sonraki aşamasında tek kavşaklı sistemde araç tiplerinin etkisi incelenecektir. Oluşturulan trafik senaryosunda mevcut araç sayısının yarısı otobüs olarak düzenlenerek sistem çalıştırılır. Simülasyondaki etkisini gözlemlemek adına otobüs sayısı gerçek hayat uygulamalarına kıyasla fazla verilmiştir.



Şekil 4.3 Otobüs dahil ağda CO2 (a) Bekleme (b) ve Seyahat (c) süreleri değişimi

Bu trafik senaryosunda ortalama olarak bekleme 84 saniye, seyahat 151,6 saniye ve CO_2 emisyonu 302710 mg elde edilmiştir. Şekil 4.3'te otobüs ilave olduğunda amaç değerlerinin düştüğü görünmektedir. Bu düşüş yol alanının sınırlı olması ve uzunluğu 16 metre olarak tanımlanmış otobüslerin daha az sayı ile trafiği tıkayabilmesi yüzündendir. Yoldaki araç sayısı daha az olduğu için toplam emisyon miktarlarında düşüş gözlemlenmektedir. Buradan çıkan sonuç trafik ağına karma bir trafik akış rejimi tanımlandığında yol ve araçlar için gerçek boyutlara mümkün olduğu kadar yakın değerler ve tiplerine göre doğru araç oranları verilmelidir. Sonraki aşamada simülasyonda ışık sürelerinin emisyona etkisi incelenecektir. Simülasyon tarafından tek kavşak ağı için 27 saniye olarak belirlenmiş ana yol yeşil ışık süresi 100 saniye olarak değiştirildiğinde elde edilen veriler ortalama olarak CO_2 emisyonu 357410 mg, bekleme süresi 109,2 saniye; seyahat süresi ise 150,3 saniye çıkmıştır.



Şekil 4.4 100 saniye için otobüs bulunan ağda CO_2 emisyonu (a), seyahat (b) ve bekleme (c) süreleri değişimi

Şekil 4.4 incelendiğinde sistemin doyuma gittiği görülmektedir. Toplam seyahat süresi yakın sonuç vermekle birlikte bekleme süresindeki artışın CO_2 emisyonuna da etki ettiği ve değerlerin arttığı görülmektedir. Seyahat süresinin sabit kalma sebebi ise iki kesişen yol özelliklerinin aynı olması ve yeşil hangi yola yanarsa yansın toplamda aynı miktarda aracın geçmesidir. Şekil 4.3 ve 4.4 beraber incelenirse ışık süresindeki artışın dalgalanma biçimini kare dalgaya yaklaştırdığı görülmektedir. Küçük hacimli seçilen ağda trafik sıkıştığı zaman yeni araç yüklenemediği ara yüz üzerinden tespit edilmiştir. Bu nedenle bekleme sürelerinin etkili incelenebilmesi için yollarda boşluk bırakılmalı ve bu yollar uzun seçilmelidir. Bir saniye örnekleme süresi ile alınan veriler incelendiğinde küçük değişimler gözlemlenirken yetersiz kaldıkları görülmüştür. Bu sebeple anlık veriler yerine toplam sürelerin trafik ağı içerisinden geçen toplam araç sayısına bölünmesi ile elde edilecek verilerin kullanılması gerekmektedir.

4.1.2 Tek kavşak optimizasyon sonuçları

Yeni amaç fonksiyonunda simülasyon çalışma süresi boyunca elde edilen amaç değerleri toplanmıştır. Bu elde edilen toplam değer doğrudan kullanılabilmektedir. Ancak kaç araç üzerinden elde edildiğinin bilinmesi gerekir. Bir saniye örnekleme süresi kullanılarak oluşturulan grafiklerde tıkalı trafikte araçların beklediği için mi yoksa daha çok araç geçtiği için mi amaç değerinin arttığı konusunda belirsizlik yaşanmıştır. Bu sebeple toplam amaç değerlerinin trafiğe yüklenen toplam araç sayısına bölünmesi ile elde edilen araç başına emisyon, bekleme ve seyahat süresi değerleri kullanılacaktır.

$$f_a(x_a) = \frac{\left(\sum_{t=1}^{3600}\sum_{i=1}^{8} x_a(i,t)\right)}{Araç\ Sayısı} \tag{4.2}$$

Literatürde bu değer üzerinden optimizasyon gerçekleştirilmesi yaygın bir uygulamadır (Dağüstü 2010, Damay 2015, Karadeniz 2016, Jiao vd. 2016, Hatri ve Boumhisi 2017). Yeni amaç fonksiyonları ile GA algoritması 14 nesilde yakınsama sağlamıştır. Algoritmanın parametresi kapatıldığından yakınsama sağladıktan sonra program 100

iterasyonu bitirmektedir. Çalışma sonucunda bulduğu değerler amaç değeri CO2 için 562550 mg; seyahat süresi için 350.7 saniye; bekleme süresi için 99.6 saniye bulunmuş ve karar değişkenimiz faz süresi 10 saniyedir. PSO algoritması kullanıldığında 3. iterasyonda aynı sonuca yakınsanmıştır. Tek kavşak için işlem maliyetinin düşük olması sebebiyle bütün ışık değerlerini tek tek deneyerek minimum alan bir program yazılmıştır. 9'dan 40'a kadar olan tam sayıları çeviren bir döngü içerisinde çalıştıran ve çıktı değerlerinin en küçüğünü kaydeden bir program, 10 saniyenin global optimum değeri olduğunu bulmuştur. Optimizasyon algoritmaları doğru sonucu vermektedir. Yalnız probleme esas olan trafik ağı yeterince karmaşık olmadığı için deneme yanılma yöntemi daha kısa zamanda sonuca ulaşmıştır.

Sonraki aşamada tek kavşaklı problemde skalerizasyon yöntemiyle algoritmalar çalıştırılacaktır. Ağırlıklar eşit alındığında CO_2 değeri Seyahat süresi ile toplanması durumunda baskın çıkmaktadır. Elde edilen değer çok büyük olduğu için ölçeklendirme yapılması gerekmektedir aksi halde sayısal olarak büyük olan değere karşı bir yanlılık oluşmaktadır. Amaç fonksiyonu olarak seçtiğimiz CO_2 ve seyahat süresi ikilisi benzer nicelikte olmadıklarından eşit ağırlık (0.5) verilmesi durumunda algoritma CO_2 amacına yanlılık gösterecektir. Skalerizasyon yöntemi kullanılarak PSO algortiması sürdürüldüğünde ışık süresi 10 saniye bulunmuştur. 4 nesilde yakınsama sağlamıştır. Aynı faz süresini bulmasının nedeni amaç fonksiyonlarının çelişmemesi olabilir. Çünkü SUMO'nun emisyon modelinde araçların hareket halinde ve dururken emisyon salınımı gerçekleştirdiği görülmektedir. Seyahat süresi hesaplanırken de hareket halindeki ve durur haldeki araçların toplamı kullanılmaktadır. Bekleme süresinin amaç değerleri arasında çatışmayı arttıracağı ön görülmektedir. Bu çalışmadan elde ettiğimiz sonuçlara göre:

- CO, HC, PMx, NOx benzer karakteristikte sonuç vermektedir. Amaç sayısını düşürerek işlem yoğunluğunu azaltmak için CO_2 amaç olarak belirlenmiştir.

- Simülasyonda araç tiplerinin etkisi ortalama hızın düşmesi ve ışıklarda bekleme sürelerinin artması olarak gözlenmektedir. Bu nedenle düzeltme katsayısı kullanılarak otomobillerden oluşan bir ağ üzerinde çalışılacaktır.
- Simülasyonda sürenin aşırı arttırılması ile bekleyen araç miktarı artmaktadır ve trafiğe doğrudan etki etmektedir. Başlangıçta tezin çalışma noktası olarak alınan ışık süreleri üzerinden çalışılmaya devam edilecek ve buralar üzerinden optimizasyon sağlanacaktır.
- Anlık değerlerin kullanımının bir manası olmadığı tespit edilmiştir. Bu sebeple amaç değerlerinde çalışma süresi boyunca açığa çıkan emisyon, seyahat süresi ve bekleme süresi bütün araçlar için toplanarak çalışma sonunda simülasyona giren araç sayısına bölünerek amaç değerleri elde edilecektir.
- Sistemde değişim gözlemlenebilmesi için toplam değerler üzerinden ortalama alınmalı ve ağın yeterince büyük ve karmaşık seçilmesi gerekmektedir.
- Amaç değerleri olarak CO_2 ve Bekleme Süresi seçilmelidir.

4.2 Çok Kavşaklı Trafik Ağı Sonuçları

Daha karmaşık bir ağ kurulmasının gerekliliği önceki bölümde ortaya konmuştur. Ayrıca literatürdeki çalışmalarda da birbirine sıralı kavşakların uygulandığı tespit edilmiştir. Bu nedenle beş kavşaklı bir sistem simülasyonda uygulanacaktır. Bu bölümde parametrelerin sisteme etkisi incelenmiştir. Sonrasında elde edilen simülasyon sonuçları Çizelgeler halinde verilmiştir.

4.2.1 Parametrelerin sisteme etkisi

Sezgisel algoritmalar parametrelere bağlı olarak farklı sonuçlar verebilmektedir. Parametreler doğrudan çalışma süresi ve performansa etki ederler. Farklı problemlerde farklı parametreler kullanılabilmektedir. Algoritma performanslarının kıyaslanabilmesi için seçilen parametreler bütün algoritmalar için sabit tutulmuştur. Literatürde karmaşıklık düzeyi daha fazla olan ağlarda bile 50 iterasyon sonrası değişimlerin çok

küçük olduğu tespit edilmiştir (Abbas, Chaudhary, Pesti ve Sharma, 2005). Teklu ve Sumalee (2007) yaptıkları çalışmalarda parametreler üzerine çalışma gerçekleştirilmesi gerektiğini vurgulamışlardır. Monte Carlo simülasyon sayısını 10 seçmişler ve 70 iterasyon için algoritmalarını çalıştırmışlardır. Robles (2012) simülasyon süresi için 60 dakikalık bir zaman dilimi seçmiştir. Ma (2012) yapılan hesaplamaların işlem maliyetinin yüksek olduğunu vurgulamıştır. Li, Yu, Tao ve Chen (2013) ışık değeri aralıklarını 12 ile 72 saniye almışlar, algoritma için de 60 popülasyon ve 200 nesil kullanmışlardır. Literatürdeki mevcut çalışmaların yanında tez çalışması kapsamında edinilen tecrübeler kullanılmıştır. Çok kavşaklı ağın çalıştırılmaya başlanmasıyla birlikte simülasyondaki bağlantı kopma ve hata oranlarının arttığı gözlenmiştir. Simülasyon süresinin uzaması ve farklı araç tipleri olan karma trafik kullanmak programda kopmalara neden olduğundan çalışmanın ilerlemesinde engel teşkil etmektedir. Bu nedenle program çalışma süresinin kısaltılması için çözüm aranmıştır. Simülasyonun örnekleme süresi, algoritmaların iterasyon sayısı ve program içerisindeki bazı metrikler incelenmiştir. SUMO yazılımının bir dezavantajı paralel işlem yapamamasıdır. Ölçüm süresi 3600 saniye olarak alınan kavşağın karakteristiklerini bozmayacak şekilde süre azaltımına gidilmesi için farklı çalışma süreleri denenmiştir. Çalışmanın bu kısmında sırasıyla 600, 900 ve 1800 saniye için simülasyon aynı koşullarda çalıştırılmış ve çıkan sonuçlarda belirlenen sinyal süreleri 3600 saniyelik kavşak ölçüm aralığı için denenmiş ve amaç değerleri kıyaslanmıştır.

Çizelge 4.1 Örnekleme Sürelerine Göre Amaç Değerleri

Kavşak Toplam Örnekleme Süresi	10 İterasyon için bulunan ışık süreleri	Ortalama Seyahat Süresi	Ortalama Seyahat Süresi (3600)
600	[10 9 9 10 40]	210.4704	487.0991
900	[9 11 10 16 25]	250.9816	478.0936
1800	[9 11 10 9 17]	335.9462	475.1202
3600	[10 12 9 12 17]	476.2886	476.2886

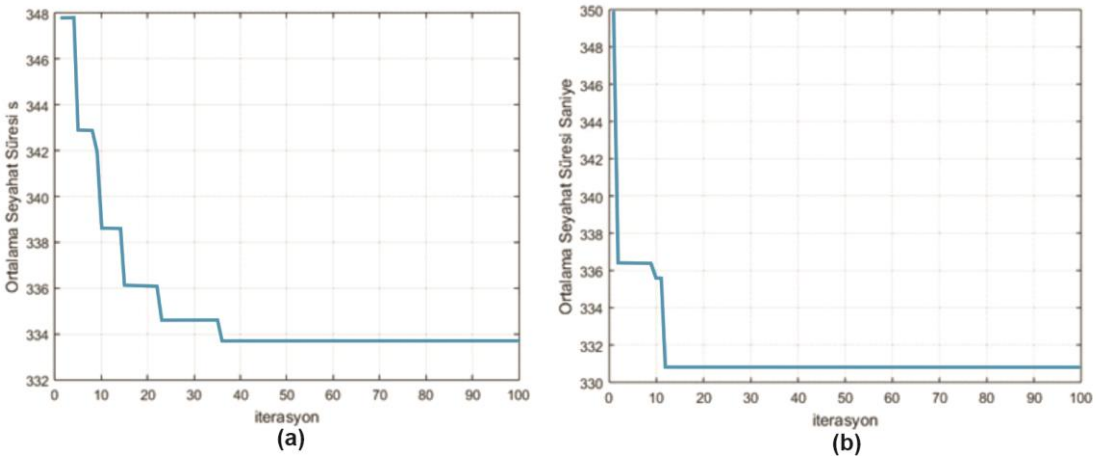
Amaç değerleri karşılaştırılabilir olmadığı için elde edilen ışık süreleri 3600 süre için test edilmiş ve sonuçlar kaydedilmiştir. Bulunan sonuçlara göre karakteristikleri bozmamak kaidesiyle en iyi sonucun 1800 Saniye yani yarım saatlik örnekleme süresinde alındığı

Çizelge 4.1'de görülmektedir. Diğer değerlerin ortalama seyahat süresi karşılıklarında artmaya sebep olurken 1800 saniyede azalmaya gitmesi sebebiyle bu değer seçilmiştir. Düzgün dağılımlı rastgele araç girişi olan sistemde trafik oluşması için bir süre beklenmesi gerekmektedir. Tek kavşaklı sistemde 500 saniyeden sonra doyuma ulaşılmıştır. Bu sebeple simülasyon sürelerinin trafik etkileri incelenmelidir. Beş kavşaklı sistem için SUMO yazılımının belirlediği sabit ışık süresi 35 saniye için simülasyon süresi ve araç değerleri Çizelge 4.2'deki gibidir:

Çizelge 4.2 Örnekleme Sürelerine Göre Giren Çıkan Araç Sayısı

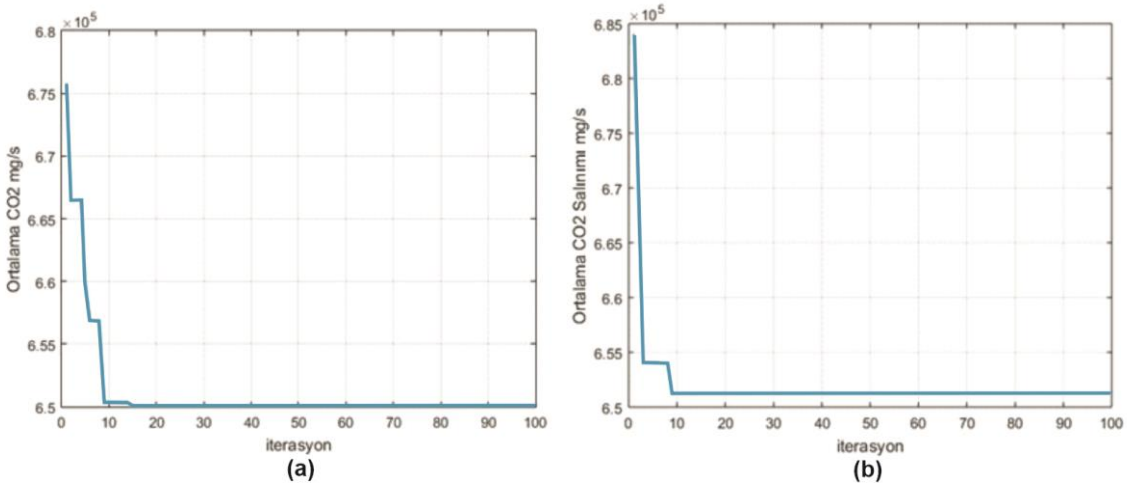
Kavşak Toplam Örnekleme Süresi	Ortalama Seyahat Süresi	Araç Girdi Çıktı Sayıları
600	223	512 - 507
900	272	853 - 764
1800	382	1535 - 1028
3600	569	3046 - 2177

Çizelge 4.2'de 600 saniye için kavşak başına 1 bekleyen araç düşmektedir. Burada 900 saniye için kavşak başına yaklaşık 18 araç, 1800 saniye için 101 araç, 3600 saniye ölçüm için 174 araç düşmektedir. Sistemimizdeki 3 şeritli 4 yolun birleşimi düşünüldüğünde kavşak başına 600 saniye için 1 araç birikmekte, 900 için 17,8 araç birikmekte, 1800 saniye için 101,4 araç birikmekte, 3600 saniye için ise 173,8 araç birikmektedir. 1800 saniye haricinde diğer simülasyon sürelerinde yeterli trafik yoğunluğu oluşmamaktadır. Bu sebeple en makul olan değerin 1800 saniye olduğu anlaşılmaktadır. Ayrıca sabit ışık sürelerine göre optimizasyon algoritmasının seyahat süresinde iyileştirmede bulunduğu burada görülmektedir. Sonraki çalışmada iterasyon sayısı azaltılacaktır.



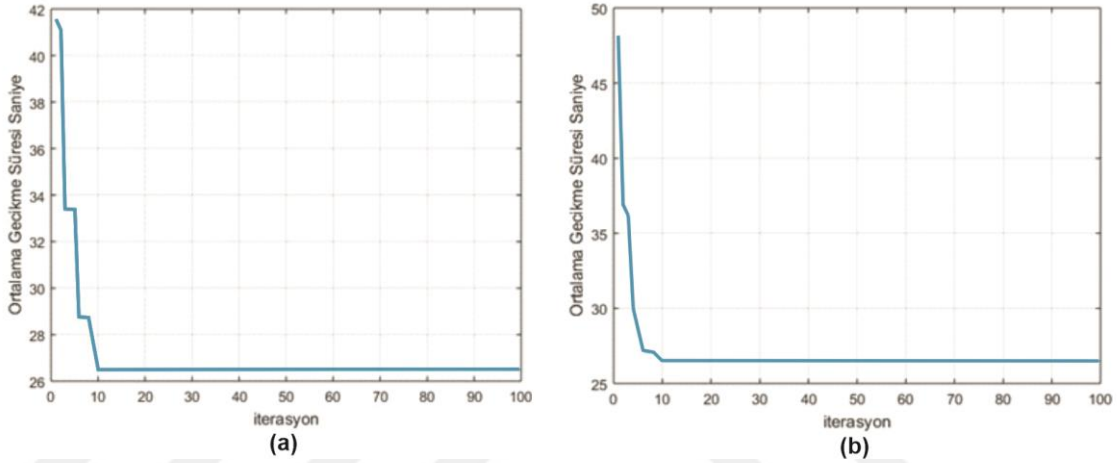
Şekil 4.5 Seyahat Süresinin GA'da (a) ve PSO'da (b) iterasyonlara göre değerinin yakınsama grafikleri

Şekil 4.5 üzerinde Ortalama Seyahat Süresi amaç değerine göre çıkan sonuç GA için 333.7 saniyedir ve 36. nesilde yakınsama sağlamaktadır. Başka bir denemede 33. nesilde 332.1 saniye değerine yakınsamıştır. Optimizasyon sonucu elde edilen ışık süreleri programdan geçirildiğinde okunan değer CO_2 için 649920 mg'dır. PSO algoritması seyahat süresinde daha iyi sonuç vermiş ve 330,8 saniye değerine yakınsamıştır. Şekil 3.17'e göre soldan sağa beş kavşak ışık süreleri için algoritma [9 11 11 12 27] saniye değerlerini elde etmiştir ve 12. İterasyonda doğru sonuca yakınsamıştır.



Şekil 4.6 CO_2 Amaç fonksiyonunun GA'da (a) ve PSO'da (b) iterasyonlara göre yakınsama grafikleri

Şekil 4.6 üzerinde Ortalama CO_2 emisyonu değerine göre algoritma çalıştırıldığında çıkan CO_2 değeri 650100 mg bulunmuştur ve 15. İterasyonda yakınsama sağlanmıştır. İlk bulduğu değere göre gelişim sağlasa da nihai değerler incelendiğinde bulunan değerin global optimum değeri bulduğu kesin olarak söylenemez. Ayrıca her çalıştırmada 37 iterasyon altında yakınsama sağladığı ve sonuç değerinin 100 iterasyona kadar değişmediği gözlemlenmektedir. PSO'da CO_2 dokuzuncu iterasyonda 651300 mg/s değerine yakınsamıştır. Işık süreleri [9 9 9 9 12] saniyedir. GA'ya göre daha kötü sonuç vermiştir. Gecikme süresi değerlendirildiğinde:



Şekil 4.7 GA'da (a) ve PSO'da (b) iterasyonlara göre gecikme süresi değerinin azalımı

Şekil 4.7'ye göre GA algoritmasının 26,5 Saniyede yakınsadığı görülmektedir. Kavşak ışık değerleri [9 9 9 9 12] saniyedir. Onuncu iterasyondan itibaren yakınsadığı görülmektedir. Karşılaştırma açısından PSO algoritmasıyla aynı trafik ağı sürülmüştür. PSO algoritması GA ile aynı değer olan 26,5 saniyeye yakınsama sağlamıştır ve 10 iterasyonda bu değeri bulmuştur. Işık süreleri aynı bulunmuştur.

Yapılan çalışmada programın çalışma süresinin büyük bölümünün amaç fonksiyonu değerlendirmesinde (function evaluation) geçmektedir. Amaç fonksiyonunun değerlendirilmesi için simülasyon içerisinde 3600 döngü adımı döndürülmektedir. Ortalama 1,7 dakika her bir simülasyon için çalışmakta ve 20 popülasyon/sürü boyutu ile 100 nesil/iterasyon için 57 saatlik bir süreye çıkmaktadır. Bu nedenle süre 1800 saniyeye

indirilmiştir. Diğer bir yapılan incelemede iterasyon sayısının arttırılmasının sonuca etki etmediği görülmektedir. Yakınsama sağladığı değerleri çok değiştirmediği gözlemlenmiş ancak her bir çalışmada farklı sonuçlar verebildiği görülmüştür. Bu nedenle iterasyon/nesil değeri 40'a indirilmiştir. Karar değişkenleri sınırları 9 ile 40 saniyeyken beş kavşaklı bir ağ için deneme yanılma metoduyla global optimum ışık değerleri bulunması tek kavşak kadar kolay değildir. Her kavşak için 31 saniyelik ışık aralığı olduğundan beş kavşak için beşinci kuvveti alınmalıdır. Buradan 28629151 değeri elde edilir. Her bir ışık kombinasyonunu denemek ortalama 2 dakika sürdüğü var sayılsa toplamda 59644 günde sonuç elde edilir. Bu gerçek hayat uygulaması için uygulanabilir bir süre değildir. Her kavşak için 9 saniye sınır değerini atadığımızda 345,6 ve 40 saniye değerini atadığımızda 385 saniye seyahat süresi elde edilmektedir. Başka bir deyişle sistem sabit ışık düzeni ile çalıştırıldığında en iyi sonucu vermemektedir. Optimizasyon algoritması optimum ışık süreleri bulmasının yanı sıra toplam çalışma zamanından da tasarruf sağlamaktadır. Trafik problemi gibi arama uzayı hakkında ön bilgi olmayan kara kutu problemlerinde stokastik arama algoritmalarının kullanılması uygun düşmektedir. Trafik ağımızda 16 adet yol tanımlanmıştır. Gidiş ve geliş olmak üzere 28 adet yol bulunmaktadır. Bazı yollar tek yönlüdür. Bizim için trafik teşkil etmeyecek çıkış yolları üzerinden ölçüm alınmayacaktır. Bu yollar sistemden doğrudan çıkış sağlayan yollardır ve buralarda araç bulunması istenen bir durumdur. Bu nedenle 28 adet olan gidiş ve geliş yol sayısı 17'ye düşürülmüştür. (4.2) formülünde i değerleri 17 adet seçilen yola göre toplanacaktır. Yeni formüle göre sadece trafik ağına giriş yolları ve diğer düğümlere bağlantı yolları hesaba katılacaktır.

4.2.2 Simülasyon sonuçları

Bu tez çalışması kapsamında kullanılan algoritmalar stokastik oldukları için iki çalıştırmada farklı sonuçlar verebilirler. Stokastik süreçler kullanıldığında yaygın uygulama Monte Carlo tekrarlarıdır. Literatürde Monte Carlo tekrarları sayısının 10 seçildiği tespit edilmiştir (Teklu ve Sumalee, 2007). Tek amaç değerleri için 10'ar kez GA ve PSO algoritması için çalıştırıldığında elde edilen değerler Çizelge 4.3'teki gibidir. Amaç olarak Bekleme süresi seçildiğinde bunun için elde edilen değerler Ortalama

Bekleme kısmında verilmiş, bulunan ışık değerleri ile trafik ağı çalıştırıldığında çıkan CO_2 emisyon değerleri de Ortalama CO_2 olarak Çizelgede belirtilmiştir. Bu değerlerin maksimum, minimum değerleri ile standart sapma ve ortalama değerleri Çizelgede verilmiştir. Aynı şekilde amaç değeri CO_2 seçildiğinde elde edilen değerler CO_2 Çizelgesine yazılmış bu değerleri sağlayan ışık değerleri sistemde çalıştırıldığı zaman verdiği bekleme süreleri Çizelgeye eklenmiştir.

Çizelge 4.3 Çok Kavşaklı Sistemde Elde Edilen Sonuçlar

	AMAÇ		Ortalama Bekleme	Ortalama CO_2
GA	Bekleme Süresi	min	**26,5**	653990
		maks	27,36	660530
		Std Sapma	0,3079	2635,2
		Ortalama	26,684	655938
	CO_2	min	27,4439	650500
		maks	35,3937	657800
		Std Sapma	2,1847	2792,6
		Ortalama	30,6312	654312
PSO	Bekleme Süresi	min	**26,5**	653980
		maks	27,2	660290
		Std Sapma	0,2826	1998,5
		Ortalama	26,64	654774
	CO_2	min	27,2935	**649500**
		maks	30,7461	651700
		Std Sapma	0,9742	900,92
		Ortalama	29,2851	650550

Birbirlerine yakın değerler elde etseler de PSO algoritmasının daha düşük standart sapma ve ortalama değerlere sahip olması sebebiyle bu problem için daha iyi sonuçlar vermiştir. Ayrıca yakınsama hızı bakımından PSO daha iyi performans sergilemiştir. Bunun nedeni PSO'nun GA gibi en iyi bireyleri tutarken aynı zamanda global en iyi bireyi de her iterasyonda seçmesi ve diğer bireyleri ona yaklaştırmasıdır. GA'da sadece ikili turnuva yapılmakta, kötü amaç değerine sahip çözümler sonraki nesile aktarılamamkta ve silinmektedir. Ayrıca global en iyi değer hesaba katılmamaktadır. Tezde bekleme süresi ile gecikme süresi ifadeleri değişmeli olarak kullanılabilmektedir.

Sonraki aşamada çok amaçlı bir fonksiyonun tek amaca çevrilmesinde etkili tekniklerden biri olan skalerizasyon yapılacaktır. Ancak bunun öncesinde amaç değerlerinin diğerine baskın olmaması için normalizasyonu gerçekleştirilecektir. Denklem (4.4)'te normalizasyon işleminin matematiksel ifadesi verilmiştir.

$$\frac{(Amaç\ Değeri - Minimum\ Değer)}{(Maksimum\ Değer - Minimum\ Değer)} \tag{4.4}$$

(4.4) formülüne göre normalize edilen amaç değerleri skalerizasyon yöntemiyle tek amaçlı hale getirilir. Minimum değerler için de Seyahat süresi 300 saniye, gecikme 20 saniye ve CO_2 640000 gram/saniye belirlenmiştir. Amaç 1 için Bekleme süresi, Amaç 2 için Ortalama CO_2 salınımı alınmıştır. Maksimum değerler için ışık süresi 35 saniye belirlendiğinde elde ettiğimiz seyahat süresi 382 saniye, CO_2 emisyonu 779000 mg/s ve gecikme süresi 95 saniye alınmıştır. Işık süresi 5 kavşaklı ağ için simülasyonun belirlediği 35 saniye sabit ışık süresidir. PSO ile ışık süreleri [9 11 10 9 10] saniye bulunmuş ve 0,07612 değerine 24. iterasyonda yakınsamıştır. Bu ışık sürelerine göre amaç fonksiyonu çıktısı 646280 mg/s CO_2 ve 28 saniye bekleme süresidir. GA ile [9 9 9 9 12] kavşak süreleri bulunmuş, 18. iterasyonda 0,08393 değerine yakınsamıştır. Işık sürelerinin amaç fonksiyonu çıktısı 653990 mg/s CO_2 değeri ve 26,91 saniye bekleme süresidir.

Çizelge 4.4 PSO ile Farklı ağırlıklar kullanılarak elde edilen skalerizasyon sonuçları

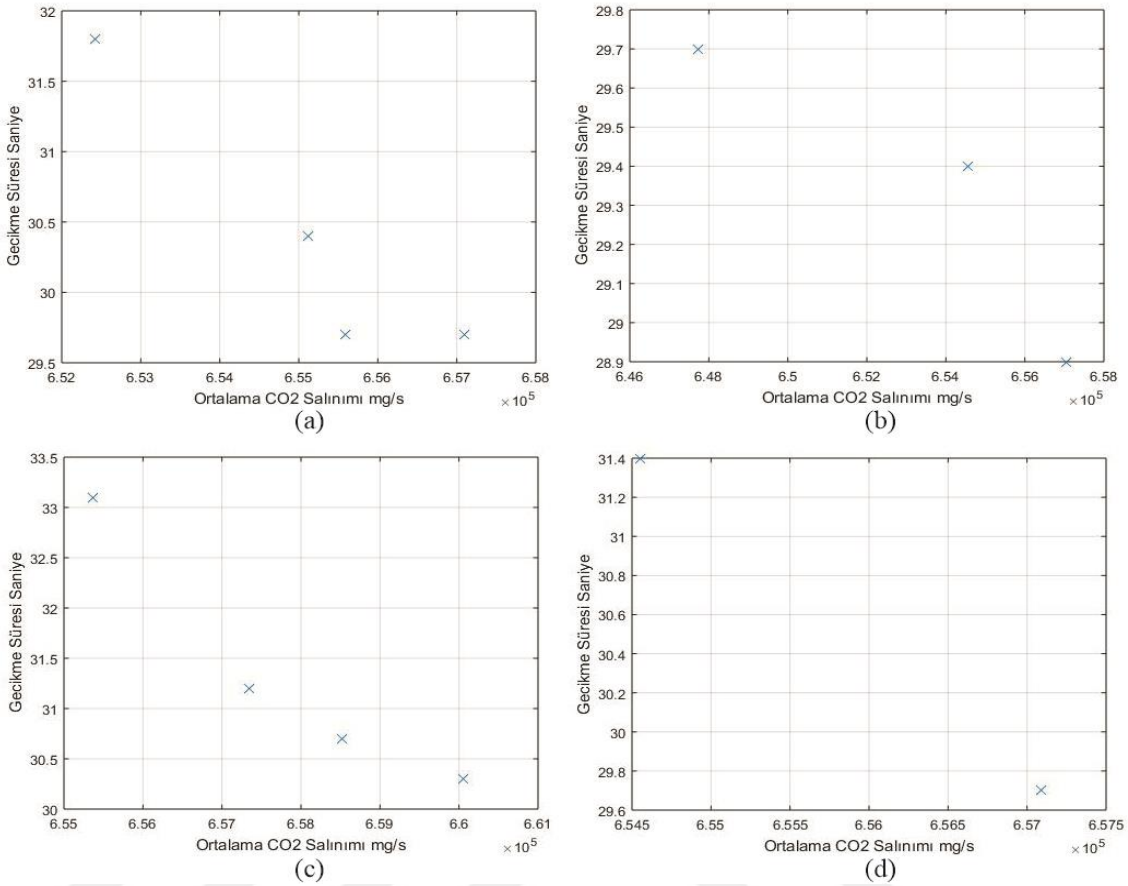
Amaç Ağırlıkları Bekleme Süresi/CO_2 Emisyonu	**Bekleme Süresi**	**CO_2**
0,1 / 0,9	27,2935	650100
0,2 / 0,8	27,2935	650100
0,3 / 0,7	26,91	653990
0,4 / 0,6	26,91	653990
0,5 / 0,5	27,9225	656500
0,6 / 0,4	26,91	653990
0,7 / 0,3	26,91	653990
0,8 / 0,2	26,91	653990
0,9 / 0,1	26,91	653990

Ağırlıkların ışık sürelerine etkisi Çizelge 4.4'te görülebilir. Burada 0,1 aralıkla farklı ağırlıklar verilerek amaç fonksiyonları PSO algoritmasında çalıştırılır. Ağırlıkların yanlılık miktarları CO_2 emisyonuna kaydığında daha iyi sonuçlar alındığı gözlemlenmektedir. Çizelge incelendiğinde alınan sonuçların aynı değerler olabileceği görülmektedir. Sonraki aşamada tek amaçlı optimizasyon problemleri için kullanılan algoritmalar yerine NSGA-II ve MOEA/D algoritmaları çalıştırılacaktır. Bu sistemler de stokastik olmaları sebebiyle 10 defa çalıştırıldıklarında elde edilen çözüm kümeleri boşluk ve hiper hacim metriğinde değerlendirilirse Çizelge 4.5'teki değerler elde edilir. Amaç fonksiyonları olarak CO_2 ve Gecikme Süresi seçilmiştir.

Çizelge 4.5 NSGA-II ve MOEA/d sonuçları

NSGA-II		MOEA/d	
Boşluk	HiperHacim	Boşluk	HiperHacim
1,3581	**60674**	1,2104	**98644**
1,2962	57207	1,3071	68342
1,0931	46342	1,3667	95693
1,152	29374	1,5556	43166
1,3333	44560	1,1878	56910
0,9028	53716	**0,8889**	51071
0,4741	38079	1	34607
1,1619	52697	0,8889	34391
1	46845	1,253	47596
1,109	32912	1,2166	64466

On monte carlo denemesi için performans değerlendirmesinde kullandığımız metrikler olan boşluk ve hiper hacim metrikleri sonuçları Çizelgede verilmiştir. Boşluk metriğinin sıfıra yakın değerde olması çözüm kümesinin iyi dağılıma sahip olduğunu, hiper hacimin büyük değerde olması çözüm kümesinin daha küçük değerler verdiğini göstermektedir.



Şekil 4.8 NSGA-II (a) ve MOEA/d (b) hiper hacim ve boşluk metriğine (c) (d) göre en iyi çözüm kümeleri

Şekil 4.8 incelendiğinde NSGA-II algoritmasının (a ve c şekilleri) daha homojen çözümler bulduğu, buna karşın MOEA/d algoritmasının (b ve d şekilleri) daha küçük amaç değerleri bulduğu görülmektedir.

Çizelge 4.6 Çok amaçlı algoritmalar ile elde edilen sonuçlar

NSGA-II	Hiper Hacim	min	29374
		max	60673
		Std Sapma	10318
		Ortalama	46241
	Boşluk	min	0,4741
		max	1,3581
		Std Sapma	0,2594
		Ortalama	1,0881
MOEA/d	Hiper Hacim	min	34391
		max	98644
		Std Sapma	22807
		Ortalama	59489
	Boşluk	min	0,8889
		max	1,5556
		Std Sapma	0,2111
		Ortalama	1,1875

Çizelge 4.6 incelendiğinde boşluk metriğine göre NSGA-II daha iyi sonuçlar vermiştir. Bu da çözüm kümesinin daha homojen olduğu anlamına gelmektedir. MOEA/d algoritması ise hiper hacim metriği için daha iyi sonuçlar bulmuştur. Bu da daha küçük değerler elde etmesi anlamına gelmektedir. SUMO yazılımının otomatik olarak belirlediği 35 saniye ışık süresinde 778980 mg CO2 ve 94,8 saniye bekleme süresi çıkmaktadır.

Çizelge 4.7 Algoritmaların İyileşme Oranları

	Emisyon	Bekleme Süresi
PSO	%16,60	%69
NSGA-II	%16,40	%68,60
MOEA/D	%16,70	%68,40
GA (tek)	%16,49	%72
PSO (tek)	%16,62	%72

Yüzde olarak bakıldığında oranların yakın görünmektedir. Bizim problemimizde amaç olarak seçtiğimiz değerleri en iyi küçülten çok amaçlı algoritmanın skalerizasyon kullanan PSO olduğu, bütün çalışma incelendiğinde tek amaçlı PSO görülmektedir.

5. TARTIŞMA ve SONUÇ

Yapılan çalışmada önce tek kavşak modeli oluşturulmuş ve basit bir model üzerinden simülasyonun çalışma yapısı incelenmiştir. Ağ oluşturulurken farklı tip araçlara farklı boyut ve hız özellikleri atanmıştır. SUMO yazılımı mevcut bir ağ üzerinden emisyon gazlarını, yakıt tüketimini, durmakta olan araç sayısını ve mevcut bir yoldaki araç sayı bilgisini sağlamaktadır. Emisyon gazları benzer karakteristikte sonuç vermektedir. İşlem maliyetini azaltmak ve doğruluk oranını arttırmak için emisyon gazlarından CO_2 amaç hesaplatılırken kullanılacaktır. İkincil amaç olarak ortalama bekleme süresi ve seyahat süresi seçilmiştir. Simülasyonda araç tiplerinin değişimi ortalama hıza ve ışıklarda bekleme sürelerine etki etmektedir. Simülasyonda ışık süresinin aşırı arttırılması ile bekleyen araç miktarı artmaktadır ve amaç değerleri yükselmektedir. Bu durum ışık süreleri üzerinden optimizasyon yapılmasına dayanak teşkil etmektedir. Sonrasında amaç fonksiyonu için 1 saniye örnekleme ile alınan değerlerin kullanımının değişimleri gözlemlemede yetersiz kaldığı tespit edilmiştir. Bu sebeple amaç değerleri hesaplama yöntemi değiştirilmiştir. Çalışma süresi boyunca açığa çıkan emisyon, seyahat süresi ve bekleme süresi bütün yollar için toplanmış ve toplam araç sayısına bölünmüştür. Yeni amaç fonksiyonları ile GA ve PSO algoritmaları çalıştırılarak 100 iterasyon/nesil 20 sürü boyutu/popülasyon ayarlarıyla çalıştırılmış ve sonuçlar elde edilmiştir. Tek kavşak probleminin basit olması sebebiyle iki algoritma aynı sonucu bulmuştur. Sonrasında skalerizasyon yöntemiyle farklı amaçlar fonksiyonları birleştirilmiştir. Skalerizasyon ile de aynı sonuç bulunmuştur. Sonrasında daha büyük boyutlara sahip ve daha karmaşık bir ağ olan beş kavşaklı ağa geçilmiştir.

Ankara ili içerisinde Keçiören Çankaya ilçelerini bağlayan bir güzergâh bu iş için seçilmiştir. Başlangıçta yine tek amaçlı optimizasyon algoritmaları GA ve PSO ile çalışılmaya devam edilmiştir. Çalışma sırasında sonuç almayı güçleştirdiği için simülasyon örnekleme süresi 60 dakikadan 30 dakikaya indirilmiş ve etkileri incelenmiştir. Parametrelerde iyileştirmeler yapılmıştır. GA ve PSO algoritmaları

emisyon, bekleme ve seyahat sürelerinden oluşan üç amaç değeri için çalıştırılarak incelenmiş ve iterasyon sayısının 40'a düşürülmesine karar verilmiştir. Son olarak ağın çıkış yolları amaç değeri hesaplamasına katılmamıştır. Toplam 17 adet yoldan ölçüm alınmıştır. Bunun nedeni söz konusu yollarda hareket halindeki araçların olmasının istenen bir durum olması ve bekleme süresini değiştirmeyecek olmalarıdır. Parametreler belirlendikten sonra sistem Monte Carlo denemeleri için çalıştırılmıştır. Simülasyon yazılımı çok kavşaklı modelde farklı araç tipleri ile olan çalışmayı kaldıramamıştır. Bu nedenle çalışmanın devamında Webster düzeltme katsayıları kullanılarak otomobillerden oluşan bir trafik üzerinde çalışılmıştır.

Emisyon modeli incelendiğinde hareket halindeki ve duran araçlar emisyon oluştuğu görülmektedir. Çok amaçlı optimizasyon problemleri çözülürken amaç değerlerinin arasında doğru orantı oluşması istenen bir durum değildir. Seçilen amaçların birbiri ile çelişmesi durumunda Pareto çözüm kümeleri oluşturulabilir. Seyahat süresi hesaplanırken hareket halindeki ve duran araçlar beraber hesaba katılmaktadır. Bu nedenle emisyon ile doğru orantıda sonuç vermektedir. Çelişme sağlamaları sebebiyle CO_2 ve Bekleme süreleri amaç olarak seçilmiş, GA ve PSO algoritmaları ile 10'ar defa çalıştırılmıştır. Her amaç için ayrı ayrı çalıştırılma yapılması gerektiği için tek amaçlı çalıştırmak zaman kaybına sebep olmaktadır. Skalerizasyon yöntemiyle çok amaçlı problemimiz tek amaca indirgenerek çalıştırılmıştır. Farklı ağırlıklarla çalışması gözlemlenen amaç fonksiyonlarımızın yanlılıklarının Bekleme süresinden yana olduğu tespit edilmiştir. Bu çalışmada PSO daha iyi sonuçlar vermiştir. Bunun nedeni PSO'nun, GA gibi en iyi bireyleri tutarken aynı zamanda global en iyi bireyi de her iterasyonda seçmesi ve diğer bireyleri ona yaklaştırmasıdır. GA'da sadece ikili turnuva yapılmakta ve global en iyi kullanılmamaktadır. Doğrudan çok amaçlı optimizasyon problemleriyle çalışmak üzere geliştirilmiş algoritmalar ile çalışmak gerekliliği doğmuştur.

Çalışmanın devamında NSGA-II ve MOEA/D algoritmaları ile 10'ar defa çalıştırılma ile sonuçlar elde edilmiştir. Çok amaçlı algoritmalar çözüm kümesi vermelerine rağmen bu sonuçlar farklı çalışmalarda çakıştığı için 10 sonuç elde edilmiştir. Boşluk metriğinde

NSGA-II daha iyi sonuç vermiştir. Bunun nedeni NSGA-II sonuçlarının düzlemde daha homojen dağılmasıdır. Bu pareto çözümlerinde istenen bir durumdur. Hiper hacim metriğindeyse MOEA/D algoritmasının NSGA-II'den daha iyi sonuç verdiği görülmektedir. Aralarında en iyi sonucu ise PSO skalerizasyon yöntemi vermiştir. Alanın büyük çıkması sabit referans noktasında Pareto cephesine daha yakın olduğunu ve değerlerin daha küçük olduğunu göstermektedir. Çözüm kümesinin bir nokta etrafında kümelenmesinin istenmediği optimizasyon problemlerinde NSGA-II tercih edilmelidir. Ancak bu çalışmada sadece en küçük değerleri bulması bakımından skalerizasyon yöntemiyle PSO algoritması iyi iş çıkarmaktadır. Bu nedenle bu problem için PSO algoritması tercih edilmelidir. Sabit ışık süresine kıyasla algoritmaların başarımları incelendiğinde yakın sonuçlar verdikleri görülmektedir.

İleriki çalışmalarda algoritma çalışma hızı, minimum değerler veya çözüm kümesinin eşit dağılımı gibi öncelikler belirlenmelidir. Daha karmaşık yapıda olan trafik ağları modellendiğinde söz konusu problem değişeceği için ilk aşamada bir yöntem kıyaslama çalışması gerçekleştirilmeli sonrasında seçilen yöntemle çalışma sürdürülmelidir. Gerçek hayat uygulaması için ise karar alırken geçen toplam süre kısaltılmaldır. Ayrıca trafiğin durumu anlık olarak değişebileceği için 1 saat ortalama değerler yerine 5 dakikalık simülasyon süresince alınan değerler üzerinden hesaplama yapılabilir. Süreye etki eden bir diğer faktör iterasyon sayısıdır. Oluşturulan trafik senaryosu için optimum değerler bulunmalı ve gerçek hayat problemine uygulanmalıdır. Literatürde amaç değerlerinin seçilme kriterleri paylaşılmamış, herkes farklı amaç değerlerini veya aynı yöntemi farklı tipte trafik senaryolarına uygulamıştır. Bu nedenle sistemler arasında kıyaslama sağlanamamaktadır. Ayrıca geçmiş çalışmaların büyük bir bölümü sabit ışık değerleri ile optimizasyon sonuçlarını kıyaslamaktadır. Benchmark problem olmaması ve herkesin sabit ışık sistemine göre kıyaslama yapması bu tip çalışmalarda gelişme sağlanmasının önüne geçmektedir.

KAYNAKLAR

Abbas, M., Chaudhary, N. A., Pesti, G. Ve Sharma A. 2005. Guidelines for Determination of Optimal Traffic Responsive Plan Selection Control Parameters, Federal Highway Administration Technical Report, No: 0-4421-2

Acosta, A. An implementation of the TraCI interface for Matlab. Interact with SUMO (Simulation of Urban Mobility). 2019. Web Sitesi: https://www.mathworks.com/matlabcentral/fileexchange/44805-traci4matlab?requestedDomain=true, Erişim Tarihi: 2019.

Anderson, J. M., Sayers, T. M. Ve Bell, M. G. H. 1998. Optimization of a Fuzzy Logic Traffic Signal Controller by a Multiobjective Genetic Algorithm, Road Transport Information and Control, Conference Publication No 454 IEE

Behrisch, M., Bieker, L. ve Erdmann, J. 2011. SUMO - Simulation of Urban Mobility - An Overview, International Conference on Advances in System Simulation, pp. 55–60.

Bodur, M.A. 2013. Akıllı Ulaşım Sistemleri, Yüksek Lisans Tezi, Bahçeşehir Üniversitesi, Fen Bilimleri Enstitüsü, Kentsel Sistemler ve Ulaştırma Yönetimi Ana Bilim Dalı, 74, İstanbul

Branke, J., Goldate, P. ve Prothmann, H. 2007. Actuated Traffic Signal Optimization using Evolutionary Algorithms, ITS European Congress, pp. 1-8.

Cameron, G., Wylie, B. J. N. ve Mcarthur, D. 1994. PARAMICS - Moving Vehicles on the Connection Machine. Proceedings of the ACM/IEEE Supercomputing Conference, 291-300.

Ceylan, H. Ve Bell, M. G. H. 2004. Traffic Signal Timing Optimisation Based on Genetic Algorithm Approach Including Drivers Routing, Transportation Research Part B, 329-342

Che, H., Zhang, H., Lin, Z., Luo, D. ve Wu, J. 2012. Urban Traffic Signal Coordinated Control Optimization with Bus Priority Based on Quantum Genetic Algorithm, Advanced Materials Research Vols. 433-440 pp 829-834

Chen, J. 2014. A Robust Multi-objective Compatible Optimization Control Algorithm for Traffic Signal Control, IEEE 17th International Conference on Intelligent Transportation Systems (ITSC).

Chen, J. ve Yuan, C. 2011. Urban Oversaturated Traffic Network Control Based on Preference Multi-Objective Compatible Optimization Control, Advanced Materials Research Vol. 317-319 pp 1373-1384

Chin, Y. K., Yong, K. C., Bolong, N., Yang, S. S. Ve Teo K. T. K. 2011. Multiple Intersections Traffic Signal Timing Optimization with Genetic Algorithm, 2011 IEEE International Conference on Control System, Computing and Engineering.

Coello Coello, C.A. and Cruz, C.N. 2005. Solving multiobjective optimization problems using an artificial immune system. Genetic Programming Evaluable Machines. 6(2), 163-190.

Dağüstü, H.Ş. 2010. Trafik Yönetiminde Kavşak Trafiğinin Kontrolü için Bir Sinyal Zamanlama Modeli, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği Ana Bilim Dalı, 137, İstanbul

Damay, N. 2015. Multiple-Objective Optimization of Traffic Lights Using a Genetic Algorithm and a Microscopic Traffic Simulator. Yüksek Lisans Tezi, Royal Institute of Technology, School of Computer Science and Communication, 107, İsveç

Deb, K. 1998. Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems, Kanpur Genetic Algorithms Laboratory (KanGAL), Technical Report No. CI-49/98.

Deb, K. 2002. A Fast and Elitist Multiobjective Genetic Algorithm NSGA-II, IEEE Transactions on Evolutionary Computation Vol. 6 No.2.

El Hatri, C., Boumhidi J. 2017. Traffic Management Model for Vehicle Re-routing and Traffic Light Control Based on Multi-Objective Particle Swarm Optimization, Intelligent Decision Technologies -1 1-10 DOI 10.3233/IDT-170288

Fellendorf, M. 1994. VISSIM: A Microscopic Simulation Tool to Evaluate Actuated Signal Control Including Bus Priority. 64th ITE Annual Meeting.

Goldberg, D. E. 1989. Genetic Algorithms in Search Optimization and Machine Learning, Addison Wesley Publishing Co., 432, Kanada

Jiao, P., Li R. ve Li, Z. 2016. Pareto front–based Multi-objective Real-time Traffic Signal Control Model for Intersections Using Particle Swarm Optimization Algorithm, Advances in Mechanical Engineering, vol. 8, no. 8, pp. 1–15.

Kalganova, T., Russel, G. Ve Cumming, A. 1999. Multiple Traffic Signal Control Using a Genetic Algorithm, Artificial neural nets and genetic algorithms, ICANNGA ’99. Proceedings of the 4th international conference held in Protorož, Slovenia, DOI: 10.1007/978-3-7091-6384-9_38

Karadeniz, A. T. 2016. Trafik Işık Optimizasyon Sistemlerinin Karşılaştırılması. Yüksek Lisans Tezi, Karabük Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Ana Bilim Dalı, 89, Karabük.

Kennedy, J. and Eberhart, R. 1995. Particle swarm optimization. IEEE International Conference on Neural Networks. 1942-1948.

Li, Y., Yu, L., Tao, S. ve Chen, K. 2013. Multi-Objective Optimization of Traffic Signal Timing for Oversaturated Intersection, Mathematical Problems in Engineering Vol 2013 Article ID 182643.

Luo, P., Ma, Q. Ve Huang, H. X. 2009. Urban Trunk Road Traffic Signal Coordinated Control Based on Multi-Objective Immune Algorithm, 2009 International Asia Conference on Informatics in Control, Automation and Robotics.

Ma, X. 2012. Multi-Criteria Evaluation of Optimal Signal Strategies Using Traffic Simulation and Evolutionary Algorithms, International Environmental Modelling and Software Society (iEMSs), International Congress on Environmental Modelling and Software Managing Resources of a Limited Planet, Sixth Biennial Meeting, Leipzig, Germany.

Marler, R. T. Ve Arora, J. S. 2004. Survey of Multi-Objective Optimization Methods for Engineering. DOI 10.1007/s00158-003-0368-6

Pardalos, P. M., Zilinkas A. Ve Zilinkas J. 2017. Non-Convex Multi-Objective Optimization, Springer Optimization and Its Applications 123, DOI 10.1007/978-3-319-61007-8_2, 13-18, İsviçre.

Rahmat, R. A. 2013. Application of Genetic Algorithm in Optimizing Traffic Control. Recent Advances in Mathematics, ISBN: 978-1-61804-158-6

Ringel, R. 1995. Schaffung eines Tools zur Modellierung und Simulation von Prozessen des Straßenverkehrs. Yüksek Lisans Tezi, Technische Universität Dresden.

Robertson, D.I. 1969. TRANSYT: a traffic network study tool. RRL Report, LR 253, Transport and Road Research Laboratory, Crowthorne.

Robles, D. 2012. Optimal Signal Control With Multiple Objectives in Traffic Mobility and Environmental Impacts. Yüksek Lisans Tezi, Royal Institute of Technology, 88, İsveç

Singh, L., Tripathi, S. Ve Arora H. 2009. Time Optimization for Traffic Signal Control Using Genetic Algorithm, International Journal of Recent Trends in Engineering, Vol. 2 No. 2.

SUMO User Documentation. 2019. Web Sitesi: http://sumo.dlr.de/wiki/SUMO_User_Documentation, Erişim Tarihi: 2019.

Teklu, F. Ve Sumalee, A. 2007. A Genetic Algorithm Approach for Optimizing Traffic Control Signals Considering Routing, Computer Aided Civil and Infrasturcture Engineering, 31-43.

Trafficware. 2001. Synchro 5.0 User's Guide.

TÜİK Motorlu Kara Taşıtları İstatistikleri. 2019. Web Sitesi: http://www.tuik.gov.tr/PreHaberBultenleri.do?id=30639 Erişim Tarihi: 2019.

Van Vliet, D. 1982. SATURN Amodern assignment model. Traffic Engineering and Control, 23, 578–81.

Vogel, A., Goerick, C. Ve Seelen, W. 2000. Evolutionary Algorithms for Optimizing Traffic Signal Operation, ESIT 2000, Almanya

Webster, F.V. 1958. Traffic Signal Settings, Road Research Laboratory Technical Paper, Vol. 39, No. 1, pp. 27-35.

While, L., Bradstreet, L. and Barone, L. 2012. A fast way of calculating exact hypervolumes. IEEE Transactions on Evolutionary Computation, 16(1), 86-95.

Yagar S. ve Han, B. 1994. A Procedure for Real-time Signal Control That Considers Transit Interference and Priority, Trans. Res.-B vol 28B, No:4 pp 315-331.

Zeng, W., He, Z. ve Chen, N. 2010. A Multi Objective Optimization Model and a Decision Making Method for Traffic Signal Control, ICCTP : Integrated Transportation Systems.

Zhang, Q. ve Li, H. 2007. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition, IEEE Transactions on Evolutionary Computation Vol. 11 No.6.

Zhou, Z. ve Cai, M. 2014. Intersection signal control multi-objective optimization based on genetic algorithm, Journal of Traffic and Transportation Engineering, Vol. 1(2), pp. 153-158.

ÖZGEÇMİŞ

Adı Soyadı	:	Adnan Şahin KARACA
Doğum Yeri	:	Ankara
Doğum Tarihi	:	25.05.1989
Medeni Hali	:	Bekar
Yabancı Dil	:	İngilizce, İspanyolca, Almanca

Eğitim Durumu

Lise : Sincan İbni Sina Lisesi (2006)

Lisans : Kırıkkale Üniversitesi Mühendislik Fakültesi Elektrik Elektronik Mühendisliği Anabilim Dalı(2012)

Yüksek Lisans : Ankara Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği Anabilim Dalı (2020)

Çalıştığı Kurum

Türk Standartları Enstitüsü (2015 - …)