GAZİ ÜNİVERSİTESİ

Gazi

1926

YAZILIM TANIMLI AĞ TABANLI NESNELERİN İNTERNETİNDE YÖNLENDİRME, KONTROLÖR VE SUNUCU YERLEŞTİRME İÇİN MİMARİ ENİYİLEMESİ

Yasin İNAĞ

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TEMMUZ 2022

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

• Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,

• Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,

• Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,

• Kullanılan verilerde herhangi bir değişiklik yapmadığımı,

• Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Yasin İNAĞ

25/07/2022

YAZILIM TANIMLI AĞ TABANLI NESNELERİN İNTERNETİNDE YÖNLENDİRME, KONTROLÖR VE SUNUCU YERLEŞTİRME İÇİN MİMARİ ENİYİLEMESİ

(Doktora Tezi)

Yasin İNAĞ

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Temmuz 2022

ÖZET

Nesnelerin İnterneti (Internet of Things - IoT) uygulamalarının sayısı arttıkça, bu uygulamalar tarafından üretilen verilerin merkezi bir bulut sunucusuna verimli bir şekilde iletilmesi zor bir konu olmaktadır. Tez kapsamında, sis hesaplama ve Yazılım Tanımlı Ağ (Software Defined Network - SDN) teknolojilerini kullanarak iletimi kolaylaştırmayı amaçlanmıştır. Bu amaçla, IoT ağları için iki yeni içerik tabanlı yönlendirme (Content Based Network - CBF) modeli önerilmiştir. İlk model, iletimi ve hesaplama gecikmesini azaltmak için sis hesaplama ve içerik tabanlı yönlendirmeden yararlanmaktadır. Birinci modele dayalı olarak, ikinci model, veri hızı ve gecikme gereksinimlerini sağlarken kritik verilerin zamanında iletimini sağlamak için önceliklendirme konseptini kullanmaktadır. Önerilen modelleri değerlendirmek ve bunların verim, gecikme ve kayıp oranı metrikleri üzerindeki etkilerini ortaya çıkarmak için kapsamlı simülasyonlar gerçekleştirilmiştir. Sonuçlara göre, önerilen modeller verimli iletim ve düşük zaman gecikmesi sağlamıştır. Ayrıca ikinci model, kritik verileri daha etkin bir şekilde iletme yeteneğine sahiptir. Önerilen modellerin etkinliğini artırmak için, Kontrolör Yerleştirme Problemi (Controller Placement Problem - CPP) için matematiksel bir model önerilmiştir. Kontrolörün hatalı olması durumunda ağın esnekliği ve güvenilirliği sağlanmalıdır. Bu nedenle, önerilen model, kontrolör arızalanması problemini dikkate alarak ağdaki kontrolör(ler)in sayısını ve konumunu optimize etmiştir. Simülasyon sonuçları, ağ sürekliliği ve iletişim güvenliğini sağlarken ortalama gecikme süresinin biraz arttığını göstermektedir. Çalışmanın devamında, veri analizi hizmeti sağlayan ilişkili Sunucuları Yerleştirme Problemi (Server Placement Problem - SPP) için tam sayılı doğrusal programlama ve sezgisel yaklaşımlı iki model önerilmiştir. Her bir sis sunucusuna farklı veri türleri iletilmektedir. İletilen veri türleri analiz edilerek yeni bir veri üretmekte ve bu veri bir başka sunucuda işlenmek üzere ağa iletilmektedir. İlişkili sis sunucu konumu için p-medyan tabanlı tam sayılı doğrusal programlama modeli ve açgözlü sezgisel model önerilmiştir. Önerilen modeller ile ortalama gecikme süresi azaltılmış ve tez kapsamında önerilen iletişim modellerinin etkinliği artırılmıştır.

Bilim Kodu	:	92407
Anahtar kelimeler	:	SDN, IoT, İçerik tabanlı yönlendirme, Sunucu yerleştirme problemi
Sayfa Adedi	:	106
Danışman	:	Doç. Dr. Mehmet DEMİRCİ

ARCHITECTURE OPTIMIZATION FOR FORWARDING, CONTROLLER AND SERVER PLACEMENT IN SOFTWARE DEFINED NETWORKING ENABLED INTERNET OF THINGS

(Ph. D. Thesis)

Yasin İNAĞ

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

July 2022

ABSTRACT

As the number of Internet of Things (IoT) applications increases, efficient data transmission to a central cloud server is a difficult issue. Within the scope of the thesis, it is aimed to facilitate transmission by using fog computing and Software Defined Network (SDN) technologies. For this purpose, two new Content Based Network (CBF) models have been proposed for IoT networks. The first model uses fog computing and content-based routing to reduce transmission and computational delay. Based on the first model, the second model uses the concept of prioritization to ensure timely transmission of critical data while ensuring the data rate and delay requirements. Extensive simulations are conducted to evaluate the proposed models and uncover their impact on throughput, delay, and loss rate requirements. According to the results, the proposed models ensure efficient transmission and low computational delay. In addition, the second model has the ability to transmit critical data more effectively. A mathematical model for the Controller Placement Problem (CPP) is proposed to increase the effectiveness of the proposed models. The flexibility and reliability of the network must be ensured in case the controller fails. Therefore, the proposed model has optimized the number and location of the controller(s) in the network, considering the controller failure problem. The simulation results show that the average latency slightly increases while ensuring network continuity and communication security. In the continuation of the study, two models with linear programming and a heuristic approach are proposed for the associated Server Placement Problem (SPP), which provides a data analysis service. Different types of data are transmitted to each fog server. The transmitted data types are analyzed, then new data is produced, which is transmitted to the network for processing on another server. A p-median based integer linear programming and the greedy heuristic models are proposed for the associated fog server placement. The average delay time has been reduced in the proposed models, and the effectiveness of the communication models proposed in the thesis has increased.

Science Code	:	92407
Key Words	:	SDN, IoT, Content based forwarding, Server placement problem
Page Number	:	106
Supervisor	:	Assoc. Prof. Mehmet DEMİRCİ

TEŞEKKÜR

Akademik gelişimime büyük katkı sağlayarak doktora çalışmalarım süresince kıymetli görüş ve yardımlarıyla beni yönlendiren danışman hocam Sayın Doç. Dr. Mehmet DEMİRCİ'ye; Doktora tez çalışmam boyunca görüşlerini esirgemeyerek değerli fikirleriyle beni yönlendiren tez izleme komitesi hocalarım Sayın Prof. Dr. Bülent TAVLI ve Sayın Prof. Dr. Suat ÖZDEMİR'e;

Kıymetli zamanlarını ayırarak değerlendirmede bulunan ve görüşlerini paylaşarak tezime katkı sunan jüri üyeleri hocalarım Sayın Prof. Dr. Şeref SAĞIROĞLU ve Sayın Dr. Öğr. Üyesi TUBA ÇAĞLIKANTAR'a;

Tez çalışmamı 1001 – Bilimsel ve Teknolojik Araştırma Projelerini Destekleme Programı kapsamında 118E212 numaralı proje ile destekleyen Türkiye Bilimsel ve Teknolojik Araştırma Kurumuna (TÜBİTAK);

Hayatım boyunca maddi manevi hiçbir desteğini esirgemeden yanımda olan aileme;

Hayatıma girdiği günden beri vermiş olduğu destek, huzur ve mutluluk için yol arkadaşım kıymetli eşim Tuğçe İNAĞ'a; sevgili kızlarım Hazal ve Defne'ye sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklamalar
GBytes	Gigabytes
GHz	Gigahertz
Mbits	Megabit per second
Ms	Milisaniye

Kısaltmalar	Açıklamalar
5G	Fifth Generration (Beşinci Nesil)
CBF-SDN	Content Based Forwarding via SDN (SDN Destekli İçerik Tabanlı Yönlendirme)
CBR	Content Based Routing (İçerik Tabanlı Yönlendirme)
CCN	Content Centric Network (İçerik Merkezli Ağ)
CPP	Controller Placement Problem (Kontrolör Yerleştirme Problemi)
GNS-3	Graphical Network Simulator – 3 (Grafik Ağ Simülasyonu-3)
ICN	Informatin Centric Networking (Bilgi Merkezli Ağ)
IoT	Internet of Things (Nesnelerin interneti)

Kısaltmalar	Açıklamalar
CF-CPP	Capacity and Fault Tolerant Controller Placement Problem (Kapasite ve Hata Toleranslı Kontrolör Yerleştirme Problemi)
NDN	Named Data Networking (Adlandırılmış Veri Ağı)
NFV	Network Function virtualization (Ağ Fonksiyonlarının Sanallaştırılması)
NOS	Network Operating System (Ağ İşletim Sistemi)
ODL	OpendayLight
OF	OpenFlow
OFM	OpenFlow Manager
OVS	OpenvSwitch
PCBF-SDN	Priority Content Based Forwarding via SDN (SDN Destekli Öncelikli Verilerin İçerik Tabanlı Yönlendirilmesi)
QoS	Quality of Services (Hizmet Kalitesi)
SDN	Software Defined Network (Yazılım Tanımlı Ağlar)
SPP	Server Placement Problem (Sunucu Yerleştirme Problemi)
ToS	Type of Service
VANET	Vehicular ad-hoc network (Araçlar Arası İletişim Ağları)
WAN	Wide Area Network (Geniş Alan Ağları)

1. GİRİŞ

Günümüzde kullanılan geleneksel ağlar esnekliğini yitirmiş, donanıma bağımlı bir hal almıştır. Sisteme yeni cihazların eklenmesi, güncellenmesi ve son teknolojilerin eklenmesi donanımsal ve yönetimsel zorluklar ortaya çıkarmıştır. Bu zorlukların çözümü için ağın yazılım ile merkezi kontrolünün sağlanması ve sanallaştırılması gibi çözümler geliştirilmiştir. Yazılım Tanımlı Ağlar (Software Defined Network, SDN) ile var olan ağ alt yapısı donanım bağımlılığını ortadan kaldırmıştır. SDN terimi ilk olarak Stanford üniversitesinde geliştirilen OpenFlow protokolü ile ortaya çıktı [1]. SDN'in temel prensibi kontrol ve veri katmanlarını birbirinden ayırmasıdır. Bu ayrım, ağ alt yapısında üretici yazılımı ve donanımı bağımlılığını ortadan kaldırmaktadır. SDN ağa esneklik kattığı gibi ağı yazılım ile merkezden kontrol etme imkânı tanıdığından ağı dinamik bir yapıya da dönüştürmektedir. Bu dinamik yapı ağın gerçek zamanlı yönetilmesine, hata kontrolünün sağlanmasına ve gecikme optimizasyonlarının sağlanmasına olanak sağlamaktadır [1].

Bulut bilişim sistemleri, Nesnelerin interneti (Internet of Things, IoT) verilerinin yüksek hızını, hacmini ve boyut sayısını ele almak için tasarlanmamıştır ve sınırlı bilgi işlem ve depolama yetenekleri nedeniyle gelecek vaat eden IoT teknolojilerinin taleplerini karşılayamamıştır. Bu sistemlerde, büyük miktarda veriyi tek bir merkezi sunucuya iletmek, trafik yoğunluğunda artış, tıkanıklık, gecikme, bant genişliği darboğazı, hizmet kalitesinde azalma (Quality of Services, QoS) ve güvenlik gibi istenmeyen sonuçlara neden olmaktadır [2,3]. Buna ek olarak, IoT uygulamaları, gerçek zamanlı olarak işlenmesi gereken coğrafi olarak dağıtılmış, geçici ve anlık veriler üretmektedir ve literatürde, bulut bilişimin bu gereksinimleri karşılayamayabileceğini göstermektedir [4,5]. IoT verilerini işlemek için dağıtılmış bir yaklaşıma ihtiyaç vardır, bu nedenle sis hesaplama kavramı bulut bilişim için yardımcı bir değerler dizisi olarak ortaya çıkmıştır [6,7].

Sis hesaplama, bilgi işlem, depolama ve ağ oluşturma süreçlerini ağın kenarına taşıyarak bulut bilişim için ek özellikler sunan yeni bir teknolojidir. Uygulamalarını ve hizmetlerini son kullanıcılara daha yakın bir yere dağıtarak gecikmeye duyarlı ve konuma duyarlı uygulamaları destekleyen tamamlayıcı bir teknolojidir [8]. Sis sunucuları, bulut sunucusunun ve son kullanıcıların ortasında bulunur ve bulut sunucusundaki yükü hafifletmek için sunucuya iletmeden önce verileri önceden işlemektedir [3,9].

IoT cihazlarından ilgili sis sunucularına veri aktarımı sırasında, ağa bağlı çok sayıda cihaz nedeniyle topoloji değişiklikleri meydana gelebilir. Çevresel koşullara bağlı olarak, bazı etkin düğümler başarısız olabilir veya yeni düğümler ağa katılıp etkinleşebilir. Böyle bir ağda anında dinamik değişiklikler meydana geldiğinde, ağın yönetimi veya kontrolü geleneksel yöntemlerle oldukça zorlaşır [10]. SDN, ağı veri düzlemi ve kontrol düzlemi olarak ayırarak, daha esnek ve yönetilebilir hale getirmekte ve bu soruna umut verici bir çözüm olarak sunulmaktadır. SDN dağıtılmış sis sunucularının merkezi kontrolünü sağlar ve içerik tabanlı yönlendirme, öncelik yönetimi, hata kontrolü ve gecikme optimizasyonlarını gerçekleştirir [1,11].

İçerik tabanlı yönlendirme (Content Centric Network, CCN), verinin içeriğine göre ağda iletilmesi gereken sunucuya iletilmesidir. SDN destekli ağlarda CCN kontrolör yazılımı veya kontrolör ile yönetilen içerik yönetim sunucular yardımı ile gerçekleştirilmektedir. CCN, paketin türüne göre yönlendirme sağlarken hatalı veri iletiminde azaltmaktadır [12].

Sis sunucu mimarisinde, farklı uygulamaların farklı gereksinimleri ve öncelikleri vardır. Bir veri iletiminde ağ tıkanıklığı olması durumunda, verilerin zamanında aktarılamaması, bir uygulamanın performansının önemli ölçüde düşmesine neden olabilir. Belirli bir düzeyde QoS memnuniyeti gerektiren kritik bir uygulama için ciddi bir başarısızlıkla sonuçlanabilir, bu nedenle, ağ tıkanıklığında bile yüksek öncelikli verilerin diğer trafikler üzerinden aktarılmasını sağlamak için verimli yönlendirme teknikleri hayati önem taşır. SDN'in, ağ akışlarını dinamik olarak ağ trafiğine uyum sağlayacak şekilde yeniden yapılandırması mümkündür. Kritik bir verinin gerekli bant genişliği, SDN denetleyicisi tarafından yönetilen bir veri merkezi ağındaki öncelik sıraları yapılandırmasıyla bant genişliği tahsisi ile de garanti edilir. Elde edilen sensör verisinin türü veya değeri bu önceliği belirlemektedir. Ağda bu tarz verilerin donanım ve yazılım destekli öncelikli iletilmesi gerçekleştirilebilir. Öncelikli verilerin iletildiği ağlarda farklı kuyruklar oluşturularak verilerin bu kuyruklara atanması gerçekleştirilir. SDN destekli ağlarda, bu kuyruk derecelendirilir. En yüksek dereceye sahip kuyruktaki veriler önce iletilir [13,14].

SDN, kontrol mantığını veri düzlemi cihazlarından ve harici ağ varlıklarından, kontrolörlere taşır. SDN'nin ana ilkesi, tüm kontrol işlevlerini veri düzlemi düğümlerinden programlanabilir bir ağ varlığı olan denetleyiciye kaydırmaktır. Kontrolör Yerleştirme Problemi (Controller Placement Problem, CPP), ağda bir veya birden çok kontrolörün

konumun belirlenmesidir. Kontrolörün ağın beyni olduğu varsayıldığından, CPP ağ performansını doğrudan etkilemektedir. Birden çok kontrolör bulunduran ağlarda, kontrolörler fiziksel olarak dağıtılmış ancak mantıksal olarak merkezileştirilmiş ve yerleştirilmiştir. Bu mimari sis hesaplama mimarilerinde daha kısa mesafe ile iletişime geçtiği için gecikmeyi azaltabilmektedir [15]. Kontrolörün servis dışı kalması durumunda kontrolöre bağlı anahtarların ağda iletişime devam edebilmesi gerekmektedir. Bu probleme literatürde hiyerarşik kontrolör mimarisi geliştirilmiştir. Hiyerarşik mimari, kontrol düzlemini birden çok seviyeye böler. Her seviyeden farklı kontrolör sorumludur ve bu kontrolörleri yöneten merkezi bir kontrolör bulunmaktadır [16,17].

Sis düğümleri olarak adlandırılan ağ kenarına yakın cihazlar, yakınındaki uç cihazlara düşük gecikme süresi ile bilgi işlem yetenekleri sunar. Ortaya çıkan sis bilişim paradigmasında, uygulama bileşenleri hem bulut veri merkezlerini hem de sis düğümlerini içeren dağıtılmış bir altyapıya yerleştirilebilir. SDN destekli sis bilişiminde sunucuların konumu performansı doğrudan etkilemektedir. Bu sebeple, ağda görev atanmış sunucuların konumunun belirlenmesi QoS hizmeti sağlamaktadır. Literatürde, Sunucu yerleştirme problemi (Server Placement Problem, SPP) olarak adlandırılan bu probleme tam sayılı doğrusal ve sezgisel algoritmalar ile çözümler önerilmektedir [18,19].

<u>Tez kapsamında ele alınan problem ve araştırmanın amacı</u>

Geleneksel IoT ağlarında, sensörlerden elde edilen veriler, içerik veya uygulama türü dikkate alınmadan verileri belirlenmiş sunucuya göndermektedir. Bu nedenle, özellikle büyük ölçekli IoT ağları için IoT cihazları arasında iletilen veri miktarı, tıkanıklık, gecikme ve\veya daha düşük QoS hizmeti gibi istenmeyen durumlarla sonuçlanabilmektedir. Belirlenmiş sunucu konumlarının güncellenmesi gelişen ve değişen ağ topolojilerinde karşılaşılan zorluklardan bir diğeridir. SDN destekli ağlarda kontrolör ve sis kenar sunucuların konumu ağ iletişim performansını doğrudan etkilemektedir. Geleneksel ağlarda, sensörlerden elde edilen veriler doğrudan ağa iletilmekte ve hedef cihaza gönderilmektedir. Tez kapsamında önerilen modelde sensör verileri ağa, içeriğine göre iletildiğin hatalı/eksik verilerin iletimi ve bu verilerin gürültüden etkilenmesi engellenmektedir.

Geleneksel ağlarda, sensörlerden elde edilen verilerin tamamı ağa iletilmektedir. Sensörler türlerine ve yapılarına göre belirli zaman aralıklarında ölçüm yaparak veri üretmektedir.

Oysa veri analizinde tüm sensör verileri kullanılmayabilir. Önerilen (içerik olarak kullanılacak) veya veri analizi için belirli zaman aralıklarında üretilen veriler ağa iletilerek gereksiz verilen iletimi azaltılmaktadır ve bu büyük ölçekli ağlarda performansı doğrudan etkilemektedir.

Geleneksel ağlarda, ağda kullanılan cihazlar iletilen verinin içeriğini bilmemektedir. Ancak CCN ağlarda veri içeriği bilgisi cihazlar tarafından bilinmektedir. Önerilen modelde, verinin içeriği kontrolör ve anahtarlar tarafından bilinmektedir. Bu sebeple ağdaki gerçekleştirilebilecek sunucu konum değişimleri merkezi yazılım ile kolayca gerçekleştirilebilmektedir.

Sensörlerden elde edilen verilerin önem derecesi farklı olabilir. Veri analizi yöntemleri sonrası belirlenmiş eşik değerlerinden yüksek değerlere sahip verilerin iletilmesine öncelik tanımlanabilir. Acil durum karar verme mekanizmalarında bu yapı kolaylık sağlamaktadır. Örneğin, hava kalitesi ölçümünde birden fazla veri türü analiz edilerek hayat yaşam kalitesi ölçülmektedir. Ancak tek bir verinin çok yüksek çıkması hayat yaşam kalitesini doğrudan etkileyebilmektedir. Bu sebeple sensörlerden elde edilen verilerin belirlenmiş eşik değerinin üstünde olduğu zaman önceliklendirilmesi gerekmektedir. Bu önceliğe sahip verilerin ağ donanımlarını daha fazla kullanması sağlanarak etkin iletişim gerçekleştirilebilir ve acil durum senaryolarına cevap verilebilir.

Literatürdeki çalışmalar incelendiğinde, SDN destekli sis mimarisinde kurulan ağlarda kontrolör ve sunucu konumu performansı doğrudan etkilediği görülmektedir. Bu sebeple, daha önce önerilen SDN destekli içerik ve öncelik tabanlı iletişim modelleri için CPP ve SPP için optimizasyon çalışmaları gerçekleştirilmiştir.

CPP için ağlarda birden fazla kontrolör olduğu durumlarda hangi konumlara yerleştirileceğinin tespiti önemli sorunlardan biridir. Ek olarak, olası bir veya birden fazla kontrolörün hizmet sağlayamaması durumunda ağın sürekliliğini ve güvenilirliğinin sağlanması gerekmektedir. Ağda var olan anahtarların bağlı bulunduğu kontrolörün servis dışı kalması durumunda ağın iletişimi nasıl devam edeceği karşılaşılan zorluklardandır. Kontrolör ile bağı kopan anahtarların yeniden ağa dahil edilmesi için hiyerarşik kontrolör modeli geliştirilmiştir. Önerilen modelde, kontrolör çökme olasılığı dikkate alınarak

doğrusal bir model önerilmiştir. Önerilen modelin NP-zor gerçekleşmesi sebebiyle de sezgisel bir modelde önerilmiştir.

Sis sunucuları bağımsız iş veya görevler atanabilmektedir. Bazı sis sunucuları replika (kopya) sunucu olarak çalışmaktadır. Dağıtık veri işleme mimarileri sis sunucularında doğrudan gerçekleştirilebilir. Büyük ölçekli veri analizinde, işlemler farklı sis sunucularında gerçekleştirilebilir. Buda sis sunucularını ilişkili hale getirmektedir. Literatürde, SPP için yapılmış çalışmalar bulunmaktadır. Ancak ilişkili sunucuların konumlarının belirlenmesi ile ilgili yapılmış çalışmalar bulunmamaktadır. Tez kapsamında önerilen mimaride, sis sunucularında sensörlerden gelen verilerin işlenmesi sonucu yeni anlamlı bir veri üretilmektedir. Bu yeni veri bir diğer sis sunucusunun girdi verisi olabilmektedir. Bu da sunucuları ilişkili hale getirmektedir. Bu probleme önerilen tam sayılı doğrusal programlama çözümüne sezgisel çözümlerde üretilmiştir.

Tezin katkıları

Tez kapsamında, Yazılım tanımlı ağ tabanlı nesnelerin internetinde yönlendirme, kontrolör ve sunucu yerleştirme problemlerine beş farklı model ile çözüm önerileri sunulmuştur. Tezin katkıları dört farklı problem için ayrı ayrı ele alınmıştır.

SDN destekli ağlarda sensörlerden elde edilen verilerin içeriğine göre yönlendirilmesi sağlanmıştır. Bu yapı ağa gereksiz verilerin iletilmesini engellemektedir. Temel ağ cihazlarının iletilen paketin içeriğini bilmesini sağlamaktadır. Veri analizi, sunucu iş atama gibi problemlere karşı merkezi yönetim imkanı sağlamaktadır.

Sensör verilerinin türüne veya belirlenmiş eşik değerinden yüksek olmasına bağlı ağda iletim önceliği verilen mimari önerilmiştir. Öncelik atanması için veriler farklı kuyruklara eklenerek iletişimi gerçekleştirilmektedir. Bu kuyruklara SDN kontrolörü ile daha fazla bant genişliği sağlanarak ağda etkin ve hızlı iletişimi sağlanmaktadır. Kritik verilerin gecikmesi veya kaybolması (ağda düşmesi) engellenmiştir.

SDN destekli ağlarda veri iletişim sürekliliği ve ağ güvenilirliği önemli bir parametredir. Bu sebeple ağda kontrolörün servis dışı kalma ihtimali dikkate alınarak birden fazla kontrolörün

yerleştirilmesi için doğrusal ve sezgisel modeller önerilmiştir. Önerilen modeller SDN destekli IoT ağlarda simüle edilmiş ve etkinliği gösterilmiştir.

Sis hesaplama mimarisinde ilişkili sunucuların ağda en uygun konumlara yerleştirilmesi sağlanmıştır. İş/görev atanmış sunucuların, sunucu – anahtar ve sunucu – sunucu konumları dikkate alınarak tam sayılı doğrusal ve sezgisel modeller önerilmiştir. Önerilen modeller SDn destekli IoT ağlarda simüle edilmiştir ve etkinliği ortalama gecikme, atlama sayısı ve link verimi metrikleri ile gösterilmiştir.

SDN destekli IoT sensör ağlarında etkin iletişim için yeni bir model önerilmiştir. Literatürde ilk defa ilişkili sunucuların konumları tartışılmıştır. İçeriğe dayalı, önceliklendirmeli verilerin iletildiği ağda CPP ve SPP için çözüm önerileri derlenmiş, yeni çözümler geliştirilmiş ve simüle edilmiştir. Önerilen modellerin ağ performansına etkisi ve literatüre katkısı farklı metriklerle ölçülerek gösterilmiştir.

Tezin Organizasyonu

Bölüm 2'de SDN teknolojisi hakkında detaylı bilgi verilmiştir. SDN destekli ağlarda içerik tabanlı yönlendirme ve öncelikli verilerin iletimi açıklanmıştır. SDN destekli ağlarda CPP ve SPP'nin terminolojisi, önemi ve literatürde önerilen çözümler belirtilmiştir.

Bölüm 3'te, Sis sunucularında öncelikli verilerin iletiminin önemi ile ilgili çalışmalar derlenmiştir. Önerilen modeller gerçek veri seti ile simüle edildiğinden toplanan veri seti açıklanmıştır. Bu bölümde içerik tabanlı yönlendirme ve önceliğe dayalı içerik tabanlı yönlendirme olmak üzere iki farklı model önerilmiş ve bu modeller detayları ile açıklanmıştır. Toplanan veriler ile önerilen modeller simüle edilmek için gerekli alt yapı kurulumu bir diğer alt bölümde detaylıca açıklanmıştır. Bu alt bölümde kullanılan SDN teknolojileri açıklanmıştır. Kullanılan kontrolör yazılımı ve akış protokolleri detaylıca gösterilmiştir. Bölüm 3'te önerilen modellerin karşılaştırılması, sağladığı avantajlar bölüm değerlendirmesi alt başlığında açıklanmıştır.

Bölüm 4'te hatalı/çökme olasılıklı kontrolör yerleştirme problemi için önerilmiş model açıklanmıştır. CPP için literatürde yapılan çalışmalar derlenmiştir. Geliştirilen modeli değerlendirmek için gerekli simülasyon ortamı kurulmuş ve detaylı açıklanmıştır. Ortaya

konulan probleme doğrusal programlama ve sezgisel modeller ile sunulan çözümler açıklanmıştır. Son olarak belirlenen hesaplama metriklerine uygun analiz sonuçları açıklanmış ve bölüm değerlendirmesi gerçekleştirilmiştir.

Bölüm 5'te ilişkili sis sunucuları konumu problemi ele alınmıştır. Literatürde yapılmış benzer çalışmalar derlenmiş ve önerilen çözüm yöntemleri açıklanmıştır. Önerilen yöntemleri test etmek için simülasyon ortamı kurulmuş ve açıklanmıştır. Daha sonra simülasyon sonuçları analiz edilmiştir. Son olarak analiz sonuçları bölüm değerlendirmesi alt başlığı altında tartışılmış ve değerlendirilmiştir.

Son olarak bölüm 6'da ortaya konan problemlere üretilen çözümler özetlenmiş ve değerlendirilmiştir. Çalışmanın literatüre katkısı açıklanmıştır.

2. YAZILIM TANIMLI AĞLAR, İÇERİK TABANLI VE ÖNCELİKLİ VERİLERİN İLETİMİ VE SUNUCU YERLEŞTİRME PROBLEMİ

Bu bölümde tez kapsamında kullanılan yazılım tanımlı ağlar, SDN destekli ağlarda içerik tabanlı ve öncelikli verilerin iletimi, kontrolör ve sunucu konumu problemleri açıklanmıştır.

2.1. Yazılım Tanımlı Ağlar (SDN)

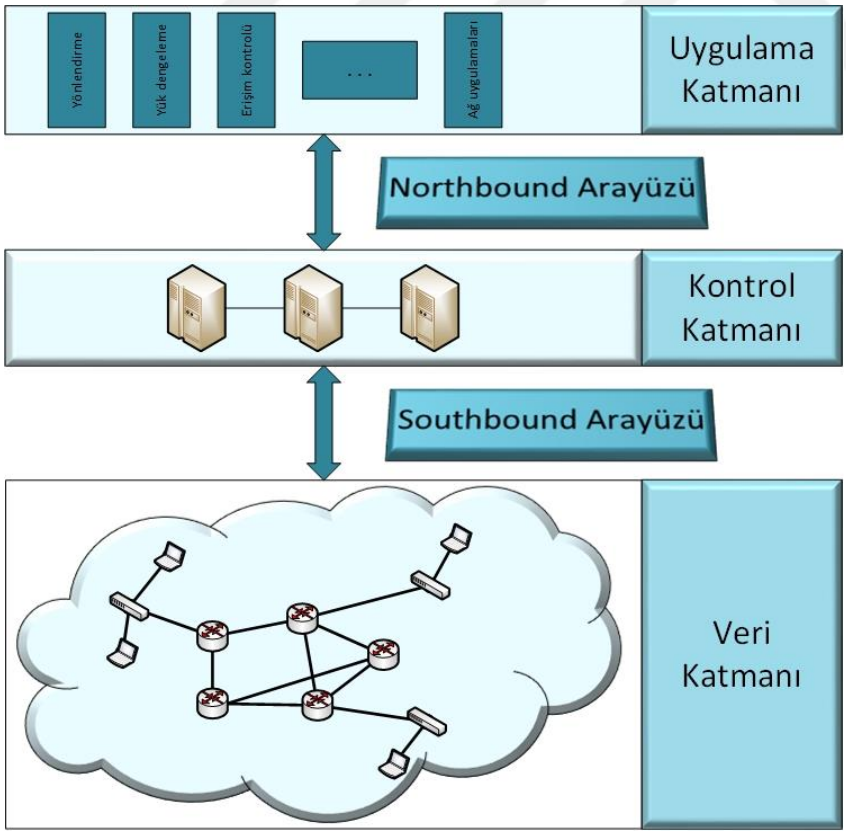
Kablosuz ağ teknolojisinin yaygınlaşması ve kullanıcı gereksinimlerin hızla artığı günümüzde var olan ağ alt yapısının eksikliklerinin giderilmesi için yeni teknolojiler geliştirilmektedir. Bu teknolojilerin başında ağ yapısının yönetilmesi ve sanallaştırmasını sağlayan yazılım tanımlı ağlar (SDN) ve Ağ Fonksiyonlarının Sanallaştırılması (Network Function virtualization, NFV) gelmektedir [1,20]. SDN terimi ilk olarak Stanford üniversitesinde geliştirilen OpenFlow protokolü ile ortaya çıktı [21]. SDN, veri düzlemi ile kontrol düzleminin ayrılması ve ağın merkezi bir kontrolör vasıtasıyla yönetilmesine dayalı bir mimari olarak geliştirilmiştir. NFV, ağ mimarisini esnek hale getirmek için donanımından bağımsız, sunucu merkezli sanallaştırma sağlar. SDN ile ağ yönetiminin sağlanmasının temel işlem adımlarından biridir. Yönlendirme, denetleme işlevlerini gerçekleştirmek için ağa esneklik kazandırmaktadır [22].

Literatürde yazılım tanımlı ağların en temel amacı, kontrol düzlemi ile veri düzleminin birbirinden ayrılması olarak ifade edilmektedir. SDN'de kontrol düzlemi ağı yöneten bileşen olduğundan ağın beyni olarak da tanımlanmaktadır. Var olan ağ alt yapısında, ağ anahtarları arasındaki haberleşme, gönderici paketin ulaştırılması gereken noktaya ulaşımının sağlanmasında kullanılması gereken yolun ağ cihazları tarafından varış adresine göre belirlenmesi ile gerçekleşmektedir. SDN tabanlı oluşturulan ağ mimarisinde, yönlendirme politikaları varış noktası yerine akış kontrolü dayalıdır. Kaynak ile hedef arasındaki iletişimde; akış soyutlaması, yönlendiriciler, anahtarlar, güvenlik duvarları vb. gibi ağ kurallarının birleştirilmesine olanak tanımaktadır [23]. Bu yapı ağda uygulanmış ve oluşturulmuş yönlendirme tablolarının aksine ağa büyük bir esneklik ve dinamiklik kazandırmaktadır.

Yazılım tanımlı ağların kontrol mantığı, SDN kontrolör veya Ağ İşletim Sistemi (Network Operating System, NOS) tarafından mantıksal olarak soyut bir ağ görünümüne dayanan

yönlendirme aygıtlarının programlanmasını sağlamaktır. NOS, yönlendirme aygıtlarını programlamayı kolaylaştırmak için gerekli kaynakları ve soyutlamaları sağlayan yazılım platformudur. Bu nedenle, geleneksel işletim sistemleri gibi çalışma mantığına sahiptir [24]. Ağı yöneten yazılımlar NOS'un üzerinde çalışır. Bu yapı SDN'in ağ alt yapısını yönetmesini sağlar.

Kontrol katmanının veri düzleminden ayrılması, ağda düşük düzeyli cihazların yapılandırılmasını, ağ politikalarının üst düzey diller ve yazılım bileşenleri ile yönetilmesini, veri transferinde oluşan sahteciliklere karşı yazılımsal çözümler üretilmesini ve ağ durumu ile ilgili bilgilerin kayıt altına alınmasını kolaylaştırmaktadır [24]. Şekil 2.1'de SDN mimarisi gösterilmektedir. Şekilde kontrol katmanı ile veri düzlemin ayrıldığı gösterilmiştir. Uygulama katmanında, SDN destekli ağlarda gerçekleştirilebilen yönetim uygulamaları gerçekleştirilmektedir. Veri katmanında, ağ cihazları bulunmaktadır. Southbound ara yüzü ile veri katmanında bulunan kontrolörler ile iletişime geçmektedir. Northbound ara yüzü, veri katmanı ile uygulama katmanı arasında iletişimi sağlamaktadır.



Şekil 2.1. SDN mimarisi

2.1.1. Kontrol katmanı

Programlanabilirlik, SDN tarafından sunulan temel paradigmadır. Ağ cihazlarının tüm bir ağının tek bir varlık olarak programlanmasına veya düzenlenmesine izin vermektedir. Başka bir deyişle, SDN kontrolörü, ağın dinamik olarak yeniden yapılandırma yeteneğine sahip mantıksal olarak merkezileştirilmiş bir zeka olarak hizmet eden kontrol düzlemini oluşturmaktadır. Kontrol düzlemi, ağ yapısının yönetilmesinde kullanılan katmandır. Bu sebeple ağın beyni olarak da tanımlanmaktadır. Yönlendirme cihazları, kontrol düzlemi elemanları tarafından programlanırlar. Tüm kontrol uygulamaları ve kontrolörler bu katmanda yer almaktadır. Geliştirilmiş birçok kontrolör yazılımı bulunmaktadır. Kullanılan yazılım alt yapısı, performans ve kullanıcı isteklerine göre farklılık göstermektedir. Literatürde NOX, POX, Floodlight, Beacon, DIFANE, OpendayLight (ODL) vb. gibi farklı kontrolör yazılımları mevcuttur [1,25-27].

Kontrol düzlemi, ağ denetimi ve çalışma mantığını uygulamak için sunulan işlevleri kullanan bir dizi uygulamayı barındıran katmandır. Yönlendirme aygıtlarının davranışını programlayan, özel talimatlar veren ve talimatları southbound API ile veri katmanına yönlendiren politikaları içermektedir.

2.1.2. Veri katmanı

Geleneksel ağ alt yapı ürünleri ve yazılım tanımlı ağ mimarisinde kullanılabilir cihazların bulunduğu katmandır. Var olan yönlendiricilere ek programlanabilir yönlendiricilerin bulunduğu ve kontrol düzlemi tarafından yönetilebilen alt yapı katmanıdır. Donanım veya yazılım tabanlı temel ağ işlevlerini yerine getiren birbirleri ile bağlantılı aygıtlarının bulunduğu katmandır. Yönlendirme aygıtları, gelen paketler üzerinde işlem yapmak veya yönlendirmek için tanımlanmıştır. Bu aygıtlar gelen paketleri, belirlenmiş portlara veya kontrolöre iletmek gibi işlemler yapar. Yönlendirme işlemini gerçekleştirmek için OpenFlow, ForCES, POF gibi farklı akış protokolleri mevcuttur ve bu protokoller kontrolörün ağ cihazlarını yönlendirmesini sağlar [28].

2.1.3. Southbound arayüzü

Yönlendirme aygıtlarının komutlarının veri katmanına iletildiği ve donanımın ile iletişimin sağlandığı geçiş katmanıdır. Yönlendirme aygıtları ile kontrol düzlemi elemanları arasındaki bağlantıyı sağlayan protokoldür. Bu amaçla kullanılan en önemli protokol OpenFlow protokolüdür. OpenFlow, kontrolör ile ağ donanım elemanları arasındaki veri akışını ve yönlendirme tablosunun güncellenmesini sağlar [21].

2.1.4. Northbound arayüzü

Ağ işletim sistemi ve bu işletim sistemi üzerinde çalışan uygulamalar ile kontrol katmanı elemanları arasındaki bağlantıyı kuran protokoldür. Uygulamalar ağı programlanmasına olanak sağlamaktadır. Yüksek seviyeli diller kullanarak tasarlanan akış kuralları ile kontrolör ile iletişime geçilir. Kontrolörde ağ cihazlarını yönlendirerek gerekli uygulamaları gerçekleştirir [23].

2.1.5. Uygulama katmanı

Uygulama katmanı, istenen ağ davranışını ve ağ gereksinimlerini SDN kontrolörüne açık ve programlı bir şekilde ileten programları içerir. Kontrolör yazılımı üzerinde gerçekleştirilen uygulamalara; yük dengeleme, güvenlik önlemleri, yönlendirme algoritmaları, içerik tabanlı yönlendirme, öncelikli verilerin yönlendirilmesi, ağ sanallaştırma uygulamaları örnek gösterilebilir [29].

2.2. SDN Destekli Ağlarda İçerik Tabanlı ve Öncelikli Verilerin İletimi

SDN destekli ağlarda, kontrolör yazılımı ile veriler içeriğine ve önceliğine göre iletilebilmektedir. Literatürde geliştirilmiş farklı yöntemler bulunmaktadır. Tezin bu bölümünde bu çalışmalar iki alt başlıkta açıklanmıştır.

2.2.1. İçerik tabanlı yönlendirme

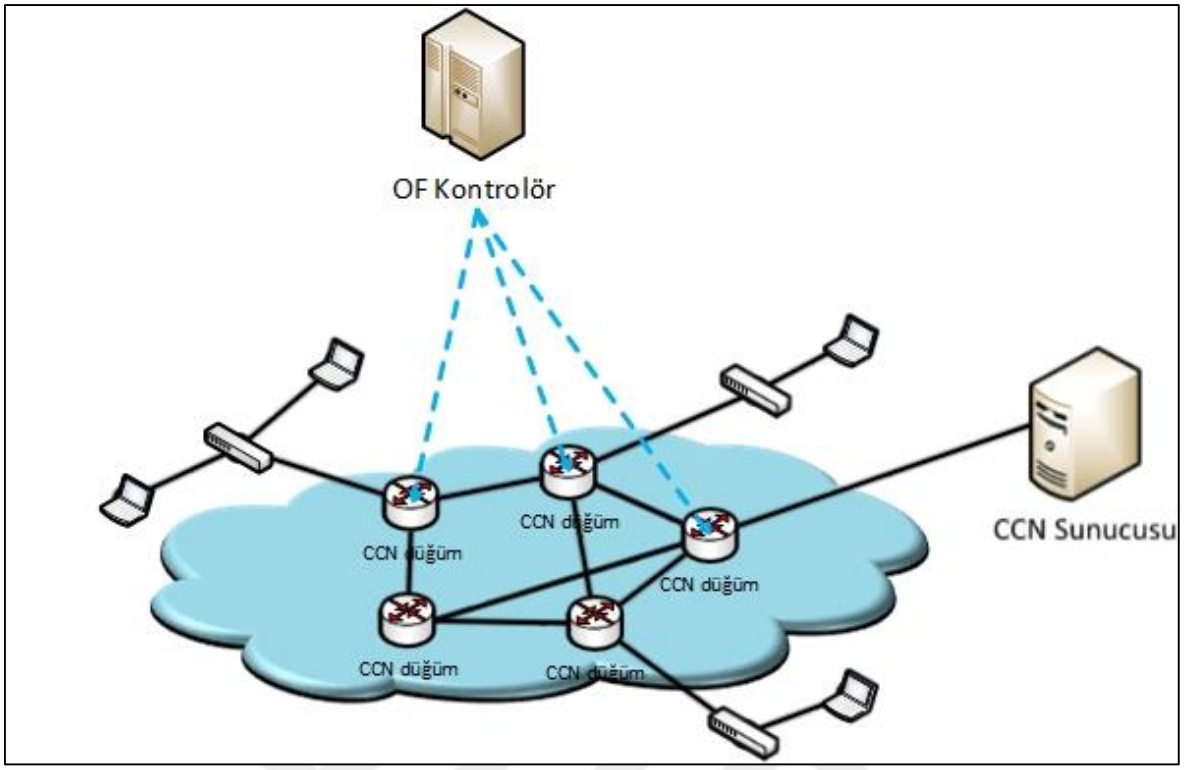
Literatürde SDN destekli ağlarda içerik tabanlı yönlendirme konusunda farklı isimlendirmeler ile gerçekleştirilmiş çalışmalar bulunmaktadır. Bunlar, İçerik Tabanlı Yönlendirme (Content Based Routing, CBR), İçerik Merkezli Ağ (Content Centric Network,

CCN), Bilgi Merkezli Ağ (Informatin Centric Networking, ICN), Adlandırılmış Veri Ağı (Named Data Networking, NDN) çalışmalarıdır. Her bir uygulamanın birbirinden farklı yönleri bulunsa da SDN çalışma mimarisi olarak benzerdirler. Şekil 2.2'de içerik tabanlı yönlendirme mimarisi verilmiştir. Kontrolör tarafından yönetilen ağlarda, içerik yönetimi harici bir sunucu ile gerçekleştirilen iletişim sayesinde gerçekleştirilir.

ICN, bilgi ve içeriği belirleyen ve yönlendirme kriteri olarak mevcut hedef adresinden ziyade içerik adını dikkate alan yeni bir ağ paradigmasıdır. SDN, esnek ve sürekli yönetime sahip bir ağ sağladığı için ICN'nin uygulanabilirliğini sağlamakta ve kolaylaştırmaktadır. ICN, içerik tabanlı yönlendirme için geliştirilmiş yeni bir paradigmadır. Bu paradigma, kullanıcıların istediği içerik veya bilgilere odaklanır. Hedef adres yerine içeriği göre yönlendirme gerçekleştirir. Ayrıca, içerik veri trafiğini sağlayan yönlendirici, içeriği önbelleğe alarak aynı içeriğe yönelik istekler oluştuğunda hızlı bir şekilde yanıt verebilir [30-33]. ICN ağlarında içeriği tanımlamak ve yönetmek için farklı bir sunucuya ihtiyaç vardır [34].

CCN, içeriğin doğrudan yönlendirilebilir ve adreslenebilir olmasını sağlayarak içerik dağıtımına sağlar. Temel amaçları, ağ ekipmanının ana bilgisayar adresi yerine içeriğin adına bağlı olarak yönlendirme kararları almasına izin vererek geleneksel ağ operasyonlarını geliştirmektir. IPv4/6'dan farklı olarak, CCN adları hiyerarşik olarak yapılandırılmıştır. Bu adlar, değiştirilebilir ve sınırsız uzunluklara sahiptir, bu da hızlı ad aramasını yeni bir sorun haline getirir [35,36]. CCN ağları, kontrolör ve OpenFlow akış protokolleri arasında bir CCNx sunucu modülüne sahiptir. Denetleyici iletişimi tanımlarken, içerik sağlayıcı CCNx sunucularına ihtiyaç duyar [37].

NDN, farklı konumlarda üretilen devasa miktarda verinin işlenmesinde devrim yaratan yeni bir ağ mimarisidir. NDN, konum tabanlı İnternet ağ sistemleri zorluklarını azaltmak için en faydalı özellik olan ağ içi bir önbellek sunar. Sisteme dahil edilen ara bellek sunucusu, ağda kullanıcıların sıklıkla kullandığı veri veya uygulamaları ön belleğe almaktadır. Bu mimari, ara belleğe alınan verilere diğer kullanıcıların doğrudan ve hızlı bir şekilde ulaşmasını sağlar. Bu yapı ağda tıkanıklığını azaltır. Ayrıca veri indirme prosedüründe ara bir bellek oluşturarak avantaj sağlar [38,39].



Şekil 2.2. SDN destekli ağlarda içerik tabanlı yönlendirme mimarisi

Sonuç olarak literatürde içerik tabanlı yönlendirme konusunda farklı öneriler ve uygulamalar bulunmaktadır. SDN destekli ağ yapısı sorunlarına farklı çözümler üretmektedir. Ortak görüşe göre, içerik tabanlı yönlendirme, kontrolör ve akış protokolleri arasına yeni bir referans sunucusu yerleştirilerek gerçekleştirilir. Sunucu, içeriğin bulunduğu konumların IP adresleri gibi fiziksel adres bilgileri tutar. Kontrolör, içerik verilerine dayalı olarak yönlendirme için sunucuya başvurmalıdır.

2.2.2. Öncelikli verilerin iletimi

SDN destekli IoT ağlarda kaynak yönetimi, ağ performansı için önemli parametrelerden biridir. Ağ cihazları ve kaynakları kontrolör desteği ile kullanıcı veya uygulama gereksinimleri karşılamak için optimize edilebilir. Bu sebeple ağda önceliğe sahip veri veya uygulamalar için farklı yöntemler geliştirilmiştir. Öncelikli verilerin iletilmesinde farklı akış kuralları ve kuyruk yapıları oluşturulur. Akış kuralları ile ağ kaynakları tahsisi gerçekleştirilirken, kuyruk yapısı oluşturularak iletişim önceliği tanınır.

Ağda öncelik tanımlaması gerçekleştirilerek kritik verilerin iletilmesi sorununa çözümler üretilmekte ve ağ performansı artırılmaktadır. Öncelik tanımlama sadece bir veri setinin

değerinden kaynaklı yapılmamaktadır. İşlem önceliği olan paketlerinde bu mimaride iletilmesi ortalama gecikme ve iş akış sürecini doğrudan etkilemektedir. SDN, öncelikli iletişim için en verimli ve yaygın kullanılan teknolojidir [40]. Veri iletiminde yoğun tıkanıklık ve düşük servis kalitesine sahip bir üniversite kampüs ağı için SDN tabanlı bir kontrol düzlemi geliştirilmiştir. Önemli olarak tanımlanan veri paketlerine öncelik verilerek, önemli hizmetler için daha yüksek servis kalitesi sağlanmıştır [41].

2.3. SDN Destekli Ağlarda Sunucu ve Kontrolör Yerleştirme Problemi

SDN destekli IoT ağlarda sunucu ve kontrolör konumu ağ performansını doğrudan etkilemektedir. Çalışmamızın bu alt bölümünde CPP ve SPP sorunları ve çözüm önerileri açıklanmıştır.

2.3.1. Kontrolör yerleştirme problemi

SDN destekli ağlarda kontrolörün konumu ağ performansını doğrudan etkilemektedir. İlk araştırmalar, tek bir denetleyicinin gerçek ağlar için yeterli olacağını gözlemledi [42]. Ancak, yapılan çalışmalarda ölçeklenebilirlik, güvenlik ve esneklik gibi sebeplerden kaynaklı birden çok denetleyiciye olan ihtiyaç ortaya çıkmıştır [43]. Yapılan çalışmalarda SDN'nin önemi ortaya konulmuştur ancak CPP gibi birçok zorluk hala devam etmektedir. CPP, bir veya birden fazla kontrolörün ağda nasıl yerleştirileceğinin belirlenmesi problemidir. Belirli sayıda düğüme sahip bir ağda, kontrolör sayısı ve konumu açık bir sorundur. Bu sorunu çözmek için genellikle gecikme, esneklik, güvenilirlik, enerji tasarrufu ve yük dengeleme gibi ölçütler kullanılmaktadır.

Literatürde CPP problemi ele alınırken ortalama gecikme süresini en aza indirgenmesi amaçlanmıştır. Kontrolörün konumu ağda gecikmeyi, enerji tüketimini ve kaynak kullanımını doğrudan etkilemektedir. Birden fazla kontrolörün yerleştirilmesi ağda esnekliği, güvenilirliği ve ağ sürekliliğini etkilemektedir. Olası bir kontrolörü hizmet sağlayamadığında diğer kontrolörlerin devreye girmesi ağda güvenliği, esnekliği ve sürekliliği sağlamaktadır. Çok kontrolör bulunduran ağlarda, her kontrolöre farklı görevler verilerek ağda dinamiklik ve esneklik elde edilebilir. Literatürde CPP için genellikle doğrusal programlama ile çözümler üretilmiştir. P-medyan, k-medyan gibi kümeleme

algoritmaları temel alınarak doğrusal modeller önerilmiştir [42,44,45]. Önerilen bu modeller genellikle NP-zor çıktığından sezgisel modeller geliştirilmiştir [46-49].

2.3.2. Sunucu yerleştirme problemi

Nesnelerin İnterneti teknolojisinin hızla gelişmesiyle, ağın ucunda oluşturulan büyük veriler, merkezi bilgi işlem merkezinin büyük miktarda veriyi verimli bir şekilde işlemesini ve kullanıcıların gerçek zamanlı gereksinimlerinin karşılanmasını zorlaştırmaktadır. Bu zorlukların üstesinden gelmek için uç bilgi işlem teknolojisi ortaya çıktı. Gerçek zamanlı iş ihtiyaçlarını sağlamak için merkezi bilgi işlem yeteneklerini ağın ucundaki sunuculara indirir. Uç bilgi işlem, sis hesaplama, bulutçuk (cloudlet) vb. gibi teknolojiler ile gerçekleştirilmektedir. Bu teknolojilerde sunucu konumlarının belirlenmesi performansı doğrudan etkilemektedir [50,51]. SPP, var olan veya yeni oluşturulacak olan ağ alt yapısında sunucuların konumlarının belirlenmesi problemdir [52]. Kenar, kontrolör, merkezi ve bulut sunucularının konumları SPP kapsamındadır.

Literatürde SPP çözümü için farklı kısıtlar dikkate alınmıştır. Ağ performansını iyileştirmek için önerilen modellerde, yük dengeleme, gecikme zamanı, bant genişliği, enerji tüketimi, donanım kaynakları kullanımı vb. kısıtlar kullanılmıştır. SDN destekli ağlarda SPP'ye tam sayılı doğrusal programlama ile çözümler önerilmiştir. Ancak genellikle NP-zor sonuç elde edildiğinden sezgisel algoritmalar ile çözümler önerilmiştir. Literatürde, p-medyan, açgözlü, k-medyan vb. gibi algoritmaları temel alan sezgiseller geliştirilmiştir [18,53-55].

3. SİS SUNUCULARINDA İÇERİK TABANLI VE ÖNCELİKLİ VERİLERİN SDN DESTEKLİ AĞLARDA İLETİMİ İÇİN YÖNLENDİRME YÖNTEMLERİ

Bulut bilişim sistemleri, IoT verilerinin yüksek hızını, hacmini ve boyut sayısını ele almak için tasarlanmamıştır. Sınırlı bilgi işlem ve depolama yetenekleri nedeniyle gelecek vaat eden IoT teknolojilerinin taleplerini karşılayamamıştır. Bu sistemlerde, büyük miktarda veriyi tek bir merkezi sunucuya iletmek, trafik yoğunluğunda artış, tıkanıklık, gecikme, bant genişliği darboğazı, hizmet kalitesinde azalma ve güvenlik gibi istenmeyen sonuçlara neden olmaktadır [2,3]. Buna ek olarak, IoT uygulamaları, gerçek zamanlı olarak işlenmesi gereken coğrafi olarak dağıtılmış, geçici ve anlık veriler üretmektedir ve literatür, bulut bilişimin bu gereksinimleri karşılayamayabileceğini göstermektedir [4,5]. IoT sensör verilerini işlemek için dağıtılmış bir yaklaşıma ihtiyaç vardır, bu nedenle sis hesaplama kavramı bulut bilişim için yardımcı bir paradigma olarak ortaya çıkmıştır [6,7].

Sis hesaplama, bilgi işlem, depolama ve ağ oluşturma süreçlerini ağın kenarına taşıyarak bulut bilişim için ek özellikler sunan yeni bir teknolojidir. Uygulamalarını ve hizmetlerini son kullanıcılara daha yakın bir yere dağıtarak gecikmeye ve konuma duyarlı uygulamaları destekleyen tamamlayıcı bir teknolojidir [56]. Sis sunucuları, bulut sunucusunun ve son kullanıcıların ortasında bulunur ve bulut sunucusu üzerindeki yükü hafifletmek için verileri sunucuya iletmeden önce ön işleme tabi tutarlar [3,9,57].

Tüm verilerin ağ üzerinden iletilmesi yerine, sis sunucularında toplanması veya veri türlerine göre önceliklendirilmesi, trafik yoğunluğunu büyük ölçüde azaltabilir ve ağın performansını artırabilir [58-60]. Örneğin, sağlık hizmetleri veya hava kalitesi izleme gibi bazı durumlarda, sis sunucuları belirli bir hizmet türü için özelleştirilebilir. Bu gibi durumlarda, coğrafi olarak dağıtılan IoT cihazları tarafından üretilen ilgili veriler, ağ üzerinden söz konusu hizmet için uzmanlaşmış ilgili sis sunucusuna yönlendirilmektedir. Bu amaçla, verileri ilgili sis sunucularına iletmek için öncelik tabanlı yaklaşımlar kullanılabilir. Veri aktarımı sırasında öncelik dikkate alınarak, önemli verilerin ilgili sis sunucusuna daha hızlı ulaşması sağlanır. Daha düşük indirme gecikmeleri, daha az bant genişliği kullanımı, gelişmiş içerik kullanılabilirliği ve düşük maliyetle sis hizmetlerinin sürdürülebilirliğinin sağlamasına yardımcı olur [61].

IoT cihazlarından ilgili sis sunucularına veri aktarımı sırasında, ağa bağlı çok sayıda cihaz nedeniyle topoloji değişiklikleri meydana gelebilir. Çevresel koşullara bağlı olarak, bazı etkin düğümler başarısız olabilir veya yeni düğümler ağa katılıp etkinleşebilir. Bu ve benzeri problemlerde ağda anında dinamik değişiklikler meydana geldiğinden, ağın yönetimi veya kontrolü geleneksel yöntemlerle oldukça zorlaşmaktadır [62]. SDN, ağı veri düzlemi ve kontrol düzlemi olarak ayırarak, daha esnek ve yönetilebilir hale getirmek amacıyla bu soruna umut verici bir çözüm olarak sunulmaktadır. Dağıtılmış sis sunucularının merkezi kontrolünü sağlarken, öncelik yönetimi, hata kontrolü, enerji ve gecikme optimizasyonlarını destekler [24].

Tezin Bu bölümünde, Sis hesaplama ve SDN teknolojilerini kullanan IoT ağlarında öncelik tabanlı veri iletme problemini ele alınmıştır. İki yeni içerik tabanlı veri iletme modeli önerilmiştir. Bu modelleri gerçekçi bir IoT tabanlı hava kalitesi izleme ağının ağ akışını kullanarak uyguladık. Şebeke akışı, sadece Türkiye'nin değil, aynı zamanda dünyanın en kalabalık şehirlerinden biri olan İstanbul'un gerçek hava kalitesi izleme veri seti kullanılarak oluşturulmuştur [63]. Geleneksel IoT ağlarında, toplanan veriler genellikle içerik veya uygulama türü dikkate alınmadan tek bir sunucuya iletilir. Önerilen modellerde trafik yoğunluğu ve bant genişliği darboğazı dikkate alınarak her biri belirli bir veri tipine ve/veya uygulamaya tahsis edilmiş sis sunucularına iletilir. Sis sunucular ana bulut sunucusuna bağlıdır ve farklı türde verilerin üretilmesinden sorumlu IoT sensörleri bulunmaktadır. Ayrıca, hangi verilerin hangi sis sunucusuna yönlendirileceğini yönetmekten sorumlu bir SDN kontrolörü bulunmaktadır.

SDN destekli İçerik tabanlı yönlendirme (Content Based Forwarding via SDN, CBF-SDN) adı verilen önerilen ilk modelimizde, SDN denetleyicisi, IoT veri içeriğine dayalı olarak yönlendirme kurallarını belirler. Bu yaklaşımın amacı, aynı tür verileri tek bir sis sunucusunda toplamak ve böylece hesaplama gecikmesini azaltmaktır. Önerilen ikinci model, SDN Destekli Öncelikli Verilerin İçerik Tabanlı Yönlendirilmesi (Priority Content Based Forwarding via SDN, PCBF-SDN), IoT verileri işlenirken iki tür kuyruk yapısı kullanılır. İlk modelde, tüm IoT verileri eşit kabul edilir ve ilgili sis sunucularına gönderilir. Önerilen ikinci modelde ise, uygulamaya ve/veya veri türüne göre IoT verilerine öncelik verilir. Bu önceliklendirmeye bağlı olarak, kritik verilerin etkin bir şekilde ve daha az gecikmeyle taşınabilmesini sağlanmaktadır. PCBF-SDN herhangi bir acil durumda sis sunucuları ile eş zamanlı ana sunucuya verinin bir kopyasını gönderir. Buradaki amaç, acil

bir durumda bulut sunucusu üzerinde hızlı karar vermektir. Önerilen modellerin avantajlarını sunmak için gerçekçi bir ağ akışı kullanılarak kapsamlı simülasyonlar gerçekleştirilmiştir. Modellerin sağladığı avantajlar tartışılmış ve açıklanmıştır.

3.1. İlgili Çalışmalar

SDN destekli ağlarda kaynak yönetimi yönlendirme ve veri önceliğine göre yapılır. Kaynak yönetimi, SDN destekli bulut sistemlerinde ağ hizmet kalitesini sağlamak için önemli bir parametre olduğu literatürde vurgulanmıştır. Farklı akış kuralları ve kuyruk yapıları oluşturmak, öncelikli verilere daha fazla kaynak tahsis ederek performansı artırır [40]. Ağ akışlarının ve paketlerinin önceliklendirilmesi, çok sayıda ağ performansı sorununun üstesinden gelmek için kullanılır.

Lara ve Quesada [41], yoğun tıkanıklık ve düşük QoS sorunlarından muzdarip bir üniversite kampüsü için SDN tabanlı bir kontrolör yazılımı geliştirilmiştir. Önemli olarak tanımlanan veri paketlerine öncelik verilerek, önemli hizmetler için daha yüksek QoS seviyeleri elde edilmiştir. Kampüs ağında trafik üç ayrı kuyruk yapısına bölünmüştür ve her bir kuyruğa önceliklerine göre farklı veri indirme hızları tanımlanmıştır. Öncelik tabanlı yönlendirme problemine en uygun çözümü bulmak için doğrusal programlama modeli geliştirmişler. İki farklı kampüs ağı önerilmiştir ve Mininet ortamında simülasyonunu gerçekleştirmişlerdir. Sonuçlar, SDN destekli ağlarda öncelikli verilerin iletiminin başarılı olduğunu göstermektedir. Yaşanan darboğaz ve ortalama veri hızında %40'lara varan iyileştirmeler elde edilmiştir.

Oh ve arkadaşları [28], SDN kontrollü ağlarda akış kurallarının dinamik değişimi sırasında meydana gelen iletim kesintileri ve paket kayıpları için bir çözüm önermişlerdir. Önerilen yöntem, önceliğe dayalı akış kontrolü diye adlandırmış ve kesintisiz veri akışları sağlamaktadır. Akış kurallarına öncelik vererek, önerilen bu yöntem akış değiştirme sürecinin karmaşıklığını azalmıştır. Bu nedenle, kontrol düzlemi işlemlerinin zaman karmaşıklığını ve paket kaybını azalttığı gözlemlenmiştir. SDN destekli anahtarlar için dinamik ve kayıpsız trafik kontrolü sunmuştur.

Lu ve arkadaşları [64], SDN destekli ağlarda acil durumlar için ağ kaynaklarının kullanımını optimize eden bir model önermişlerdir. Acil durumlarda öncelikli olan süreç veya veri iletimi

için QoS sağlanması amaçlanmıştır. Operasyonların öncelik ve temel gereksinimlerine göre ağı çoklu akış yollarına bölmektedir. Ağda trafik yükünü dengeleyen model akış kontrol mekanizması ile desteklenmiştir. Önerilen modelde, çok amaçlı optimizasyon ile düğüm ve önceliklendirme gerçekleştirilmiştir. Ağda farklı rotalar oluşturulmuş ve her bir rotaya yük dengelemeye uygun bant genişliği tahsis edilmiştir. Önerilen modelin farklı iletişim yollarını hesaplayarak maksimum verim sağladığı deneysel sonuçlar ile gösterilmiştir.

Shinkuma ve arkadaşları [65], SDN destekli ağlarda gerçek zamanlı verileri öncelikli iletmeyi sağlayan bir model önermişlerdir. Önerilen modelde, makine öğrenmesi yöntemleri kullanılarak verilerin ve akışların önemi belirlenmiş ve matematiksel optimizasyonlar gerçekleştirilmiştir. Yapılan çalışmada, veriye veya akışa bir önem puanı belirlenmiştir ve bu puan verinin önceliğinin belirlenmesinde kullanılmıştır. Önerilen model mobil trafik log kayıtlarından oluşan gerçek bir veri seti kullanılarak simüle edilmiştir. Önerilen model ile rastgele seçim yapılan model kıyaslanmış ve başarısı gösterilmiştir. Önerilen modelde kontrolör yazılımını geliştirmek için makine öğrenmesi teknikleri kullanılmıştır.

Qin ve arkadaşları [66], heterojen kablosuz ağ senaryolarında farklı IoT görevleri atanmış sis sunucularına, dinamik olarak farklılaştırılmış kalite seviyelerine ulaşmak için yazılım tanımlı bir ağ modeli tasarlamışlar. Bu model, katmanlı bir SDN denetleyicisi ile gerçekleştirilmiştir. Önerilen model Multinetwork Information Architecture (MINA) mimarisinin geliştirilmesi ile gerçekleştirilmiştir. Önerilen model, görev bazlı heterojen geçici yollar sağlayarak akış planlaması gerçekleştirmektedir. Mevcut IoT ağlarında kullanımı için ağ analiz ve genetik algoritmalar kullanılmıştır. Önerilen genişletilmiş MINA, elektrikli araçların, elektrikli şarj sitelerinin ve akıllı şebeke altyapılarının geniş ölçekli entegrasyonu senaryosunda uygulanmıştır. Simülasyon sonuçları, Önerilen modelin, IoT çoklu ağ yeteneklerinin verimli bir şekilde kullanılmasını destekleyebileceğini göstermiştir. Deng ve Wang [67], SDN destekli ağlarda sis sunucularında gerçekleştirilen uygulamaya duyarlı yeni bir model önermişlerdir. Önerilen model Qin ve arkadaşlarının [66], önerdiği MINA modelinin geliştirilmesi ile elde edilmiştir ve Apalication-aware QoS Routing Algorithm (AQRA) diye isimlendirilmiştir. AQRA, yüksek öncelikli IoT uygulamalarının gereksinimlerini karşılamak için geliştirilmiştir. MINA'ya oranla uçtan uca akış performansı, gecikme ve paket kaybı oranları sırasıyla %10,75, %11,88 ve %10,82 oranlarında iyileştiği belirtilmiştir. Çalışma süresi olarak AQRA %38,56 oranında MINA'dan daha kısa sürede gerçekleşmektedir.

Diro ve arkadaşları [68], ağ performansını artırmak için SDN destekli ağlarda öncelik tabanlı yeni bir model önermişlerdir. Makalede, farklı öncelik seviyelerine sahip IoT verileri için farklı akış yolları oluşturulmuştur. IoT verilerinin aktarılacağı akış şeması, verilerin önceliğine göre belirlenmiştir. Öncelikli veriler doğrudan iletilirken, öncelikli olmayan veriler ayrı bir kuyrukta tutulur. Gerçekleştirilen simülasyonda, geleneksel ağ mimarisi ile öncelikli ve öncelikli olmayan verilerin iletişimi verim, paket kayıp oranı metrikleri ile kıyaslanmıştır. Elde edilen sonuçlar SDN destekli ağlarda öncelikli verilerin iletişiminde başarılı sonuçlar ortaya koymuştur. Önerilen model, acil (öncelikli) paketleri ileterek verim ve kayıp oranı avantajları elde etmiştir.

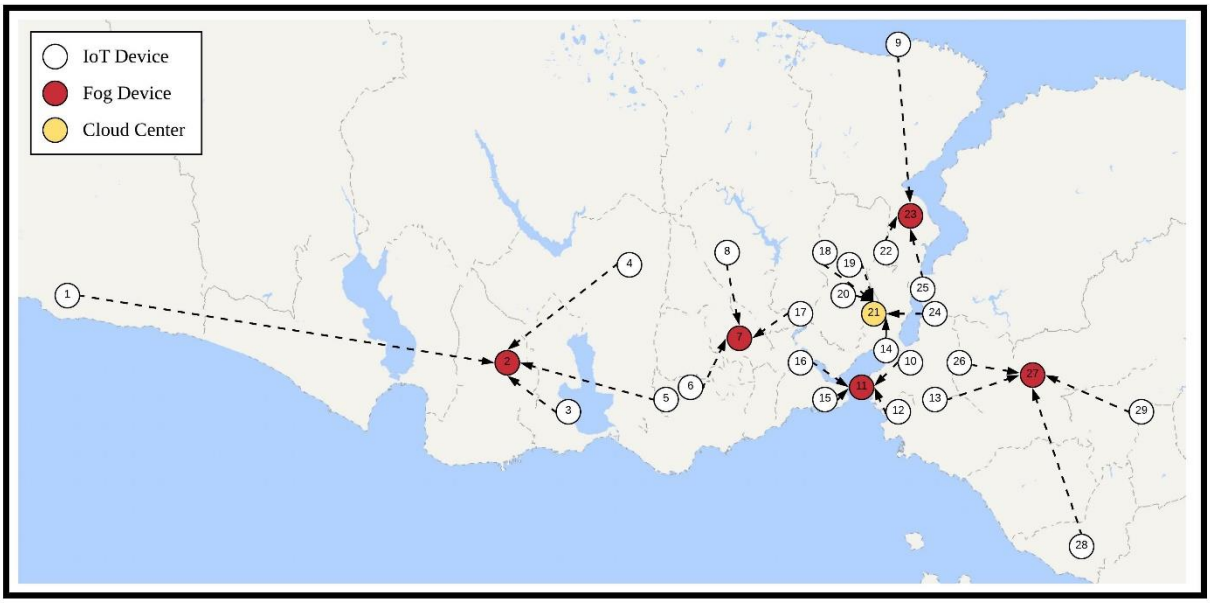
Bardalai ve arkadaşları [69], IoT cihazlarından elde edilen verilerin sağlık sistemlerindeki önceliğine ve önemine göre OpenFlow (OF) akış kontrollü yönlendirme yöntemleri önermişlerdir. Yazarlara göre, Covid-19 sonrası sağlık verilerinin analiz edilmesi ve iletilmesinin önemi oldukça artmıştır. Bu nedenle, bu veriler veri kaybı olmadan ve öncelikli olanların ağ altyapısında minimum gecikme ile iletilmesi sağlık verileri analizi için kritik öneme sahiptir. Sağlık verilerini işlemek için OF akış kuralları ile yönetilen kuyruk yapısı kullanan OF tabanlı bir trafik şekillendirme modeli olan OpenHealthQ önerilmiştir. OpenHealthQ, ağın ucundaki SDN tabanlı sis düğümlerine ve dağıtılmış bulut mimarisinde tasarlanmış bilgi işlem altyapısına güvenli, isteğe bağlı (öncelikli) ve düşük maliyetli erişim sağlamaktadır. İletişim, verinin önem derecesine göre çoklu kuyruklardan biri üzerinden gerçekleşmektedir. Önerilen model, Best Effort yaklaşımı ile karşılaştırılmıştır. Simülasyon sonuçlarına göre, yüksek öncelikli verilerin iletiminde verim ve gecikme iyileştirilmiştir.

Ghazi ve arkadaşları [70], acil durum mesajlarına öncelik veren ve bunları ağa ileten Araçlar Arası İletişim Ağları (Vehicular ad-hoc network, VANET) tabanlı çalışmaları incelemiş ve derlemişlerdir. Çalışmada, öncelikli verilerin ayrıcalıklı iletilmesinin önemi belirtilmiş ve VANET uygulamalarının simüle edilebileceği NS2, NS3, mininet, Qualnet, MATLAB, NetSim ve OMNet++ uygulamaları tanıtılmıştır. Ağlarda artan veri trafiği, öncelikli veri iletiminin önemini ortaya çıkarmaktadır. Ayrıca, kontrolör yazılımının dinamik olarak uygulanabilmesi büyük bir avantaj sağlamaktadır. Bu nedenle ağlar değişen veri önceliklerine ve sistemlerine göre yeniden düzenlenebilir. SDN destekli araç ağlarında, acil ve öncelikli durumlarda önceliğe dayalı çok yollu modelin kullanılması önerilmektedir. Çok yollu kuyruk modelin uygulanmasıyla ağ sürekliliği, esneklik ve hizmet kalitesinde artış sağlanır [71,72].

3.2. Veri Seti

Önerilen modellerin etkinliğini göstermek için gerçekçi bir uygulama senaryosu geliştirilmiştir. Türkiye Çevre, Şehircilik ve İklim Değişikliği Bakanlığı veri portalından sağlanan hava kalitesi verileri kullanılarak bir veri seti oluşturulmuştur. Bakanlık tarafından izlenen kirleticiler arasında azot dioksit (NO2), kükürt dioksit (SO2), 10 mikrometre boyutundan küçük partikül madde (PM10), ozon (O3) ve karbon monoksit (CO) seçilmiştir.

Elde edilen bu veri seti kullanılarak bir hava kalitesi tahmin senaryosu oluşturuldu. Veri seti işlenerek gerçekçi bir sensör ağ akışı oluşturuldu. Ardından, önerilen modelin etkinliğini göstermek için önerilen modeller kullanılarak ağ akışı simüle edildi. Türkiye Çevre, Şehircilik ve İklim Değişikliği Bakanlığı ülke geneline yayılmış 339 hava kalitesi izleme istasyonundan oluşan bir hava kalitesi izleme ağına sahiptir. İstanbul, Türkiye'nin en kalabalık şehri ve en fazla istasyon sayısına sahip olduğu için çalışmamızda İstanbul hava kalitesi ölçüm verileri kullanılmıştır. Sonuç olarak, Şekil 3.1'de gösterildiği gibi, çalışma için İstanbul'dan 29 istasyonun hava kalitesi verileri seçilmiştir. Şekilde beyaz daireler IoT cihazlarını, kırmızı düğümler sis cihazlarını ve sarı düğüm bulut merkezini simgelemektedir.



Şekil 3.1. İstanbul hava kirliliği sensör konumları

Veri analizi sonucunda hava kalitesini belirleyen veriler etki değerlerine göre sınıflandırılmaktadır. Daha yüksek kirletici konsantrasyonları insanlar için daha yüksek risk oluşturmaktadır. Çizelge 3.1'de gösterildiği gibi, hava kalitesini belirleyen verilerin değer

aralıkları 5 seviyeye ayrılmıştır. Seviye 1 en düşük etkiyi, seviye 5 ise en yüksek etkiyi göstermektedir. Çizelge 3.2'de toplanan 2.300.681 paket verinin sayısal değerleri ve hangi sis sunucusunda olduğu gösterilmektedir.

Çizelge 3.1. Verilerin etki değer aralıkları

	Seviye 1		Seviye 2		Seviye 3		Seviye 4		Seviye 5	
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
CO	0	4999	5000	9999	10000	14999	15000	19999	20000	~
NO2	0	49	50	99	100	149	150	199	200	~
O3	0	59	60	119	120	179	180	239	240	~
PM10	0	24	25	49	50	74	75	99	100	~
SO2	0	174	175	349	350	524	525	699	700	~

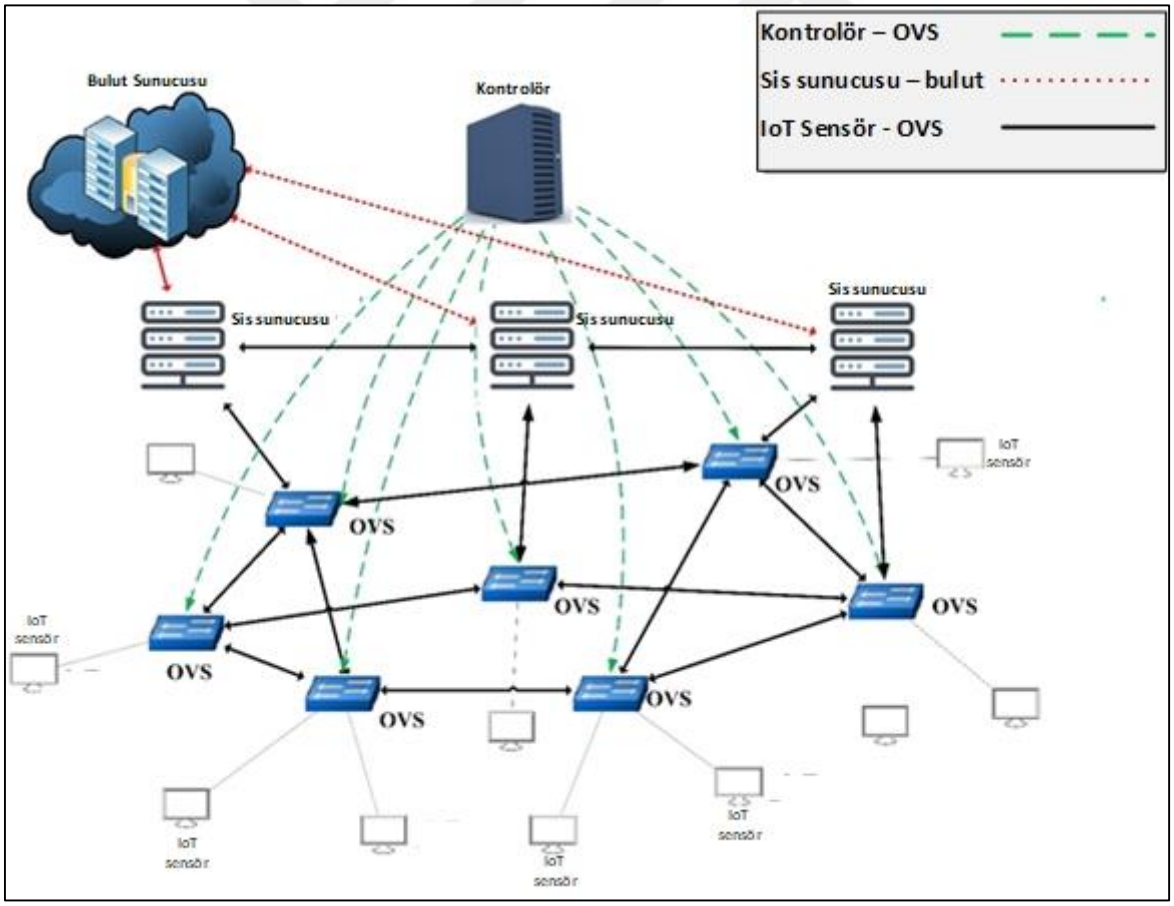
Çizelge 3.2. Sensörlerden toplanan verilerin öncelik seviyeleri ve paket sayıları

	Kirletici madde	Öncelik Seviyesi				
		Seviye 1	Seviye 2	Seviye 3	Seviye 4	Seviye 5
Sis sunucusu 1	CO	165 878	0	0	5	0
	NO2	43 550	17 884	4 676	406	31
	O3	31 570	876	0	0	18
	PM10	55 918	19 289	4 677	1 340	841
	SO2	49 497	0	0	0	1
	Toplam	346 413	38 049	9 353	1 751	891
	Oran (%)	87,38	9,60	2,36	0,44	0,22
Sis sunucusu 2	CO	32 546	0	0	1	0
	NO2	159 883	61 251	9 837	802	173
	O3	29 459	2 486	8	0	9
	PM10	36 275	9 541	1 904	482	220
	SO2	0	0	0	0	0
	Toplam	258 163	73 278	11 749	1 285	402
	Oran (%)	74,86	21,25	3,41	0,37	0,12
Sis sunucusu 3	CO	16 680	0	0	0	0
	NO2	52 497	11 339	2 131	245	27
	O3	118 354	24 951	580	5	763
	PM10	37 968	14 235	7 351	3 088	2 757
	SO2	65 402	0	0	0	0
	Toplam	290 901	50 525	10 062	3 338	3 547
	Oran (%)	81,17	14,10	2,81	0,93	0,99
Sis sunucusu 4	CO	49 393	0	0	1	0
	NO2	62 232	3 630	103	0	6
	O3	39 598	8 671	361	2	104
	PM10	124 205	105 986	37 252	12 644	11 197
	SO2	65 271	1	0	0	2
	Toplam	340 699	118 288	37 716	12 647	11 309
	Oran (%)	65,44	22,72	7,24	2,43	2,17
Sis sunucusu 5	CO	24 646	0	0	0	0
	NO2	42 384	6 558	552	15	66
	O3	35 663	4 717	175	1	54
	PM10	56 049	16 605	5 436	1 947	1 720
	SO2	202 509	0	0	0	9
	Toplam	361 251	27 880	6 163	1 963	1 849
	Oran (%)	90,52	6,99	1,54	0,49	0,46
Sis sunucusu 6	CO	66 631	0	0	1	0
	NO2	50 800	14 333	1 316	38	1
	O3	28 816	4 032	13	0	3
	PM10	56 502	13 535	6 876	2 575	2 725
	SO2	33 009	0	0	0	3
	Toplam	235 758	31 900	8 205	2 614	2 732
	Oran (%)	83,84	11,34	2,92	0,93	0,97
Toplam	Toplam	1 833 185	339 920	83 248	23 598	20 730
	Oran (%)	80	15	4	1	1

3.3. Önerilen Modeller

Geleneksel IoT ağlarında sensörler, bulundukları ortamdan gerekli ölçümleri yaparak veri toplar ve veriyi talep eden uygulamanın türü veya içeriğine bakmaksızın tek bir sunucuya gönderirler. Bu nedenle, özellikle büyük ölçekli IoT ağları için IoT cihazları arasında iletilen veri hacmi, tıkanıklık, daha yüksek iletim gecikmeleri ve düşük QoS gibi istenmeyen sonuçlara sebep olmaktadır. SDN destekli sis tabanlı IoT ağları için iki yeni içerik tabanlı veri iletme modeli önerilmiştir.

Şekil 3.2'de önerilen modellerin SDN mimarisi gösterilmiştir. Kontrolör ile yönetilen ağda verilerin saklandığı bulut sunucusu ve sis sunucuları bulunmaktadır. Ağ alt katmanında SDN ile yönetilebilen OpenvSwitch (OVS) anahtarlar ve bu anahtarlara bağlı son kullanıcılar ile sensörler bulunmaktadır.



Şekil 3.2. Önerilen modellerin SDN destekli IoT/sis ağ mimarisi

Önerilen modeller, IoT ağının birkaç sis sunucusuna (düğümlere) sahip olduğunu ve bunların her birinin belirli bir veri tipine ve/veya uygulamaya tahsis edildiğini varsaymaktadır. Ayrıca modellerde paketlerin kendi sis sunucularına ulaşmasından SDN kontrolörü sorumludur. Önerilen modeller, İstanbul ilinin hava kalitesi ölçümlerinden elde edilen gerçekçi ağ akışı kullanılarak uygulanmış ve değerlendirilmiştir.

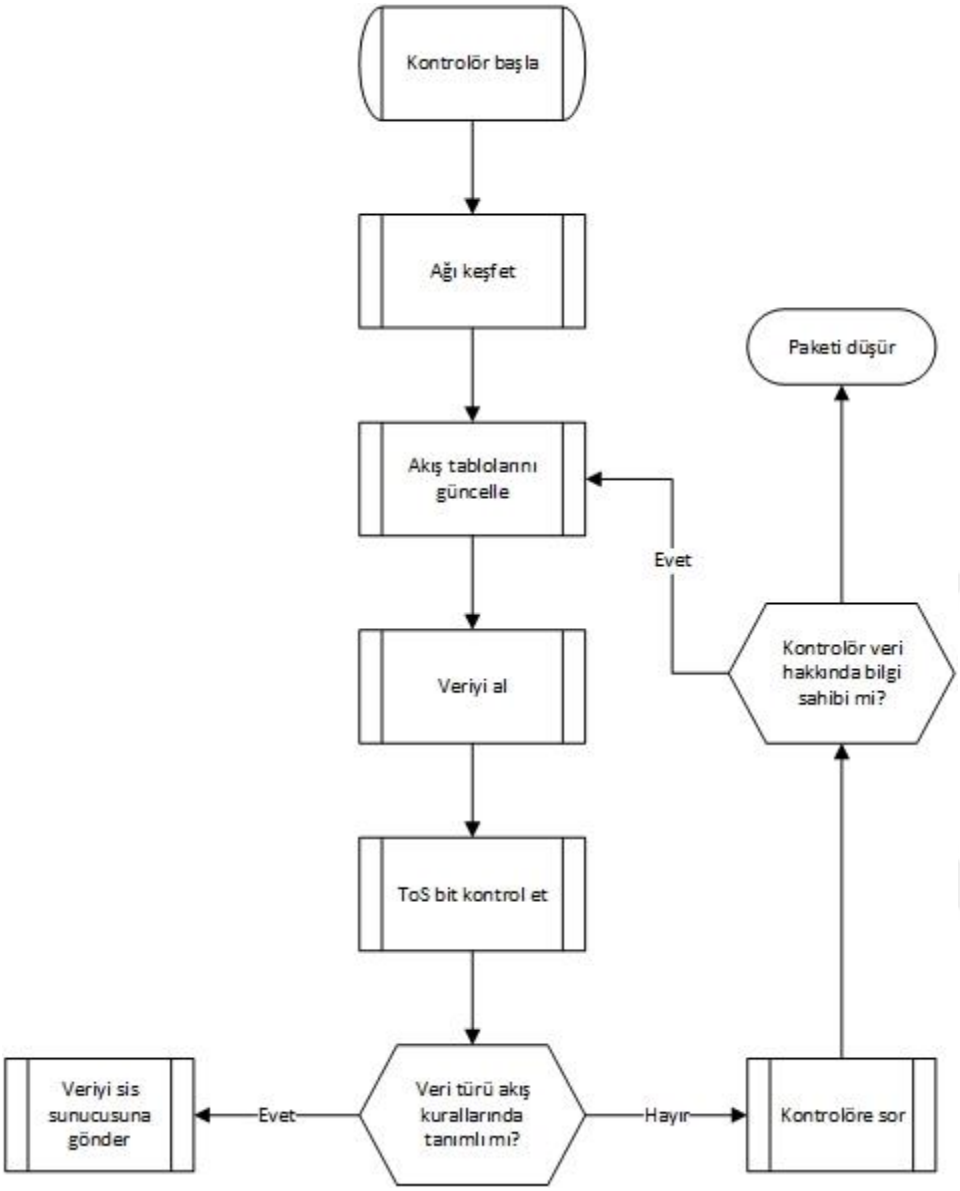
Tezin bu alt bölümünde önerilen CBF-SDN ve PCBF-SDN modelleri detayları ile açıklanmıştır.

3.3.1. Önerilen SDN destekli içerik tabanlı yönlendirme yöntemi (Content based forwarding via SDN, CBF-SDN)

Iot sistemlerinde, birden fazla sensör bir bilgisayara bağlanabilmektedir ve bu sistemlere binlerce bilgisayar (son kullanıcı) dâhil edilebilmektedir. Bu sistemler dinamiktir bu sebeple ihtiyaçlar zaman içinde değişebilmektedir. Gereksinimler değiştiğinde, her bilgisayar, sensörler ve anahtarlar yeniden optimize edilmelidir. Bu işlem, büyük sistemlerde tekrarlanan ve zahmetli bir iş yükü oluşturmaktadır. Önerilen modelde, yönlendirme anahtarlarda kontrolörün desteği ile gerçekleştirilmektedir. Tüm sistem, kontrolör yazılımında yapılacak basit bir merkezi güncelleme ile ihtiyaca göre değiştirilebilmektedir.

CBF-SDN ile sensörlerden elde edilen veriler, veri tipine göre etiketlenerek ağa iletilmektedir. Verileri içerik türüne göre etiketlemek, ağdaki anahtarların ve denetleyicilerin verileri tanımlamasını sağlamaktadır. Etiketler denetleyici tarafından bilinmektedir ve SDN özellikli veri düzlemi cihazlarına iletilmektedir. Öte yandan, geleneksel ağ yapısında, ağ katmanında veri paketinin içeriği, ağ cihazları tarafından bilinmemektedir. Bu, önerdiğimiz modeli geleneksel modellerden ayıran en önemli farktır.

Ağa bağlı cihazlar denetleyici tarafından iletişime geçirildiğinden ilk adımda sadece cihazların birbirleri ile haberleşmesi sağlanmaktadır. Bu nedenle, veriler iletilmeden önce, anahtarlar üzerindeki akış tablolarında paketler hakkında her hangi bir bilgi bulunmamaktadır. Sensörlerden elde edilen veriler ağ cihazlarına iletilmeden etiketlenmektedir. Bu etiketleme Hizmet Türü (Type of Service, ToS) bit değerlerinin değiştirilmesi ile gerçekleşmektedir. Yani, her veri türüne göre farklı ToS bit değerleri kullanılmaktadır ve bu değerler kontrolör tarafından bilinmektedir.



Şekil 3.3. Önerilen CBF-SDN modelinin akış şeması

Şekil 3.3'te CBF-SDN modelinin akış şeması verilmiştir. İlk adım olarak kontrolör ağda iletişimi başlatır. SDN destekli ağlarda çekirdek ağda bulunan cihazların iletişimini kontrolör sağlamaktadır. Ağda bulunan cihazları keşfederek tüm cihazlardaki akış tablolarını güncellemektedir. Bir sonraki adımda sensörden gelen verinin ToS biti kontrol edilir. ToS bitine göre veri türü bilgisinin akış kurallarında var olup olmadığı kontrol edilir. Akış kurallarında bu bilgi mevcutsa veri belirlenmiş sis sunucusuna iletilir bilgi mevcut değilse kontrolöre veri içeriği sorulur. Kontrolör veri içeriği bilgisine sahipse ağdaki tüm akış tablolarını günceller ve iletişim devamını sağlayarak verinin belirlenmiş sunucuya iletilmesini sağlar. Kontrolör veri içeriği hakkında bilgi sahibi değilse paket ağdan düşürülür.

Önerilen CBF-SDN modeli, sis sunucularının yerini belirleme, ağı optimize etme, en kısa yolu belirleme ve seçme vb. gibi ağ performansını geliştirmede avantajlar sağlamaktadır. Denetleyicinin ve SDN destekli anahtarların, verilerin içeriğini tanıması ağ yönetiminde birçok avantaj sağlamaktadır. Bu avantajlar aşağıdaki gibi sıralanabilir;

1. Sensörlerden elde edilen verilerin tamamı veri işleme için kullanılmayabilir. CBF-SDN modelinde sensörlerden elde edilen verilerin belirli eşik değerlerinden yüksek olması istenebilir. Buda ağa kullanılmayacak verilerin iletilmesini engeller. Önemsiz/ gereksiz verilerin iletilmemesi ağda verimi artırmaktadır.

2. Acil durum senaryosu, iletilen sensör verisi belirtilen eşik değerinden daha yüksek olması hızlı karar almayı gerektirebilir. Örneğin CO gazının belirli seviyeden yüksek olması canlıların yaşamsal fonksiyonlarını olumsuz etkileyebilmektedir. Bu tarz acil durumlarda CO verisinin diğer veriler ile işlenerek anlamlı sonuç çıkarmasını beklemek yerine direk karar verilmesi sağlanabilir. Önerdiğimiz model, örnekteki gibi tanımlanacak acil durum senaryolarında kullanılabilir.

3. Etiketleme veri türüne göre yapıldığından, sensörler veya çevresel faktörler tarafından hasar gören veriler kontrolör tarafından tanımlanamadığı için gereksiz yere sunuculara iletilmesi önlenmektedir.

4. İçeriğe dayalı yönlendirmede, kullanıcı isteklerine göre belirlenen veri türünün değiştirilmesi, belirli veri türlerinin sis sunucularına aktarılması veya ağ altyapısında değişiklik yapılması hızlı ve dinamik bir şekilde gerçekleştirilmektedir.

5. Sensörler belirli zaman aralıklarında veri üretir. Veri işlemede, bu zaman aralığı farklı olabilir. Örneğin, CO sensörü her 5 dakikada bir veri oluşturur, ancak veri işleme için yalnızca her 30 dakikada bir oluşturulan veriler kullanılıyor olabilir. Kontrolör yazılımı ile sensörlerden elde edilen tüm veriler değil sadece kullanılacak olan veriler iletilebilmektedir. Kullanılmayan veriler denetleyici tarafından iletilmediğinden, ağ performansı artmaktadır.

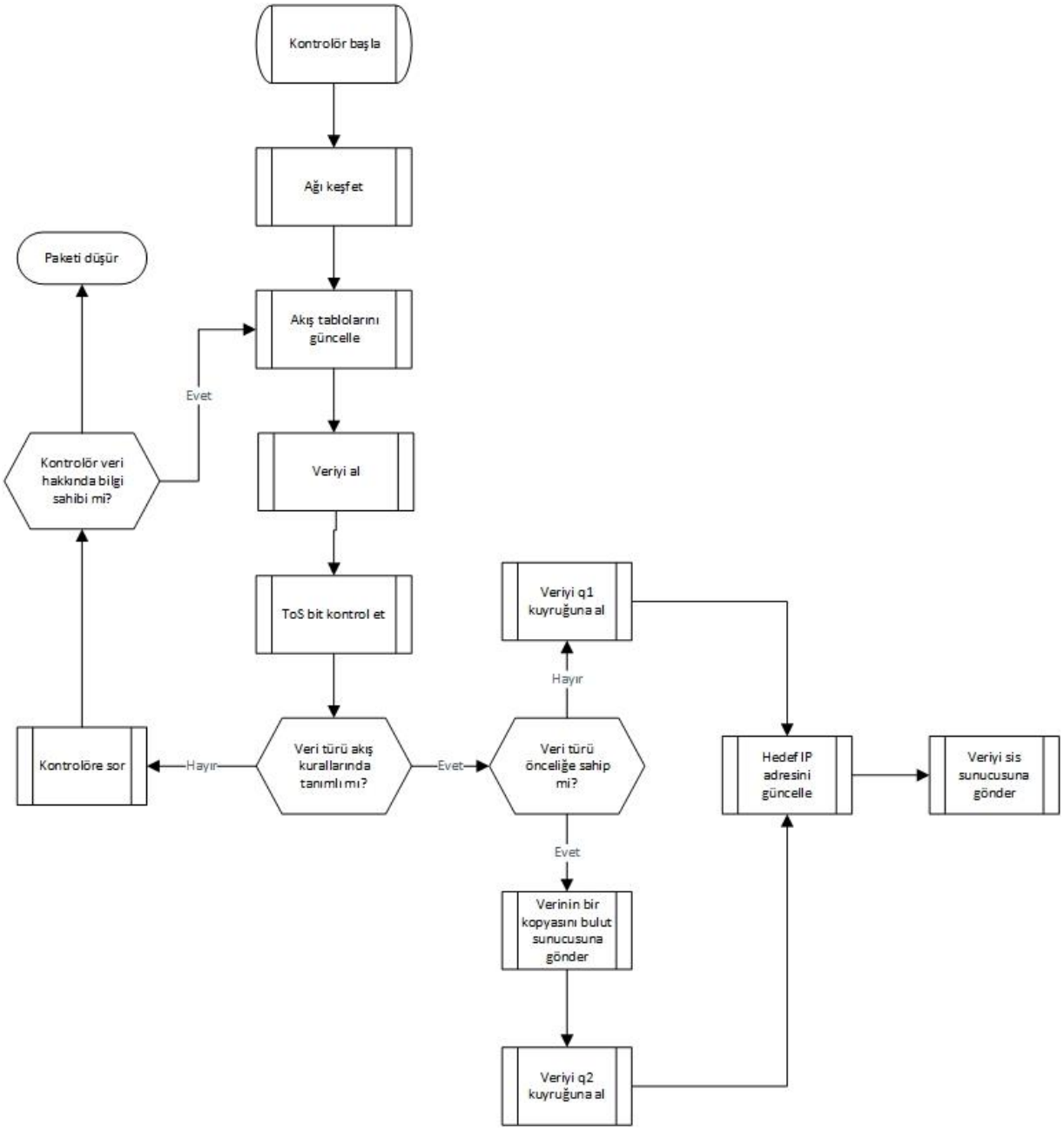
CBF-SDN'de gönderilmemiş verilerin veri işleme için önemli olma olasılığı göz ardı edilmiştir. Bu nedenle, önerdiğimiz ikinci modelde, yalnızca önemli verilerin (öncelikli) bir kopyası bulut sunucusuna iletilmektedir, böylece veri kaybı nedeniyle oluşabilecek hatalı analiz sonuçları azaltılmaktadır.

3.3.2. Önerilen SDN destekli öncelikli verilerin içerik tabanlı yönlendirme yöntemi (Priority content based forwarding via SDN, PCBF-SDN)

Bazı veri türlerinin analiz sonucuna etkisi diğer veri türlerine göre daha fazla olduğu tespit edilmiştir (Bkz. Çizelge 3.2). Bu nedenle verilerin sayısal değerleri, analiz sonuçlarını etkilediği ölçüde verilerin önceliğinin tanımlanmasında kullanılmıştır. Sonuç olarak, veri

analizi sonuçları üzerinde yüksek etkiye sahip verilerin ayrı bir kuyruğa aktarılması ve bu kuyruğa ayrıcalık tanınması, analiz sonuçlarını ve ağ performansını etkilemektedir. CBF-SDN yapısına öncelik kuyruğu eklenerek genişletilmiş ve bu yeni mimariye PCBF-SDN ismi verilmiştir.

Belirlenen eşik değerinden daha yüksek değere sahip olan veri türleri analiz sonuçlarını doğrudan etkilemektedir. PCBF-SDN'de, bu veri türleri ayrı bir kuyruğa yerleştirilerek ağda daha yüksek öncelik ve daha fazla bant genişliği ile iletilmektedir. PCBF-SDN, bu verilerin olası acil durum iletimi için daha hızlı kararlar almakta ve sis sunucusunu atlayarak verileri doğrudan ana sunucuya iletmektedir. Örneğin, CO gazı sayısal değeri 15.000'den yüksek olduğunda, ortamın yaşam kalitesi önemli ölçüde azalmaktadır. Bu nedenle, 15.000’den daha büyük değere sahip veriler sadece sis sunucusuna değil, aynı zamanda ana sunucuya da iletilmektedir. Bu, acil durumlarda kararların daha hızlı alınmasını sağlamaktadır.



Şekil 3.4. Önerilen PCBF-SDN modelinin akış şeması

Şekil 3.4'te PCBF-SDN modelinin akış şeması verilmiştir. İlk adım olarak kontrolör ağda iletişimi başlatır. SDN destekli ağlarda çekirdek ağda bulunan cihazların iletişimini kontrolör sağlamaktadır. Ağda bulunan cihazları keşfederek tüm cihazlardaki akış tablolarını güncellemektedir. Bir sonraki adımda sensörden gelen verinin ToS biti kontrol edilir. ToS bitine göre veri türü bilgisinin akış kurallarında var olup olmadığı kontrol edilir. Akış kurallarında bu bilgi mevcut değilse kontrolöre veri içeriği sorulur. Kontrolör veri içeriği bilgisine sahipse ağdaki tüm akış tablolarını günceller ve iletişim devamını sağla. Kontrolör veri içeriği hakkında bilgi sahibi değilse paket ağdan düşürülür. Akış kurallarında sensör

gelen veri bilgisi mevcut ise verinin önceliğinin var olup olmadığı kontrol edilir. Veri herhangi bir önceliğe sahip değilse kuyruk-1'e aktarılır daha sonra hedef IP'si güncellenerek sis sunucusuna iletilir. Veri önceliğe sahipse bir kopyası bulut (ana) sunucuya iletilir ve kuyruk-2'ye aktarılır. Kuyruk-2'ye aktarılan veri hedef IP'si güncellenerek sis sunucusuna iletilir.

Daha önce belirtildiği gibi PCBF-SDN, CBF-SDN modelinin genişletilmiş versiyonudur. Bu nedenle, çalışmanın geri kalanında PCBF-SDN modeli teknik detaylarla açıklanmıştır. Şekil 3.2'te görüldüğü gibi, önerilen model SDN kontrolör tarafından yönetilmektedir. Ağda cihazların iletişimi, ağ cihazlarına akış protokollerinin iletilmesi ve paketlerin yönlendirmesi kontrol birimi tarafından gerçekleştirilmektedir. Ayrıca model, paketlerin (içeriğe dayalı olarak) hangi sis sunucusuna iletileceğine ve bir kopyanın ana sunucuya iletilip iletilmeyeceğine karar vermektedir.

PCBF-SDN modelinde yönlendirme, veri analistlerinin taleplerine göre merkezi, dinamik ve yazılım desteği ile kolayca güncellenebilmektedir. Başka bir deyişle, bazı verilerin önemi veri analizinin sonuçlarına göre değiştiğinde, önceliği değişebilir ve bu değişim kontrolör yazılımının güncellenmesi ile akış tablolarında değiştirilebilir.

```
Algorithm 1: PCBF-SDN modelinin algoritması
  Input: Sensör verisi p, ToS bit değeri T, Paket zaman aşımı t, Anahtarların akış tablosu ft
  Output: Hedef IP dest-ip, Kuyruklar q1 q2
  başlangıç;
  while t ≠ 0 do
    | if ft, T hakkında bilgi sahibi then
    |   | dest-ip = ft ile hedef IP güncelle;
    | else
    |   | kontrolöre sor;
    |   | if Kontrolör paket hakkında bilgi sahibi then
    |   |   | akış tablolarını güncelle;
    |   | else
    |   |   | Paketi ağdan düşür;
    |   | end
    | end
  end
  if p öncelikli veri then
    | p'nin bir kopyasını ana sunucuyu gönder;
    | p'yi q2 kuyruğuna aktar;
  else
    | p'yi q1 kuyruğuna aktar;
  end
  p'yi hedef sis sunucusuna gönder;
```

Şekil 3.5. Önerilen PCBF-SDN modelinin akış kuralları

3.4. Deneysel Çalışmalar

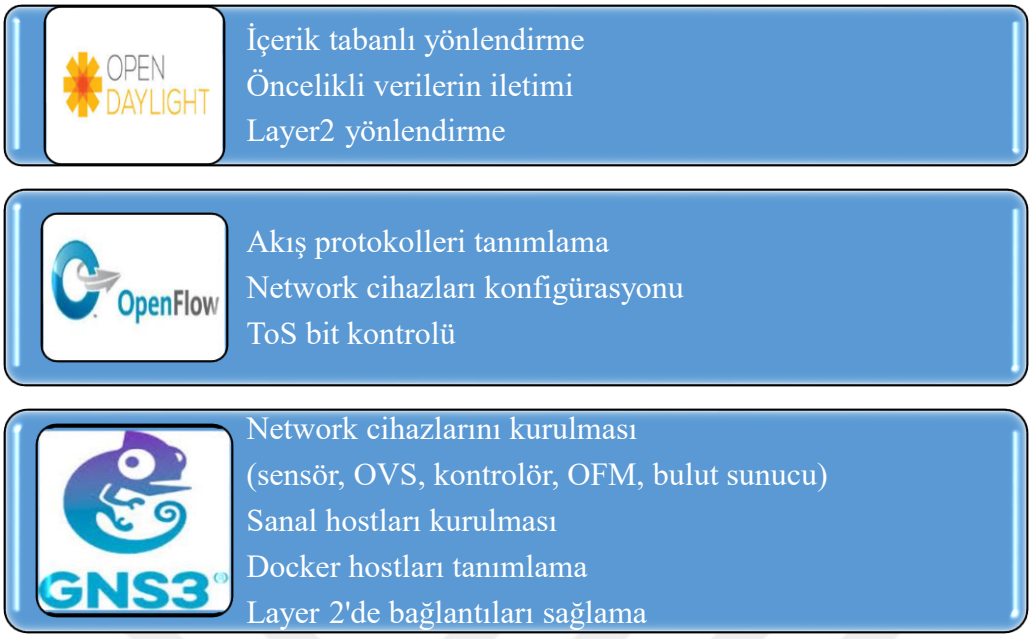
Tezin bu alt bölümünde önerilen modellerin performans analizi için gerekli deneysel ortam kurulmuştur. IoT sensörlerinden elde edilen verilerin SDN destekli sis sunucularında işlenmesini ve ağ performansı incelenmiştir. Altı farklı sis sunucusunda toplanan veriler analiz edilerek, analiz sonuçları ana sunucuya iletilmektedir. Veri analizi işlemleri ana sunucuda toplanan verilerle gerçekleştirilmektedir. Sensörlerden elde edilen verilerin tek bir sunucuda toplanması ve işlenmesi ile karşılaştırılmaktadır. Önceki bölümde belirtildiği gibi, sensörlerden gelen veriler analiz sonucu üzerindeki etkiye göre öncelikli ve öncelikli olmayan diye iki ayrı gruba ayrılmıştır. Önerilen kuyruk yapısının kullanılmasının etkileri ve veri önceliklendirmenin ağ performansı üzerindeki etkisi incelenmiştir.

3.4.1. Deney ortamı

Önerilen modellerin test edilebilmesi için gerekli platform bu bölümde açıklanmıştır. Önerdiğimiz iki model ve geleneksel IoT ağ yapısı karşılaştırılmaktadır. Geleneksel IoT modelinde, sensörlerden gelen veriler doğrudan sunucuya iletilmektedir. CBF-SDN modelinde, sensörlerden gelen veriler içeriğine göre yönlendirilmektedir ve sis sunucularında işlendikten sonra ana sunucuya iletilmektedir. Öte yandan, PCBF-SDN modelinde, sensörlerden gelen veriler önceliklerine ve içeriklerine göre sis sunucularına yönlendirilerek veri analiz işlemleri gerçekleştirilmektedir. Daha sonra bu veriler, ana sunucuya iletilmektedir. Ek olarak PCBF-SDN modelinde, yüksek öncelikli sensör verilerinin bir kopyası sis sunucusunda işlenmeden doğrudan ana sunucuya iletilmektedir.

Şekil 3.6'da deneysel kurulumu gerçekleştirilen platformlarda uygulanan adımlar gösterilmiştir. Önerilen modeller uygulama katmanında ODL kontrolör yazılımı ile gerçekleştirilmiştir. OF ile akış kontrolleri kontrol katmanında gerçekleştirilmiş. Veri katmanı cihazları GNS3 platformunda simüle edilmiştir.

Test ortamı, her biri farklı sanal makinelerde kurulu kontrolör sunucusu, OpenFlow Manager (OFM) sunucusu, verilerin saklandığı ana sunucu ve GNS3 platformunun kurulduğu sunuculardan oluşmaktadır. Sanallaştırma ile 4 farklı sanal sunucu bağlantısı sağlanmıştır.

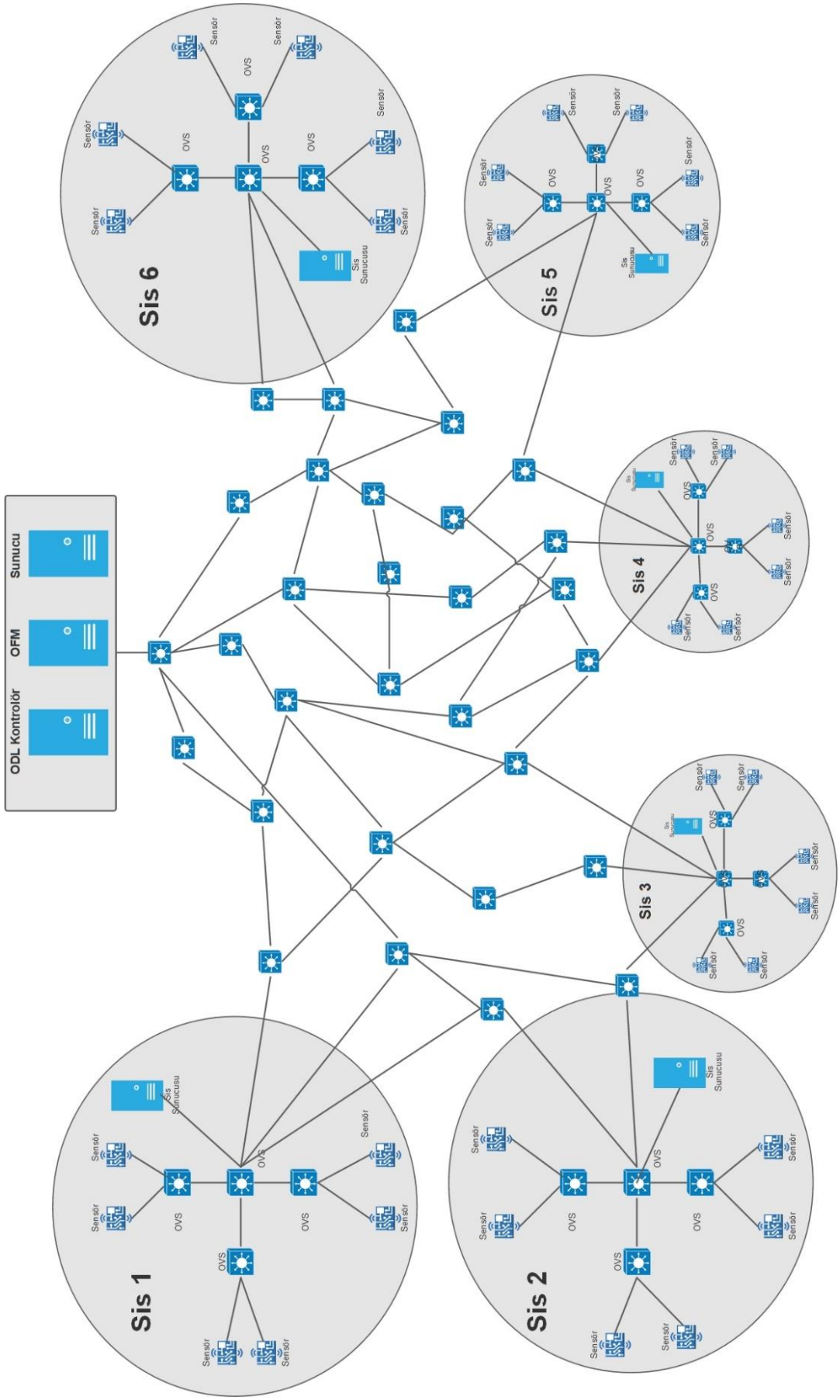


Şekil 3.6. Önerilen SDN destekli modellerin kontrolör, openflow ve GNS3 içerikleri

GNS3 kurulu sunucuda ağ alt yapısı (veri katmanı) kurulmuştur. Bu ortamda, 70 OpenvSwitch (OVS), 6 sis sunucusu, bu sunucuların her birine bağlı 6 bilgisayar (son kullanıcı) ve her bilgisayara bağlı sensörlerden oluşmaktadır.

OVS, hem akış protokolü olarak OpenFlow'u kullanan hem de bir denetim programı olan OFM tarafından yönetilebilen açık kaynaklı, çok katmanlı ve SDN destekli sanal bir anahtardır [73]. OFM, OF topolojilerini görselleştirmek, istatistik toplamak ve akış kodları yazmak için kullanılmaktadır [74]. Ağ simülasyon ortamı Şekil 3.7'te gösterilmiştir.

Önerilen modeller Graphical Network Simulator-3'de (GNS3) simüle edilmiştir. GNS3 2.2.5 versiyonu kullanılmıştır. GNS3, test ve uygulama geliştirme amacıyla kullanılan açık kaynaklı bir yazılımdır ve Cisco IOS yazılımlarını desteklemektedir [75]. VMware ve VirtualBox gibi sanallaştırma yazılımlarıyla oluşturulan sanal makineleri desteklemekte ve uyumlu bir şekilde çalışmaktadır. Ayrıca, Wireshark gibi ağ izleme, paket yakalama programlarını desteklemektedir. GNS3'te önerilen modelleri test edebilmek için Docker container'larda OVS anahtarları kuruldu [76]. OFM, ODL kontrolör yazılımı ile paralel çalışan bir uygulamadır. Bu nedenle OFM, VMware ile oluşturulmuş linux işletim sistemine sahip sanal bir makinede kurulmuş sunucuda çalışmaktadır [77].



Şekil 3.7. Önerilen modellerin GNS3 simülasyon ortamındaki ağ topolojisi

Sensörlerin bağlı olduğu linux işletim sistemine sahip bilgisayarlar, Docker container'lara kuruldu. Docker, linux veya Windows işletim sistemlerinde oluşturulan sanal makinelere yüklenebilen ve MS Windows, linux ve macOS işletim sistemleri kurulabilen açık kaynaklı bir sanallaştırma aracıdır [78]. GNS3 ile entegre edilebilen bu yazılım, diğer sanallaştırma programlarının aksine, sistem araçlarını paylaşımlı kullandığı için daha hızlı ve dinamiktir.

Veriler Netcat yardımcı program yazılımı ile ToS bitleri değiştirilerek ağa iletilmektedir. Netcat, iki bilgisayar arasında TCP veya UDP bağlantısı kurabilen ve açık bir bağlantı noktası üzerinden iletişim sağlayabilen bir programdır. Netcat ile bilgisayarlar arası dosya transferi gerçekleştirilebildiği gibi, hedef IP, hedef bağlantı noktası, ToS bit vb. bilgiler değiştirilebilmektedir [79].

Literatürede, ODL, POX, Ryu vb. gibi birçok SDN kontrolörü kullanılmaktadır. Bu çalışmada, ODL kontrolörü kullanılmıştır. Çünkü OFM, OVS ve GNS3 platformları ile uyumludur. ODL, Windows işletim sistemine sahip sanal bir makineye kuruldu. Bu sanal makine GNS3 ile entegre edilerek ağ yönetimi ODL kontrolörü ile gerçekleştirildi. ODL kontrolörü ağdaki aygıtları ve iletim yollarını tanımlayarak ağdaki aygıtların iletişim kurmasını sağlamaktadır [25,80,81].

ODL denetleyicisi en kısa yolları bulmak için Dijkstra algoritmasını kullanmaktadır ve her anahtardaki her veri türü için bir sonraki atlamaya karar vermektedir. Ardından, yönlendirme kararlarını uygulamak için tüm anahtarlara akış kurallarını gönderir. Sistemdeki her anahtar akış tablolarındaki kurallara göre yönlendirme gerçekleştirir [82]. Paketler bir anahtara ulaştığında, veri türlerini temsil eden önceden tanımlanmış ToS bit'e dayalı olarak akış tablolarındaki kurallardan biriyle eşleştirilmektedir. Kuralda tanımlanan ilgili yönlendirme eylemi, paketin veri türünün önceliğine bağlı olarak doğru kuyruğa alınmasını da içermektedir. Paket mevcut bir kuralla eşleştirilemez ise, kontrolöre gönderilmektedir. Bu noktada, kontrolör ToS bit değerine göre verinin hedef IP'sini belirleyerek anahtarlara iletmektedir ve verinin doğru sunucuya iletilmesi gerçekleştirilmektedir.

Bu çalışma, 64 bit linux işletim sistemine, 3.4 GHz işlemciye ve 32 GB RAM'e sahip bir sunucuda gerçekleştirilmiştir. GNS3 ile tasarlanan ve simüle edilen topolojide, Docker container'da çalışan 70 OVS ve 36 sanal bilgisayar, VMware üzerinde çalışan 4 sanal sunucu bulunmaktadır. Sensörlerin her biri saatte 90 paket veri üretmektedir ve ortalama paket

boyutu 75 byte olarak kabul edilmiştir. CBF-SDN modelinde bant genişliği 10 Mbit/s olarak belirlenirken, PCBF-SDN modelinde kuyruk 1 için 6 Mbit/s ve kuyruk 2 için 4 Mbit/s olarak belirlenmiştir.

3.4.2. Hesaplama metrikleri

Bu alt bölümde, bu çalışmada tartışılan üç modeli değerlendirmek için kullanılan metrikler açıklanmaktadır. Çizelge 3.3'te, verilerin öncelikli kuyruğa alınması için belirlenmiş eşik değerleri gösterilmektedir.

Çizelge 3.3. Simülasyon ortamında kullanılan veri seti için eşik değerleri

Veri türü	Eşik değeri
CO	>1500
NO2	>200
O3	>120
PM10	>100
SO2	>175

Geleneksel IoT ağı ve önerilen modelleri simüle etmek için belirlenen metrikler:

- Verim (Throughput), birim zamanda alınan ve iletilen toplam paket boyutudur.
- Ortalama gecikme süresi, verinin kaynaktan hedefe ulaşıncaya kadar geçen süredir.
- Ortalama kayıp oranı, hedefe ulaşamayan paket sayısının toplam paket sayısına oranıdır.

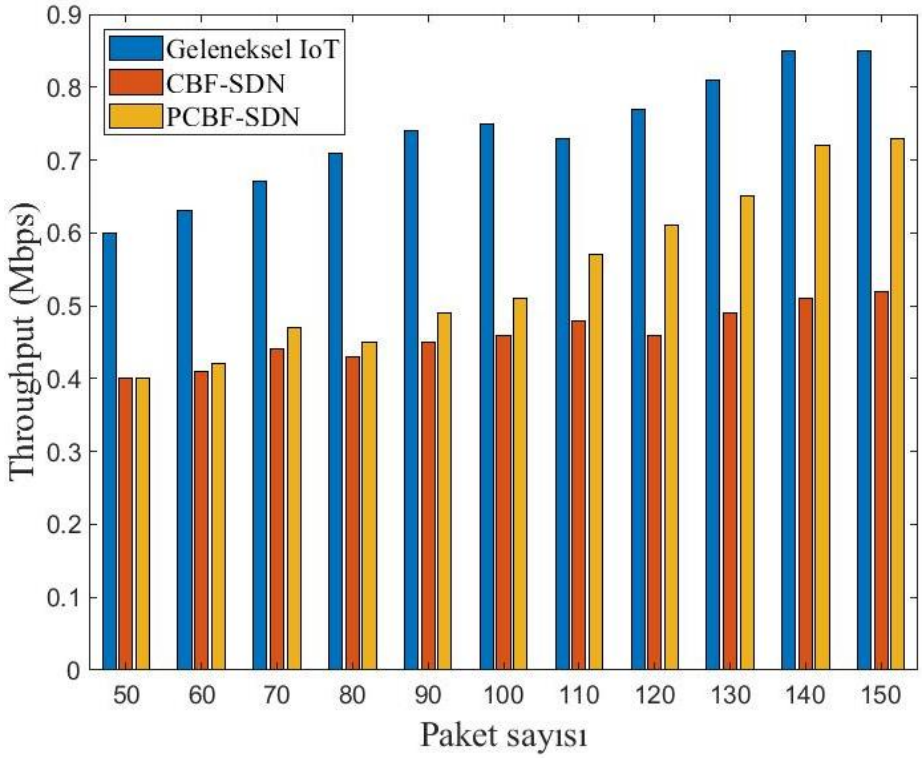
Bu üç metriğe ilişkin simülasyon deneylerinden elde edilen sonuçlar, bir sonraki bölümde sunulmuş ve tartışılmıştır.

3.4.3. Analiz sonuçları

Önerilen modeller kurulan deneysel ortamda test edilmiştir ve önceki bölümde açıklanan hesaplama metrikleri ile analiz edilmiştir. Önerilen iki model geleneksel IoT ağ mimarisi ile karşılaştırılmış ve sonuçlar aşağıda verilmiştir.

Şekil 3.8'da gösterildiği gibi, üç modelin verim değerleri iletilen veri paketleri sayısına göre karşılaştırılmıştır. Önerilen CBF-SDN ve PCBF-SDN modellerindeki verim değerleri geleneksel IoT ağlarından daha düşüktür. Önerilen modellerde, sensörlerden gelen veriler

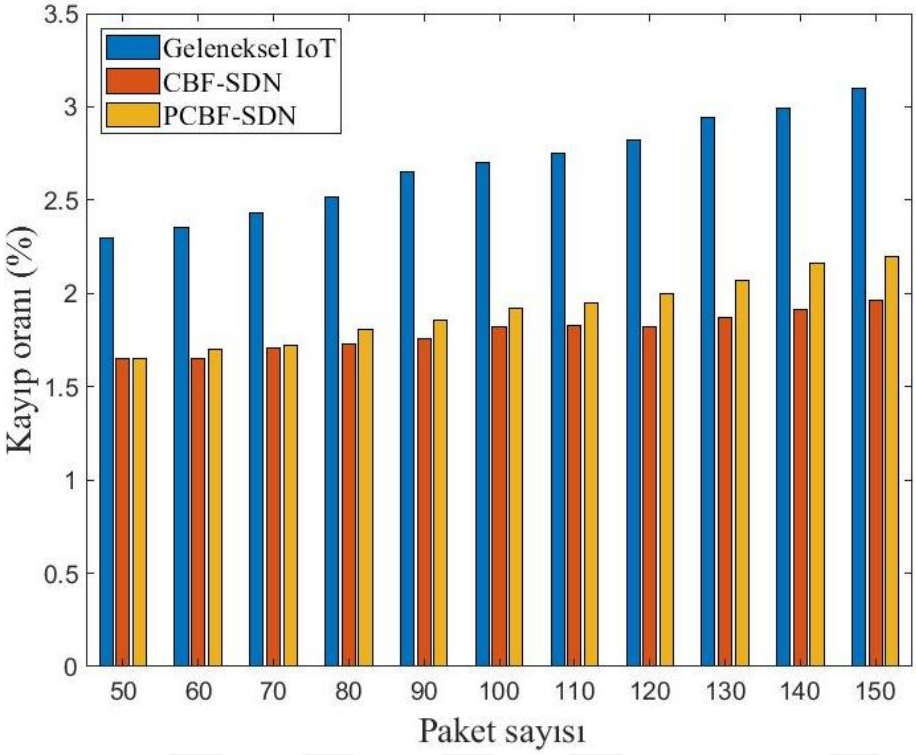
sis sunucusunda analiz edilmekte ve kontrolör yazılımı ile içeriği kontrol edilmektedir. Bu sebeple ağa iletilen paket sayısı geleneksel ağlarda gönderilen paket sayısından daha azdır. CBF-SDN modelinde ağa iletilen paket sayısı daha düşüktür. Bununla birlikte, PCBF-SDN'de, toplam bant genişliği öncelikli veri ve önceliksiz verile için iki ayrı kuyruğa bölünmektedir. Bu sebeple herhangi bir kuyrukta gönderilen paket sayısı değişkenlik göstermektedir. Ek olarak öncelikli verilerin bir kopyası farklı bir sunucyada gönderildiğinden ağa gönderilen paket sayısı artmaktadır. Yani, hem sis sunucularından hem de sensörlerden eşzamanlı olarak iletildikleri için toplam paket sayısı artmıştır (sadece öncelikli veriler için). Bu sebeplerden kaynaklı CBF-SDN modelinde verim PCBF-SDN modeline göre daha düşük olduğu gözlemlenmiştir.



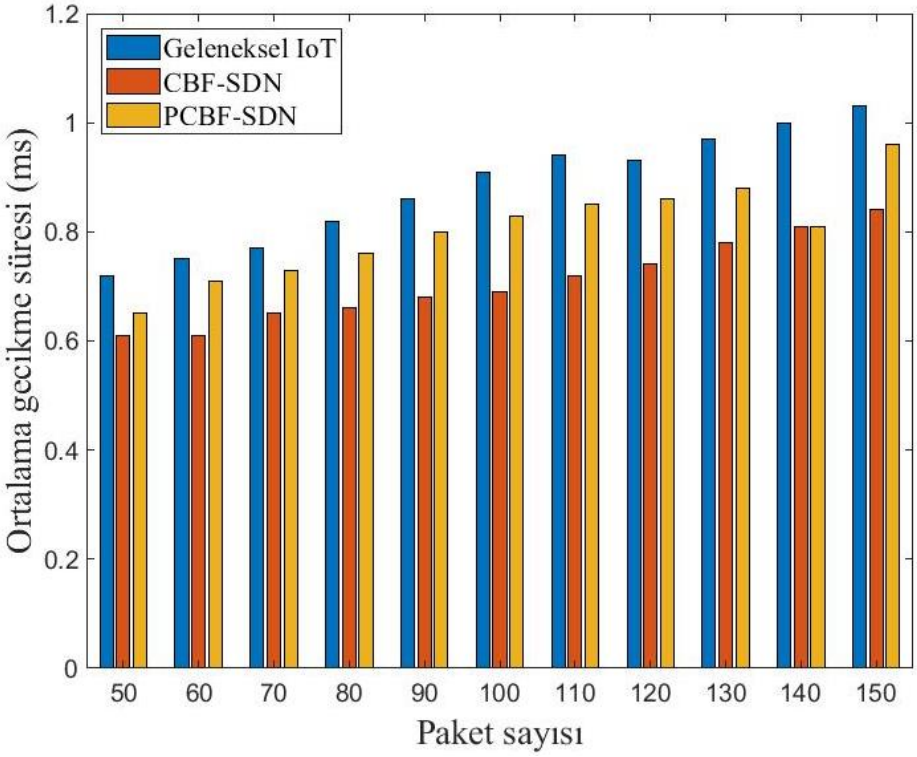
Şekil 3.8. Paket sayısı ve verim (Throughput) arasındaki ilişki

Şekil 3.9 ve Şekil 3.10'da gösterdiği gibi, birim zamanda sensörlerden gönderilen paket sayısı ne kadar fazla olursa, ortalama paket kaybı oranı ve gecikmesi o kadar yüksek olmaktadır. Önerilen modeller, paket kaybı ve ortalama gecikme süresi açısından daha düşük oranlara sahiptir. Başka bir deyişle, önerilen modellerde, veriler önce sis sunucularında işlenmektedir, bu sebeple ağa gönderilen ortalama paket sayısı azalmaktadır. Paketler içeriğine göre yönlendirildiğinden ağda gürültü ve çevresel etkilerden daha az etkilenmektedir. PCBF-SDN modeli, önceliksiz verilere daha az bant genişliği ayırmakta ve

öncelikli verilerin bir kopyasının sunucuya doğrudan iletilmesini sağlamaktadır. Bu sebeple CBF-SDN modelinden daha yüksek değerlere sahip olduğu gözlemlenmiştir.

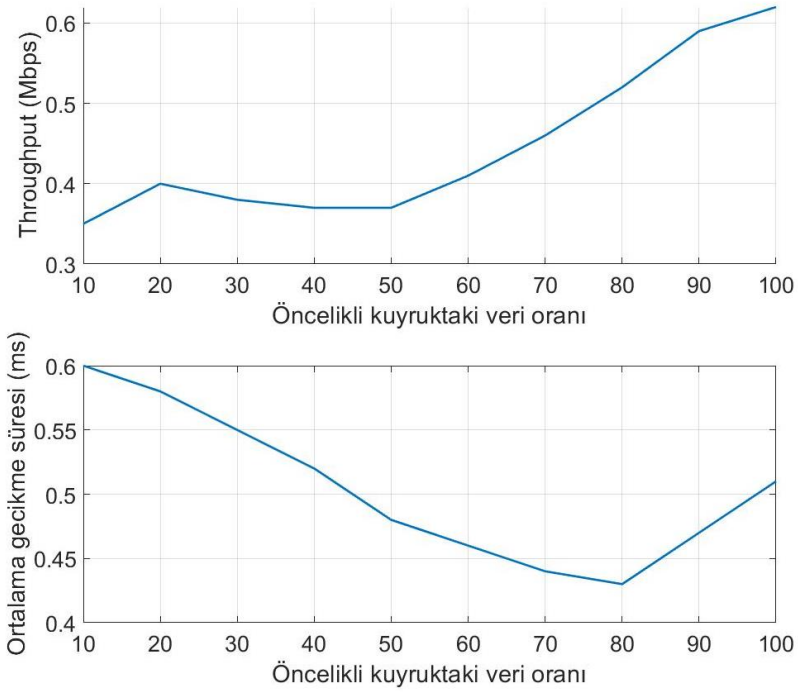


Şekil 3.9. Paket sayısının kayıp oranına etkisi



Şekil 3.10. Paket sayısının ortalama gecikme süresine etkisi

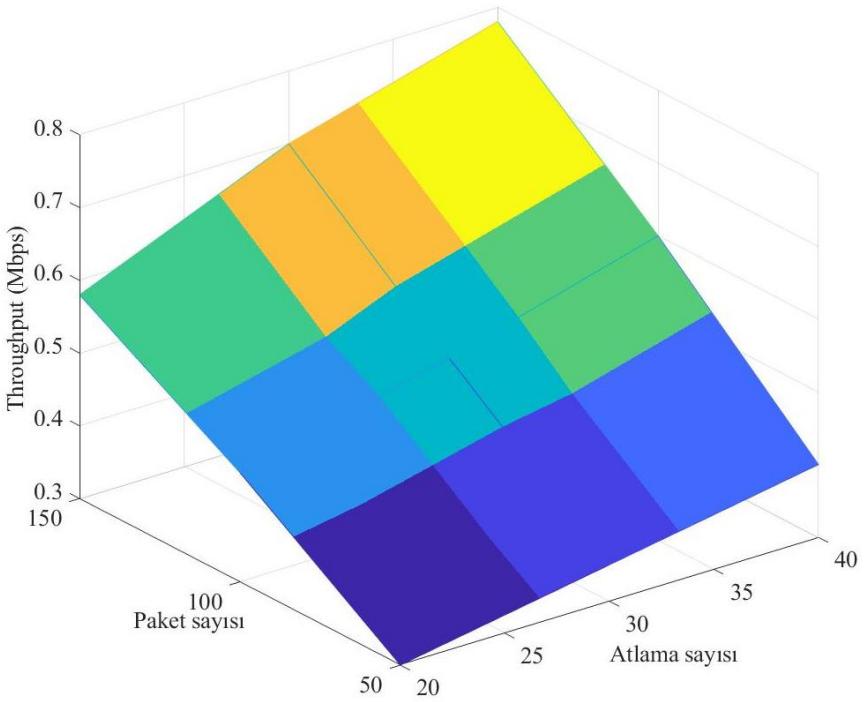
PCBF-SDN'de veri iletim performansını etkileyen faktörler; öncelik kuyruğundaki veri oranı, atlama sayısı, paket sayısı ve paket boyutudur. Öncelik kuyruğundaki veri oranının verim ve gecikme üzerindeki etkisi Şekil 3.11'de gösterilmiştir. Bant genişliği iki bölüme ayrıldığından ve öncelik kuyruğu için daha geniş bir alan tanımlandığından, öncelik kuyruğundaki veri oranı % 50 veya üzerinde olduğunda verimin artmaya başladığı görülmektedir. Öncelikli kuyruktaki veri oranı düşük olduğunda, diğer kuyruktaki veri oranı daha yüksek olacağından ve öncelikli kuyruğa daha geniş bir bant aralığı tanımlandığından, ortalama bant genişliğinin daha düşük olduğu görülmektedir. Örneğin, veri oranı öncelikli kuyrukta % 50'yi aştığı durumda, öncelikli kuyruğa aktarılan bant genişliği daha fazla olduğundan ağda ortalama bant genişliği daha yüksek olmaktadır. Buda verimin daha yüksek olmasını sağlamıştır. Öte yandan, öncelik kuyruğundaki veri oranı % 80'e ulaşana kadar ortalama gecikme süresi azalmaktadır, çünkü bu aşamadaki öncelikli kuyruğun bant genişliği daha yüksektir. Ancak, bu kuyruktaki verilerin oranı % 80'i aştığında ortalama gecikme süresi artmaktadır. Bunun nedeni ise, gecikmedeki artışın her zaman bant genişliği kullanımından etkilenmesidir.



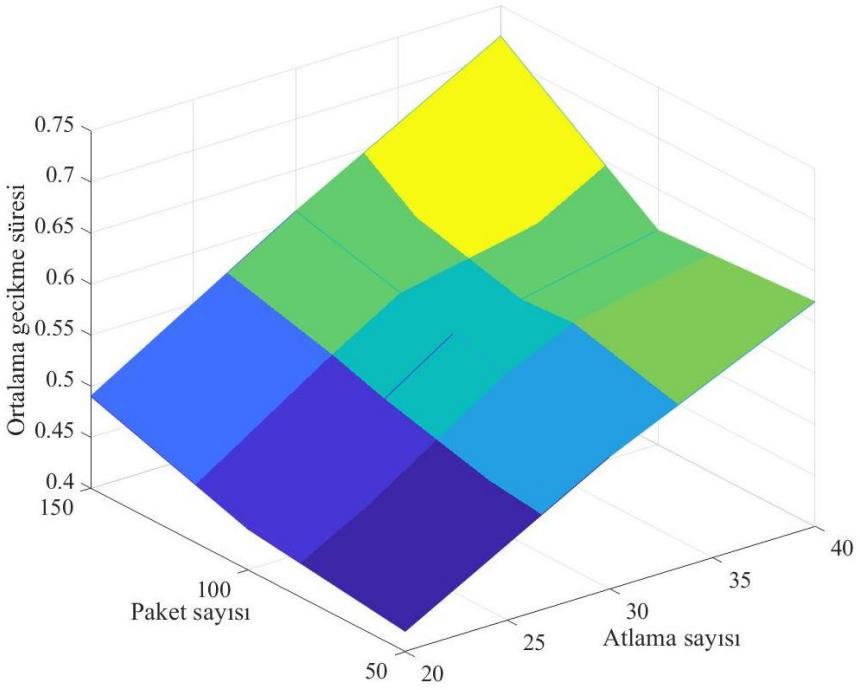
Şekil 3.11. Öncelikli kuyruktaki veri oranın ortalama gecikme süresi ve verim üzerine etkisi

Atlama ve paket sayısının verim üzerindeki etkisi Şekil 3.12'de, ortalama gecikme süresi üzerindeki etkisi ise Şekil 3.13’te gösterilmiştir. Önerilen PCBF-SDN modelinde atlama

sayısı arttıkça, verim ve ortalama gecikme süresi de artmaktadır. Verimdeki artış, hedef sunucuya alternatif yolun artmasından etkilenirken, ortalama gecikme süresi atlanan anahtar sayısından daha fazla etkilendiği gözlemlenmiştir. Atlama sayısının etkisi incelendiğinde, ana sunucunun konumunun performans üzerinde önemli bir etkisi olduğu gözlemlenmiştir. IoT sunucularının sensörlere olan ve kullanılan topolojide sensörlerin birbirlerine olan uzaklığı atlama sayısını ve ortalama gecikme süresini etkilemektedir. Sensörlerden ve sis sunucularından ağa gönderilen paket sayısı arttıkça ve atlama sayısı sabit kaldıkça verim artmaktadır. Bununla birlikte, ortalama gecikme süresi doğrusal olarak artmamaktadır. Saniyede gönderilen paket sayısının artması ile verimin artması beklenirken, ortalama gecikme süresindeki artış bant genişliğinin verimli kullanımından etkilenmektedir. Özelikle PCBF-SDN modelinde öncelikli verilerin iletildiği kuyruğa ayrılan bant genişliği oranı bu süreyi etkilemektedir. Daha önce bu oranın performansa olan etkisi (Bkz. Şekil 3.11) açıklanmıştı.

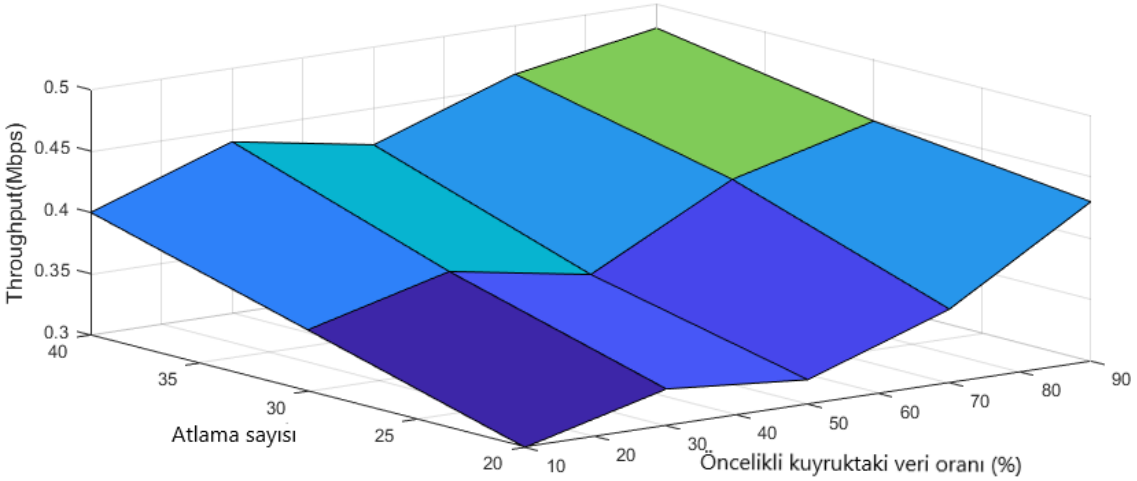


Şekil 3.12. Paket sayısı ve atlama sayısının verim üzerindeki etkisi

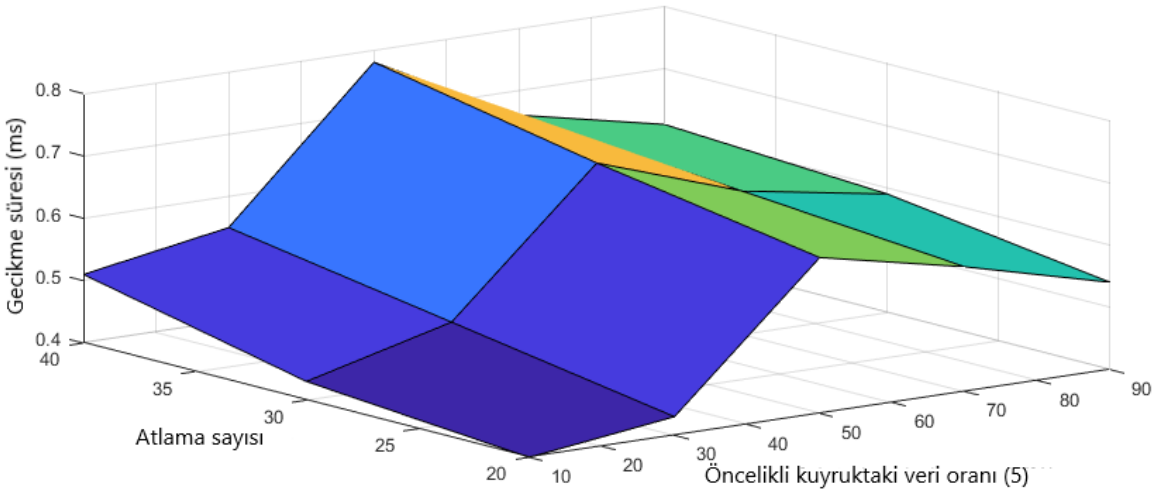


Şekil 3.13. Paket sayısı ve atlama sayısının gecikme süresi üzerindeki etkisi

Son olarak Şekil 3.14 ve Şekil 3.15’te atlama sayısının ve öncelikli verilerin bulunduğu kuyruktaki veri oranının verim ve ortalama gecikme süresi üzerindeki etkisi gösterilmiştir. Atlama sayısı arttıkça ve öncelikli verilerin iletildiği kuyruktaki veri oranı sabit olduğunda, verim ve ortalama gecikme birlikte artmaktadır. Ayrıca, atlama sayısı arttıkça ve öncelik kuyruğundaki veri oranı % 50'ye ulaştığında, ortalama gecikme süresi artmaktadır. Ancak, öncelikli verilerin iletildiği kuyruktaki veri oranı % 50'yi aştığında, ortalama gecikme süresi azalmaktadır. Bu azalma, saniyede iletilen veri sayısı arttıkça, öncelikli verilerin iletildiği kuyruktaki paket sayısının daha fazla bant genişliğine sahip olması nedeniyle gerçekleştiği gözlemlenmiştir.



Şekil 3.14. Atlama sayısı ve öncelikli kuyruktaki veri oranının verime etkisi



Şekil 3.15. Atlama sayısı ve öncelikli kuyruktaki veri oranının gecikme süresine etkisi

3.5. Bölüm Değerlendirmesi

Hava kalitesini ölçmek için elde edilen veriler, bir IoT ağı gibi simüle edilmiştir. Simülasyon verilerinin toplandığı gerçek sensör konumlarına uygun gerçekleştirilmiştir. Yapılan analizler sonucunda verilerin öncelik değerleri belirlenmiştir. Önerilen model, bu verilerle ortalama gecikme ve verim açısından geleneksel ağlarla karşılaştırılmıştır. Çizelge 3.4'te gösterildiği gibi, geleneksel ağlarda ortalama 1,1 ms'lik, CBF-SDN modelinde ortalama 1,02 ms gecikme gözlemlenmiştir. Ayrıca PCBF-SDN modelinde öncelikli verilerin ortalama

gecikmesi 0,8 ms, öncelikli olmayan verilerin 1,3 ms ve tüm verilerin ortalama gecikmesinin 1,23 ms olduğu gözlemlenmiştir. Öncelikli verilerde daha az gecikmeye sahip olmasının faydaları önceki bölümlerde açıklanmıştır. Bu gecikmeler, öncelikli verilere tahsis edilen daha geniş bant genişliğinden ve öncelikli verilerin toplam verilere oranından etkilenir.

Çizelge 3.4. Önerilen modellerin ortalama gecikme süreleri

Model		Ortalama gecikme süresi (ms)
Geleneksel IoT		1,10
CBF-SDN		1,02
PCBF-SDN	Öncelikli veri	0,8
	Öncelikli olmayan veri	1,3
	Tüm veriler	1,23

CBF-SDN modelinde kontrolörün tanımlayamadığı kayıp veri oranı yaklaşık %3'tür. Sensörlerden gelen veriler sis sunucusunda analiz edildiğinden, bu ağa iletilen paket sayısını etkiler. Bu nedenle, CBF-SDN modelindeki verim değerleri geleneksel IoT'lerden daha düşüktür. Önerilen modelimizin (CBF-SDN) daha az ortalama gecikmeye sahip olmasının ana nedeni, önerilen modelimizin bozuk/eksik verileri tespit etme ve bunları ağa iletmemesinden kaynaklanmıştır. Ağa iletilen veriler azaldıkça, verim azalmakta ve dolayısıyla ortalama gecikme süresi de azalmaktadır.

Çizelge 3.5’te üç modelin verim değerleri ve paket sayısı karşılaştırılmıştır. Geleneksel ağlarda ortalama 2.03 Mbitlik bir aktarım hızı ve saniyede 15 paket iletimi gözlemlenmiştir. Ancak CBF-SDN modelinde ortalama 1,86 Mbitlik aktarım hızı ve saniyede 13 paket iletimi gözlemlenmiştir. Ayrıca PCBF-SDN modelinde 1,69 Mbitlik ortalama verim ve saniyede 17 paket iletimi gözlemlenmiştir.

CBF-SDN modelindeki verim değerleri geleneksel IoT'dekinden daha düşüktür. Verilerin sis sunucusunda analiz edilip ağa iletilmesinden kaynaklı paket sayısı değişmektedir. Sensörlerden gelen bozuk/eksik veriler ağa iletilmediğinden saniyede iletilen ortalama paket sayısı azalmıştır. Ayrıca PCBF-SDN modelinde toplam bant genişliği öncelikli ve öncelikli olmayan kuyruklara ayrıldığından ve öncelikli kuyruğa %60 olarak tanımlandığından verim azalmıştır. Verim, belirlenmiş bant genişliği oranı ile doğru orantılı olarak azalmıştır.

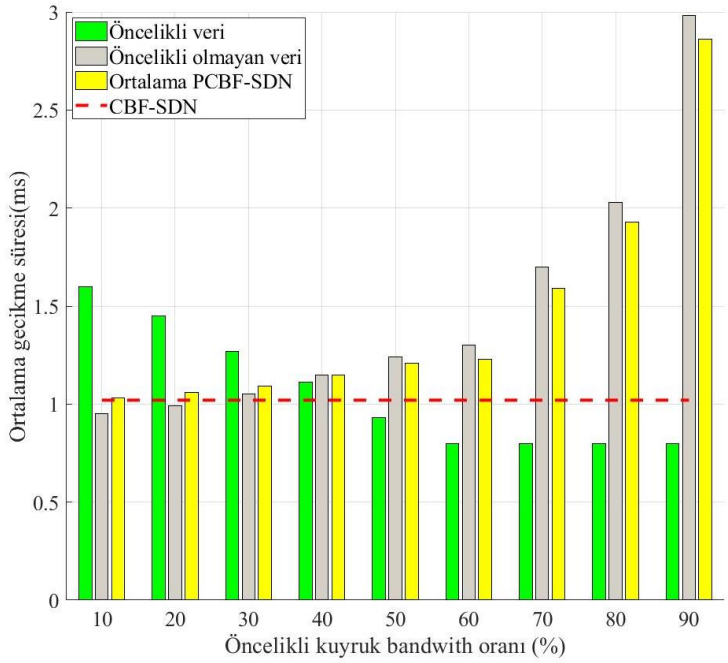
Önceki bölümde açıklandığı gibi (sadece öncelikli veriler için) verilerin bir kopyasını ana sunucuya ilettikleri için toplam paket sayısı da artmıştır.

Çizelge 3.5. Önerilen modellerin verim ve paket sayısı

Model	Verim (Mbits)	Paket sayısı
Geleneksel IoT	2,03	15
CBF-SDN	1,86	13
PCBF-SDN	1,69	17

Şekil 3.16’te önerilen modellerin ortalama gecikmesi, öncelikli verilerin atandığı kuyruğa tahsis edilen bant genişliğine göre karşılaştırılmıştır. Toplam bant genişliği, öncelikli ve öncelikli olmayan kuyruklara bölünmüştür. Şekil 3.16’te kuyruklara tahsis edilen bant genişliği yüzdesel olarak ifade edilmiştir. Bu nedenle, kalan kısım öncelikli olmayan kuyruğa tahsis edilen bant genişliğini ifade etmektedir. Daha önce öncelikli verilerin toplam verilere oranının %10,67 olduğu belirtilmişti.

Öncelikli verilerin iletildiği kuyruğa ayrılan bant genişliği arttıkça, ortalama gecikme artarken öncelikli verilerin gecikme süresi azalmıştır. Önerilen model için ideal bant genişliği oranı %40 olduğu gözlemlenmiştir. Çünkü öncelikli verilerin iletildiği kuyruğa atanan bant genişliği oranı %40’ı aştığı durumlarda gecikme süresinde herhangi bir iyileşme olmadığı gözlemlenmiştir. Önerilen modelin en uygun şekilde kullanılabilmesi için öncelikli verilerin iletildiği kuyruğa en uygun bant genişliği belirlenmelidir.



Şekil 3.16. Ortalama gecikme süresi ve öncelikli kuyruğun bant genişliği oranı ilişkisi

SDN destekli IoT ağlarında sensör verilerinin verimli bir şekilde iletilmesi için sis hesaplama tabanlı bir mimari önerilmiştir. Bu mimaride iki veri yönlendirme modeli önerilmektedir. İlk model, SDN kullanarak içeriğe dayalı veri iletiminden yararlanırken, ikinci modelde veri iletimi sırasında öncelik kavramını kullanmaktadır. Performans değerlendirme sonuçları, önerilen sis tabanlı mimari modelinin büyük ölçekli IoT ağlarında gecikme ve bant genişliği tüketimini azalttığını göstermektedir. Ayrıca, SDN destekli ağlarda veri iletiminde öncelik kavramı kullanılarak kritik verilerin zamanında teslimi de sağlanmıştır.

4. KONTROLÖR YERLEŞTİRME PROBLEMİNE ÖNERİLEN ÇÖZÜM YÖNTEMLERİ

SDN destekli IoT ağlarda içerik tabanlı ve öncelikli veri iletimi modellerini önerdik ve deneysel çalışmalar ile avantajlarını önceki bölümde sunduk. Önerilen modellerde performansı etkileyen önemli parametrelerden biride kontrolörün ağda bulunduğu konumdur. Bu sebeple, tezin bu bölümünde, SDN'de kontrolör yerleştirme problemi için matematiksel bir model önermekteyiz. 3. Bölümde önerilen ve deneysel ortamda uygulaması gerçekleştirilen SDN destekli ağ mimarisi için kontrolör konumunun etkisi incelenmiştir.

SDN destekli ağlarda, kontrolör sayısı, kontrolörler arası ilişki ve kontrolör konumu performansı doğrudan etkilemektedir. Birden fazla kontrolör bulunduran sistemlerde sunucu hata toleransı dikkate alınarak gerekli kontrolör sayısını ve konumunu belirleyen matematiksel bir model önerilmiştir. Modelin amacı, kontrolör kapasite ve kontrolörün servis dışı kalma olasılığı kısıtlamalarını göz önünde bulundurarak, ağın sürekliliğini sağlamak ve ortalama gecikmeyi en aza indirmektir. Simülasyon sonuçları, ağın sürekliliğini sağlarken ortalama gecikme süresinin az da olsa arttığını göstermektedir. Önerilen model, çeşitli ağlar tarafından SDN'i uygulamak veya yeni bir SDN ağı planlamak için kullanılabilir.

Geleneksel IP ağları, ağ anahtarları üzerinde çalışan kontrol ve ağ protokolleri aracılığıyla yönetilmektedir. Bu mimari, her cihazda ayrı ayrı uygulanması gerektiğinden ve her cihaza özel komutlar gerektiğinden ağ yönetimi karmaşık bir hale gelmektedir [83]. Geleneksel ağ mimarilerinde, kontrol düzlemi ve veri düzlemi ayrılmasına olanak sağlayan anahtarlar kullanılmamaktadır. Bu sebeple, ağlarda değişen trafik koşulları, ağ hataları, güvenlik tehditleri vb. gibi ihtiyaçlara dinamik ve esnek çözümler getirilememektedir. SDN, ağlara esneklik ve yazılımsal çözümler üreten ve dinamik ağ yönetimini kolaylaştıran yeni gelişmiş bir teknolojidir [20,24,84,85].

SDN, veri merkezi ağlarında ve Geniş Alan Ağlarında (Wide Area Network-WAN) giderek daha fazla kullanılmaktadır. Google gibi şirketler, dünya genelindeki veri merkezlerinde SDN kullanmaktadır [86]. Son yıllarda SDN'in, 5G [87], uç bilgi işlem [18], nesnelerin interneti [88], mobil/kablosuz ağlar [89], ağ işlevi sanallaştırması (NFV) [90] gibi birçok yeni nesil ağ teknolojisi uygulamalarında kullanımı önem kazanmıştır. SDN'i önemli hale

getiren ana fikir, veri ve kontrol düzlemlerinin birbirinden ayırması ve merkezi bir kontrol düzlemini sağlamasıdır. SDN kontrol düzlemi, tüm ağı izleyerek ve yapılandırarak, yeni bir yönlendirme protokolünü test etmeyi veya yeni bir trafik mühendisliği politikasını tek bir kontrol noktasından uygulamayı kolaylaştırmaktadır.

İlk araştırmalar, tek bir denetleyicinin gerçek ağlar için yeterli olacağını gözlemledi [42]. Ancak, yapılan çalışmalarda ölçeklenebilirlik, güvenlik ve esneklik gibi sebeplerden kaynaklı birden çok denetleyiciye olan ihtiyaç ortaya çıkmıştır [43]. Yapılan çalışmalarda SDN'nin önemi ortaya konulmuştur ancak birden fazla kontrolör kullanılan sistemlerde yeni zorluk ortaya çıkmıştır. CPP, bir veya birden fazla kontrolörün ağda nasıl konumlandırılacağı problemidir. Belirli sayıda düğüme sahip bir ağda, kontrolör sayısı ve konumu açık bir sorundur. Literatürde bu sorunu çözmek için genellikle gecikme, esneklik, güvenilirlik, enerji tasarrufu ve yük dengeleme gibi ölçütler/kısıtlar kullanılmıştır. Kontrolör konumu belirlemede gecikme önemli ölçütlerden biridir. Gelişen ve hızla artan veri trafiğinde zaman çok önemli bir metrik haline gelmiştir. Bu sebeple, bu çalışmada gecikmeyi en optimum seviyede sağlamayı amaçlamaktayız.

Çok kontrolörlü ağlarda, herhangi bir veya birden fazla kontrolör servis dışı kaldığında ağın nasıl yönetileceği önemli bir diğer sorundur. Çalışmamızda kontrolör konumları belirledikten sonra olası gerçekleşebilecek kontrolör hatalarında ağın optimum gecikme ile sürekliliğinin sağlanması amaçlanmıştır. Önerdiğimiz model, Capacity and Fault Tolerant Controller Placement Problem (CF-CPP, Kapasite ve Hata Toleranslı Kontrolör Yerleştirme Problemi) diye isimlendirilmektedir.

CF-CPP, başlangıçta k-median kümeleme algoritması kullanılarak kontrolör sayısı ve konumları belirlenmiştir. Kontrolör sayısı belirlenirken kontrolör hata toleransı kısıt olarak dikkate alınmıştır. Ağ iletimi gerçekleşirken, herhangi bir kontrolör servis dışı kaldığında hangi anahtarların hangi kontrolörlere bağlanması gerektiği önerdiğimiz lineer programlama ile çözümlenmektedir. Kontrolör sayısı ve kontrolördeki boş port sayısı kısıtları ile minimum gecikme hedeflenmiştir. Ağın sürekliliğini sağlamayı ve ortalama gecikme süresini en aza indirgemeyi sağlarken kontrolör sayısı ve hata toleransının etkilerini inceledik.

CF-CPP'nin ana katkıları aşağıda kısaca vurgulanmaktadır.

- SDN destekli ağlarda kontrolör konumu belirleme ve olası kontrolör hatası durumu için çözümler sunulmuştur.
- CF-CPP, matematiksel olarak formüle edilmiş ve simülasyon ortamında test edilmiştir. Bu çalışmanın asıl amacı ortalama gecikmeyi minimuma indirmek olsa da, ağda gerçekleşebilecek hatalara karşı ağa esneklik ve dinamiklik kazandırmaktır.
- Ağın esnekliğini ve sürekliliğini sağlamak için k-median kümeleme algoritması tabanlı doğrusal programlama ile yeni bir model önerilmiştir.
- Kontrolör konumları belirlenmiş bir ağda olası kontrolör arızası durumunda, bu kontrolöre bağlı anahtarların sistemdeki diğer kontrolörlere atanması sağlanarak ağın sürekliliği sağlanmıştır.

4.1. İlgili Çalışmalar

SDN destekli ağlarda CPP önemli problemlerden biridir ve literatürde geniş bir şekilde incelenmiştir. Yerleştirme gerçekleştirilirken farklı amaçlar belirlenmektedir. Bunlar, minimum gecikme, maliyet, güvenilirlik, dayanıklılık vb. gibi amaçlardır. Bu amaçları sağlarken farklı kısıtlar kullanılmıştır. Kontrolör sayısı, yük dengeleme, kontrolör yazılım güncelleme maliyeti vb. gibi kısıtlar kullanılmıştır [46-49,83,91].

Literatürde, Kontrolör ve anahtarlar arasındaki gecikmeyi optimize eden ve CPP'ye çözüm sunan çalışmalar incelenmiş ve Çizelge 4.1'de gösterilmiştir.

Literatürde kontrolör yerleştirme problemini ilk kez Heller ve arkadaşları [42] ortaya koydular. Kontrolörlerin sayısı ve konumu belirlemek için k-means kümeleme algoritması kullandılar. Bu çalışmada, anahtarlar ve kontrolör arasındaki gecikmeyi optimize ederek CPP'ye çözüm sundular. Kontrolör sayısını artırmanın gecikmeyi azalttığı gözlemlenirken, kontrolör hata toleransı ve kapasite sınırı dikkate alınmamıştır.

Çizelge 4.1. Kontrolör yerleştirme problemi çalışmaları

Kaynak	Kısıt	Amaç	Optimizasyon yöntemi	Ağ performansı			Geliştirilen modelin avantajları
				Süreklilik	Güvenilirlik	Çökme/hata olasılığı	
[42]		Kontrolör sayısı, Gecikme optimizasyonu	K-means		✓		Sistemde gerekli kontrolör sayısını ve konumunu tespit etmek için k-means kümeleme algoritması ile model önerilmiştir.
[92]	Ağ cihazları arasındaki mesafe	Kontrolör sayısı, Yeniden kontrolör yerleşimi	K-median	✓	✓	✓	Simülasyon sonuçları, ağda performans artışı sağlandığını ve yeniden atamanın ortalama gecikmeye etkisinin çok az olduğunu göstermektedir.
[44]	Kontrolör sayısı ve kapasitesi	Minimum gecikme, kontrolör sayısı, Yük dengeli kontrolör yapısı	Doğrusal programlama, K-median	✓	✓		Gecikme, kontrolör kapasitesi ve sayısını belirleyen bir model önerdi. Simülasyon sonuçları, önerilerinin kontrolör sayını ve kontrolör yükünü azalttığını göstermiştir.
[93]	Alt ağ sayısı	Alt-ağlara bölme, Minimum gecikme, Yük dengeleme	Genişletilmiş K-medyan	✓			Büyük ölçekli ağları alt-ağlara böldüler ve alt-ağlarda bir kontrolör atayarak gecikmeyi optimize ettiler.
[94]	Alt ağ sayısı	Alt-ağlara bölme, Minimum gecikme	Genişletilmiş K-medyan				Genişletilmiş k-means algoritmasını ile önerilen modelin 2347 kez daha hızlı sonuç verdiğini gösterdiler
[95]	Ağ alt sayısı	Minimum gecikme, Kontrolör sayısı	Spektral kümeleme		✓		Büyük ölçekli ağları alt-ağlara böldüler ve alt-ağlarda bir kontrolör atayarak gecikmeyi, verimi optimize ettiler.

Çizelge 4.1. (devamı) Kontrolör yerleştirme problemi çalışmaları

Kaynak	Kısıt	Amaç	Optimizasyon yöntemi	Ağ performansı			Geliştirilen modelin avantajları
				Süreklilik	Güvenilirlik	Çökme/hata olasılığı	
[96]	Kontrolör sayısı	Minimum gecikme Yük dengeleme	k-means		✓		Optimal sayıdan daha fazla kontrolör kullanmanın gecikme süresini artırdığını ve gereksiz maliyet oluşturduğunu göstermiştir
[97]	Kontrolör sayısı	Minimum gecikme	k-means, Aç gözlü	✓	✓	✓	Kontrol katmanında önerilen çoklu kontrolör mimarisi ile ağ güvenliği ve sürekliliği dikkate alınarak konum belirlenmiştir.
[98]	Kontrolör sayısı	Minimum gecikme	NDP, K-means	✓	✓	✓	Geniş alan ağlarında NDP-merkezi ve NDP-küme algoritmaları ile alt ağlara ve merkezi ağa kontrolör yerleştirilmesi gerçekleştirilmiştir.
[99]	Kontrolör sayısı Kontrolör kapasitesi	Minimum gecikme	Aç gözlü, Kaba kuvvet, Rastgele yerleşim	✓	✓	✓	SDN destekli ağlarda, kontrolörün arızalanması durumda ağda sürekliliği sağlayan model önerilmiştir. Geliştirilen aç gözlü yaklaşım, rastgele yerleştirme ve kaba kuvvet algoritmasına kıyasla daha iyi sonuçlar sağlamıştır.
[100]	Kontrolör kapasitesi	Minimum gecikme	Sırt çantası, Sezgisel algoritma	✓	✓		Önerilen model, ateş böceği, parçacık sürü ve k-means algoritmaları ile karşılaştırılmıştır. Önerilen model, diğer algoritmalardan daha iyi performans göstermesine ek olarak, en düşük çalışma süresini elde etmiştir.
[101]	Ağ topolojisi Yanıt süresi	QoS Kontrolör sayısı	Aç gözlü sezgisel, bölümleyici kümeleme ve primal-dual	✓	✓		Simülasyon sonuçları, önerilen açgözlü sezgisel yöntemin diğer iki yöntemden ve temel yaklaşımlardan daha iyi performans gösterdiğini göstermektedir.

Kobo ve arkadaşları [92], yazılım tanımlı kablosuz sensör ağılarında uygun kontrolör yerleştirme ve herhangi bir kontrolör arızası durumunda kontrolör değiştirilmesini ön gören modeller önerdiler. Öneriler gerçek veri seti ile deneysel ortamda test edilmiştir. Önerilen modelde, bir cihazın en az bir denetleyiciye bağlanması gerektiği, ancak herhangi bir zamanda en fazla bir denetleyici tarafından kontrol edildiği kabul edilmiştir. Kontrolör yerleşimi, ağ cihazları ile kontrolör arasındaki ve yerel ile merkezi kontrolörler arasındaki gecikmeyi azaltmaktadır. Bu çalışmada, yerel kontrolörler ve merkezi kontrolörler atanması için iki ayrı model önerilmiştir. Yerel kontrolör yerleşimi için k-median algoritması kullanılırken, yeniden atama ve merkezi kontrolör yerleşimi için genişletilmiş k-medyan algoritması kullanılmıştır. Kontrolör arızası sonrası, cihazların yeni bir kontrolöre atanması sağlanmış ve yaşanan gecikme dikkate alınmıştır. Bu değişim ağda kısıtlı bir gecikmeye sebep olmuştur. Sonuç olarak, K-median algoritmasını kullanarak minimum gecikmeyi ve esnekliği sağlayan bir çözüm önerdiler. Simülasyon sonuçları, yerel ve merkezi kontrolör kullanılarak daha iyi performans elde edildiği ve kontrolör değişikliğinin ortalama gecikmeye etkisinin çok az olduğunu göstermektedir.

Guang Yao ve arkadaşları [44] gecikme, kontrolör yükü ve kapasitesini dikkate alan bir model önerdi. Bu model, K-medyan (k-center) algoritmasının probleme uyarlanması ile elde edilmiş Capacitated Controller Placement Problem (CCPP) olarak isimlendirilmiştir. Minimum kontrolör sayısını belirlemek için cihazlar arası mesafeyi dikkate alan doğrusal programlama kullanılmıştır. Simülasyon sonuçları, önerilerinin kontrolör sayını ve kontrolör yükünü azalttığını göstermiştir.

Kuang ve arkadaşları [93], geniş alan ağlarında gecikme ve güvenilirliği hesaba katarak, kontrolör konumu belirlemede hiyerarşik k-means algoritmasını kullanan bir çözüm önerdiler. Büyük ölçekli ağları alt-ağlara böldüler ve her bir alt ağa bir kontrolör atayarak gecikmeyi optimize ettiler. Deneysel sonuçlar önerilen algoritmanın, kontrolör ve anahtarlar arasındaki gecikmeyi azalttığı ve kontrolör yüklerini dengeli dağıttığını göstermektedir.

Wang ve arkadaşları [94], CPP'ye ağı alt bölümlere ayırıp k-means algoritması temelli yeni bir yöntem önerdiler. Çalışmalarında problemi iki alt başlıkta incelidir. Bunlar, büyük ölçekli ağı daha küçük alt ağlara bölmek ve her alt ağa bir kontrolör ile merkezi bir kontrolör atamaktır. Önerilen model kapsamlı ve tekrarlı şekilde simüle edilmiştir. Simülasyon sonuçları, önerilen algoritmanın, standart K-means algoritmasına kıyasla, merkez ve

düğümleri arasındaki maksimum gecikmeyi önemli ölçüde azalttığını göstermiştir. Maksimum gecikmenin, standart k-means algoritmasıyla elde edilen ortalama gecikme süresinden 2.437 kat daha kısa olduğu gözlemlenmiştir.

Xiao ve arkadaşları [95], büyük ölçekli ağlarda CPP problemine spektral kümeleme algoritması ile çözüm önermişlerdir. Ağı alt ağlara bölerek kontrolör konumu belirlenerek ağda gecikme optimize edilmiştir. Jimenez ve arkadaşları [96], kontrolör yerleştirme problemini minimum gecikme ve yük dengeli kontrolörler yerleştirerek çözen k-kümeleme algoritması temelli yeni bir model önermişler. Sonuçlar, optimal sayıdan daha fazla kontrolör kullanmanın gecikme süresini artırdığını ve gereksiz maliyet oluşturduğunu göstermiştir.

Hu ve arkadaşları [97], CPP'ye çözüm olarak uygulanan rastgele yerleşim, açgözlü yaklaşım ve k-means tabanlı yerleştirme algoritmalarını karşılaştırmıştır. Kontrol katmanında önerilen çoklu kontrolör mimarisi ile ağ güvenliği ve sürekliliği dikkate alınarak konum belirlenmiştir. Kontrolör yerleştirme stratejilerinin SDN performansı için çok önemli olduğunu ve açgözlü bir algoritmanın optimuma yakın yerleştirme çözümleri sağladığını göstermektedir.

Alenazi ve Çetinkaya [98], CPP'ye çözüm olarak Nodal Disjoint Path (NDP) yaklaşımlı model önermişlerdir. Geniş alan ağlarında NDP-merkezi ve NDP-küme algoritmaları ile alt ağlara ve merkezi ağa kontrolör yerleştirilmesi gerçekleştirilmiştir. Analiz sonuçları, NDP-küme, k-medyan ve k-merkez algoritmaları ile karşılaştırılmıştır. NDP-merkezi algoritması tarafından kontrolör yerleştirilmesinin, kontrolör hataları karşısında daha iyi ağ esnekliği sağladığını göstermektedir. NDP-küme algoritmasının k-medyan algoritması ile karşılaştırılabilir bir gecikme performansına sahip olduğunu ve daha yüksek ağ esnekliği sağladığını göstermektedir.

SDN, geleneksel ağlarda karşılaşılan sorunlara çözümler üretmekle birlikte yeni sorunları da ortaya çıkarmaktadır. Kontrolör sayısı ve konumu belirlemede güvenilirlik sağlamak önemli bir sorundur. Ağlarda bir veya daha fazla kontrolörün servis dışı kalması karşılaşılan problemlerden biridir [83,102,103].

Yannan ve arkadaşları [99], kontrolör konumu belirlemede güvenilirlik ve sürekliliği tanıttılar. Yazarlara göre, kontrolörün konumu ve güvenilirliği ağ performansını artırmaktadır. SDN destekli ağlarda, kontrolörün arızalanması ile ağ başarısız olabilir. Böyle bir sorunla karşılaşmamak için ağ mimarisinin hataya dayanaklı olması gerekmektedir. Kontrolörün arızalanması beyinsiz bir ağ anlamına gelmektedir. Yazarlar, SDN'de güvenilir kontrolör yerleşimi için rastgele yerleştirme, kaba kuvvet ve açgözlü (greedy) algoritması önerdiler. Greedy algoritması, rastgele yerleştirme ve kaba kuvvet algoritmasına kıyasla daha iyi sonuçlar sağladı. Önerilen modelde, ağ oluşturulurken maksimum sayıda kontrolör ve her anahtarın bağlı olduğu birden fazla kontrolör noktası belirleyerek problemi çözmeye çalışmışlardır. Her ne kadar bu yöntem probleme çözüm üretse de donanım maliyetini artırmaktadır. Ağ performans metriklerinden ortalama gecikme süresi bu çalışmada dikkate alınmamıştır. Ağ esnekliği, güvenilirliği ve sürekliliği sağlanırken donanımsal maliyet ve ortalama gecikme süresi artmıştır.

Torkamani-Azar ve Jahanshahi [100], uçtan uca gecikmeleri azaltmak için kontrolörlerin kapasitesini dikkatte alan yeni bir model önermişler. Sırt çantası algoritmasını temel alan Garter Snake Optimization Capacitated Controller Placement Problem (GSOCCPP) isimli sezgisel bir algoritma önerilmişler. Bu algoritma, minimum gecikmeleri elde etmek için makul miktarda hesaplama süresi kullanır. Topology-Zoo veri setleri ile simüle edilen GSOCCPP, ateş böceği, parçacık sürü ve k-means algoritmaları ile karşılaştırılmıştır. Önerilen model, diğer algoritmalardan daha iyi performans göstermesine ek olarak, en düşük çalışma süresini elde etmiştir. Ayrıca, önerilen bu çözüm, farklı ağ topolojilerinde kontrolör yerleşimi için diğer algoritmalara kıyasla daha verimli bir bellek tüketimine de sahiptir.

Cheng ve arkadaşları [101], Kontrolör konumu belirlerken QoS sağlayan bir model önermişlerdir. Ağ topolojisi ve yanıt süresi kısıt olarak belirlendiğinde, kontrolör sayısı ve konumunu belirleyen sezgisel modeller geliştirilmiştir. Açgözlü, bölümleyici kümeleme ve primal-dual algortiması tabanlı üç farklı sezgisel yaklaşım geliştirmişler. Önerilen modeller. Topology-Zoo veri setleri ile simüle edilmiş. Simülasyon sonuçları, önerilen açgözlü sezgisel yöntemin diğer iki yöntemden ve temel yaklaşımlardan daha iyi performans gösterdiğini göstermektedir.

4.2. Önerilen Model

SDN destekli ağlarda kontrolör yerleştirme, sayısı ve servis dışı kalması önemli bir sorundur. Tezin bu bölümünde belirtilen sorunlara çözüm üreten bir model tanıtılmaktadır. Önerdiğimiz model k-medyan kümeleme algoritması kullanılarak kontrolör konumu ve sayısını belirlemektedir. Ağ iletimi gerçekleştikten sonra olası bir/birden fazla kontrolör hizmet sağlayamadığında ağın sürekliliğini sağlamak için servis dışı kalan kontrolöre bağlı anahtarların sistemde var olan diğer kontrolörlere atanmasını sağlayan bir model önerilmektedir.

K-medyan, k-küme merkezlerini, merkez ile kümedeki diğer tüm noktalar arasındaki mesafelerin toplamını en aza indiren bir kümeleme algoritmasıdır. K-medyan, k-means yaklaşımının geliştirilmiş bir modeli olarak kabul edilmektedir ve merkez ile küme noktaları arasındaki mesafeyi en aza indirmektedir [92].

K-medyan modeli:

Ağ grafiği $G(V,E)$, kenar ağırlıkları gecikmeleri temsil etmektedir, $d(v,s)$, $v \in V$ düğümünden $s \in V$ 'ye en kısa yolu temsil etmektedir, $n = |V|$, düğüm sayısıdır. S', tüm olası kontrolör yerleşim noktaları S seçilen noktalardır. $|S'| = K$ yerleştirilecek kontrolörlerin sayısıdır, kontrolörlerin yerleştirilmesi için ortalama gecikme $L_{avg}(S')$ aşağıdaki şekilde formüle edilmiştir.

$$L_{avg}(S') = \frac{1}{n}\sum_{v \in V} \min_{s \in S} d(v,s) \tag{4.1}$$

K-medyan ile kontrolör sayısı belirlenirken kontrolör hata toleransı dikkate alınmıştır. Hata toleransı, sistemde var olan kontrolörlerin servis dışı kalma olasılığı yüzdesidir. Bu oran ile ağda en az kaç adet kontrolör olması gerektiği hesaplanmaktadır. Deneysel çalışmalar ile bu oranın ortalama gecikme süresi üzerine etkisi incelenmiştir. Elde edilen sonuçlar ile SDN destekli ağlarda işleyişe ve isterlere en uygun şekilde kontrolör sayısı belirlenmektedir.

k, kontrolör sayısını, p, kontrolörün port sayısını, a, anahtar sayısını (switch node), ht, hata tolerans oranını ifade etmektedir.

$$m \geq \frac{a}{p} \qquad m \in \text{tamsayı} \tag{4.2}$$

$$k \geq m(1 + ht) \qquad k \in tamsayı \tag{4.3}$$

Önerdiğimiz modelde, veri iletişimi gerçekleştikten sonra kontrolörün servis dışı kalması sorununa lineer programlama ile çözüm üretilmiştir. Servis dışı kalan kontrolöre bağlı olan anahtarların hangi kontrolöre bağlanarak ağın sürekliliğini sağlayacağı tanımlanmıştır. Kontrolörlerin boşta bulunması gereken port sayısı bu olasılık yüzdesi ile hesaplanmaktadır. Dolayısı ile ağda bulunan kontrolör sayısı artırılırken, her kontrolör için boş port sayısı da aynı oranla artırılmaktadır. Boşta kalan anahtarlar, en az gecikme ile sürekliliği sağlamak için var olan kontrolörlere atanmaktadır.

CF-CPP'nin matematiksel modeli aşağıdaki gibidir.

İndisler

i : anahtar sayısı

j : kontrolörün sayısı

Parametreler

d_{ij} : i anahtarından j kontrolörüne gidiş maliyeti

M_j: j kontrolör kapasitesi

Değişkenler

x_{ij} : i anahtarından j kontrolörüne gidilip gidilmemesi (0-1) değişken

b_j: j kontrolörünün bozulup bozulmadığını belirleyen (0-1) değişkeni

Amaç fonksiyonu:

$$Min \sum_{i \in I} \sum_{J \in j} d_{ij} * x_{ij} \tag{4.4}$$

Kısıtlar:

$$\sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{4.5}$$

$$\sum_{i \in I} x_{ij} \leq M_j(1 - b_j) \qquad \forall j \in J \tag{4.6}$$

$$\sum_{j \in J} b_j = 1 \tag{4.7}$$

$$x_{ij}, b_j = \{0, 1\} \tag{4.8}$$

Amaç fonksiyonu Eş. (4.4), anahtar ve kontrolör arasındaki toplam maliyeti minimum yapmayı hedeflemektedir. Eş. (4.5), her bir anahtarın yalnızca bir kontrolöre atanmasını sağlayan kısıttır. Eş. (4.6), seçilen kontrolörün kalan kapasitelerinin aşılmasını engelleyen kısıttır. Eş. (4.7) ise kontrolörlerden yalnızca birinin servis dışı kalacağını diğerlerinin aksamadan çalışacağını sağlayan kısıttır. Eş. (4.8), değişken özelliklerini tanımlayan kısıttır.

Önerilen model SDN destekli ağlarda CPP'ye kontrolör servis dışı kalma olasılığını hesaplayarak yeni bir öneri getirmektedir. Bu model, kontrolör ile anahtarlar arasındaki iletişimi minimum gecikme ile sağlarken, kontrolör kapasitesi (port) ve kontrolör servis dışı kalma olasılığı kısıtlarını dikkate almaktadır. Ağda en az kaç adet kontrolör olacağı ve kontrolör sayısının artırılmasının sistemin sürekliliği ve ortalama gecikme süresi üzerine etkileri simüle edilmiş ve incelenmiştir. Deneysel incelemeler, başlangıç durum ve önerilen doğrusal modelin ürettiği sonuçlara göre ayrı ayrı simüle edilerek gerçekleştirilmiştir ve karşılaştırılmıştır.

Önerilen modelin avantajları;

- Kontrolörün hizmet sağlayamama olasılığı dikkate alınarak gerekli kontrolör sayısını ve konumunu belirlemektedir.
- Anahtarlar ile kontrolörler arasındaki minimum gecikmeyi sağladığından iletişim performansını artırmaktadır.
- Servis dışı kalan kontrolöre bağlı olan anahtarların minimum gecikme ile var olan kontrolörlere aktarılması ile sistemin sürekliliğin sağlanmaktadır.

Modelin dezavantajları;

- Ağ performansını (minimum gecikme) sağlarken ve kontrolör servis dışı kalma olasılığı hesaplanarak kontrolör sayısı belirlendiğinden kontrolör sayısı artmaktadır. Bu sebeple, donanımsal maliyet artırmaktadır.

4.3. Deneysel Çalışmalar

Bu alt bölümde, kontrolör servis dışı kaldığında ağın sürekliliğini sağlamak ve CPP'ye çözüm sunmak için önerdiğimiz CF-CPP modelinin ortalama gecikme süresine ve kontrolör sayısına olan etkisi incelenmektedir. K-medyan ile anahtar sayısını karşılayan en az sayıda kontrolör için ortalama gecikme hesaplanmakta ve bu süreler kontrolör servis dışı kaldığında gerçekleşen ortalama gecikme süresi ile karşılaştırılmaktadır.

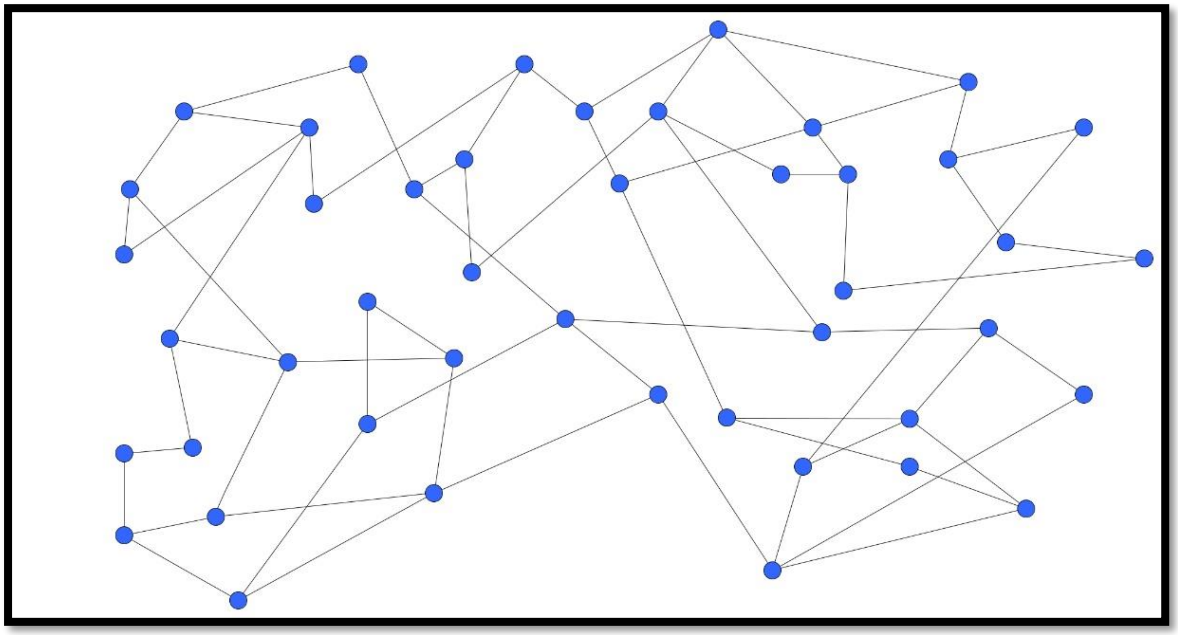
4.3.1. Hesaplama metrikleri

Önerilen modeli değerlendirmek için literatürde yaygın olarak kullanılan metrikler kullanılmıştır. Bu metrikler:

- Gecikme (Latency), CPP'de dikkate alınan temel faktörlerden biridir. Düğümler arasındaki mesafeye bağlıdır ve kontrolör ile anahtarlar arasındaki yanıt süresidir.
- Optimal kontrolör sayısı, ağ performansını artırmak ve QoS sağlamak için anahtar sayısına ve kontrolörün konuma bağlı olarak belirlenmesi gereken sayıdır.
- Hata toleransı, anahtarların kontrolörler ile olan iletişim kopma olasılığıdır. Genellikle kontrolör hatalarından kaynaklanmaktadır. Ağ performansı ve maliyet için en iyi konumda ve en az sayıda kontrolör yerleştirmek önemlidir.

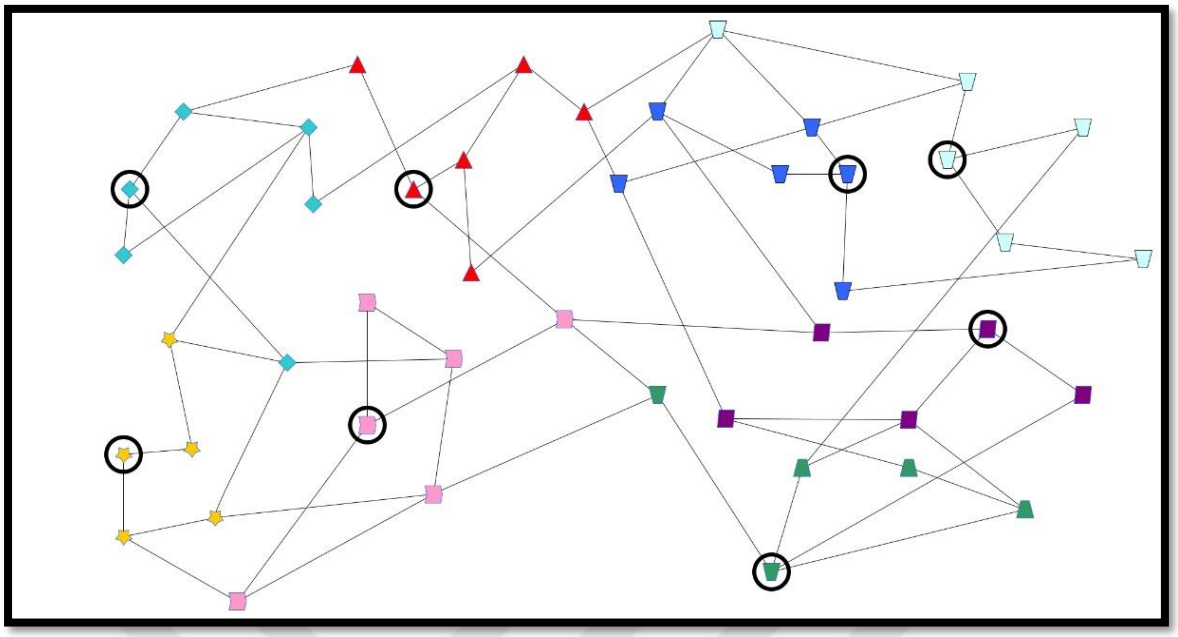
4.3.2. Analiz sonuçları

CF-CPP, Şekil 4.1'de gösterilen 45 düğüm ve 61 kenardan oluşan bir çizelge ile GNS3 platformunda test edilmiştir. Düğümler anahtarları temsil etmektedir. Kontrolörlerin 6 portu olduğu varsayılmıştır. Bu sebeple kontrolörlere bağlanabilecek maksimum anahtar sayısı 6'dır.



Şekil 4.1. Ağ modeli

K-medyan kümelemede ağın kaç alt kümeye ayrılacağını belirtmek gerekmektedir. Eş. (4.2)'de her anahtarın en az bir kontrolöre atanması için gerekli sayı hesaplanmaktadır. Kontrolör sayısı, toplam anahtar sayısının, kontrolör port kapasitesine oranından büyük olan en küçük tam sayı seçilmelidir. Kontrolör port kapasitesi 6 olarak kabul edilmiştir. Bu sebeple, 45 anahtarlı test ortamımız için kontrolör sayısı 8 olmalıdır.



Şekil 4.2. 8-Medyan ile optimal yerleştirme

CF-CPP'de, kontrolör sayısını hesaplarken Eş. (4.3)'te görüldüğü gibi hata tolerans oranını katmaktayız. Hata toleransı, sistemde yerleri ve sayısı belirlenmiş kontrolörlerin servis dışı kalma olasılığıdır. Bu sebeple, kontrolör sayısı belirlenirken bu olasılığına göre kontrolör sayısı artırılmaktadır. Buradaki asıl amaç kontrolör servis dışı kaldığında, boşta kalan her bir anahtarı atayacak en az bir kontrolör bulunmasını sağlamaktır. Çizelge 4.2'de hata tolerans oranına göre kontrolör sayıları gösterilmiştir.

Çizelge 4.2. Hata tolerans oranına göre kontrolör sayısı

Hata tolerans oranı (%)	0	10	20	30	40
Kontrolör sayısı	8	9	10	11	12

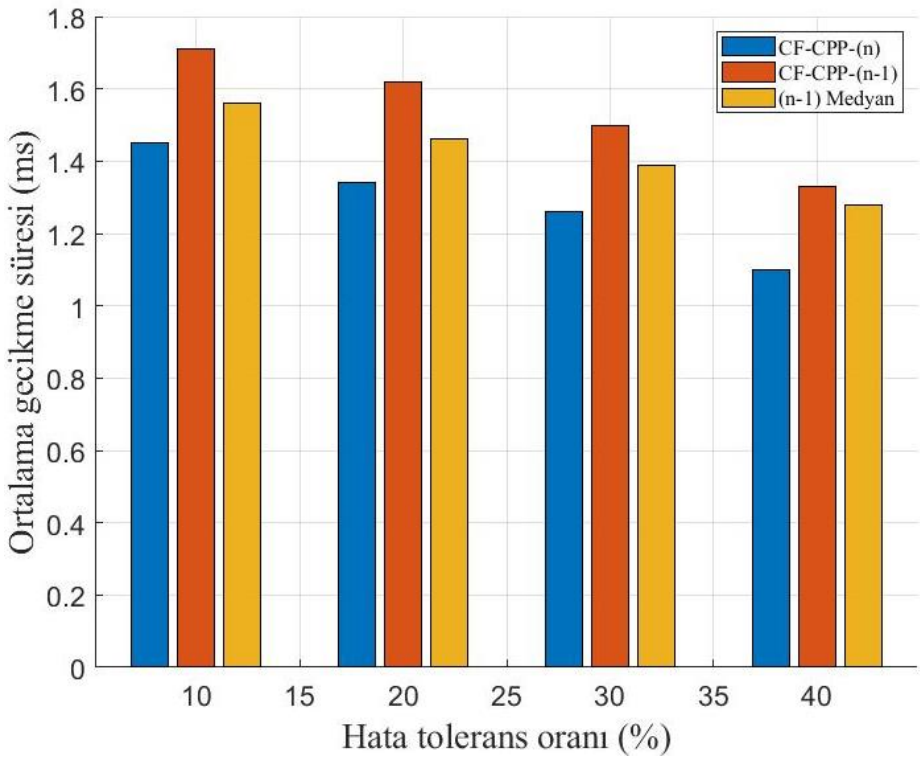
CF-CPP'nin asıl amacı ağın sürekliliğini ve güvenilirliğini sağlamaktır. CF-CPP-(n) Önerilen modelde hiçbir kontrolörün çökmediği durumdur. CF-CPP-(n-1) ise bir kontrolörün servis dışı kaldığı durumu temsil etmektedir. Şekil 4.3'te CF-CPP-(n), CF-CPP-(n-1) ve (n-1) medyanın ortalama gecikme süresine etkisi gösterilmiştir. CF-CPP hata toleransı ile birlikte hesaplanan sayıda kontrolörün yerleştirilmesini sağlayan modeldir. İlk yerleştirme k-medyan ile gerçekleştirilmektedir.

CF-CPP-(n-1), ağın sürekliliğini sağlamak için önerdiğimiz lineer modeldir. Bu modelde ağda iletişim gerçekleşirken atanmış herhangi bir kontrolörün devre dışı kalması durumunda,

ona bağlı olan anahtarların minimum gecikme ve kontrolör port kısıtı ile en uygun kontrolöre atanmasıdır. Ortalama gecikme hesaplanırken hangi kontrolörün bozulacağı belli değildir. Bu sebeple tüm olasılıklar denenerek ortalaması alınmıştır.

(n-1) medyan, sistemde CF-CPP ile hesaplanmış ve bir kontrolörün bozulduğu varsayılmıştır. Ağ yeniden (n-1) adet kontrolör için k-medyan ile kontrolör konumu belirlenerek ortalama gecikme hesaplanmıştır. (n-1)'de asıl amaç ağda bir kontrolör çöktüğü zaman yeniden tüm anahtarları kontrolörlere atamaktır. Sırayla tüm atanmış kontrolörler sistemden atılarak gecikme hesaplanmış ve ortalaması alınmıştır.

Şekil 4.3'te görüldüğü üzere, hata tolerans oranı artıkça ortalama gecikme süresi azalmaktadır. Çünkü hata tolerans artıkça ağa atanan kontrolör sayısı artmaktadır. CF-CPP-(n-1)'de iletişimin ortalama gecikme süresi (n-1) medyan'a göre daha yüksek çıkmaktadır. Ancak önerilen modelde ağın sürekliliği sağlanırken, yeniden atama yapıldığında ağda iletişim kesintisi ve veri kaybı gerçekleşmektedir.



Şekil 4.3. Hata tolerans oranına göre ortalama gecikme süreleri

4.4. Bölüm Değerlendirmesi

SDN destekli ağlarda, kontrolör sayısı, kontrolörler arası ilişki ve kontrolör konumu performansı doğrudan etkilemektedir. Çok kontrolörlü ağlarda, herhangi bir veya birden fazla kontrolörün servis dışı kalması durumunda ağın nasıl yönetileceği önemli bir sorundur. Çalışmamızda kontrolör konumları belirledikten sonra olası gerçekleşebilecek kontrolör hatalarında ağın optimum gecikme ile sürekliliğinin sağlanması amaçlanmıştır. Önerdiğimiz model, Capacity and Fault Tolerant Controller Placement Problem (CF-CPP, Kapasite ve Hata Toleranslı Kontrolör Yerleştirme Problemi) diye isimlendirilmiştir. Birden fazla kontrolör bulunduran sistemlerde sunucu hata toleransı dikkate alınarak gerekli kontrolör sayısını ve konumunu belirleyen matematiksel bir model önerdik. Modelin amacı, kontrolör kapasite ve kontrolörün servis dışı kalma olasılığı kısıtlamalarını göz önünde bulundurarak, ağın sürekliliğini sağlamak ve ortalama gecikmeyi en aza indirmektir. Simülasyon sonuçları, ağın sürekliliğini sağlarken ortalama gecikme süresinin az da olsa arttığını göstermektedir. Önerilen model, çeşitli ağlar tarafından SDN'i uygulamak veya yeni bir SDN ağı planlamak için kullanılabilir.

CF-CPP, başlangıçta k-median kümeleme algoritması kullanılarak kontrolör sayısı ve konumları belirlenmiştir. Kontrolör sayısı belirlenirken kontrolör hata toleransı ve kontrolör boş port sayısı kısıt olarak dikkate alınmıştır. Ağ iletimi gerçekleşirken herhangi bir kontrolör servis dışı kaldığında hangi anahtarların hangi kontrolöre bağlanması gerektiği önerdiğimiz lineer programlama ile çözümlenmektedir. Ağın sürekliliğini ve ortalama gecikme süresini en aza indirgemeyi sağlarken kontrolör sayısı ve hata toleransının etkilerini inceledik. Sonuçlara göre, sürekliliği sağlamak için kontrolör hata toleransı artırılabilir. Önerdiğimiz model ile ortalama gecikme süresi az artmasına rağmen, ağın güvenilirliği ve sürekliliği önemli ölçüde artmaktadır. Ek olarak, elde edilen sonuçlar ile dinamik bir şekilde ağın performansı iyileştirilmektedir.

5. İLİŞKİLİ SİS SUNUCULARI YERLEŞTİRME PROBLEMİNE ÖNERİLEN ÇÖZÜM YÖNTEMLERİ

Nesnelerin internetine bağlı cihaz sayısı hızla artığı günümüze bu cihazlardan elde edilen verilerde hızla artmaktadır. Bölüm 3'te önerilen modellerde sunucu konumu bu verilerin işlenmesinde ve ağ performansında kritik bir önem taşımaktadır. Önerilen modellerde veriler işlem/ görev atanmış sunuculara iletilmekteydi. Bu sebeple sunucuların IoT cihazlarına olan konumu ortalama gecikme, verim vb. QoS gereksinimlerini doğrudan etkilemektedir.

Tezin bu bölümünde, IoT sensör verileri ilgili sis sunucularında işlenmekte ve ortalama gecikme için optimize edilmektedir. Öncelikle her bir sis sunucusuna farklı bir görev atanır. Daha sonra her görev için belirlenen aynı sensör verileri bu sunuculara iletilir ve işlenir. Bu sis sunucularında görev gerçekleştirildikten sonra yeni bir veri tipi oluştur ve bu veri başka bir sis sunucusunun girdi verisi olabilir. Bu yapı sis sunucuları arasında bir ilişki oluşturmaktadır. İlişkili sunucuların konumu, sadece IoT sensörlerine olan mesafeye değil aynı zamanda ağ alt yapısında bulunan diğer sis sunucuları konumuna da bağlıdır.

SDN destekli sis hesaplama yöntemlerinde, sis sunucularının konumunun belirlenmesi performansı etkilemektedir. Literatürde SDN kontrolörünün, sis sunucularının konumlarının belirlenmesi üzerine çalışmalar mevcuttur. Literatürde ki çalışmaların çoğu, sensörlerden elde edilen verilerin tek bir sunucuda işlem görmesinin getirdiği dezavantajları ortadan kaldırmak için belirli bölgelerde aynı işlemleri yapan kopya (replica) sunucuların yerleştirilmesi ile performans ölçütlerinin iyileştirilmesi üzerine gerçekleştirilmiştir [104-108].

Farklı noktalarda bulunan sensörlerden elde edilen veriler, işlenmesi gereken sunuculara iletilmektedir. Sis sunucularına iletilen veriler işlendikten sonra elde edilen sonuçlar başka bir analiz veya yeni bir veri işleme için diğer sis sunucularının girdi verisi olabilmektedir. Kısaca açıklamak gerekirse, hiyerarşik ve dağıtık veri işlemeye uygun sis sunucu sistemi tasarlanmaktadır.

Sis sunucularının konumunun belirlenmesinde, belirli veri tiplerinin aynı sunucuya iletilmesi ve minimum gecikme, maksimum verim ve sunucular arası iletim dikkate alınarak gerçekleştirilmiştir. İlgili sunucu yerleştirme problemi için p-medyan yaklaşımına sahip tam

sayılı doğrusal programlama modeli önerilmiştir. Büyük ağlarda daha kısa sürede çözüm üretebilmek için açgözlü yaklaşımlı sezgisel bir model önerilmiştir. Sezgisel model, büyük ölçekli ağlarda çok daha kısa sürede çözümler üretmiş, ancak ortalama gecikme süresini kısmen artırmıştır.

5.1. İlgili Çalışmalar

Literatürde SDN ve sis hesaplama teknolojilerini birleştirerek sunucu yer seçiminde QoS ihtiyaçlarına daha etkin çözümler üreten çalışmalar incelenmiştir. Ağ alt yapılarında sis sunucuları, ana sunucu ve kontrolör sunucu konumlarının belirlenmesi ağ performansını doğrudan etkilemektedir. Literatürde hizmet sağlayan sunucuların konumunun belirlenmesi problemleri incelenmiştir. Bunlar, SDN kontrolörünün konumu, sis sunucularının konumu ve ana (main) sunucularının konumunun belirlenmesidir. Tüm bu problemlerin çözümü ağ optimizasyonunu gerçekleştirerek ağ performansını artırmaya yöneliktir ve çözüm önerileri benzerlik taşımaktadır. Yapılan çalışmalarda SDN destekli ağlarda kontrolör sunucusunun konumunun belirlenmesi üzerine birçok çalışma mevcuttur. Kontrolör sunucusunun konumunun belirlenmesi ana sunucusunun veya servis hizmeti veren sunucuların konumunun belirlenmesi ile benzer bir problemdir [46,109].

Literatürde, bir ağ altyapısına kritik hizmetler sağlayan öğelerin yerleştirilmesinin, söz konusu altyapının sağladığı QoS üzerinde etkileri olduğu fikri, SDN araştırmalarında yaygın olarak görülmektedir. SDN destekli ağlarda sis sunucularının konumlarının belirlenmesine farklı yöntemler kullanılmaktadır. Bu yöntemler ile ağ performansı iyileştirmeye çalışılmıştır. Kümeleme algoritmaları, sezgisel algoritmalar, yapay sinir ağları, makine öğrenmesi, derin öğrenme kullanılan yöntemlerdendir Literatürde yaygın olarak sis sunucularının kopyasının oluşturularak yerleştirilmesi problemi incelenmiştir. Sis sunucularının farklı işlemler gerçekleştirecek şekilde tasarlanan çalışmalar oldukça azdır [110]. Sis hesaplama, depolama ve ağ oluşturma kaynaklarına sahip küçük tesisler olan sis düğümlerinden oluşur ve son kullanıcılar için yerel işlem sağlamak üzere çeşitli konumlara yerleştirilir [104,105].

Literatürde gerçekleştirilen sunucu yerleştirme problem çalışmaları Çizelge 5.1’de derlenmiştir.

Çizelge 5.1. Sunucu yerleştirme problemi çalışmaları

Kaynak	Kısıt	Amaç	Konumu belirlenen sunucu tipi	Sunucu tipi		Yöntem		Önerilen model
				ilişkili	ilişkisiz	Doğrusal prog.	Sezgisel alg.	
[106]	Kapasiteli sunucu, Sabit kapasiteli sunucu	Yük dengeleme, minimum gecikmeli	Sis kenar, bulutcuk		✓	✗	✗	k-medyan ve açgözlü algoritmaları ile yeni bir yöntem önermişlerdir.
[107]	Sunucu sayısı	Yük dengeleme, minimum gecikmeli	Kopya proxy		✓	✗	✗	k-medyan ve açgözlü algoritmaları ile yeni bir yöntem önermişlerdir.
[108]	Yük dengeleme, minimum gecikmeli	Enerji tüketimi	Sis	✓		✗	✗	sezgisel bir model olan Energy and Demand Tradeoff Algorithm (EDTA) önerilmiştir. EDTA ile son kullanıcılar için önemli ölçüde enerji tasarrufu sağladığını göstermektedir.
[111]	Sunucu sayısı, Minimum gecikme, güvenlik	Enerji tüketimi	Sis		✓	✗	✗	Karışık tamsayı programlama ve karınca-koloni optimizasyonu ile geliştirilmiş sezgisel bir algoritma kullanılmıştır.
[112]		Minimum gecikme, Sunucu sayısı	Sis, bulut		✓	✗		Gecikmeyi ve kaynak kullanımı optimize eden doğrusal programlama çözümü sunmuşlardır. mevcut algoritmalara kıyasla %20 daha başarılı olduğu belirtilmiştir.
[113]	Sunucu sayısı	Minimum gecikme, Enerji tüketimi	Sis, QoS sunucusu	✓		✗	✗	Sis hesaplama tabanlı kurulan ağlarda, farklı sis katmanları kurulması önerilmiştir.
[114]		Sistem maliyeti, Ortalama gecikme	Sis, Bulut		✓	✗		Önerilen model Alternating Direction Method of Multipliers (ADMM) diye adlandırılmıştır. Analiz sonuçları, sistem maliyeti, veri oranı, ortalama gecikme ve kullanıcı başı veri oranı metrikleri ile karşılaştırılmıştır.

Çizelge 5.1. (devamı) Sunucu yerleştirme problemi çalışmaları

Kaynak	Kısıt	Amaç	Konumu belirlenen sunucu tipi	Sunucu tipi		Yöntem		Önerilen model
				ilişkili	ilişkisiz	Doğrusal prog.	Sezgisel alg.	
[115]	Sunucu kapasite ve sayısı	Maksimum kullanıcı sayısı, Ortalama gecikme	Sis, kopya sunucu	✓		✗	✗	Model gerçek veri seti ile simüle edilmiş ve son kullanıcı sayısı, sis sunucu sayısı, sunucu kapasitesi metrikleri ile analiz edilmiştir. Büyük ölçekli ağlarda önerilen sezgisel model 3 kat daha iyi sonuç üretmiştir.
[116]		İş yükü, Ortalama gecikme	Mobil-kenar ve kopya	✓		✗	✗	Shanghai Telecom veri seti ile modeller uygulanmıştır. Önerilen model, K-means, rastgele yerleşim modelleri karşılaştırılmış ve daha başarılı sonuçlar elde edilmiştir.
[117]	Kapasite ve maliyet	Sunucu sayısı, Ortalama gecikme	Mobil ve kenar		✓	✗	✗	Önerilen model ağ büyüklüğü, gecikme ve sis sunucu sayısı metrikleri ile karşılaştırılmıştır ve rastgele yerleştirme algoritması ile karşılaştırıldığında dağıtılan sis sunucu sayısı %16,7 oranında azalmıştır.
[19]	Kullanıcı tahsisi	Yük dengeleme, Ortalama gecikme	Sis, Mobil		✓	✗		K-means, ağırlık öncelikli algoritma temelli önerilen model, rastgele erişim ila karşılaştırılmıştır. Simülasyon sonuçları, sunucu konumları belirlenirken, mobil cihazların gecikme sürelerinin etkili bir şekilde azaldığını göstermektedir.
[118]	QoS	Ortalama gecikme, Sunucu ve kullanıcı sayısı	Bulutçuk, mobil kenar	✓		✗	✗	Problemin çözümü için tam sayılı doğrusal model ve açgözlü sezgisel bir yaklaşım önerilmiştir. Önerilen model [115]'te önerilen VSVBP ile karşılaştırılmıştır. Analiz sonuçları, önerilen modelin VSVBPden farklı olarak dinamik QoS sağladığını, gecikmeyi ve sunucu çalışma zamanını iyileştirdiğini göstermektedir.

Çizelge 5.1. (devamı) Sunucu yerleştirme problemi çalışmaları

Kaynak	Kısıt	Amaç	Konumu belirlenen sunucu tipi	Sunucu tipi		Yöntem		Önerilen model
				ilişkili	ilişkisiz	Doğrusal prog.	Sezgisel alg.	
[119]	Ağ büyüklüğü	Minimum gecikme, Sunucu sayısı	Sis, kenar ve mobil	✓		✗		Önerilen model, ağ büyüklüğü ve ortalama gecikme metrikleri ile rastgele yerleşim algoritması ile karşılaştırılmıştır. Önerilen model, ortalama ağ gecikmesinde, büyük ölçekli ağlarda daha başarılı sonuçlar elde etmiştir.
[120]	Ortalama gecikme	Enerji tüketimi, Sunucu sayısı	Bulutçuk		✓	✗		Önerilen model k-means algoritmasının geliştirilmesi ile gerçekleştirilmiştir.
[121]	Kapsama alanı	Kullanıcı sayısı, Ortalama gecikme	Mobil cihaz, Kenar ve sis	✓		✗		EACP-CA, kapsama alanı ve ortalama kullanım değeri metrikleri ile k-means kümeleme yaklaşımı ile karşılaştırılmıştır. Analiz sonuçları, EACP-CA'nın k-means'e daha başarılı sonuçlar elde ettiğini göstermektedir

Xu ve arkadaşları [106], sis sunucu konum belirleme problemini tamsayı doğrusal programlama ile çözümlemektedir. Kapasiteli sis-kenar-bulutçuk (cloudlet) yerleştirme ve dinamik istek atama problemini çözmektedirler. Ortalama erişim süresini en aza indiren kapasiteli sis-bulutçuk sunucuların yerleştirme problemine çözüm önerilmiştir. Geliştirilen doğrusal programlama isterleri tam karşılayamadığından sezgisel model geliştirilmiştir. Tüm sis sunucularının aynı kapasiteye sahip olduğu kabul edilmiştir. Bununla birlikte, mobil kullanıcılar genellikle hareketli durumda olmasına rağmen çalışmada kullanıcılar sabit kabul edilmiştir. Bulut bilişim sunucu konumu problemine k-medyan ve açgözlü algoritmaları ile yeni bir yöntem önermişlerdir. Simülasyon sonuçları, önerilen algoritmaların umut verici ve ölçeklenebilir olduğunu göstermektedir.

Q Shi ve arkadaşları [107], İçerik dağıtım ağlarında ihtiyaç duyulan kopya-proxy sunucuların yerleştirme problemini incelediler. Ortalama gecikmeyi azaltmak ve sunucu yük dengelemeyi sağlamayı amaçlamışlardır. Özelikle büyük ölçekli ve yüksek hacimli ağlarda çok sayıda sunucu ihtiyacı doğmaktadır. Bu sebeple, çoğaltılmış (kopyalanmış) web sunucuları yerleştirme stratejisi için açgözlü algoritma temelli sezgisel model önermektedirler. Önerilen model Capacitated Facility Location Problem with Edge Capacity (CFLPEC) olarak isimlendirilmiştir. CFLPEC, ağı bölütlemek için k-medyan algoritmasını ve her bir alt ağa sunucu atamak için açgözlü sezgisel algoritmasını kullanmaktadır. Her bir alt ağ için yapılan atamalar, alt ağ sınır noktaları için yeniden değerlendirilerek birleştirilmiştir. CFLPEC, simülasyon ortamında farklı büyüklükte ağlar için gerçekleştirilmiştir ve analiz sonuçları önerilen algoritmanın uygulanabilirliğini ve etkinliğini göstermektedir.

da Silva and Nelson [108], düşük gecikmeli kısıtlamalarla ağır iş yüklerini işleyebilen sis düğümü konum sorununa bir çözüm önermektedir. Önerilen yaklaşım enerji tüketimini en aza indirgemek için sis sunucularının konumunu belirlemektir. Problem için çözüm önerileri, çok kriterli karma tamsayı doğrusal programlamadır ve küçük ölçekli ağlarda başarılı performans göstermiştir. Buna ek olarak büyük ölçekli ağlarda yerleştirme problemi için sezgisel bir model önermişler ve Energy and Demand Tradeoff Algorithm (EDTA) olarak isimlendirmişlerdir. Gerçek bir veri seti kullanılarak elde edilen sonuçlar, EDTA'nın çok kriterli karma tamsayı doğrusal programlama sonuçlarına benzer sonuçlar elde ettiğini göstermiştir. EDTA ile son kullanıcılar için önemli ölçüde enerji tasarrufu sağladığını göstermektedir.

Huang X. Arkadaşları [111], en az sayıda sis sunucusu kullanarak enerji tüketimini en aza indirirken, ortalama gecikme süresi ve güvenlik kısıtlarını karşılayan bir sezgisel model önerilmiştir. Önerilen modelde, karışık tamsayı programlama ve karınca-koloni optimizasyonu ile geliştirilmiş sezgisel bir algoritma kullanılmıştır. Atanacak sis sunucusu sayısı tam sayı olacağından karışık tamsayı programlama kullanılmıştır. Kapsamlı simülasyon sonuçları, gecikme ve güvenlik kısıtlamaları ile sis sunucusu konum sorunlarını çözmek için yeni bir strateji sağladığını göstermiştir.

Shah ve Wong [112], sis-bulut bilişiminde kaynak tahsisi stratejisi önermektedir. Gecikmeyi ve kaynak kullanımı optimize eden doğrusal programlama çözümü sunmuşlardır. Deneysel sonuçları, gecikmeye duyarlı IoT uygulamaları için düşük gecikmeli sis bilişim hizmetlerini sağladığını göstermektedir. Önerilen modelin, literatürdeki mevcut algoritmalara kıyasla %20 daha başarılı olduğu belirtilmiştir.

Souza ve arkadaşları [113], sis sunucularının kapasite gereksinimlerini belirlerken yaşanan gecikmeyi en aza indirerek sis bulut senaryolarında hizmet tahsisini optimize etmekte ve konumunu belirlemektedir. Sis hesaplama tabanlı kurulan ağlarda, farklı sis katmanları kurulması önerilmiştir. Ağı alt bölümlere ayıraran benzer çalışmaların aksine bu çalışmada sis mimarisini katmanlara ayırarak çözüm önerilmiştir. Ağ katmanlara bölmek ve yönetmek için önerilen model, enerji ve gecikmeyi iyileştirirken daha fazla donanıma ihtiyaç duyulmuştur.

Liu ve arkadaşları [114], toplam sistem maliyetini ve gecikmeyi en aza indirmeyi amaçlayan bir optimizasyon problemi olarak kullanıcı planlamasını ve sis sunucusu konum tahsis problemini formüle etmişlerdir. Önerilen model Alternating Direction Method of Multipliers (ADMM) diye adlandırılmıştır. Analiz sonuçları, sistem maliyeti, veri oranı, ortalama gecikme ve kullanıcı başı veri oranı metrikleri ile karşılaştırılmıştır. ADMM, Non-Orthogonal Multiple Access (NOMA) ile karşılaştırılmış ve daha başarılı sonuçlar elde edilmiştir.

Lai ve arkadaşları [115], sis sunucu tahsis problemini bir kutu paketleme problemi olarak formüle etmektedir. Maksimum sayıda kullanıcıya hizmet sağlayacak konumları, gecikmeyi optimize ederek belirlemektedir. Önerilen model hedef programlama ile geliştirilmiş sonuçlar NP-zor çıktığından açgözlü ve rastgele erişim temelli sezgisel algoritmalar

geliştirilmiştir. Model gerçek veri seti ile simüle edilmiş ve son kullanıcı sayısı, sis sunucu sayısı, sunucu kapasitesi metrikleri ile analiz edilmiştir. Büyük ölçekli ağlarda önerilen sezgisel model 3 kat daha iyi sonuç üretmiştir.

Wang ve arkadaşları [116], sis sunucuların yerleşimini çok amaçlı bir optimizasyon problemi olarak formüle etmektedir. Ardından, en uygun çözümü bulmak için karma tam sayılı programlama modeli önerilmiştir. Sunucu iş yükü dengeleme ve gecikme optimize edilmiştir. Shanghai Telecom veri seti ile modeller uygulanmıştır. Önerilen model, K-means, rastgele yerleşim modelleri karşılaştırılmış ve daha başarılı sonuçlar elde edilmiştir.

Chen ve arkadaşları [117], farklı sistem gereksinimlerine iki verimli yöntem tasarlanmıştır ve bulut uygulamasının çalışmalarındaki sis-kenar bulutçuk(cloudlet) sunucuların yerleşimi tarihsel istatistiklere göre belirlemişlerdir. İlk algoritmada, kapasite ve maliyetler kısıtlarına göre sezgisel bir yerleştirme modeli önerilmiş ve ikinci algoritmada sunucu sayısını en aza indirmek ve ortalama gecikmeyi azaltmak için tam sayılı doğrusal programlama formüle edilmiştir. Önerilen model ağ büyüklüğü, gecikme ve sis sunucu sayısı metrikleri ile karşılaştırılmıştır ve rastgele yerleştirme algoritması ile karşılaştırıldığında dağıtılan sis sunucu sayısı %16,7 oranında azalmıştır.

Literatürde mobil kullanıcılarının, sis sunucularında işlem yapabilmesi için geliştirilmiş sunucu konumu belirleme algoritmaları sensör verileri içinde kullanılabilmektedir. Sabit mobil kullanıcılar, sensörler gibi veri ilettiğinden ve hizmet aldığından sis sunucu konum belirleme algoritmaları ile benzerdir. Bu sebeple literatürde sabit mobil cihazlar için geliştirilmiş sunucu konum belirleme çalışmaları da incelenmiştir.

Jia ve arkadaşları [19], bulut sunucu yerleştirme ve kullanıcı tahsisi sorununa odaklanmaktadır. Bulut uygulamalarının hangi sis sunucularına nasıl dağıtılacağına ve kullanıcıları sis sunucularına nasıl atayacaklarına karar vermek için yeni bir yöntem önermektedirler. Önerilen modelde, sunucu yük dengeleme ve gecikme optimize edilmiştir. K-means, ağırlık öncelikli algoritma temelli önerilen model, rastgele erişim ila karşılaştırılmıştır. Simülasyon sonuçları, sunucu konumları belirlenirken, mobil cihazların gecikme sürelerinin etkili bir şekilde azaldığını göstermektedir.

Lai ve arkadaşları [118], mobil kullanıcılar için hizmet kalitesi (QoS) ölçütlerini dikkate alarak QoS uç kullanıcı konum belirleme problemine dönüştürmektedir. Problemin çözümü için tam sayılı doğrusal model ve açgözlü sezgisel bir yaklaşım önerilmiştir. Önerilen model, ortalama gecikme, sunucu sayısı ve kullanıcı sayısı kısıtları ile rastgele erişim, en uygun yerleşim ve [115]'te önerilen Ariable Sized Vector Bin Packing (VSVBP) algoritmaları ile karşılaştırılmıştır. Analiz sonuçları, önerilen modelin VSVBP'den farklı olarak dinamik QoS sağladığını, gecikmeyi ve sunucu çalışma zamanını iyileştirdiğini göstermektedir.

Liang ve Li [119], mobil cihazların coğrafi konumuna göre, ağı birden çok kümeye bölerek ve hizmet sunucularını belirlenen kümelerin merkezlerine en yakın düğüme yerleştirmektedir. K-means algoritması temelli önerilen model hizmet sağlayan sunucuları en az gecikme ve sunucu sayısı ile yerleştirmeyi amaçlamaktadır. Önerilen model, ağ büyüklüğü ve ortalama gecikme metrikleri ile rastgele yerleşim algoritması ile karşılaştırılmıştır. Önerilen model, ortalama ağ gecikmesinde, büyük ölçekli ağlarda daha başarılı sonuçlar elde etmiştir.

Shen ve arkadaşları [120], kümeleme algoritması ile cloudlet yerleştirme için dinamik bir yöntem önermektedirler. Ortalama gecikme süresi kısıt olarak belirlenirken, enerji tüketimi ve kullanılan sis sunucu sayısı optimize edilmiştir. Önerilen model k-means algoritmasının geliştirilmesi ile gerçekleştirilmiştir.

Zhang ve arkadaşları [121], mobil cihazların konumuna dayalı uyarlanabilir bir kümeleme yöntemi önermektedirler. Ek olarak, dinamik değişiklikleri gerçekleştirmek için bulut uygulamasının başlangıç konumu ve hedef konumu belirlenmektedir. Önerilen model, Enhanced Adaptive Cloudlets Placement approach based on Covering Algorithm (EACP-CA) olarak isimlendirilmiştir. EACP-CA, kapsama alanı ve ortalama kullanım değeri metrikleri ile k-means kümeleme yaklaşımı ile karşılaştırılmıştır. Analiz sonuçları, EACP-CA'nın k-means'e daha başarılı sonuçlar elde ettiğini göstermektedir.

Ağ alt yapılarında sis, kenar bulutçuk (cloudlet) gibi servis hizmeti sağlayan sunucuların konumu ağ performansını etkilemektedir. Literatürde bu alanda yapılmış çalışmalar bulunmakla birlikte SDN destekli ağlarda yapılmış çalışma çok azdır. Ek olarak farklı sunucularda farklı hizmet birimleri oluşturan ve bu birimler arasında ilişki kuran çalışma

bulunmamaktadır. Çalışmamızda sensörlerden gelen verilerin işlenmesi gereken sunucuya iletilmesi, iletilen sunucuda gerçekleştirilen veri işleme sonrası elde edilen anlamlı verinin bir başka sunucuya iletilmesi SDN destekli bir algoritma ile gerçekleştirilmektedir. Ortalama gecikme süreleri doğrusal programlama ile çözümlenmiş ve en uygun sunucu konumları belirlenmiştir.

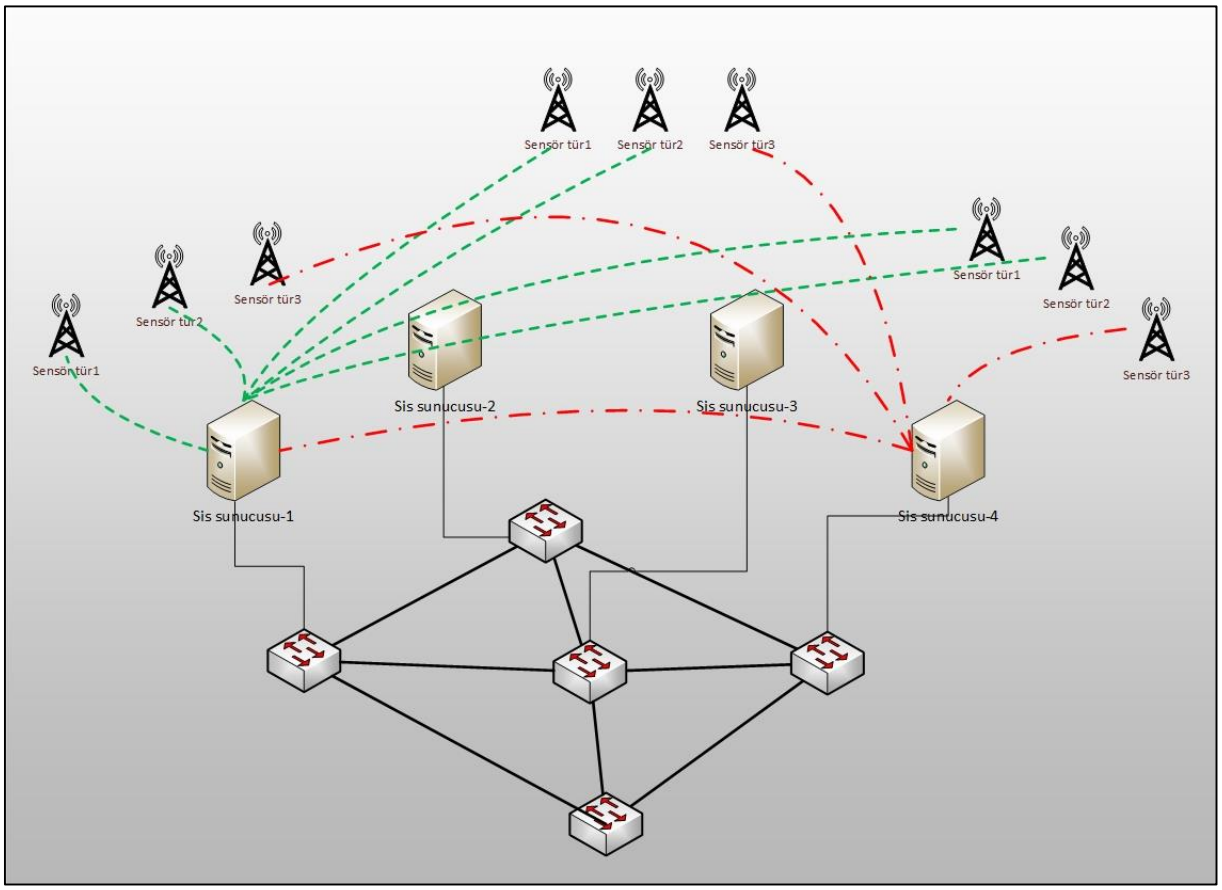
5.2. Problemin Tanımı

Önceki bölümde açıklandığı gibi, IoT ağlarda Sis-kenar mimarileri yaygın olarak kullanılmaktadır. Literatürde ağ performansını etkileyen sebeplerden birinin sis-kenar sunucularının konumu olduğu önceki bölümde açıklanmıştı. Sunucuların, veri üreten sensörlere göre olan konumu veri iletimini, gecikme süresini verimi doğrudan etkilemektedir. Önerilen model Şekil 5.1'de gösterilmiştir.

Geleneksel ağ mimarisinde sensörlerden elde edilen veriler merkezi bir sunucuda toplanmaktadır. Veri işleme, veri depolama ve hizmet sağlama vb. gibi işlemler gerçekleştirilmektedir. Sensör verilerinin tamamının tek bir sunucuya iletilmesi çok fazla verinin ağda iletilmesine ve ağ trafiği oluşmasına sebep olmaktadır. Ağ trafiğini azaltmak için sis mimari modeli kullanılabilir. Sis mimarisinde, ağın kenarlarında daha küçük işlemci ve depolama kapasitesine sahip cihazlardan oluşmaktadır. Sis mimarisi performansı artırırken yeni sorunları da ortaya çıkarmaktadır. Sis sunucuların konumun belirlenmesi ortaya çıkan yeni problemdir. Sis sunucuları farklı iş yüklerine sahip olabileceği gibi kopya sunuculardan da oluşabilir. Kopya sis sunucularının konumu da performansı etkilemektedir. İlgili çalışmalarda açıklandığı gibi, ağ belirli kriterlere göre kümelere ayrılarak her kümeye bir sis sunucusu yerleştirilmiştir. Kopya sis sunucusu yerleştirmede sunucuların birbirleri ile ilişkisi veya bağı bulunmamaktadır. Çalışmamızda belirtilen problemde ise her sis sunucunda farklı veri türlerini işlemektedir ve sis sunucuları birbirleri ile ilişkilidir. Yani her sis sunucusu farklı veri türlerini işleyerek sonuç üretmektedir. Sunucuda sensörlerden gelen veriler işlenerek elde edilen anlamlı sonuç veya yeni veri türü sistemde bulunan başka bir sis sunucusunun girdi verisi olmaktadır.

Şekil 5.1'de gösterilen topolojide her sunucu belirlenmiş veri seti kümesini işlemek ile görevlendirilmiştir. Sensörlerden gelen veriler belirlenmiş altkümelere ayrılarak bu sunuculara iletilmektedir. Sunucularda gerçekleştirilen analiz sonucu elde edilen yeni veri

türleri de veriler kümesine dahil edilmektedir. Bu yapı eş zamanlı veri işleme, farklı analiz sonuçlarını elde etme vb. gibi veri analizinde kolaylıklar sağlamaktadır aynı zamanda ağ altyapısında daha az ve yönetilebilir veri iletimi gerçekleştirerek performansın artırılması sağlamaktadır. Sunucu seçimi belirlenirken farklı kısıtlar oluşturulmaktadır. Bu kısıtlar tam sayılı doğrusal programlama yardımı ile çözümlenerek optimum sonuç elde edilmesi amaçlanmaktadır.



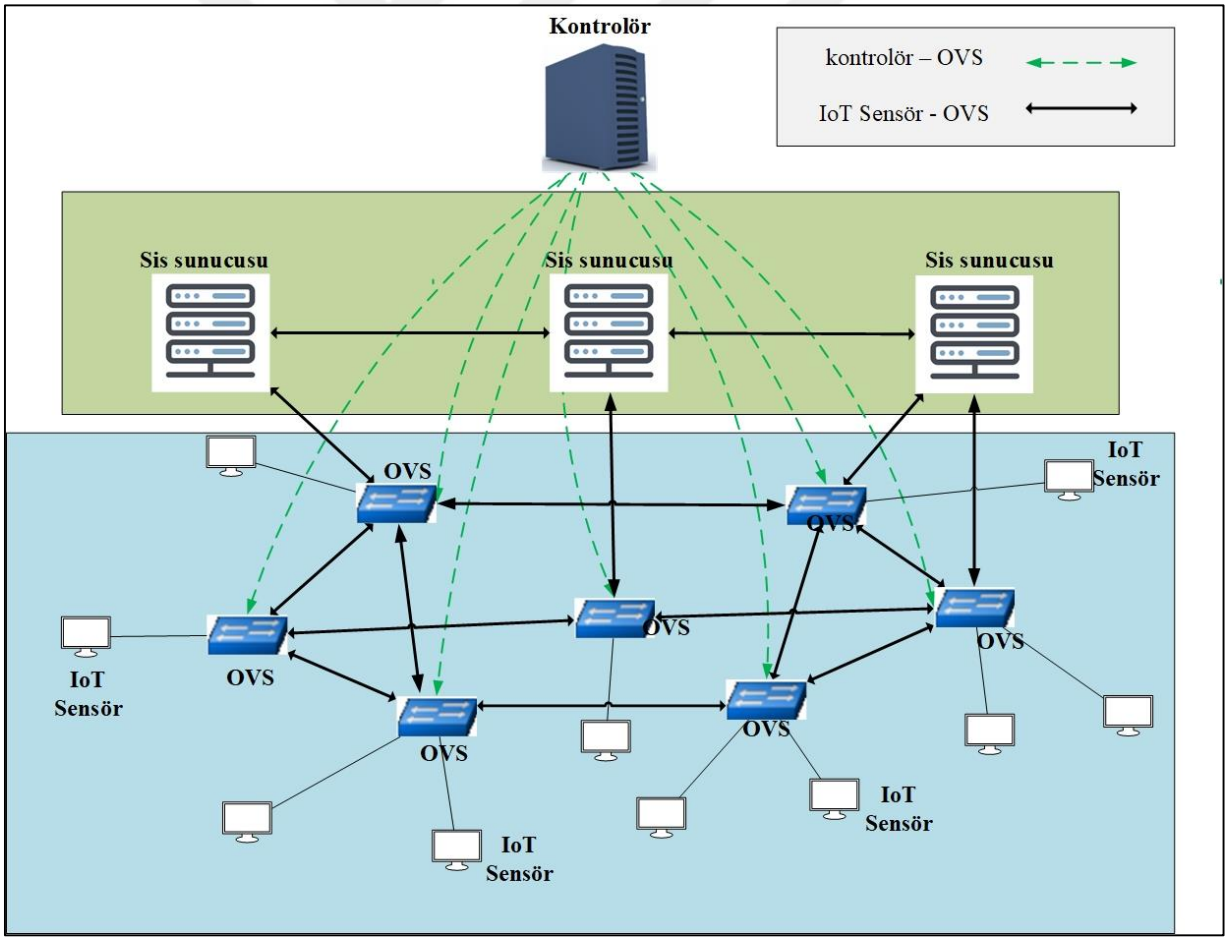
Şekil 5.1. SDN destekli IoT ağlarında ilişkili sunucu konumu mimarisi

Şekil 5.1'de, sensör tür 1 ve sensör tür 2 verileri sis sunucusu-1'de işlenmektedir. Ağda bu verileri üreten tüm sensörler verileri doğrudan belirlenmiş sis sunucu-1'e yönlendirmektedir. Sis sunucu-1'den üretilen yeni veri ve sensör tür3 verileri sis sunucu-4'te işlenmektedir. Sis mimarisi ile kurulan ağda sis sunucusu-1 ve sis sunucu-4 ilişkili sunucu olmaktadır. Bu sebeple, ağda konum belirleme gerçekleştirilirken sunucuların bir birleri ile olan konumu da öenm arz etmektedir.

Ağ altyapılarında sis, kenar bulutları sağlayan sunucuların konumu ağ performansını etkiler. Literatürde SDN destekli ağlarda farklı sunucular üzerinde farklı hizmet birimleri oluşturan

ve bu birimler arasında ilişki kuran çalışmalar yetersizdir. Çalışmamız sensör verilerini belirtilen hizmeti sağlayan sis sunucusuna yönlendirmektedir. Bu sunucularda gerçekleştirilen veri işleme sonrasında elde edilen yeni veriler başka bir sunucuya iletilir. Tüm bu yönlendirmeler kontrolör yazılımı ile gerçekleştirilir. Ortalama gecikme süreleri tam sayılı doğrusal programlama ile analiz edilmiş ve enuygun sunucu konumları belirlenmiştir.

Önerilen yüksek seviyeli sistem mimarisi Şekil 5.2'de gösterilmiştir. Her sunucuya belirli bir veri kümesini işleme görevi verilmiştir. Sensörlerden gelen veriler alt kümelere ayrılarak bu sunuculara iletilir. Sunucular üzerinde yapılan analizlerden elde edilen yeni veri türleri de veri seti içerisinde eklenmektedir. SDN ile yönetilen bu yapı akan veri işleme gerçekleşirken de kendini güncelleyebilir. Ayrıca ağ altyapısında daha az ve yönetilebilir veri iletimi gerçekleştirilerek performansın artırılması hedeflenmektedir.



Şekil 5.2. Önerilen SDN destekli sis ağ modeli mimarisi

Önerilen model aşağıdaki kısıtlara çözümler üretmektedir.

1. Ağ ilk oluşturulduğunda sunucuda işlenmek üzere herhangi bir veri veya veri seti belirlenmemiştir. Bu sebeple hangi verilerin hangi sunucuya iletileceği belli değildir.
2. Minimum gecikme, sensör ve sunuculardan elde edilen verilerin hedef sunucuya iletilmesine kadar geçen sürenin optimize edilmesidir.
3. Kısıt 1'de açıklandığı gibi ilişkili sunucuların konumunun belirlenmesi, belirlenen veri kümeleri bazı sunucularda elde edilen analiz sonucuna bağlıdır. Bu nedenle sunucuların birbirleri ile olan ilişkisi konum belirlenmede diğer bir kısıttır.

Belirlenen kısıtlar uygulanarak gecikmeyi en aza indirmek için sistemdeki aday sunucu konumları arasından en uygun sunucunun seçilmesi amaçlanmaktadır. Tüm kısıtlar ile birlikte en ideal çözüm için doğrusal programlama kullanılmıştır.

5.3. Önerilen Model

Bu bölümde, kullanılan tam sayılı doğrusal programlama ve geliştirilen sezgisel algoritma açıklanmıştır. Önerilen model için kurulan sanal ağ mimarisi simüle edilerek sensör - sunucu ve sunucu - sunucu arasında veri transferi gerçekleştirildiğinde yaşanan gecikmeler hesaplanmıştır. Elde edilen sonuçlara göre sunucu konumları belirlenmesi için literatürde yaygın kullanılan p-medyan modeli ve yeni bir matematiksel model kullanılmıştır.

P-medyan problemi, konum belirlemede yaygın olarak kullanılan yöntemlerden biridir. Bu model, tüm talebe hizmet vermenin toplam ağırlıklı mesafesinin en aza indirilmesi için bir ağ üzerindeki p tesislerinin konumunu belirlemektedir. Genel olarak, her bir talep düğümüne en yakın tesis tarafından hizmet verildiği ve bu tür bir talebe hizmet etmenin ağırlıklı mesafesinin, en yakın tesise olan mesafenin o düğümdeki talep miktarı ile çarpımı olduğu varsayılır.

Hakimi [122], belirli bir konum belirleme sorununa yönelik en az bir optimal çözümün tamamen düğüm bölgelerinden oluştuğunu kanıtladı. P-medyan problemine optimal bir çözüm arayışı, sonlu düğüm kümesiyle sınırlandırılabilir. Teitz ve Bart'ın [123], p-medyan problemi için yeni bir yöntem önermektedir. Bu yöntem, başlangıç çözümü olarak rastgele seçilmiş bir p-düğüm konfigürasyonu ile başlar. Daha sonra, yapılandırmanın bir parçası olmayan aday düğümlerden biri, mevcut tesis konumlarından herhangi birinin yerini aldığında çözümün iyileştirilip iyileştirilemeyeceğini kontrol edilmektedir. Daha başarılı

sonuç elde edilirse düğüm değiştirilerek en optimum çözüm aranmaktadır. Yer değiştirmelerde daha başarılı bir sonuç üretilmediği sürece bu döngü devam eder.

İlk defa ReVelle ve Swain [124], p-medyan problemi için bir matematiksel model önermişlerdir. Modelde küçük değişiklikler yapılsa da, kurulan tüm modeller büyük oranda orijinali ile aynıdır. Bu çerçevede ReVelle ve Swain tarafından kurulan matematiksel modeli temel alan, Rolland ve arkadaşlarının önermiş olduğu p-medyan modeli aşağıdaki gibidir [125].

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} d_{ij} z_{ij} \tag{5.1}$$

Kısıtlar:

$$\sum_{j=1}^{n} z_{ij} \qquad \forall i \tag{5.2}$$

$$z_{ij} \leq y_j \qquad \forall i, \forall j \tag{5.3}$$

$$\sum_{j=1}^{n} y_j = p \tag{5.4}$$

$$z_{ij} \in \{0,1\} \qquad \forall i, \forall j \tag{5.5}$$

$$y_j \in \{0,1\} \qquad \forall j \tag{5.6}$$

Karar değişkenleri:

$$z_{ij} = \begin{cases} 1, & eğer\ i\ verisi\ j\ sunucusuna\ atanırsa \\ 0, & diğer\ durum \end{cases}$$

$$y_j = \begin{cases} 1, & eğer\ j\ düğümünde\ bir\ sunucu\ seçilirse \\ 0, & diğer\ durum \end{cases}$$

n, toplam veri gönderecek nokta sayısı, d_{ij}, i noktası ile j noktası arasındaki en kısa mesafe, p yerleştirilecek olan sunucu (medyan) sayısıdır. Eş. 5.1, amaç fonksiyonudur ve belirlenecek sunucu ile veri gönderecek sensörler veya sunucular arasında oluşacak olan maliyeti minimize etmektedir. Eş. 5.2, her veri üreten birimin yalnız bir sunucudan hizmet

almasını sağlanmaktadır. Eş. 5.3, sunucu seçilmeyen herhangi bir düğüme talep noktası atanmasını engellemektedir. Eş. 5.4 ise seçilecek olan sunucu sayısını p adet ile sınırlı kalmasını sağlamaktadır.

SDN destekli Sis mimarilerinde p-medyan ile sunucu yer seçimi problemi literatürde de gösterildiği gibi NP-zor bir problemdir. Bu sebeple daha hızlı sonuçlar üreten sezgisel, genetik vb. gibi yaklaşımlar geliştirilmiştir. Literatürde p-medyan problemine çözüm olarak, genetik algoritmalar [126-129], tabu arama [130,131], sezgisel [132,133], açgözlü sezgisel [134-136] vb. yöntemler önerilmiştir.

Çizelge 5.2. Önerilen tam sayılı doğrusal programlama modelinde kullanılan değişkenler

i	Sensör kümesi $i \in I$
j	Sensör küme elemanı $j \in J$
k, m	Aday sunucu düğümleri $k, m \in K$
d_{ijk}	i sensor kümesi elamanı j'nin, k sunucusuna olan maliyet
C_{km}	k sunucusunun seçilmiş m sunucusuna olan maliyet
p	Seçilmiş sunucu sayısı
X_{ik}	i sensor kümesinin k sunucusuna atanıp atanmadığı
y_{km}	k sunucusunun m sunucusuna atanıp atanmadığı
z_i	i sensor setinin atandığı sunucu $z \in Integer$
v_k	k sunucusunun seçilip seçilmediği

Amaç fonksiyonu:

$$\min \sum_{k \in K} \sum_{j \in J} \sum_{i \in I} x_{ij} * d_{ijk} + \sum_{m \in K} \sum_{k \in K} y_{km} + c_{km} \tag{5.7}$$

Kısıtlar:

$$\sum_{k \in K} x_{ik} = 1 \qquad \forall i \tag{5.8}$$

$$x_{ik} \leq v_k \qquad i \in I, k \in K \tag{5.9}$$

$$\sum_{k \in K} v_k = p \tag{5.10}$$

$$\sum_{i \in I} x_{ik} = 1 \qquad \forall k \tag{5.11}$$

$$z_i = \sum_{k \in K} x_{ik} * k \qquad \forall i, z_i \in Integer \tag{5.12}$$

$$(1 - y_k) \geq |k - z_i| + |m - z_{i+1}| \tag{5.13}$$

$$x_{ik}, y_{km}, v_k = 0,1 \tag{5.14}$$

i, her sunucuya iletilecek sensörler alt kümesidir, j ise bu kümelerin her bir elemanını temsil etmektedir. Eş. 5.7, belirlenecek sunucu konumu ile sensör ve sunucular arasındaki iletişim maliyetini en aza indirir. Eş. 5.8, her sensör setinin bir sunucuya atanmasını sağlar. Eş. 5.9 sensörlerin seçili olmayan sunucuya atanmasını engeller. Tutarlılığı sağlamak için kullanılır. Eş. 5.10, p adet sunucunun seçilmesine izin verir. Eş. 5.11 aynı kümedeki farklı sensor türlerinin aynı sunucuya atanmasını sağlar. Her sunucuya yalnızca bir sensör seti atanır. Eş. 5.12, sensör kümesinin hangi sunucuya atandığını gösteren tamsayı değişkenini belirtir. Eş. 5.13, seçili olmayan sunucular arasındaki veri akışını engeller.

Önerdiğimiz ve matematiksel olarak tanımladığımız tam sayılı doğrusal programlama modeli GAMS uygulaması ile gerçekleştirilmiştir. İlgili çalışmalarda açıklandığı gibi sunucu konum problemi NP-zordur. Bu nedenle açgözlü bir sezgisel yöntem geliştirilmiştir. Açgözlü algoritma problemimize uyarlanarak bir çözüm üretildi. Açgözlü algoritma, p-medyan tesis yerleşimi problemlerini çözmek için sıklıkla kullanıldığı için seçilmiştir [137-140].

Şekil 5.3'te gösterildiği gibi, sensör ve sunucu verilerinin iletileceği sunucular önerilen açgözlü sezgisel algoritma ile belirlenmiştir. İlk olarak, her bir sunucuya iletilecek olan veri

alt kümesi M, sensörler I ve gecikme matrisi d(,) belirlenir. Ardından, öğelerin her bir alt kümesi için yeni bir matris yeni_d() elde edilir. Buradaki amaç, sadece yeni bir sunucu seçecek olan sensörlerin gecikme süresini tutan matrisi elde etmektir. Bu şekilde, ilişkili sunucu konumu tespiti aşamalı olarak gerçekleştirilir. Bu işlemler, sırasıyla her bir m alt kümesi için gerçekleştirilir. Son olarak 12. satırdaki açgözlü yaklaşımıyla sunucu konumu belirlenir. 13. satırda belirlenen bu sunucu konumu, sensör setine dahil edilir. Çünkü daha önce de açıklandığı gibi bu sunucu, modelin devamında sensörler gibi veri üretiyor ve bu veri seçilecek yeni sunucunun girdi verisi olabilir.

Algoritma: Önerilen aç gözlü sezgisel modelin algoritması

1: **Girdi:** Sensörler I, Sensör alt kümeleri i, Gecikme zamanı matrisi d(,)
2: **Çıktı:** Seçilen sunucular s
3: **for each** i ∈ I
4: j ← 0
5: **for** i=0 to i < |I|
6: **if** (i ∈ m) **ise**
7: for s=0 to s < |S|
8: yeni_d(j,s) = d(i,s)
9: **end for**
10: j ← j+1
11: **end for**
12: $\min_{i \in I} \sum_{s \in S} yeni\_d(i,s)$
13: Güncelle I ← I ∪ {s}
14: **end for**
15: **Return** s ∈ S

Şekil 5.3. Açgözlü sezgisel modelin sözde kodu

5.4. Deneysel Çalışmalar

İlişkili sunucu konumu problemi için önerilen modeller için gerçekleştirilen deney ortamı ve elde edilen analiz sonuçları bu alt bölümde açıklanmıştır.

5.4.1. Deney ortamı

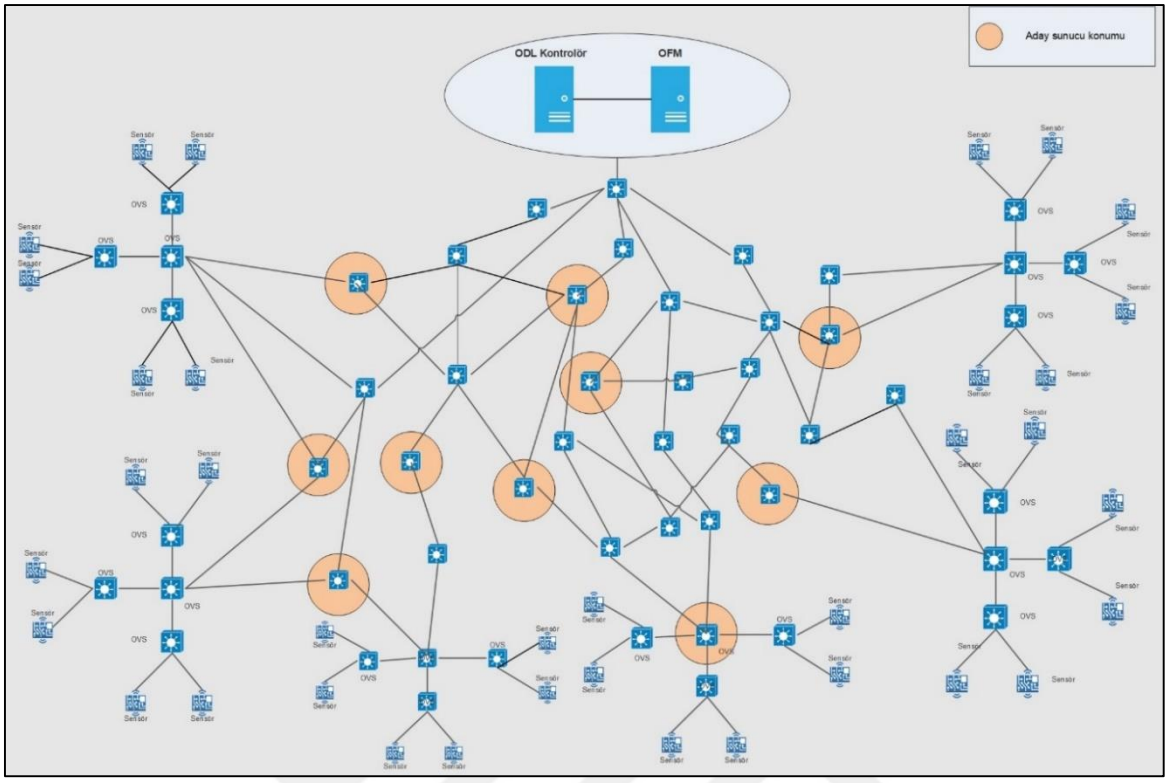
Çalışmanın bu bölümünde daha önce tanımlanan probleme çözüm önerisi olarak sunulan iki model tanıtılmıştır. Bu modeller, tam sayılı doğrusal programlama ve sezgisel yaklaşım ile

geliştirilmiştir. Önerilen modeller, SDN destekli ağ mimarisi ile simüle edilmiştir. Tezin bu alt bölümünde, Önerilen modellerin simülasyonu için kurulmuş platform açıklanmaktadır.

Önerilen modeller GNS3’te simüle edilmiştir. GNS3 2.2.5 versiyonu kullanılmıştır. Wireshark ağ izleme, paket yakalama programı ile veriler incelenmiştir. GNS3'te önerilen modelleri test edebilmek için Docker container’larda OVS anahtarları kuruldu. Linux işletim sistemine sahip sanal bir makinede OFM kuruldu. Sensörlerin bağlı olduğu linux işletim sistemine sahip bilgisayarlar, docker container’lara kuruldu. Veriler iperf3 yardımcı program yazılımı ile ağa iletilmektedir. Bu çalışmada, ODL kontrolör yazılımı kullanılmaktadır. Çünkü OFM, OVS ve GNS3 platformları ile uyumludur. ODL, windows işletim sistemine sahip sanal bir makineye kuruldu. Bu sanal makine GNS3 ile entegre edilerek ağ yönetimi ODL ile gerçekleştirilmiştir

Simülasyon ortamı Şekil 5.4’te gösterilmiştir. Test ortamında 10 aday sunucusu konumu belirlenmiştir, bu sunucuların 4 tanesi farklı görevlerin atanması için seçilecektir. 6 farklı veri tipi üreten toplam 36 sensör sisteme veri üretmektedir. Sensörler iperf yardımcı yazılımı ile veri üreterek ağa iletmektedir. Simülasyon GNS3 ve ağ alt elemanlarının bulunduğu bir sunucu, bir OFM sunucusu ve bir ODL kontrolör sunucusu olmak üzere 3 sanal sunucudan oluşmaktadır. GNS3 kurulu sanal sunucuda, 70 OVS ve 36 sensör kurulmuştur.

Deneysel çalışma, 64 bit linux işletim sistemine, 3.4 GHz işlemciye ve 32 GB RAM'e sahip bir sunucuda gerçekleştirilmiştir. Sensörlerin her biri saatte 90 paket veri üretmektedir ve ortalama paket boyutu 75 byte ve bant genişliği 10 Mbit/s olarak belirlenmiştir.



Şekil 5.4. Önerilen modelin test ortamı

5.4.2. Analiz sonuçları

SDN destekli sanal anahtarlar ve sunucular ile oluşturulan topoloji GNS3 ile simüle edilerek gecikme süreleri hesaplanmıştır. Sensör verilerinin ve sunucular üzerinde yapılan analizlerden elde edilen verilerin, sis sunucularına iletilmesindeki gecikmeler hesaplanarak tam sayılı doğrusal model ve sezgisel algoritma ile optimum sunucu konumu belirlenmektedir. Önerilen modeller MATLAB ile gerçekleştirilmiştir.

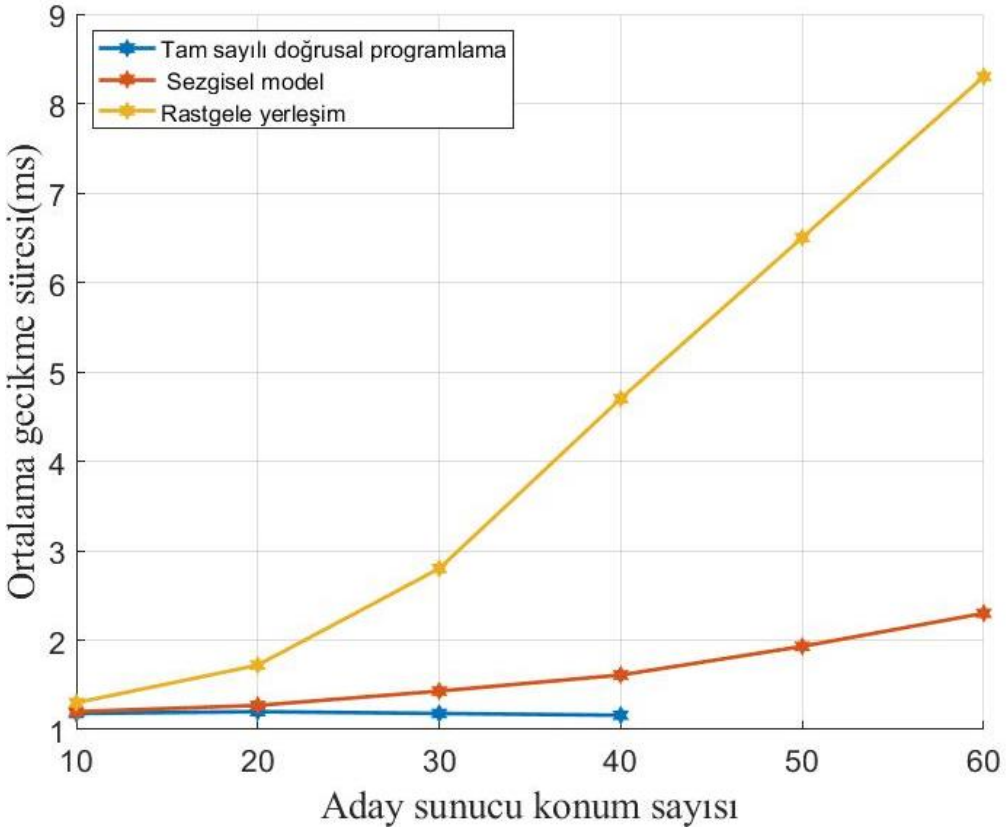
Çizelge 5.3'te önerilen modellerin çalışma süreleri verilmiştir. 70 OVS ile oluşturulan topolojide belirlenmiş 20 aday sunucu konumu içinden üç farklı konum belirlendiğinde tam sayılı doğrusal programlama 86.61 saniyede çözüm üretirken önerilen sezgisel model 3.7 saniyede çözüm sunmuştur. Aday sunucu konum sayısı 40 olduğu zaman sezgisel model 13.2 saniyede çözüm üretmektedir.

Çizelge 5.3. Önerilen modellerin ortalama çalışma süreleri

Aday sunucu sayısı	Tam sayılı doğrusal programlama	Açgözlü sezgisel model
20	86.61s	3.7s
40	30780s	13.2s

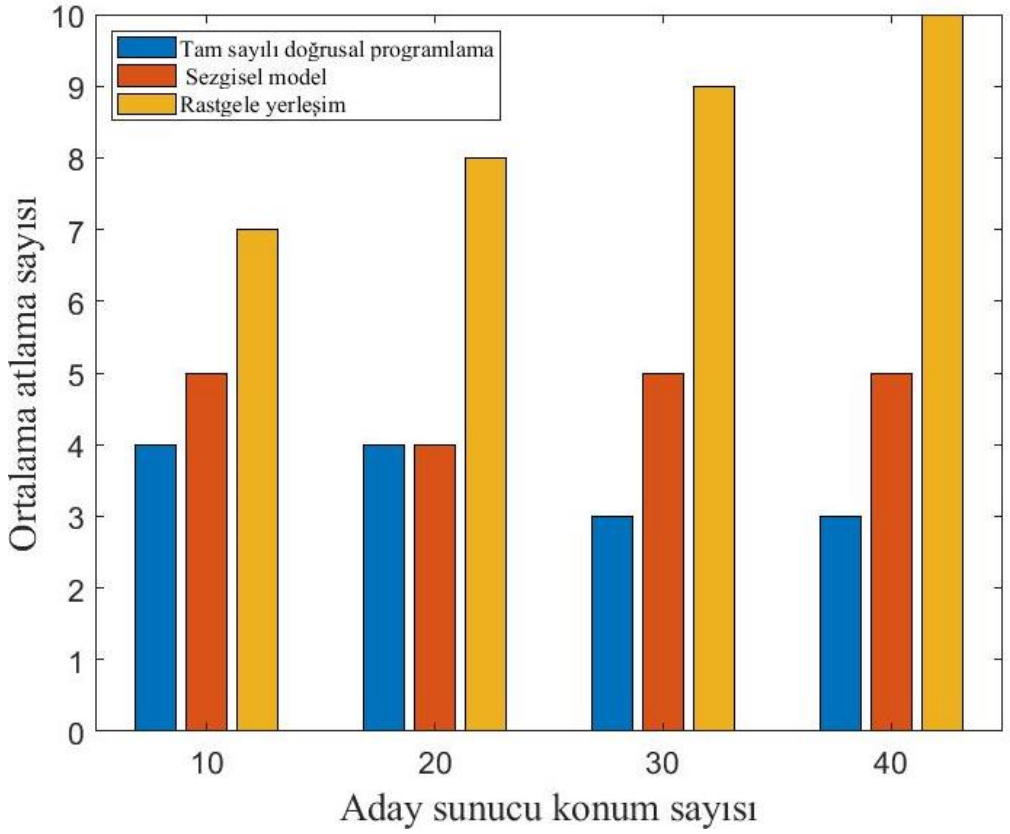
Literatürde açıklandığı gibi, p-medyan tabanlı sunucu yerleştirme problemi NP-zor bir problemdir. Aday sunucu sayısı S, belirlenen sunucu sayısı m olsun. Tam sayılı doğrusal model (S*(S-1)*(S-2)…(S-m+1)) durumunu değerlendirir. Sezgisel algoritma (S+(S-1)+(S-2)+...+(S-m)) durumunu değerlendirir. Örneğin, geliştirdiğimiz tam sayılı doğrusal model 36 sensör noktası, 8 ilişkili alt küme verisi ve 70 aday sunucu konumu ile test edilirse. Matematiksel model yaklaşık 380 milyar durumu değerlendirir ve sezgisel model 532 durumu değerlendirir.

Şekil 5.5'de, ortalama gecikme, aday sunucu sayısına göre hesaplanmıştır. Her bir durum için seçilecek sunucu sayısı üç olarak belirlenmiştir. Ortalama gecikme süresi, geliştirilen tam sayılı doğrusal programlama, önerilen sezgisel model ve rastgele yerleştirme modelleri karşılaştırılmıştır. Aday sunucu sayısı artmasına rağmen tam sayılı doğrusal modelin ortalama gecikme süresi azalmaktadır. Ağda konumlandırma seçeneğinin artması daha iyi bir konumun seçilmesini sağlamıştır. Ancak, sezgisel modelde ortalama gecikme süresi artmıştır. Açgözlü sezgisel model, her adımda en uygun sunucu konumunu seçer. Bu nedenle aday sunucu sayısı arttıkça en iyi durumdan sapmaktadır. Rastgele yerleşim modelinde konumlar belirlendikten sonra gerçekleşen ortalama gecikme süresi önerilen modellerden çok daha yüksek çıkmıştır. Buda önerdiğimiz modellerin verimini ortaya koymaktadır.



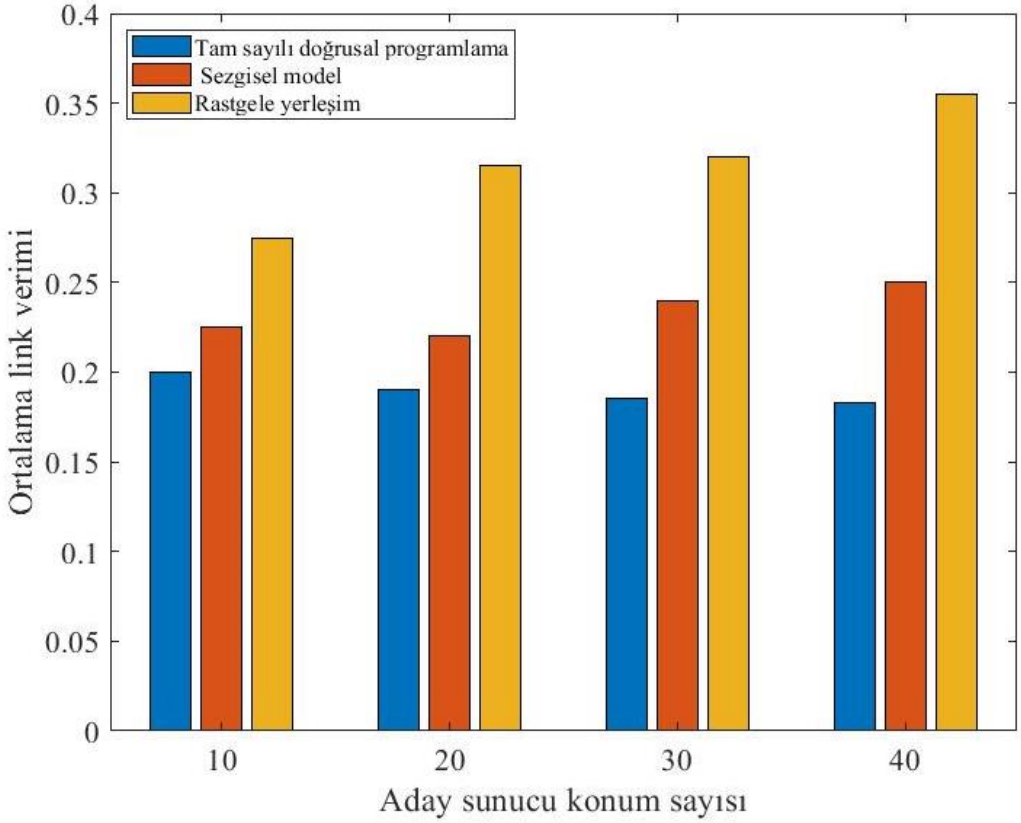
Şekil 5.5. Ortalama gecikme süresi ile aday sunucu konumlarının sayısı arasındaki ilişki

Şekil 5.6'da aday sunucu konum sayısına göre belirlenen sunuculara erişimde gerçekleşen atlama sayısı gösterilmiştir. Rastgele yerleştirme modelinde ortalama atlama sayısı, aday sunucu konum sayısı ile doğru orantılı bir şekilde artmıştır. Önerilen sezgisel modelde atlama sayısı aday sunucu konumu sayısından daha az etkilendiği görülmektedir. Tam sayılı doğrusal modelde ise aday sunucu konum sayısı ile ters orantılı bir ilişki gözlemlenmiştir. Önerilen modellerin rastgele yerleştirme yöntemine göre daha az atlama sayısına sahip olduğu gözlemlenmiştir.



Şekil 5.6. Ortalama atlama sayısı ile aday sunucu konumlarının sayısı arasındaki ilişki

Şekil 5.7'de önerilen modeller ve rastgele yerleştirme modeli için link kullanım oranı incelenmiştir. Deney ortamında 70 OVS kullanıldığından link kullanım oranı literatürdeki verim çalışmalarına göre görece düşük çıkmıştır. Önerilen modellerin link kullanım oranı rastgele yerleştirme yönteminden daha düşük olduğu gözlemlenmiştir. Atlama sayısının daha düşük olması, verinin sensörlerden belirlenmiş sunucuya iletiminde daha az link kullandığı göstermektedir. Bu sebeple ortalama link verimi daha düşük çıktığı gözlemlenmiştir.



Şekil 5.7. Ortalama link verimi (Utilization) ile aday sunucu konumlarının sayısı arasındaki ilişki

5.5. Bölüm Değerlendirmesi

Tezin bu bölümünde, SDN destekli IoT ağlarında, ilişkili sis sunucularının konumunu belirlemek için tam sayılı doğrusal ve sezgisel modeller önerilmiştir. IoT sensörleri, verileri sis sunucusuna yönlendirilir. Problemimizde her bir sis sunucusuna farklı görevler atanmıştır. Her sunucuya farklı veri türleri iletilir. Atanan işlemden sonra üretilen yeni veriler başka bir sis sunucusunun giriş verileri olabilir.

Bu nedenle, sis sunucuları birbirleri ile ilişkilidir. İlgili sunucuların konumunu belirlemek için p-medyan ile geliştirilen tam sayılı doğrusal bir model sunulmuştur. Bu model, en uygun sis sunucusu konumlarını belirlemektedir. Ancak literatürde ve analizlerde görüldüğü gibi sunucu yerleştirme problemi NP-zordur. Bu nedenle açgözlü sezgisel yaklaşımla bir model geliştirilmiştir. Sezgisel model, sunucu konumlarını daha kısa sürede bulmaktadır. Ancak, büyük ağlarda sensörler ve sunucular arasındaki ortalama gecikme süresi artmaktadır.

Önerilen modellerin geçerliliğini desteklemek için ortalama gecikme süresi, atlama sayısı ve link verimi metrikleri incelenmiştir. Ortalama gecikme süresi en düşük tam sayılı doğrusal programa ile önerilen modelde gerçekleşirken rastgele yerleşimde bu süre en yüksek seviyelere çıkmıştır. Atlama sayısı ve link veriminde de önerilen modeller daha başarılı sonuçlar üretmiştir. Bu veriler, önerdiğimiz modellerin verimini ortaya koymaktadır.

SDN destekli, IoT ve sis ağı topolojilerinde sunucuların konumu ağ performansını etkiler. Ağı SDN ile yönetmek, belirlenen sunuculara göre yeniden yönlendirme yapmayı kolaylaştırmıştır. Sunucular ilişkili olduğu için ağda iletişim sürekliliği zorlaşır. SDN yazılımı ile uzaktan, dinamik ve hızlı bir şekilde iletişim sağlanmaktadır.

SDN destekli kenar veri işleme için önerdiğimiz modellerde, sistem kullanıcıları veri setine, veri türüne, analiz yöntemine, IoT sensörlerinin konumuna, sunucu kapasitesine ve ağ topolojisine göre kontrolör yazılımını güncelleyerek kullanabilmektedir. Hava kalite ölçüm verileri ile test ettiğimiz model farklı ortamlar içine kullanılabilmektedir. Örneğin; sürücü davranış analizinde birçok farklı veri kullanılarak ölçüm yapılmaktadır [141]. Hava durumu sürücülerin davranışını etkileyen faktörlerden biridir. Sürücü demografik değişkenleri, gelir seviyesi, çalışma şartları davranışı etkileyen diğer faktörlerdir. Hava durumu veri seti, sıcaklık, yağış türü, yağış oranı gibi verilerden oluşmaktadır. Bu veriler ayrı bir sunucu toplanıp analiz edildikten sonra kategorize edilip sürücüye ait diğer veriler ile gerekli ilişkisi kurulmak üzere bir diğer sunucuya iletilebilir. Bu yapı daha önce deneysel ortamda test edildiği ve açıklandığı gibi ağ performansını artıracaktır.

6. SONUÇLAR VE ÖNERİLER

Bu alt başlıkta tez kapsamında yapılan çalışmalar, önerilen modeller, literatüre katkılar, karşılaşılan zorluklar ve gelecek çalışma önerileri açıklanmıştır.

6.1. Sonuçlar ve Değerlendirme

İnternete bağlı cihaz sayısı hızla artmaktadır. Bu artış yüksek hacimli veri oluşmasına sebep olmaktadır. Yüksek hacimli verilerin ağda iletişimin sağlanması, hizmet kalitesinin artırılması ve ağ maliyetlerinin optimize edilmesi güncel bir sorundur. Tez kapsamında gerçekleştirdiğimiz çalışmada, yüksek hacimli IoT sensör verilerin etkin iletişimini sağlamak için iki yeni model önerilmiştir. Hizmet kalitesini artırmak için kontrolör ve sunucu yerleştirme problemlerini çözen doğrusal ve sezgisel modeller önerilmiştir. Yüksek hacimli ve çok sayıda IoT cihazından oluşan sistemleri yönetmek için SDN teknolojisi kullanılmıştır.

Sis sunucularına veri işleme hizmeti sağlayan uygulamalar yerleştirildiği varsayılmıştır. Her bir sis sunucusuna farklı veri türleri iletilmektedir. İletilen veri türleri analiz edilerek yeni bir veri üretmektedir ve bu veri bir başka sunucuda işlenmek üzere ağa iletilmektedir. Dağıtık veri işleme modelinin ağ cihazları ile gerçekleştirilmesine benzer bir uygulama ve ağ topolojisi kurulmuştur. Bu fikir ile sensörlerden elde edilen verilerin tamamı değil sadece ihtiyaç duyulanlar ağa iletilmektedir.

Tez kapsamında, IoT sensörlerinden elde edilen veriler, SDN kontrolör yazılımı ile içeriğine ve önceliğine göre ağa aktarılmaktadır. CBF-SDN modelinde veriler sensörlerden ağa iletilirken içeriğine göre aktarılmakta, PCBF-SDN modelinde hem içeriğine hem önceliğine göre aktarılmaktadır. Önerdiğimiz bu modeller ile sensörlerde oluşan verilerin, ağ cihazları tarafından tanınmasını sağlanmıştır. Kontrolör yazılımı ile yönlendirme sağladığımızdan hizmet sağlayan sunucuların gereksinimine uygun verilerin ağa iletilmesini gerçekleştirerek ağa iletilen toplam veri hacmi iyileştirilmiştir. Aynı zamanda verilerin önceliğine göre kontrolör desteği ile farklı iletişim teknikleri uygulayarak acil durum senaryolarına çözümler üretilmiştir.

Yüksek hacimli IoT ağlarda iletişimi SDN kontrolör yazılımı ile gerçekleştirdiğimizden, kontrolör ve sunucu konumu problemleri de incelenmiştir. CPP ve SPP, SDN destekli ağlarda, hizmet kalitesini etkileyen önemli parametrelerden biridir. Tez kapsamında, CPP için doğrusal programlama ile kontrolör arızalanma ve kapasitesi kısıtları ile ağın sürekliliğini ve güvenilirliğini sağlayan bir model önerilmiştir. Farklı görev atanmış ve ilişkili sis sunucularının konumunu belirlemek için tam sayılı doğrusal ve sezgisel iki model önerilmiştir. İlişkili sunucu konumu problemi literatürde ilk defa incelenmiştir.

Önerdiğimiz tüm modelleri gerçek ağ benzetimi ile simüle ettik. CBF-SDN ve PCBF-SDN modellerini gerçek veri setleri ile analiz edilmiştir. CPP ve SPP problemlerine çözüm önerilerimizi aynı veri seti ve topolojisi ile geliştirip analiz edilmiştir.

Önerilen modeller ve avantajları

Tez kapsamında yönlendirme problemine 2, kontrolör yerleştirme problemine 1 ve sunucu yerleştirme problemine 2 olmak üzere mimari eniyilemesi için beş yeni model önerilmiştir. Önerilen modeller, yeni veri seti ile GNS3 ortamında simüle edilmiştir. Simülasyon sonuçları literatür ışığında değerlendirilmiş ve katkıları açıklanmıştır. Önerilen yeni modellerin avantajları aşağıda tek tek açıklanmıştır.

CBF-SDN, Sensörlerden elde edilen veriler içeriğine göre etiketlenerek ağa iletilen modeldir. CBF-SDN modeli, sensörlerden elde edilen hatalı verilerin ağa iletilmesini engellemektedir. Kontrolör desteği ile ağ cihazlarının veri içeriğini bilmesini sağlamakta ve dinamik yönlendirme adımlarının gerçekleştirilmesine olanak sağlamaktadır. Sağladığı bir diğer avantaj gereksiz ve sensörler tarafından hatalı üretilen verilerin ağa iletilmesini engellemektedir.

PCBF-SDN, öncelikli verileri iletebilmesi için CBF-SDN modelinin geliştirilmiş versiyonudur. Sensörlerden elde edilen verilerin belirlenmiş eşik değerinden yüksek olması veya veri türünün önemli olduğu durumlarda daha etkin iletimini sağlamaktadır. Bu verilerin ağda daha etkin iletimi için ayrı bir kuyruk oluşturulmuş ve bu kuyruğa daha fazla ağ kaynağı tahsis edilmiştir. Öncelikli verilerin iletimi ortalama gecikme süresi azaltılarak gerçekleştirilmiştir. Model kontrolör yazılımı ile gerçekleştirildiğin öncelikli veriler için

farklı akış kuralları uygulanabilir. SDN'in ağa sağladığı dinamiklikten kaynaklı önceliği belirleyen metrikler değiştirilerek güncellenebilir.

CF-CPP, kapasite kısıtı ve kontrolörün bozulma ihtimali dikkate alınarak kontrolör yerleştirme problemine çözüm olarak sunulmuştur. Daha önce önerdiğimiz modellerin etkinliğini artırmak için sunulmuştur. SDN destekli bir ağ topolojisi kurulurken olası kontrolör hatalarını engellemek, kontrolör iletişimden kaynaklı ortalama gecikme sürelerini azaltmak için k-means tabanlı doğrusal programlama modeli geliştirilmiştir. Önerilen model, ağda sürekliliği ve iletişim güvenliğini sağlamaktadır.

SDN destekli IoT ağ modeli gerçek veri seti ile simüle edilmiştir. Veriler İstanbul ilinde 29 farklı noktada bulunan sis sunucularından elde edilmiş. Sis sunucu konumu ağ performansını ve önerdiğimiz modellerin etkinliğini doğrudan etkilemektedir. Literatürde ilk defa veri analizi hizmeti sağlayan ilişkili sis sunucuları konumu belirleyen model önerilmiştir. İlişkili sis sunucu konumu için p-medyan tabanlı tam sayılı doğrusal programlama modeli önerilmiştir. Büyük hacimli ağlar için NP-zor çıktığından açgözlü sezgisel model geliştirilmiştir. Önerilen modeller ile ortalama gecikme süresi azaltılmış ve tez kapsamında önerilen iletişim modellerinin etkinliği artırılmıştır.

Kısaca özetlemek gerekirse, Sensörlerde oluşan verilerin analizinde, veri kaybını engellemek, önceliğine göre iletişimi sağlamak, ağ cihazları tarafından içeriği bilinen verilerin kontrolör yazılımı ile içeriğine uygun iletimini sağlamak için iki yeni model önerilmiştir. Önerdiğimiz modellerin ağ performansını artırmak için ilişkili sunucu ve kontrolör yerleştirme algoritmaları geliştirip hizmet kalitesi artırılmıştır.

<u>Tezin literatüre katkıları</u>

- SDN destekli ağlarda, veriler sensörlerden elde edildiği gibi etiketlendiğinden harici isim tanımlama sunucularına ihtiyaç duymamaktadır. Literatürdeki diğer önerilere göre daha düşük donanım ve yönetim maliyeti sağlamaktadır.
- İçerik yönetimi, ağa iletilen paketin ToS bit değeri değiştirilerek tanımlandığından ağ cihazları veri içeriği hakkında bilgi sahibi olmaktadır. Bu özellik ile kontrolör, veri işlemenin ve yönlendirilmesinin kolaylaştırılmasını sağlamaktadır. Aynı zamanda ağa esneklik ve dinamiklik sağlamaktadır.
- Öncelikli verilerin iletiminde önerilen model ile acil durum senaryolarına çözüm üretilmektedir.

- Kontrolör arızalanma ihtimali düşünülerek doğrusal programlama ile yeni bir model önerilmiştir.
- Literatürde ilk defa veri analizi hizmeti sağlayan ilişkili sis sunucuları konumu belirleyen modeller önerilmiştir.
- Sis sunucu yerleştirme problemine tam sayılı doğrusal programlama ve sezgisel yaklaşımlı iki yeni model önerilmiştir.

6.2. Karşılaşılan Zorluklar

Bu tez kapsamında karşılaşılan zorluklar;

- SDN ile ilgili akademik çalışmaların olmasına rağmen teknik detayları açıklayan, uygulanmasında yol gösteren kaynakların az olması.
- Birden fazla teknoloji veya uygulama aracının kullanılması ve bunların haberleştirilmesinin sağlanması
- Kullanılan araçlardan birinin güncellenmesi sonucu kurulan deneysel ortamın istikrarlı çalışmaması
- Tek bir donanım üzerinde sanallaştırma ve docker ile çok fazla sistemin kurulması uygulanması
- Gerçek ve büyük veri seti ile çalışmasının verdiği test süresi uzunluğu
- Docker konteynırlara kurulan OVS'lerin manuel konfigürasyonu
- Donanımsal ağ adaptörünün sanallaştırma kapasitesinin yetersizliği
- ODL kontrolörünün cisco araçlarını (OVS, host vb.) yönetmedeki uyum sorunları

olarak açıklanabilir.

6.3. Gelecek Çalışma Önerileri

Öncelikli verilerin iletilmesinde önerilen PCBF-SDN modelinde kenar sunucularında gerçekleştirilen analiz sonuçları elde edilen eşik değerlerinin kontrolör yazılıma doğrudan iletilmesi sağlanarak kendi kendini geliştiren ve yöneten yapay zeka destekli bir kontrolör yazılımı geliştirilebilir. Özelikle akan veri analizinde zamana bağlı değişimler ile anlık karar güncellemeleri gerçekleştirilebilir.

İçerik tabanlı yönlendirmede hatalı ve eksik verilerin ağa iletilmesi engellenmiştir. Bu veriler farklı bir sunucuda toplanarak gerçekleştirilecek veri analizi ile tahmin edilerek ağ performansına ve veri analizine etkisi incelenebilir.

Kontrolör yerleştirme probleminde çoklu kontrolörlü sistemlerde hiyerarşik ve bağlantılı kontrolör mimarilerinin kullanılabilir ve farklı metrikler ile karşılaştırılabilir.

Veri analizinde kullanılmak için önerilen ilişkili sunucu konum modelleri farklı sezgisel veya doğrusal modeller ile geliştirilebilir. Önerilen model farklı senaryolar ve veri setleri ile incelenerek farklı alanlarda kullanılması önerilebilir. Büyük veri teknolojisi ile ilişkisi kurularak sağlayacağı avantajlar incelenebilir.

İlişkili sunucu konumda gerekli minimum sunucu sayısı ve yük dengeli sunucu iletişimi incelenebilir. Literatürde kopya sunucu konumu yerleştirme çözümleri ile birleştirilerek işlem yüküne göre sanallaştırılmış sunucular kullanılabilir. Önerilen model, işlemci yükü ve enerji kullanımı metrikleri ile analiz edilerek değerlendirilebilir.

KAYNAKLAR

1. Kreutz, D., Ramos, F.M., Verissimo, P., Rothenberg, C.E., Azodolmolky, S.and Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE,* 103(1), 14-76.

2. Kawabata, A., Chatterjee, B.C.and Oki, E. (2018). Participating-domain segmentation based server selection scheme in delay-sensitive distributed communication approach. IEEE Access 7 (2019): 20689-20697.

3. Ray, P.P. (2018). A survey on Internet of Things architectures. *Journal of King Saud University-Computer and Information Sciences,* 30(3), 291-319.

4. Okay, F.Y. and Ozdemir, S. (2018). Routing in fog-enabled IoT platforms: A survey and an SDN-based solution. *IEEE Internet of Things Journal,* 5(6), 4871-4889.

5. Pop, P., Raagaard, M.L., Gutierrez, M.and Steiner, W. (2018). Enabling fog computing for industrial automation through time-sensitive networking (TSN). *IEEE Communications Standards Magazine,* 2(2), 55-61.

6. He, T., Toosi, A.N.and Buyya, R. (2019). Performance evaluation of live virtual machine migration in SDN-enabled cloud data centers. *Journal of Parallel and Distributed Computing,* 131, 55-68.

7. Iorga, M., Feldman, L., Barton, R., Martin, M.J., Goren, N.S.and Mahmoudi, C. (2018). Fog computing conceptual model.

8. Ghalehtaki, R.A., Kianpisheh, S.and Glitho, R. (2019). A bee colony-based algorithm for micro-cache placement close to end users in fog-based content delivery networks. *16th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2019.*

9. Atlam, H.F., Walters, R.J.and Wills, G.B. (2018). Fog computing and the internet of things: A review. *big data and cognitive computing,* 2(2), 10.

10. Mahmud, R., Kotagiri, R.and Buyya, R. (2018). Fog computing: A taxonomy, survey and future directions. *Internet of everything*. Springer. 103-130.

11. Bellavista, P., Berrocal, J., Corradi, A., Das, S.K., Foschini, L.and Zanni, A. (2019). A survey on fog computing for the Internet of Things. *Pervasive and mobile computing,* 52, 71-99.

12. Jmal, R. and Fourati, L.C. (2017). An OpenFlow architecture for managing content-centric-network (OFAM-CCN) based on popularity caching strategy. *Computer Standards & Interfaces,* 51, 22-29.

13. İnağ, Y., Güzel, M., OKAY, F.Y., Demirci, M.and Özdemir, S. (2022). Priority enabled content based forwarding in fog computing via SDN. *Turkish Journal of Electrical Engineering and Computer Sciences,* 30(4), 1439-1459.

14. Son, J. and Buyya, R. (2018). Priority-aware VM allocation and network bandwidth provisioning in software-defined networking (SDN)-enabled clouds. *IEEE Transactions on Sustainable Computing,* 4(1), 17-28.

15. Qin, Q., Poularakis, K., Iosifidis, G.and Tassiulas, L. (2018). SDN controller placement at the edge: Optimizing delay and overheads. *IEEE INFOCOM 2018-IEEE Conference on Computer Communications. IEEE, 2018.*

16. Liyanage, K.S.K., Ma, M.and Chong, P.H.J. (2018). Controller placement optimization in hierarchical distributed software defined vehicular networks. *Computer Networks,* 135, 226-239.

17. Hou, X., Muqing, W., Bo, L.and Yifeng, L. (2019). Multi-controller deployment algorithm in hierarchical architecture for SDWAN. *IEEE Access,* 7, 65839-65851.

18. Baktir, A.C., Ozgovde, A.and Ersoy, C. (2017). How can edge computing benefit from software-defined networking: A survey, use cases, and future directions. *IEEE Communications Surveys & Tutorials,* 19(4), 2359-2391.

19. Jia, M., Cao, J.and Liang, W. (2015). Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. *IEEE Transactions on Cloud Computing,* 5(4), 725-737.

20. Benzekki, K., El Fergougui, A.and Elbelrhiti Elalaoui, A. (2016). Software-defined networking (SDN): a survey. *Security and communication networks,* 9(18), 5803-5833.

21. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Turner, J. (2008). OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM computer communication review,* 38(2), 69-74.

22. Akyildiz, I.F., Wang, P.and Lin, S.-C. (2015). SoftAir: A software defined networking architecture for 5G wireless systems. *Computer Networks,* 85, 1-18.

23. Casado, M., Foster, N.and Guha, A. (2014). Abstractions for software-defined networks. *Communications of the ACM,* 57(10), 86-95.

24. Kreutz, D., Ramos, F.M., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S.and Uhlig, S. (2014). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE,* 103(1), 14-76.

25. Salman, O., Elhajj, I.H., Kayssi, A.and Chehab, A. (2016). SDN controllers: A comparative study. *18th mediterranean electrotechnical conference (MELECON). IEEE, 2016.*

26. Mamushiane, L., Lysko, A.and Dlamini, S. (2018). A comparative evaluation of the performance of popular SDN controllers. *Wireless Days (WD). IEEE, 2018.*

27. Zhu, L., Karim, M.M., Sharif, K., Xu, C., Li, F., Du, X.and Guizani, M. (2020). SDN controllers: A comprehensive analysis and performance evaluation study. *ACM Computing Surveys (CSUR),* 53(6), 1-40.

28. Oh, B.-H., Vural, S., Wang, N.and Tafazolli, R. (2018). Priority-based flow control for dynamic and reliable flow management in SDN. *IEEE Transactions on Network and Service Management,* 15(4), 1720-1732.

29. Cox, J.H., Chung, J., Donovan, S., Ivey, J., Clark, R.J., Riley, G.and Owen, H.L. (2017). Advancing software-defined networks: A survey. *IEEE Access,* 5, 25487-25526.

30. Salsano, S., Blefari-Melazzi, N., Detti, A., Morabito, G.and Veltri, L. (2013). Information centric networking over SDN and OpenFlow: Architectural aspects and experiments on the OFELIA testbed. *Computer Networks,* 57(16), 3207-3221.

31. Jmal, R. and Fourati, L.C. (2020). Distributed software defined information centric networking. *International Journal of High Performance Computing and Networking,* 16(1), 14-25.

32. Fazea, Y. and Mohammed, F. (2021). Software Defined Networking based Information Centric Networking: An Overview of Approaches and Challenges. *International Congress of Advanced Technology and Engineering (ICOTEN),* 1-8, IEEE.

33. Guesmi, T., Kalghoum, A., Alshammari, B.M., Alsaif, H.and Alzamil, A. (2021). Leveraging software-defined networking approach for future information-centric networking enhancement. *Symmetry,* 13(3), 441.

34. Li, P., Muqing, W., Ning, W.and Hongbao, L. (2015). Supporting information-centric networking in SDN. *International Journal of Future Computer and Communication,* 4(6), 386.

35. Bannour, F., Souihi, S.and Mellouk, A. (2020). Adaptive distributed SDN controllers: application to content-centric delivery networks. *Future Generation Computer Systems,* 113, 78-93.

36. Alhisnawi, M. (2021). Forwarding Information Base Design Techniques in Content-Centric Networking: A Survey. *Next Generation of Internet of Things*. Springer. 157-174.

37. Nguyen, X.N., Saucez, D.and Turletti, T. (2013). Efficient caching in content-centric networks using OpenFlow. *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2013.*

38. Naeem, M.A., Rehman, M.A.U., Ullah, R.and Kim, B.-S. (2020). A comparative performance analysis of popularity-based caching strategies in named data networking. *IEEE Access,* 8, 50057-50077.

39. Amadeo, M., Campolo, C., Ruggeri, G., Molinaro, A.and Iera, A. (2020). Understanding Name-based Forwarding Rules in Software-Defined Named Data Networking. *2020 IEEE International Conference on Communications (ICC).*

40. Alomari, A., Subramaniam, S.K., Samian, N., Latip, R.and Zukarnain, Z. (2021). Resource management in SDN-based cloud and SDN-based fog computing: taxonomy study. *Symmetry,* 13(5), 734.

41. Lara, A. and Quesada, L. (2017). Priority-based routing in a campus network using SDN. *2017 IEEE 9th Latin-American Conference on Communications (LATINCOM).*

42. Heller, B., Sherwood, R.and McKeown, N. (2012). The controller placement problem. *ACM SIGCOMM Computer Communication Review,* 42(4), 473-478.

43. Hu, T., Guo, Z., Yi, P., Baker, T.and Lan, J. (2018). Multi-controller based software-defined networking: A survey. *IEEE Access,* 6, 15980-15996.

44. Yao, G., Bi, J., Li, Y.and Guo, L. (2014). On the capacitated controller placement problem in software defined networks. *IEEE Communications Letters,* 18(8), 1339-1342.

45. Sallahi, A. and St-Hilaire, M. (2014). Optimal model for the controller placement problem in software defined networks. *IEEE communications letters,* 19(1), 30-33.

46. Das, T., Sridharan, V.and Gurusamy, M. (2019). A survey on controller placement in sdn. *IEEE Communications Surveys & Tutorials,* 22(1), 472-503.

47. Isong, B., Molose, R.R.S., Abu-Mahfouz, A.M.and Dladlu, N. (2020). Comprehensive review of SDN controller placement strategies. *IEEE Access,* 8, 170070-170092.

48. Killi, B.P.R. and Rao, S.V. (2019). Controller placement in software defined networks: A comprehensive survey. *Computer Networks,* 163, 106883.

49. Lu, J., Zhang, Z., Hu, T., Yi, P.and Lan, J. (2019). A survey of controller placement problem in software-defined networking. *IEEE Access,* 7, 24290-24307.

50. Li, Y. and Wang, S. (2018). An energy-aware edge server placement algorithm in mobile edge computing. *In 2018 IEEE International Conference on Edge Computing (EDGE) (pp. 66-73). IEEE.*

51. Mukherjee, A., De, D.and Roy, D.G. (2016). A power and latency aware cloudlet selection strategy for multi-cloudlet environment. *IEEE Transactions on Cloud Computing,* 7(1), 141-154.

52. Osanaiye, O., Chen, S., Yan, Z., Lu, R., Choo, K.-K.R.and Dlodlo, M. (2017). From cloud to fog computing: A review and a conceptual live VM migration framework. *IEEE Access,* 5, 8284-8300.

53. Gravalos, I., Makris, P., Christodoulopoulos, K.and Varvarigos, E.A. (2018). Efficient network planning for internet of things with QoS constraints. *IEEE Internet of Things Journal,* 5(5), 3823-3836.

54. Goudarzi, M., Wu, H., Palaniswami, M.and Buyya, R. (2020). An application placement technique for concurrent IoT applications in edge and fog computing environments. *IEEE Transactions on Mobile Computing,* 20(4), 1298-1311.

55. Peng, K., Qian, X., Zhao, B., Zhang, K.and Liu, Y. (2020). A new cloudlet placement method based on affinity propagation for cyber-physical-social systems in wireless metropolitan area networks. *IEEE Access,* 8, 34313-34325.

56. Yousefpour, A., Ishigaki, G., Gour, R.and Jue, J.P. (2018). On reducing iot service delay via fog offloading. *IEEE Internet of Things Journal*, 5(2), 998-1010.

57. Samann, F.E.F., Zeebaree, S.R.and Askar, S. (2021). IoT provisioning QoS based on cloud and fog computing. *Journal of Applied Science and Technology Trends*, 2(01), 29-40.

58. da Silva, H.W., Barbalho, F.R.and Neto, A.V. (2019). Cross-layer multiuser session control for optimized communications on SDN-based cloud platforms. *Future Generation Computer Systems*, 92, 1116-1130.

59. Inagaki, Y., Shinkuma, R., Sato, T.and Oki, E. (2019). Prioritization of mobile IoT data transmission based on data importance extracted from machine learning model. *IEEE Access*, 7, 93611-93620.

60. Siasi, N. and Jaesim, A. (2020). Priority-aware SFC provisioning in fog computing. *17th Annual Consumer Communications & Networking Conference (CCNC), IEEE, 1-6.*

61. Eki, R.L. and Kumar, S. (2011). Method and system for priority based routing: Google Patents.

62. Kirkpatrick, K. (2013). Software-defined networking. *Communications of the ACM*, 56(9), 16-19.

63. İnternet: SİM (Sürekli İzleme Merkezi): T. C. Çevre ve Şehircilik Bakanlığı. URL: https://www.havaizleme.gov.tr/, Son Erişim Tarihi: 15.05.2022.

64. Lu, Y., Fu, B., Xi, X., Zhang, Z.and Wu, H. (2017). An SDN-based flow control mechanism for guaranteeing QoS and maximizing throughput. *Wireless Personal Communications*, 97(1), 417-442.

65. Shinkuma, R., Yamada, Y., Sato, T.and Oki, E. (2020). Flow control in SDN-Edge-Cloud cooperation system with machine learning. *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS).*

66. Qin, Z., Denker, G., Giannelli, C., Bellavista, P.and Venkatasubramanian, N. (2014). A software defined networking architecture for the internet-of-things. *2014 IEEE network operations and management symposium (NOMS).*

67. Deng, G.-C. and Wang, K. (2018). An application-aware QoS routing algorithm for SDN-based IoT networking. *2018 IEEE Symposium on Computers and Communications (ISCC).*

68. Diro, A.A., Reda, H.T.and Chilamkurti, N. (2018). Differential flow space allocation scheme in SDN based fog computing for IoT applications. *Journal of Ambient Intelligence and Humanized Computing*, 1-11.

69. Bardalai, P., Medhi, N., Bargayary, B.and Saikia, D.K. (2021). OpenHealthQ: OpenFlow based QoS management of Healthcare Data in a Software-Defined Fog environment. *ICC 2021-IEEE International Conference on Communications.*

70. Ghazi, M.U., Khattak, M.A.K., Shabir, B., Malik, A.W.and Ramzan, M.S. (2020). Emergency message dissemination in vehicular networks: A review. *Ieee Access,* 8, 38606-38621.

71. Kadhim, A.J. and Seno, S.A.H. (2019). Energy-efficient multicast routing protocol based on SDN and fog computing for vehicular networks. *Ad Hoc Networks,* 84, 68-81.

72. Secinti, G., Darian, P.B., Canberk, B.and Chowdhury, K.R. (2018). SDNs in the sky: Robust end-to-end connectivity for aerial vehicular networks. *IEEE Communications Magazine,* 56(1), 16-21.

73. İnternet: OpenvSwitch. URL: https://docs.openvswitch.org/en/latest/, Son Erişim Tarihi: 10.11.2021.

74. İnternet: OpenFlow. URL: https://github.com/mininet/openflow-tutorial/wiki, Son Erişim Tarihi: 10.05.2022.

75. İnternet: GNS3 Tutorial. URL: https://docs.gns3.com/1PvtRW5eAb8RJZ11maEYD9_aLY8kkdhgaMB0wPCz8a38/index.html, Son Erişim Tarihi: 10.05.2022.

76. İnternet: GNS3 Virtual Server. URL: https://docs.gns3.com/1Bn-s1Izkjp13HxcPF4b8QSGfkWJYG_dpMt9U1DQjvZ4/, Son Erişim Tarihi: 10.12.2021.

77. İnternet: OpenDaylight. URL: https://www.opendaylight.org/about, Son Erişim Tarihi: 20.05.2022.

78. İnternet: Docker. URL: https://docs.docker.com/engine/docker-overview/, Son Erişim Tarihi: 11.05.2022.

79. İnternet: Netcat. URL: https://nmap.org/, Son Erişim Tarihi: 15.05.2022.

80. Stancu, A.L., Halunga, S., Vulpe, A., Suciu, G., Fratu, O.and Popovici, E.C. (2015). A comparison between several Software Defined Networking controllers. *12th international conference on telecommunication in modern satellite, cable and broadcasting services (TELSIKS), 223-226, IEEE.*

81. Khondoker, R., Zaalouk, A., Marx, R.and Bayarou, K. (2014). Feature-based comparison and selection of Software Defined Networking (SDN) controllers. *2014 world congress on computer applications and information systems (WCCAIS).*

82. İnternet: OpenDaylight MD-SAL. URL: https://wiki.opendaylight.org/view/OpenDaylight_Controller:MD-SAL, Son Erişim Tarihi: 20.05.2022.

83. Singh, A.K. and Srivastava, S. (2018). A survey and classification of controller placement problem in SDN. *International Journal of Network Management,* 28(3), e2018.

84. Nunes, B.A.A., Mendonca, M., Nguyen, X.-N., Obraczka, K.and Turletti, T. (2014). A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications surveys & tutorials,* 16(3), 1617-1634.

85. Xia, W., Wen, Y., Foh, C.H., Niyato, D.and Xie, H. (2014). A survey on software-defined networking. *IEEE Communications Surveys & Tutorials,* 17(1), 27-51.

86. Michel, O. and Keller, E. (2017). SDN in wide-area networks: A survey. *2017 Fourth International Conference on Software Defined Systems (SDS), 37-42, IEEE.*

87. Gupta, A. and Jha, R.K. (2015). A survey of 5G network: Architecture and emerging technologies. *IEEE access,* 3, 1206-1232.

88. Bera, S., Misra, S.and Vasilakos, A.V. (2017). Software-defined networking for internet of things: A survey. *IEEE Internet of Things Journal,* 4(6), 1994-2008.

89. Haque, I.T. and Abu-Ghazaleh, N. (2016). Wireless software defined networking: A survey and taxonomy. *IEEE Communications Surveys & Tutorials,* 18(4), 2713-2737.

90. Li, Y. and Chen, M. (2015). Software-defined network function virtualization: A survey. *IEEE Access,* 3, 2542-2553.

91. Wang, G., Zhao, Y., Huang, J.and Wang, W. (2017). The controller placement problem in software defined networking: A survey. *IEEE Network,* 31(5), 21-27.

92. Kobo, H.I., Abu-Mahfouz, A.M.and Hancke, G.P. (2019). Efficient controller placement and reelection mechanism in distributed control system for software defined wireless sensor networks. *Transactions on Emerging Telecommunications Technologies,* 30(6), e3588.

93. Kuang, H., Qiu, Y., Li, R.and Liu, X. (2018). A hierarchical K-means algorithm for controller placement in SDN-based WAN architecture. *10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA).*

94. Wang, G., Zhao, Y., Huang, J., Duan, Q.and Li, J. (2016). A K-means-based network partition algorithm for controller placement in software defined network. *IEEE International Conference on Communications (ICC).*

95. Xiao, P., Qu, W., Qi, H., Li, Z.and Xu, Y. (2014). The SDN controller placement problem for WAN. *IEEE/CIC International Conference on Communications in China (ICCC).*

96. Jimenez, Y., Cervello-Pastor, C.and Garcia, A.J. (2014). On the controller placement for designing a distributed SDN control layer. *IFIP Networking Conference.*

97. Hu, Y.-N., Wang, W.-D., Gong, X.-Y., Que, X.-R.and Cheng, S.-D. (2012). On the placement of controllers in software-defined networks. *The Journal of China Universities of Posts and Telecommunications,* 19, 92-171.

98. Alenazi, M.J. and Cetinkaya, E.K. (2020). Resilient placement of SDN controllers exploiting disjoint paths. *Transactions on Emerging Telecommunications Technologies,* 31(2), e3725.

99. Hu, Y., Wang, W., Gong, X., Que, X.and Cheng, S. (2014). On reliability-optimized controller placement for software-defined networks. *China Communications,* 11(2), 38-54.

100. Torkamani-Azar, S. and Jahanshahi, M. (2020). A new GSO based method for SDN controller placement. *Computer Communications,* 163, 91-108.

101. Cheng, T.Y., Wang, M.and Jia, X. (2015). QoS-guaranteed controller placement in SDN. *IEEE Global Communications Conference (GLOBECOM).*

102. Fonseca, P.C. and Mota, E.S. (2017). A survey on fault management in software-defined networks. *IEEE Communications Surveys & Tutorials,* 19(4), 2284-2321.

103. Chen, J., Chen, J., Xu, F., Yin, M.and Zhang, W. (2015). When software defined networks meet fault tolerance: A survey. *International conference on algorithms and architectures for parallel processing.*

104. Marín-Tordera, E., Masip-Bruin, X., García-Almiñana, J., Jukan, A., Ren, G.-J.and Zhu, J. (2017). Do we all really know what a fog node is? Current trends towards an open definition. *Computer Communications,* 109, 117-130.

105. Zhou, S., Sun, Y., Jiang, Z.and Niu, Z. (2019). Exploiting moving intelligence: Delay-optimized computation offloading in vehicular fog networks. *IEEE Communications Magazine,* 57(5), 49-55.

106. Xu, Z., Liang, W., Xu, W., Jia, M.and Guo, S. (2015). Efficient algorithms for capacitated cloudlet placements. *IEEE Transactions on Parallel and Distributed Systems,* 27(10), 2866-2880.

107. Shi, Q., Wang, X.and Lin, W. (2018). Placement Strategy for Replicated Servers in CDN. *37th Chinese Control Conference (CCC), 6410-6416, IEEE.*

108. da Silva, R.A. and da Fonseca, N.L. (2020). Location of Fog Nodes for Reduction of Energy Consumption of End-User Devices. *IEEE Transactions on Green Communications and Networking,* 4(2), 593-605.

109. Benamrane, F., Ros, F.J.and Mamoun, M.B. (2016). Synchronisation cost of multi-controller deployments in software-defined networks. *International Journal of High Performance Computing and Networking,* 9(4), 291-298.

110. Herrera, J.L., Foschini, L., Galán-Jiménez, J.and Berrocal, J. (2020). The Service Node Placement Problem in Software-Defined Fog Networks. *IEEE Symposium on Computers and Communications (ISCC).*

111. Huang, X., Yang, Y.and Wu, X. (2019). A Meta-Heuristic Computation Offloading Strategy for IoT Applications in an Edge-Cloud Framework. *3rd International Symposium on Computer Science and Intelligent Control.*

112. Shah-Mansouri, H. and Wong, V.W. (2018). Hierarchical fog-cloud computing for IoT systems: A computation offloading game. *IEEE Internet of Things Journal,* 5(4), 3246-3257.

113. Souza, V.B.C., Ramírez, W., Masip-Bruin, X., Marín-Tordera, E., Ren, G.and Tashakor, G. (2016). Handling service allocation in combined fog-cloud scenarios. *IEEE international conference on communications (ICC).*

114. Liu, Y., Yu, F.R., Li, X., Ji, H.and Leung, V.C. (2018). Distributed resource allocation and computation offloading in fog and cloud networks with non-orthogonal multiple access. *IEEE Transactions on Vehicular Technology,* 67(12), 12137-12151.

115. Lai, P., He, Q., Abdelrazek, M., Chen, F., Hosking, J., Grundy, J.and Yang, Y. (2018). Optimal edge user allocation in edge computing with variable sized vector bin packing. *International Conference on Service-Oriented Computing,* 230-245, Springer, Cham.

116. Wang, S., Zhao, Y., Xu, J., Yuan, J.and Hsu, C.-H. (2019). Edge server placement in mobile edge computing. *Journal of Parallel and Distributed Computing,* 127, 160-168.

117. Chen, L., Wu, J., Zhou, G.and Ma, L. (2018). QUICK: QoS-guaranteed efficient cloudlet placement in wireless metropolitan area networks. *The Journal of Supercomputing,* 74(8), 4037-4059.

118. Lai, P., He, Q., Cui, G., Xia, X., Abdelrazek, M., Chen, F., Yang, Y. (2019). Edge user allocation with dynamic quality of service. *International Conference on Service-Oriented Computing,* 86-101, Springer, Cham.

119. Liang, T.-Y. and Li, Y.-J. (2017). A location-aware service deployment algorithm based on k-means for cloudlets. *Mobile Information Systems,* 2017.

120. Shen, C., Xue, S.and Fu, S. (2019). ECPM: an energy-efficient cloudlet placement method in mobile cloud environment. *EURASIP Journal on Wireless Communications and Networking,* 2019(1), 141.

121. Zhang, Y., Wang, K., Zhou, Y.and He, Q. (2018). Enhanced adaptive cloudlet placement approach for mobile application on spark. *Security and Communication Networks,* 2018.

122. Hakimi, S.L. (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations research,* 12(3), 450-459.

123. Teitz, M.B. and Bart, P. (1968). Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations research,* 16(5), 955-961.

124. ReVelle, C.S. and Swain, R.W. (1970). Central facilities location. *Geographical analysis,* 2(1), 30-42.

125. Rolland, E., Schilling, D.A.and Current, J.R. (1997). An efficient tabu search procedure for the p-median problem. *European Journal of Operational Research,* 96(2), 329-342.

126. Tuba, E., Strumberger, I., Bacanin, N.and Tuba, M. (2018). Bare bones fireworks algorithm for capacitated p-median problem. *International Conference on Swarm Intelligence,* 283-291, Springer, Cham.

127. Shi, J., Zheng, X., Jiao, B.and Wang, R. (2019). Multi-Scenario Cooperative Evolutionary Algorithm for the β-Robust p-Median Problem with Demand Uncertainty. *Applied Sciences,* 9(19), 4174.

128. Silva, S.d.S., Costa, M.G.F.and Costa Filho, C.F. (2019). Customized genetic algorithm for facility allocation using p-median. *Federated Conference on Computer Science and Information Systems (FedCSIS),* 165-169, IEEE.

129. Herda, M. (2017). Parallel genetic algorithm for capacitated p-median problem. *Procedia engineering,* 192, 313-317.

130. Chang, J., Wang, L., Hao, J.-K.and Wang, Y. (2020). Parallel iterative solution-based tabu search for the obnoxious p-median problem. *Computers & Operations Research,* 105155.

131. Montoya, M.R., Velázquez, R.G., Analco, M.E., Flores, J.L.M.and Loranca, M.B.B. (2019). Solution Search for the Capacitated P-Median Problem using Tabu Search. *International Journal of Combinatorial Optimization Problems and Informatics,* 10(2), 17-25.

132. Adeh, I.M., Baroughi, F.and Alizadeh, B. (2017). A modified particle swarm optimization algorithm for general inverse ordered p-median location problem on networks. *Facta Universitatis Series Mathematics and Informatics,* 32(4), 447-468.

133. Mu, W. and Tong, D. (2018). A spatial-knowledge-enhanced heuristic for solving the p-median problem. *Transactions in GIS,* 22(2), 477-493.

134. Gokalp, O. (2020). An iterated greedy algorithm for the obnoxious p-median problem. *Engineering Applications of Artificial Intelligence,* 92, 103674.

135. Colmenar, J.M., Greistorfer, P., Martí, R.and Duarte, A. (2016). Advanced greedy randomized adaptive search procedure for the obnoxious p-median problem. *European Journal of Operational Research,* 252(2), 432-442.

136. Lin, G. and Guan, J. (2018). A hybrid binary particle swarm optimization for the obnoxious p-median problem. *Information Sciences,* 425, 1-17.

137. Katayama, N. (2019). A combined fast greedy heuristic for the capacitated multicommodity network design problem. *Journal of the Operational Research Society,* 70(11), 1983-1996.

138. Ali, J. and Dyo, V. (2017). Coverage and mobile sensor placement for vehicles on predetermined routes: A greedy heuristic approach.

139. Sahoo, J. and Glitho, R. (2016). Greedy heuristic for replica server placement in cloud based content delivery networks. *IEEE Symposium on Computers and Communication (ISCC),* 302-309, IEEE.

140. Zhang, T., Ji, X.and Xu, W. (2019). AHV-RPL: Jamming-Resilient Backup Nodes Selection for RPL-Based Routing in Smart Grid AMI Networks. *the International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness,* 235-251, Springer, Cham.

141. Terzi, R., Sağıroğlu, Ş., Cocu, O., Arkan, R., Tosun, M.and Tulgar, Y. (2021). New driver behaviour models for fleet management based on big data analytics. *Journal of the Faculty of Engineering and Architecture of Gazi University*, 36(1), 543-557.

GAZİ GELECEKTİR...