HÜCRESEL AĞLAR İÇİN YENİ BİR AKTİF KUYRUK YÖNETİMİ ALGORİTMASI TASARIMI

Muhammet ÇAKMAK

2021
DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ

Tez Danışmanı
Dr. Öğr. Üyesi Zafer ALBAYRAK

HÜCRESEL AĞLAR İÇİN YENİ BİR AKTİF KUYRUK YÖNETİMİ ALGORİTMASI TASARIMI

Muhammet ÇAKMAK

T.C.
Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalında
Doktora Tezi
Olarak Hazırlanmıştır

Tez Danışmanı
Dr. Öğr.Üyesi Zafer ALBAYRAK

KARABÜK
Ocak 2021

Muhammet ÇAKMAK tarafından hazırlanan "HÜCRESEL AĞLAR İÇİN YENİ BİR AKTİF KUYRUK YÖNETİMİ ALGORİTMASI TASARIMI" başlıklı bu tezin Doktora Tezi olarak uygun olduğunu onaylarım.

Dr. Öğr. Üyesi Zafer ALBAYRAK .....
Tez Danışmanı, Bilgisayar Mühendisliği Anabilim Dalı

Bu çalışma, jürimiz tarafından oy birliği ile Bilgisayar Mühendisliği Anabilim Dalında Doktora tezi olarak kabul edilmiştir. 04/01/2021

Ünvanı, Adı SOYADI (Kurumu)	İmzası
Başkan : Prof. Dr. Necmi Serkan TEZEL (KBÜ)	.....
Üye : Doç. Dr. Ahmet ZENGİN (SAÜ)	.....
Üye : Doç. Dr. Hakan KUTUCU (KBÜ)	.....
Üye : Doç. Dr. Turgut ÖZTÜRK (BTÜ)	.....
Üye : Dr. Öğr. Üyesi Zafer ALBAYRAK (KBÜ)	.....

.

KBÜ Lisansüstü Eğitim Enstitüsü Yönetim Kurulu, bu tez ile, Doktora derecesini onamıştır.

Prof. Dr. Hasan SOLMAZ .....
Lisansüstü Eğitim Enstitüsü Müdürü

"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."

Muhammet ÇAKMAK

ÖZET

Doktora Tezi

HÜCRESEL AĞLAR İÇİN YENİ BİR AKTİF KUYRUK YÖNETİMİ ALGORİTMASI TASARIMI

Muhammet ÇAKMAK

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi Zafer ALBAYRAK
Ocak 2021, 73 sayfa

Günümüzde internet içeriğinin çoğu hücresel ağlar ile sağlanmaktadır. Hücresel ağlar yüksek kaliteli video, ses, veri indirme, bulut tabanlı uygulamalar, akıllı ev yönetimi, otonom araç kontrolü, artırılmış gerçeklik ve sanal gerçeklik gibi uygulamalara erişim ve yönetimde kilit bir role sahiptir. Hücresel ağların kullanım sahasının genişlemesi, internet kullanıcılarının sayısındaki hızlı artış ve mobil internet kullanımında artan talepler karşısında hücresel ağlarda kaynakların yönetimi daha fazla önem kazanmıştır. Hücresel ağ operatörleri, ağ donanımı geliştiren sanayi kuruluşları ve araştırmacılar hücresel ağ sistemlerinin geliştirilmesi, performanslarının iyileştirilmesi ve artan mobil internet taleplerinin karşılanması için yeni çözümler üretmektedir. Hücresel Uzun Vadeli Evrim (LTE - Long Term Evolution) ağında, veri iletiminde kapasiteyi aşan ani bir artış olduğunda kullanıcı kuyruklarında aşırı paket yığılması oluşur. Eğer kullanıcı kuyruk boyutları yeterli değilse kuyruklardaki fazla paketler düşer. Bu durum ağdaki kaynakların boşuna harcanmasına, uçtan uca verimin

azalmasına ve kuyruk gecikmelerine neden olur. Ağdaki meydana gelebilecek tıkanıklık sorunlarının çözülmesi için kuyruk yönetim algoritmaları kullanılır. Kuyruk yönetim algoritmaları ağda yaşanabilecek muhtemel tıkanıklıkları önceden tespit ederek tıkanıklık oluşmadan ağı kontrol eder ve tıkanıklık oluşması durumunda tıkanıklıktan kurtulma mekanizmalarını çalıştırır. Tıkanıklık sorunlarının çözümü için uygun algoritmanın seçimi, ağın çalışmasına doğrudan etki etmektedir.

Bu tez çalışmasında, hücresel LTE ağlarında yaşanan tıkanıklık ve darboğaz problemlerinin çözümü için Radyo Bağlantısı Kontrolü (RLC - Radio Link Control) katmanında çalışan yeni bir aktif kuyruk yönetim algoritması geliştirilmiştir. Geliştirilen algoritma iki adımda çözüm üretmektedir. İlk adımda kuyruk doluluğunu kontrol edilmesini sağlayan sanal kuyruk yapısını kullanılmaktadır. Sanal kuyruk, gerçek ortalama kuyruk boyutunu kontrol ederek ağın ani salınımlardan korunmasını sağlamaktadır. İkinci adımda ise Evrimleşmiş Düğüm B'deki (eNodeB - Evolved Node B) kullanıcı kuyruklarında yaşanan kuyruk taşması problemini çözmek için paket düşürme olasılık değeri ağdaki yük durumuna göre adaptif şekilde yeniden hesaplamaktadır. Geliştirilen Adaptif Sanal Kuyruk Tabanlı RED (AVRED-r) algoritması uçtan uca ortalama verim, gecikme, paket teslim oranı ve adalet indeksi değerleri açısından Rastgele Erken Tespit (RED - Random Early Detection), DropTail, Geliştirilmiş Orantılı İntegral Denetleyici (PIE - Proportional Integral Controller Enhanced), Paket Sınırlı İlk Giren İlk Çıkış Sırası (pFIFO - Packet Limited First In First Out Queue) ve Kontrollü Gecikme (CoDel -Controlled Delay) algoritmalarına göre daha iyi sonuç üretmektedir.

Geliştirilen algoritma RLC tamponun farklı yük değerlerinde dengeli çalışmasını sağlamaktadır. Ayrıca Paket Veri Yakınsama Protokolü (PDCP - Packet Data Convergence Protocol) katmanına iletilecek olan Servis Veri Birimlerinin (SDU - Service Data Unit) RLC tamponun tıkanıklık seviyesine göre iletilmesine katkı sunmaktadır.

Anahtar Sözcükler : LTE ağı, QoS, ns-3, Simülasyon, Kuyruk yönetim algoritmaları
Bilim Kodu **:** 92407

ABSTRACT

Ph.D. Thesis

MODELLING A NEW ACTIVE QUEUE MANAGEMENT ALGORITHM FOR CELLULAR NETWORKS

Muhammet ÇAKMAK

Karabük University
Institute of Graduate Programs
Department of Computer Engineering

Thesis Advisor:
Assist. Prof. Dr. Zafer ALBAYRAK
January 2021, 73 pages

Most internet content is provided by cellular networks today. Cellular networks play a key role in accessing and managing high-quality video, audio, data download, cloud-based applications, smart home management, autonomous vehicle control, augmented reality and virtual reality. The management of resources in cellular networks has gained more importance in the face of the expansion of the usage area of cellular networks, the rapid increase in the number of internet users and the increasing demands in mobile internet usage. Cellular network operators, industry organizations that develop network hardware, and researchers are creating new solutions for the development of cellular network systems, improving their performance and meeting the increasing mobile internet demands.

In Long Term Evolution (LTE) network, when there is a sudden increase in data transmission that exceeds the capacity, excessive packet congestion occurs in user

queues. If the user queue sizes are not enough, the excess packages in the queues will drop. This results in wasted resources on the network, reduced end-to-end throughput, and queue delays. Queue management algorithms are used to solve congestion problems that may occur in the network. Queue management algorithms detect possible congestions in the network in advance and control the network before congestion occurs, and in case of congestion, it operates the mechanisms to get rid of congestion. Choosing the proper algorithm for the solution of congestion problems directly affects the performance of the network.

In this thesis, a new active queue management algorithm working on the Radio Link Control (RLC) layer has been developed to solve congestion and bottleneck problems in cellular LTE networks. The developed algorithm finds solutions in two steps. In the first step, the virtual queue structure is used to control the queue occupancy. The virtual queue ensures that the network is protected from sudden oscillations by controlling the real average queue size. In the second step, the packet drop probability value is adaptively recalculated according to the load on the network to solve the queue overflow problem in the user queues.

The proposed Adaptive Virtual Queue Based RED (AVRED-r) algorithm gives better results than RED (Random Early Detection), DropTail, Proportional Integral Controller Enhanced (PIE), Packet Limited First In First Out Queue (pFIFO-) and Controlled Delay (CoDel) algorithms in terms of end-to-end average throughput, delay, Packet Delivery Fraction (PDF) and fairness index values.

The proposed AVRED-r algorithm ensures stable operation of the RLC buffer at different load values. It also contributes to the transmission of the Service Data Units (SDU) to be transmitted to the Packet Data Convergence Protocol (PDCP) layer according to the congestion level of the RLC buffer.

Key Word : LTE network , QoS, Ns-3 simulation, Queue management algorithms
Science Code : 92407

TEŞEKKÜR

Bu tez çalışmasının planlanmasında, araştırılmasında, yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığım, yönlendirme ve bilgilendirmeleriyle çalışmamı bilimsel temeller ışığında şekillendiren değerli hocam Dr. Öğr. Üyesi Zafer ALBAYRAK'a teşekkürlerimi sunarım.

Tez çalışmamdaki önemli katkıları sebebiyle tez izleme komitesi ve tez savunma jürisindeki hocalarıma teşekkürü bir borç bilmekteyim. Sevgili alime ve dostlarıma manevi desteklerinden dolayı müteşekkirim.

İÇİNDEKİLER

Sayfa

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

SİMGELER

q : Kuyruğun anlık boyutu

avg : Kuyruğun ortalama boyutu

wq : Kuyruğun ortalama ağırlığı

Pa : Paket düşme olasılığı

$Minth$: Minimum eşik değeri

$Maxth$: Maksimum eşik değeri

q : Kuyruk boyutu

p_{max} : Maksimum kuyruk değeri

r : Algortimanın versiyon değeri

KISALTMALAR

AQM : Aktif Kuyruk Yönetim Algoritması

AMPS : Gelişmiş Cep Telefonu Sistemi

TACS : Toplam Erişim İletişim Sistemi

1G : 1. Nesil

2G : 2. Nesil

3G : 3. Nesil

4G : 4. Nesil

5G : 5. Nesil

GPRS : Genel Paket Radyo Servisi

EDGE : GSM Evrimi için Gelişmiş Veri Hızı

UMTS : Evrensel Mobil Telefon Sistemi

HSPA : Yüksek Hızlı Paket Erişimi

HPSA+ : Gelişmiş Yüksek Hızlı Paket Erişimi

WiMAX : Mikrodalga Erişimi için Dünya Çapında Birlikte Çalışabilirlik

LTE	: Uzun Vadeli Evrim
LTE-A	: Uzun Vadeli Evrim Gelişmiş
IoT	: Nesnelerin İnterneti
QoS	: Hizmet Kalitesi
3GPP	: Üçüncü Nesil Ortaklık Projesi
MAC	: Medya Erişim Kontrolü
RLC	: Radyo Bağlantı Kontrolü
PDCP	: Paket Veri Yakınsama Kontrolü
SDU	: Hizmet Veri Birimleri
TCP/IP	: İletim Denetimi Protokolü / Internet Protokolü
RED	: Rastgele Erken Tespit Algoritması
HOL	: Satır Başı Paket Gecikmesi
VQ	: Sanal Kuyruk
RQ	: Gerçek Kuyruk
DFCSD	: Gecikmeli Akış Kontrol Algoritması
SFB	: Stochastic Fair Blue (Stokastik Açık Mavi)
SAE	: Sistem Mimarisi Evrimi
PIE	: Geliştirilmiş Orantılı İntegral Denetleyici
pFIFO	: Paket Sınırlı İlk Giren İlk Çıkış Sırası
ColDel	: Controlling Delay (Kontrollü Gecikme)
SAE	: Sistem Mimarisi Evrimi
EPC	: Evrimleşmiş Paket Çekirdeği
E-UTRAN	: Gelişmiş Evrensel Karasal Radyo Erişim Ağı
MME	: Mobilite Yönetimi Varlığı
SGW	: Servis Ağ Geçidi
RTT	: Gidiş-Dönüş Süresi
PDN	: Paket Veri Ağları
IMS	: IP Multimedya Alt Sistemi
VoIP	: İnternet Üzerinden Ses Protokolü
RTP	: Gerçek Zamanlı Aktarım Protokolü
UDP	: Kullanıcı Datagram Protokolü
OFDMA	: Dik Frekans Bölmeli Çoklu Erişim
SC-FDMA	: Tek Taşıyıcılı Frekans Bölmeli Çoklu Erişim

BÖLÜM 1

GİRİŞ

Hücresel iletişim teknolojisi son yıllarda çok hızlı bir dönüşüm geçirmektedir [1]. Bu hızlı dönüşümdeki temel etken, hücresel ağlarda artan veri gereksinimidir. Hücresel iletişim teknolojisindeki değişimler nesiller olarak ifade edilmektedir. Her neslin onu diğer nesillerden ayıran bazı yeni teknikleri, yapısı ve kapasite değerleri vardır. Birinci Nesil (1G) mobil iletişim teknolojisi, analog sinyallerin iletişimi ile başlar. 1G teknolojisinde Amerika'da Gelişmiş Cep Telefonu Sistemi (AMPS - Advanced Mobile Phone System) kullanılırken, Avrupa kıtasında Toplam Erişim İletişim Sistemi (TACS - Total Access Communication System) kullanılmıştır. İkinci Nesil (2G) ile mobil iletişim dijital sinyallere dönüşmüştür. 2G ile yüksek ses kalitesi, şifreleme ve mesaj verilerinin iletimi gibi yenilikler ortaya çıkmıştır [2]. Genel Paket Radyo Servisi (GPRS- Genel Paket Radyo Servisi) 2.5G ve GSM Evrimi için Gelişmiş Veri Hızı (EDGE - Enhanced Data for GSM Evolution) 2.75G ile mobil iletişimde veri aktarımına başlanmıştır. Evrensel Mobil Telefon Sistemi (UMTS - Universal Mobile Telecommunications System) olarak bilinen Üçüncü Nesil (3G) ile hızlı veri iletimi, görüntülü görüşme ve mobil internet erişimi hız kazanmıştır [3]. 3.5G olarak da bilinen Yüksek Hızlı Paket Erişimi (HSPA - High Speed Packet Access) ve 3.75G olarak da bilinen Gelişmiş Yüksek Hızlı Paket Erişimi (HPSA+ - Evolved High Speed Packet Access) ile daha hızlı internet için düşük gecikme süresi ve yüksek hızlı veri elde edilmiştir. Dördüncü nesil (4G) teknolojiler olan Mikrodalga Erişimi için Dünya Çapında Birlikte Çalışabilirlik (WiMAX - Worldwide Interoperability for Microwave Access), LTE ve Uzun Vadeli Evrim Gelişmiş (LTE A - Long Term Evolution Advanced) yüksek çözünürlüklü TV, video konferans, 3D TV gibi yüksek hızlı uygulamalar için kullanılabilmektedir [4]. Beşinci Nesil (5G) ile nesnelerin interneti (IoT - Internet of Things), akıllı ev, sürücüsüz araba, yüksek çözünürlüklü ve çok hızlı veri aktarımı, sanal gerçeklik ve artırılmış gerçeklik, Endüstri 4.0 ve uzaktan cerrahi uygulamalar gibi alanları geliştirmeye devam etmektedir [5–9].

Hızla değişim geçiren hücresel ağlarda artan kaynak gereksinimlerinin karşılanması için hücresel ağ operatörle ses, video, multimedya hizmeti ve gittikçe yaygınlaşan IoT gibi birçok uygulama için abonelerine en iyi Hizmet Kalitesini z(QoS - Quality of Service) sağlaması gerekmektedir [10]. Hücresel ağlarda yüksek kaliteli video ve ses akışı, yüksek hızlı veri indirme, bulut tabanlı uygulamalara erişim, akıllı ev ve otonom araç kontrolü, artırılmış ve sanal gerçeklik uygulamalarına hızlı erişim gibi konularda artan talepler, hücresel ağ şebekelerin geliştirilmesindeki ana etkenleri oluşturmaktadır. Yaşanan bu değişimler karşında hücresel ağlarda kaynakların yönetimi daha fazla önem kazanmaktadır. Bu nedenle hücresel ağ operatörleri, ağ donanımı geliştiren sanayi kuruluşları ve araştırmacılar, hücresel ağ sistemlerinin geliştirilmesi ve performanslarının iyileştirilmesi için yeni çözümler önermektedirler [11].

Hücresel LTE ağında, veri iletiminde kapasiteyi aşan ani bir artış olduğunda eNodeB kullanıcı kuyruklarında aşırı paket yığılması oluşur. Eğer eNodeB'deki kullanıcı kuyruk boyutları yeterli değilse kuyruklardaki fazla paketler düşer [12]. LTE ağında fiziksel katman, Medya Erişim Kontrolü (MAC-Media Access Control), RLC, PDCP ve daha üst katmanlarda veri akış hızının artırılması için çok sayıda araştırma yapılmıştır [13–16]. RLC, veri paketlerinin güvenli şekilde iletilmesini sağlayan bir LTE protokolüdür [17]. RLC, MAC katmanın üzerinde PCDP katmanın altında bulunur. RLC katmanında RLC SDU'lar segmentlere ayrılır ve birleştirme işlemi yapılır [18]. Veriler, Kullanıcı Ekipmanına (UE - User Equipment) iletilmeden önce RLC katmanında tutulur. RLC katmanındaki yaşanan tıkanıklık ve taşma durumunda veri gecikmeleri kaçınılmazdır. Yaşanan taşma ve gecikmeler, gecikmeye karşı duyarlı hizmetler (örneğin ses) için QoS değerlerinin sağlanmasını önler [19]. LTE RLC katmanında yaşanan tıkanıklık, kanal iletişim kalitesini doğrudan etkiler. Zayıf kanal iletişim kalitesi durumunda düşük hızlı modülasyon şeması seçilir. Ayrıca bu durum paketlerin bölümlenmesine neden olabilir [20]. Kanal durumu RLC katmanındaki SDU'ları iletmek için uygun değilse, alt katmandaki RLC varlıkları bölümleme ve birleştirme için beklemek zorunda kalmaktadır. Bu durum LTE ağlarındaki arabelleklerde tıkanıklık ve zaman kaybına neden olmaktadır.

Kuyruk yönetim algoritmaları ağda yaşanan kuyruk taşma problemlerini çözmek için geliştirilmiştir [21]. Kuyruk yönetimi için geliştirilen algoritmalar tampondaki sıkışma ve taşma durumunda çalışmaktadır [22]. Bu durum sürekli gerçekleştiğinde düşürülen paketler ağ trafiğini olumsuz etkilemektedir. Ayrıca kullanıcılar için gerekli olan QoS değerlerinin sağlanmasında zorluklar yaşanmaktadır. Aktif Kuyruk Yönetim Algoritmaları (AQM) ağdaki tıkanıklığı göndericiyi kapalı veya açık bildirim mekanizması ile haberdar ederek önler [23]. Kapalı geri bildirim sistemi [22,24] olasılık değerini kullanarak yönlendirici kuyruğundan paket düşürür. Göndericinin paket gönderim oranını azaltılmasını sağlayarak yönlendiricideki veri trafiğini azaltır. Buradaki problem bu mekanizma ile düşürülen paketlerin yeniden iletilmesinin zaman almasıdır. Bu durum ağ kaynaklarının tüketilmesine sebep olur. Açık tıkanıklık bildirimi [25] mekanizması kaynağı bilgilendirmek için TCP onay paketinin başlığındaki ikili bildirim yapısını kullanır. Kaynak, tıkanıklık geri bildirimini kullanarak paketlerin kuyruktan düşmemesi için pencere boyutunu küçültür. Ancak ikili bildirim yapısı, yüksek trafik durumunda ağdaki tıkanıklığı engelleyememiştir.

1.1. ÇALIŞMANIN AMACI

Yukarıdaki sorunlar göz önüne alındığında, Hücresel LTE ağında yaşanan tıkanıklık ve darboğaz problemlerinin çözümü için RLC katmanında çalışan aktif kuyruk yönetim algoritmasının sistem verimliliğinin artırılması bu tez çalışmasının ana konusunu oluşturmaktadır. Önerilen algoritma, ağdaki uçtan uca verim ve paket gecikmesi değerlerini kontrol ederek RLC tamponun farklı yük değerlerinde dengeli çalışmasını sağlamaktadır. Ayrıca PDCP katmanına iletilecek olan SDU'ların RLC tamponun tıkanıklık seviyesine göre iletilmesine katkı sunmaktadır. Yukarıda belirtilen iyileştirmeleri sağlayabilmek için sanal kuyruk kontrolü önerilmiş ve paket düşürme olasılık değeri ağın trafik yoğunluğuna göre adaptif şekilde ayarlanmıştır.

1.2. LİTERATÜR İNCELEMESİ

Bu bölümde LTE ağında yaşanan kuyruk taşması, tıkanıklık ve gecikme sorunları ve kuyruk yönetim algoritmalarının verimliliği ile ilgili yapılan literatür çalışması verilmiştir.

Adesh vd. eNodeB'deki tıkanıklık seviyesinin tahmini için paket programlayıcı sıkışıklık geri besleme mekanizması yöntemini önermiştir [12]. Önerilen yöntemde paket göndericisi tıkanıklık penceresini farklı trafik yük durumlarına göre değiştirir. Kuyruk gecikmesi ve kuyruk taşması sorununun azaltmak için paket gecikme tahmininin erken tanımlanması, ağ tıkanıklığının tahmini ve sıkışıklık penceresi boyutunu ayarlanması art arda yapılmıştır. Paketin QoS değeri kullanılarak paketin Geçiş Kontrol Protokolü (TCP - Transmission Control Protocol) ya da Kullanıcı Datagram Protokolü (UDP - Kullanıcı Datagram Protocol) olarak tanımlanması sağlanır. Paket düşüşlerine dayanıksız olan TCP için düşük, dayanıklı olan UDP için ise yüksek paket düşürme olasılıkları belirlenmiştir. Önerilen yöntem kullanıcılara kaynak atamasında öncelik yapısını kullandığı için kaynakların dağıtımında adaletsizliğe neden olmaktadır. Yüksek öncelik metriğine sahip kullanıcılar kaynakları sürekli kullandığı için düşük öncelik metriğine sahip kullanıcılar için kaynak atanmasında gecikmeler yaşanmaktadır.

Kumar vd. eNodeB RLC katmanında değişken trafik yüklerinde darboğaz ve sürekli paket düşürülmesi sorununun önlenmesi için akıllı Aktif Kuyruk Yönetimi (AQM - Active Queue Management) olan smRED algoritmasını önermiştir [26]. Önerilen şema, RED'deki varyans değerinin ayarlanmasıyla çalışmaktadır. Önerilen çalışmada paket planlayıcısının tek hücreli ve çok hücreli ağlarda hücresel geçiş ve hücresel geçiş olmadan verim, gecikme ve sapma değerleri üzerine etkisi incelenmiştir. RED'in ayarlanan varyans değeri *i*, düşük ve yüksek yük durumlarında farklı değerler almıştır. Bu yöntem istenilen değerlerin yanında istenilmeyen paket düşürme değerlerini de hesaplayarak ağda gecikmeye neden olmaktadır. Ayrıca üretilen paket düşürme değerinin hangisinin seçileceği için bir çözüm sunmamaktadır.

Wang vd., önerilen MAC planlayıcı ile Garantili Bit Hızı (GBR - Guaranteed Bit Rate) akışları için QoS gereksinimlerini sağlamaya çalışmıştır. Ayrıca önerilen yöntem GBR olmayan akışların veri iletimini de iyileştirmiştir. Geliştirilen algoritmanın çalışma prensibi her bir akışın kanal kalitesini, kuyruk durumunu, paket gecikmesini ve QoS sınıfı tanımlayıcısını dikkate alarak işlem yapmaktır [27]. Önerilen yöntemde aynı anda kontrol edilmek istenen parametreler kaynak atamasında gecikmelere sebep

olmaktadır. Önerilen çalışma, veri trafiği akış performansını artırmasına rağmen ortalama gecikme değerini azaltmaktadır.

Qui vd., LTE ağılarındaki akışlar için gerçek ve sanal tabanlı işaretleme şemalarını karşılaştırmışlardır. Daha yüksek öncelikli akışlar için kullanılan sanal kuyruk yapısının ağda daha az kaynak kullandığını göstermişlerdir [28]. Tasarlanan Sanal Kuyruk (VQ - Virtual Queue) tabanlı algoritmanın, Gerçek Kuyruk (RQ - Real Queue) tabanlı algoritmaya kıyasla tıkanıklık ve düşük kuyruk gecikmesi açısından daha iyi sonuçlar vermesine rağmen, kullanıcılar arasında adalet sağlamadığı gözlenmiştir.

Dong vd., hücresel ağlardaki büyük tampon kullanımının neden olduğu yüksek gecikme problemini çözmeye çalışmışlardır [29]. Önerdikleri yöntem ile kuyruktaki bekleme sürelerini sınırlandırdılar. Düşük gecikme sürelerine ihtiyaç duyan uygulamalarda hızı kontrol etmek için Gecikmeli Akış Kontrol Algoritması (DFCSD) adında bir iletim kontrol protokolü önerdiler. Önerilen DFCSD algoritmasının amacı uzun ve kısa süren akışlar rekabet ettiğinde hem kısa hem de uzun akışlar için TCP performansını iyileştirerek ağdaki gecikmeyi kontrol etmektir. Önerilen yöntemde TCP mekanizması ile akışlar üzerinde iyileştirmeler sağlansa da kuyruk yönetim algoritmaları tamamen işlev dışı bırakılmıştır.

Abdullah vd., LTE ağının TCP performansını iyileştirmek için yeni bir yaklaşım önerdiler [30]. Önerilen yaklaşım, eNodeB'de arabelleğe alma boyutunun kontrol edilmesine dayanmaktadır. Önerilen yöntemde eNodeB'nin bellek boyutu aktif kullanıcılar arasında bölünür. Ayrıca TCP kullanıcıları için tıkanıklık penceresi boyutu dinamik olarak ayarlanır. Önerilen yöntemdeki sorun, bellek boyutunun kullanıcılar arasında bölünmesiyle bellekte kullanılamayan çok miktarda boş alanın oluşmasıdır.

Diğer taraftan aktif kuyruk yönetimi algoritmaları için literatür çalışmaları incelendi. AQM algoritmaları sezgisel, optimizasyon ve kontrol teorisi olarak sınıflandırılmaktadır [31]. Gentle RED [32], adaptif RED [33], dinamik RED [34], hiperbola RED [35], GREEN [36], BLUE [37], Stokastik Açık Mavi (SFB- Stochastic Fair Blue) [38],vb. sezgisel yöntemin iyi bilinen örnekleridir. Bu önerilen algoritmalar, farklı yük durumları için verimlilik, paket kaybı, ağ kullanımı, adillik ve

uyarlanabilirlik gibi özellikleri geliştirmeyi amaçlamıştır. Gentle RED algoritmasında paket düşme ihtimalini yumuşak bir eğimle değiştirilir. Böylece ağdaki paket düşüşlerinin önüne geçilir. Gentle RED'in temel problemi parametre ayarlamasıdır. Farklı ağ koşullarına göre farklı parametrelerin ayarlanması gerekmektedir. Aktif RED (ARED - Active RED), çarpımsal artış-çarpımsal düşüş yaklaşımını kullanarak maksimum paket düşme olasılığı Pmax'ı uyarlanabilir şekilde ayarlar. ARED farklı parametre yapılandırmalarına duyarlı olmasına rağmen ağ ortamı karmaşık olduğunda performansı RED'den üstün değildir. BLUE tıkanıklık kontrol sistemini modellemek için bağlantının boş olma durumunu ve paket kayıp değerlerini kullanır. BLUE'da paket kaybı değerleri düşük olmasına rağmen uzun süren gecikme durumları tıkanıklığa neden olur. Adil Rastgele Erken Düşürme (FRED - Fair Random Early Drop) [39], RED algoritmasındaki adil olmayan akış yapısını azaltmak amacı ile geliştirilmiştir. FRED, filtre edilmiş bir grup bağlantıya geri besleme vererek, tıkanıklık uyarısı yapma problemini çözmeye çalışmıştır. Stabilize Rastgele Erken Düşürme (SRED - Stabilized Random Early Drop) [40], yüksek iletim hızını ve kaynakların eşit dağıtılmasını hedeflemektedir. SRED kuyruk boyunu serbest bir değere sabitlemeye çalışır. Ayrıca yüke bağlı olarak paketleri düşürmeye devam eder. Ağdaki akışları takip ederek ağın analizini yapar. SFB, BLUE algoritmasını temel alarak alır ve TCP akışlarını, tepkisiz akışlardan korumayı hedefler. SFB algoritması SRED ve BLUE'dan farklı olarak kuyruk tıkandığında paketleri düşürmek için basit bir geri bildirim yaklaşımı kullanır.

Kontrol teorik yaklaşımının amacı, AQM yapısını kullanarak TCP'nin modellenmesidir. Böylece istikrarlı ve hızlı tepki veren bir sistem kullanımına imkân sağlanır [34]. TCP tabanlı tıkanıklık kontrolü için birçok algoritma geliştirilmiştir. Gecikme tabanlı tıkanıklık kontrolü [32, 33], alıcı pencere kontrolü [42,43] kayıp tabanlı tıkanıklık kontrolü [44,45] ve hız tabanlı tıkanıklık kontrolü [46,47] gibi algoritmalar TCP tabanlı tıkanıklık kontrolü sistemlerindendir. Bu sistemler, AQM parametrelerini analitik olarak belirleyerek sistemin çalışmasını sağlar[48]. Ayrıca kontrol teorik yaklaşımı için Orantısal Türev (PD) denetleyicisi [49], Orantısal İntegral (PI) denetleyicisi [50] ve bulanık mantık denetleyicisi [51] gibi şemalar önerilmiştir.

Kablolu ve kablosuz ağlarda önerilen darboğaz sorununun çözümü için PIE [52] ve ColDel [53] şemaları önerilmektedir. ColDel, paketlerin kuyruktaki gecikmesini kontrol eder. Paket iletim süresi belirtilen hedef değerini aştığında tıkanıklık algılanır. Tüm paketler hedef gecikme süresi içinde oluncaya kadar kalan paketler kontrol edilir. PIE'de paketlerin uçtan uca hız değeri ölçülür. Hareketli paket düşürme değerinin artış ve azalışı kontrol edilir.

Tampon sıkışıklığı problemini azaltmak için [43]'de Dinamik Alıcı Pencere Ayarı (DRWA) yöntemi önerilmiştir. DRWA yöntemi, alıcı penceresi boyutunu (rwnd) minimum gidiş-dönüş süresinin (RTT – Round Trip Time) taban değerine göre hesaplamaktadır. Hesaplanan RTT değeri paketin geçerli RTT'si ile karşılaştırır. Daha sonra rwnd boyutu paketin tanıtım başlığı aracılığıyla iletir. Gönderici minimum tıkanıklık penceresi boyutuna ve rwnd boyutuna göre paketleri oluşturur. [54]'de tıkanıklık bilgisini göndericiye iletmede DRWA'dan etkilenererek tasarlanan Passive INverse feedbacK (PINK) algoritmasını geliştirmiştir. PINK'de rwnd boyutu, aktif kullanıcı sayısının bant genişliğindeki gecikme durumuna bağlı olarak tahmin edilmektedir. Her iki yöntemin de dezavantajı bu yöntemlerin kaynaklar için kübik veya yüksek hızlı TCP gibi yöntemlerle karşılaştırıldığında performansının daha az olmasıdır.

Literatür çalışması incelenmesinden sonra farklı trafik yüklerinde çalışabilen ve kullanıcılar arasında adil bir kaynak paylaşımı yapan tekniklerin eksik olduğu sonucuna varılmıştır. Önerilen yöntemler kuyruk taşması ve sıkışıklık problemlerinin iyileştirilmesini sağlasa da asıl amaç paket düşüşünün kontrol edilerek tüm trafik yükleri için çalışabilecek bir mekanizmanın oluşturulabilmesidir.

Elde edilen literatür sonuçlarına göre LTE ağlarında tıkanıklığa sebep olan veri akışları önlenmelidir. Ayrıca RLC arabelleğindeki kuyruk taşması sorunlarının çözülmesi ve kuyruktaki gecikmeden kaynaklanan paket kayıpların önlenmesi gerekmektedir. Bu sorunların çözümü için sanal kuyruk tabanlı AVRED-r algoritmasını öneriyoruz. Önerilen AVRED-r, klasik RED algoritmasındaki ortalama kuyruk uzunluğu (qav) yerine sanal ortalama kuyruk uzunluğunu (Vqav) kullanmaktadır. *Vqav*, eNodeB tamponundaki paket değişimlerini takip ederek ağı kuyruk gecikmesi ve kuyruk

taşması sorunlarına karşı korumaktadır. Ayrıca klasik RED'in doğrusal düşürme fonksiyonunu değiştirerek farklı trafik yükleri için ayarlanabilir parametre değerleri (r=2,3,4,5) kullanılmaktadır. Önerdiğimiz adaptif *r* parametresi ortalama kuyruk uzunluğu değerini değiştirerek düşük, orta ve yüksek trafik yoğunluğunda ağdaki uçtan uca verim, paket gecikmesi ve adalet değerleri arasında denge sağlamaktadır. Önerilen şema RLC tamponun farklı yük değerlerinde dengeli çalışmasını sağlar. Ayrıca PDCP katmanına iletilecek olan SDU'ların RLC tamponunun tıkanıklık seviyesine göre iletilmesine katkı sunmaktadır.

1.3. TEZE GENEL BAKIŞ

Hazırlanan tez çalışması altı bölüm şeklinde düzenlenmiştir. Birinci bölümde, literatür taraması ve teze genel bir giriş yapılarak tez çalışmasının amacı açıklanmıştır.

İkinci bölümde, LTE ağının genel yapısı, işleyişi ve ağ katmanları anlatılmıştır.

Üçüncü bölümde, kuyruk yönetim algoritmalarının genel yapısı ve işleyişi açıklanmıştır.

Dördüncü bölümde, ağ simülatörleri incelenmiştir. Özellikleri bakımından simülatörler karşılaştırılarak simülatör seçim süreci anlatılmıştır.

Beşinci bölümde, geliştirilen yöntemin detayları açıklanmış ve deneysel çalışma gösterilmiştir.

Altıncı bölümde, çalışma sonucunda elde edilen veriler yorumlanmış ve tartışılmıştır.

BÖLÜM 2

LTE AĞI

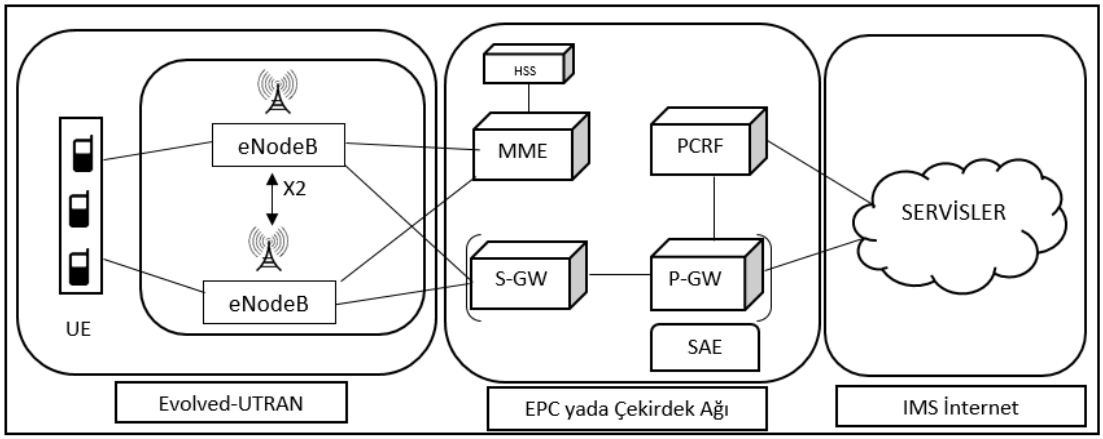
LTE ağlarının amacı, esnek bant genişliği dağıtımlarını destekleyen yüksek veri hızı, düşük gecikme süresi ve paket iletimi açısından optimize edilmiş bir radyo erişim teknolojisi sağlamaktır. Ayrıca LTE ağları yeni ağ mimarilerinin kullanımı, sorunsuz hareketlilik, yüksek QoS değerleri, minimum gecikme ve paket anahtarlamalı trafiği desteklemek amacıyla tasarlanmıştır [55].

LTE ağlarında gönderici tarafından oluşturulan paketler, eNodeB aracılığı ile hedeflere ulaştırılır. Hücresel ağ içerisindeki mobil kullanıcılar eNodeB'e hava arayüzü üzerinden bağlanmaktadır. eNodeB'nin kaynak tahsisi öncesinde mobil kullanıcıların çeşitli parametreleri göz önünde bulundurulur [56]. eNodeB'de çok fazla veri akışı varsa MAC katmanındaki çalışan zamanlayıcı yüksek metriklere sahip kullanıcılara daha fazla kaynak atamaktadır. Bu durumda düşük metrikli kullanıcıların paketleri kullanıcı kuyruklarında birikmeye başlar ve taşma meydana gelir. Paket düşüşleri, düşürülen paketlerin yeniden iletilmesi ve ağ kaynaklarının boşuna işgal edilmesine sebep olur.

Paket anahtarlamalı yaklaşımı kullanan LTE ağları, ses yoluyla iletilen paket bağlantıları dahil tüm hizmetlerin desteklenmesine izin verir. LTE ağları yalnızca iki tür düğüm yapısını kullanarak son derece basitleştirilmiş düz bir mimariye (SAE) dayanır. SAE ile LTE ağlarında eNodeB ve Mobilite Yönetimi Varlığı /Servis Ağ Geçidi (MME / SGW - Mobility Management Entity / Serving Gateway) yapıları kullanılmaya başlamıştır. Böylece 3G sistemindeki çok daha fazla ağ düğümünün kullanıldığı hiyerarşik yapıdan daha basit bir yapıya geçilmiştir [57]. LTE için spesifikasyonları Üçüncü Nesil Ortaklık Projesi (3GPP - 3rd Generation Partnership Project) tarafından üretilmektedir. Her biri kararlı ve açıkça tanımlanmış bir dizi özellik içeren sürümler şeklinde düzenlenir [58].

2.1. LTE AĞ MİMARİSİ

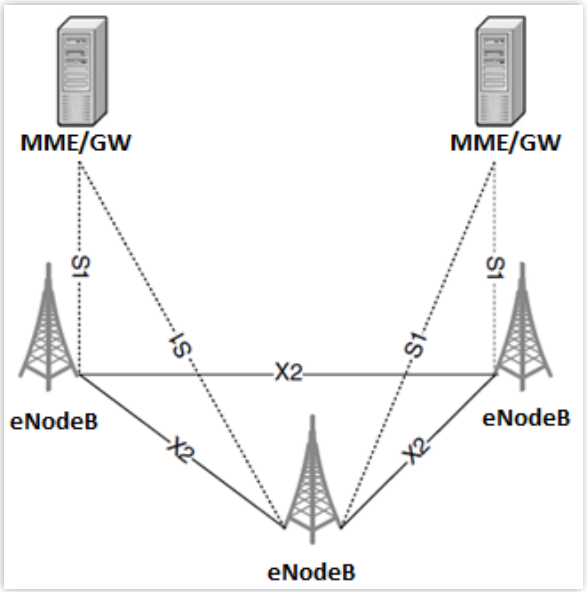
LTE ağ yapısı, SAE olarak bilinen basit ve sade mimari yapısına dayanmaktadır. LTE yüksek hızda veri ve sinyal iletimi ve sorunsuz bir hareket desteği sağlamaktadır. Şekil 2.1'de gösterildiği gibi, esas olarak çekirdek ağ yani Evrimleşmiş Paket Çekirdeği (EPC - Evolved Packet Core) ve Evrensel Karasal Radyo Erişim Ağı (E-UTRAN - Evolved Universal Terrestrial Radio Access Network) olarak adlandırılan iki bölümden oluşmaktadır [10].



Şekil 2.1. LTE ağ mimarisi [59].

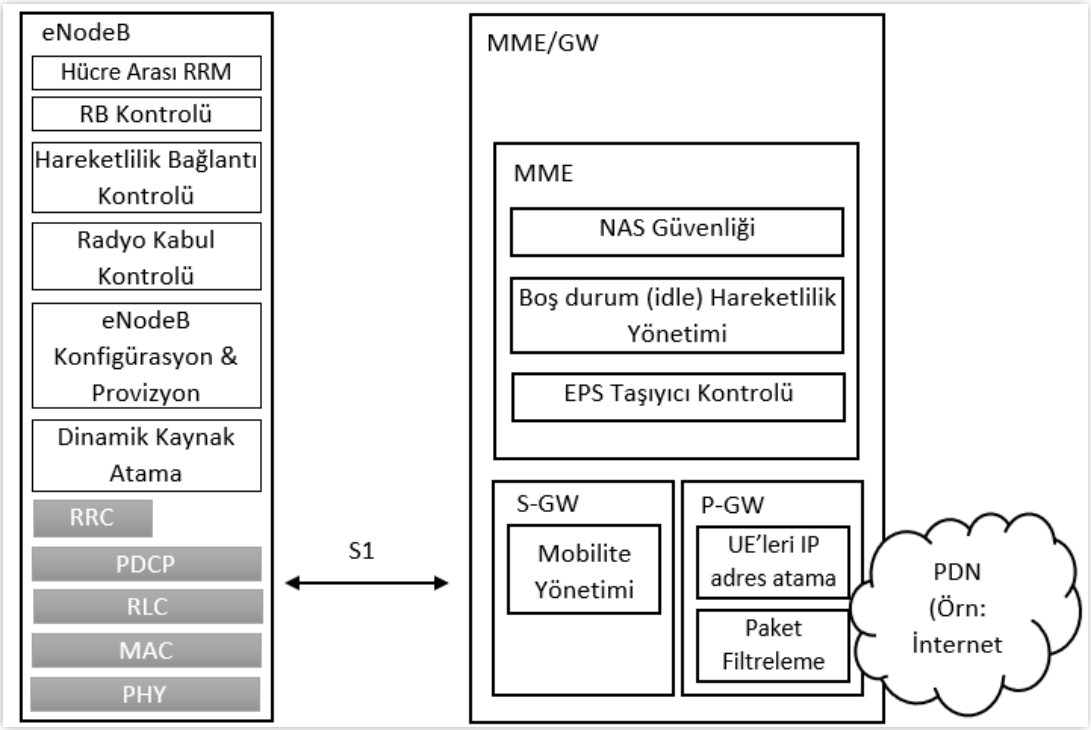
EPC yapısı IP multimedya alt sistemine (IMS - IP Multimedia Subsystem) bağlıdır. Ev Abone Sunucusu (HSS - Ev Abone Sunucusu), EPC'de veri tabanı görevi gören ana IMS veri tabanıdır. EPC'nin Mobilete Yönetim Varlığı (MME - Mobile Management Entity), Servis Ağ Geçidi (SGW – Serving Gateway) ve Paket Veri Ağ Geçidi (PGW - PDN Gateway) olmak üzere üç ana fonksiyonel birimi vardır [60]. MME arabirimi LTE ağında bağlantı kurulduktan sonraki kullanıcı hareketliliğinin takip edilmesi ve çağrı işlemlerinin kontol edilmesinden sorumludur [61]. SGW'nin ana işlevi, LTE düğümleri arasında kullanıcı veri paketlerini yönlendirmek ve iletmektir. Ayrıca LTE ve diğer 3GPP teknolojileri arasında geçişi yönetmektir [62]. PGW, LTE ağını Internet'in geri kalanıyla birbirine bağlayan bir fonksiyona sahiptir [63].

LTE erişim ağı, UE (son kullanıcı) ve eNodeB (baz istasyonu) olarak iki tür düğüm barındırır. eNodeB'ler Şekil 2.2'de gösterildiği gibi X2 arabirimi aracılığıyla birbirine bağlıdır. eNodeB'ler S1 arabirimi aracılığıyla MME/GW'ye bağlanırlar [64]. S1 arabirimi MME/GW ve eNodeB'ler arasında çoktan çoğa ilişkiyi destekler. eNodeB ve MME/GW arasındaki işlevsel ayrım Şekil 2.3'de gösterilmektedir. S-GW veri paketlerini eNodeB'ye ileten ve alan yerel bir bağlantı aracı olarak işlev görür. P-GW birimi İnternet ve IMS gibi harici Paket Veri Ağları (PDN - Packet Data Network) ile arayüz kısmını oluşturur. P-GW ayrıca adres atama, politika uygulama, paket filtreleme ve yönlendirme gibi çeşitli IP işlevlerini de gerçekleştirir [65]. MME'nin ana işlevleri çağrı, yeniden iletiminin kontrolü ve yürütülmesi, izleme alanı listesi yönetimi, dolaşım, kimlik doğrulama ve yetkilendirmedir. MME ayrıca P-GW/S-GW'nin seçimi, taşıyıcı yönetimi, güvenlik görüşmeleri ve UE'ye erişilebilirlikten sorumludur. eNodeB'nin ana işlevleri başlık sıkıştırma, şifreleme ve paketlerin güvenilir şekilde teslim edilmesidir.



Şekil 2.2. LTE ağında haberleşme [17].

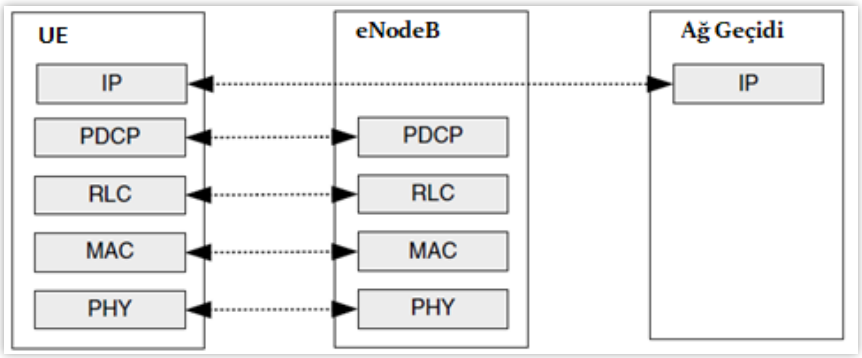
Kontrol düzleminde eNodeB, veri giriş kontrolü ve radyo kaynağı yönetimi gibi işlevleri içerir. LTE ağlarda erişim için kullanılan tek bir düğümün faydası, gecikme süresi ve işlem yükünün birden çok eNodeB'ye dağıtılmasıdır.



Şekil 2.3 eNodeB ve MME/GW arasındaki iletişim [5].

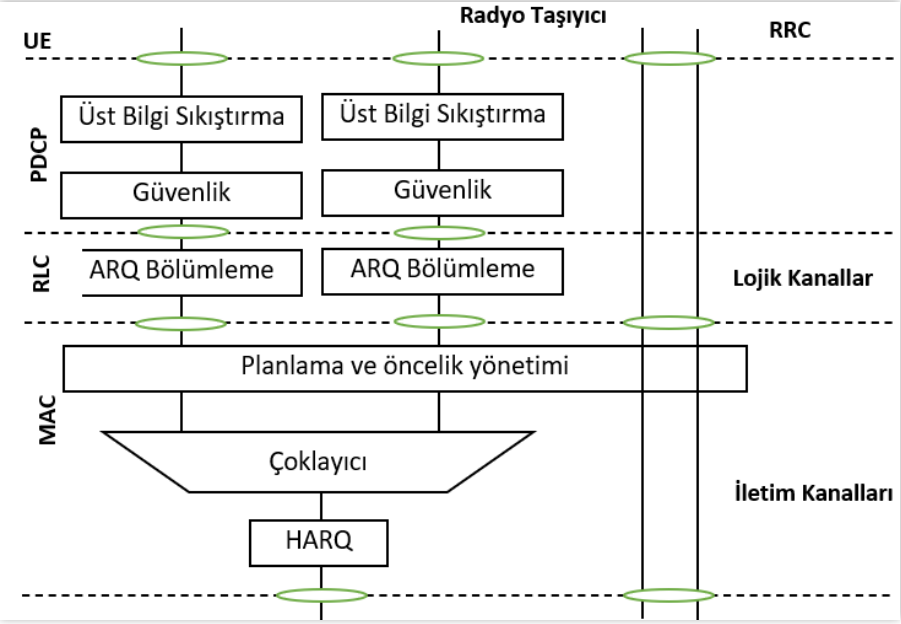
2.1.1 Kullanıcı ve Kontrol Düzlem Protokol Yapısı

Kullanıcı düzlemindeki E-UTRAN'nın protokol yığını Şekil 2.4'te verilmiştir. Kullanıcı düzlem yapısında E-UTRAN eNodeB'de sonlandırılan PDCP katmanını da yapısında bulundurur. Bu mimaride paketlerin bölümleme işlemleri SDU boyutuna, kanal durum bilgisine, programlayıcı (shedular) yapısına ve paket taşıma şekline göre yapılmaktadır [66].



Şekil 2.4. Kullanıcı düzlem protokolü [65].

Ses ve video paketleri için bölümleme (segmentation) işleminin yapılmasına izin verilmez. Böylece daha az protokol yükü oluşur. Veri paketleri için bölümleme tek tek paketlerin boyutundan ziyade aktarım tamponundaki veri miktarına göre yapılır. Bu şekilde paket başına daha az ek yük oluşturulur. Ayrıca sistemde çalışan algoritmanın verimi artar. Fiziksel katmandan sonra EUTRAN'nın çalışma performansını katman 2'nin (layer 2) mantıksal kanal akışı belirler [67]. Kullanıcı Ekipmanında katman 2'nin işleyiş yapısı Şekil 2.5'de gösterilmiştir.



Şekil 2.5. Kullanıcı ekipmanında katman 2'nin temel yapısı [62].

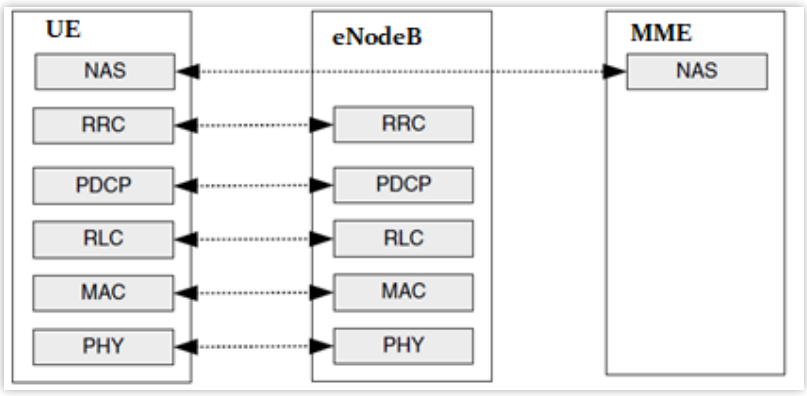
Katman 2 protokolleri bölümleme, programlama ve taşıma formatı seçiminin tüm kritik kararları için kanal durum bilgisini sağlar. İnternet Üzerinden Ses Protokolü (VoIP) olarak teslim edilen ses hizmeti için PDCP'de başlık bilgilerinin sıkıştırması iletim için gereklidir. Ses paketinin başlık bilgi yükü Gerçek Zamanlı Aktarım Protokolü (RTP), Kullanıcı Datagram Protokolü (UDP) ve IP Protokolü tarafından oluşturulan başlık bilgilerine kıyasla küçüktür. Her paket aralığında başlığın sadece dinamik kısımları iletilir.

EUTRAN'da güvenlik kontrolü PDCP katmanında da devam etmektedir. Paketlerin PDCP katmanından geçişi esnasında paketteki uzun kuyruk numaraları korunur. Böylelikle PDCP katmanında güvenlik sağlanır. Bu durum ayrıca her paketin ayrı ayrı

işlenmesine imkan sağlar. RLC katmanında paketlerin bölümleme ve pencereleme işlemleri yapılır. Ayrıca fiziksel katmandan tekrar iletilmesi gereken paketlerin yeterli olmaması durumunda SDU'ların yeniden iletilmesini sağlar.

LTE'de Dik Frekans Bölmeli Çoklu Erişim (OFDMA) ve Tek Taşıyıcılı Frekans Bölmeli Çoklu Erişim (SC-FDMA) teknikleri hem zaman hem de frekans alanları için kullanılır. OFDMA ve SC-FDMA yüksek çözünürlükle kanala erişimi mümkün kılar. MAC katmanı üst katmanlardan gelen trafik akışlarının kalite ve öncelik talepleri ile ilgilenir. Ayrıca fiziksel kaynakların kanal koşullarına göre adil ve verimli paylaşımını gerçekleştirir. EUTRAN'da yeniden iletimleri çok hızlıdır (minimum 8 ms'lik döngülerde). MAC katmanında çalışan programlayıcı (scheduler) yeniden iletimlerde tahsis edilen kaynakların sıklığını değiştirebilmektedir.

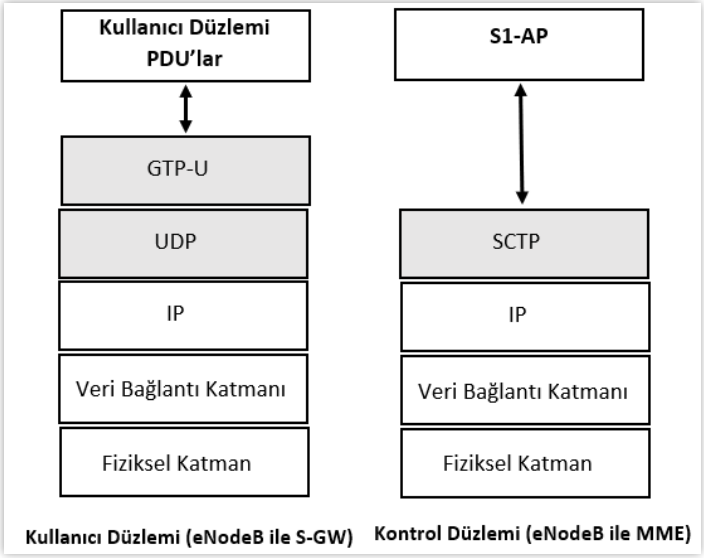
Şekil 2.6'da kontrol düzlemi protokol yığınını gösterilmektedir. Radyo Kaynak Kontrol (RRC) katmanı tarafından sistem bilgisi yayını, sayfalama, radyo taşıyıcı kontrolü, bağlantı yönetimi, mobilite işlemleri ve UE ölçüm raporlaması gibi işlemler gerçekleştirilir [68]. Erişilemeyen Tabaka (NAS) protokolü MME ve UE alanlarında sonlandırılır. NAS protokolü Gelişmiş Paket Sistemi (EPS) yönetimi, kimlik doğrulama ve güvenlik kontrolü vb. görevlerini yürütmektedir [69].



Şekil 2.6. Kontrol düzlem protokol yığını [70].

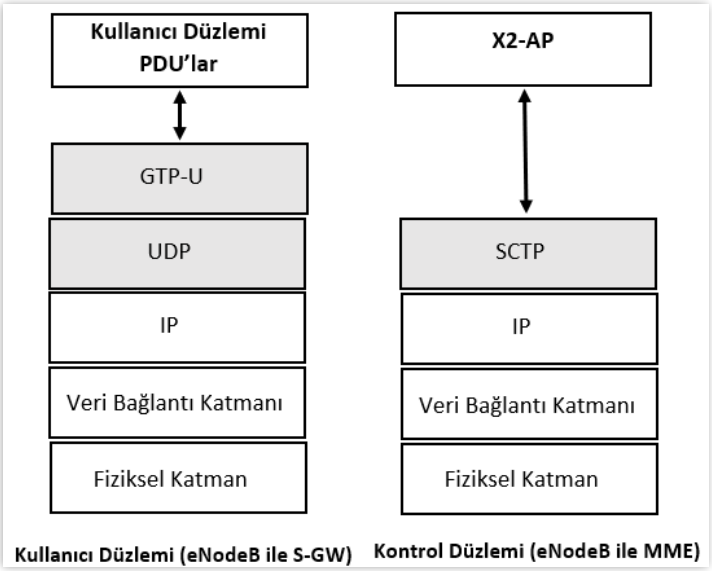
2.1.2 S1 ve X2 Arayüz Kullanıcı ve Kontrol Düzlem Protokol Yapısı

S1 ve X2 arayüz protokol yığınları sırasıyla Şekil 2.7 ve 2.8'de gösterilmektedir. S1 kullanıcı düzlemi arayüzü (S1-U) eNodeB ve S-GW arasında tanımlanır [71]. S1-U arayüzü UDP / IP aktarımı için GPRS protokolü Kullanıcı Veri Tüneli (GTP-U - GPRS Tunnelling Protocol) yapısını kullanmaktadır.



Şekil 2.7. S1 arayüzü kullanıcı ve kontrol düzlemleri.

S1-U arayüzü eNodeB ile S-GW arasında kullanıcı düzlemindeki PDU'larının garantisiz olarak teslim edilmesini sağlar [72]. GTP-U her uç nokta kümesi arasında birçok tünele izin veren basit bir IP tabanlı tünelleme protokolüdür. S1 kontrol düzlemi arayüzü (S1-MME) eNodeB ve MME arasında tanımlanmaktadır. Kullanıcı düzleminde çalışan S1-MME arayüzü IP aktarımını sağlar. Sinyal mesajlarının güvenilir bir şekilde taşınması için IP ile birlikte Akış Kontrolü Aktarım Protokolü (SCTP) kullanılır [70]. SCTP protokolü TCP'ye benzer şekilde çalışır. SCTP tıkanıklık kontrolü yaparak iletilerin güvenilir ve sıralı şekilde aktarımını sağlar. S1 uygulama protokolü (S1-AP) ve X2 uygulama protokolü (X2-AP) uygulama katmanında sinyalizasyon için kullanılır [65].



Şekil 2.8. X2 arayüzü kullanıcı ve kontrol düzlemleri.

BÖLÜM 3

KUYRUK YÖNETİM ALGORİTMALARI

Bir bilgisayar ağında çok sayıda uç sistemi ağı aynı anda kullanabilmektedir. Yönlendiriciler iletebildikleri veri miktarından daha fazla veri transferi yaptıklarında aşırı yüklenme veya ağ tıkanıklığı problemleri oluşmaktadır. Ağda yaşanan tıkanıklık paket kayıplarını ve gecikmeleri artırır. Bu paket kayıpları istenilmeyen bir durumdur. Eğer bir paket kaynağa varmadan düşürülürse kaynaklar boşuna harcanmış olur [73].

Ağdaki tıkanıklığını iyileştirmek için tıkanıklık anında tepki veren tıkanıklık kontrol mekanizmaları kullanılmaktadır. Kuyruk yönetim algoritmaları ağda yaşanan taşma problemlerini çözmek için geliştirilmiştir [21]. Kuyruk yönetiminin amacı, hangi paketlerin düşürüleceğini seçmek ve bunun ne zaman uygun olduğunu belirleyerek kuyruk uzunluğunu ve potansiyel olarak ağı meşgul eden akışları kontrol etmektir. Ağda muhtemel tıkanıklık sorunlarının çözümü için uygun algoritmanın seçimi, ağın çalışmasına doğrudan etki edebilmektedir [25].

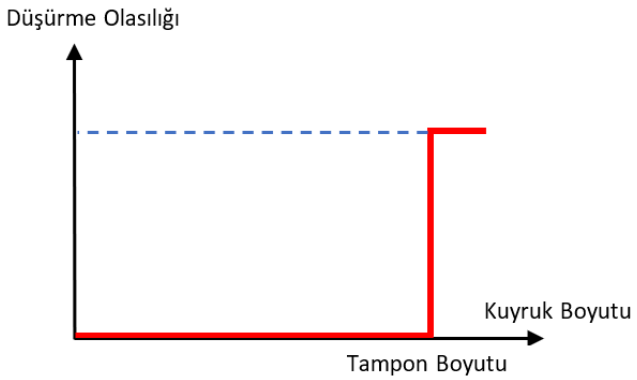
Aktif Kuyruk Yönetim Algoritmaları (AQM) ağdaki tıkanıklığı tanımlar ve göndericiyi kapalı veya açık bildirim mekanizması ile haberdar eder. AQM tekniklerinin temel hedefi, ağdaki ortalama kuyruk boyutunu küçük tutmaktır [21]. AQM'ler ortalama kuyruk boyutunu kontrol altında tutar ve paketleri düşürmeden tampondaki birikmeyi önler. Ayrıca global senkronizasyon ve TCP'de zaman aşım sorununa karşı çözüm üretirler [21]. AQM'ler göndericinin paket gönderim oranının düşürülmesini sağlayarak yönlendiricideki veri trafiğini azaltır. AQM algoritmaları, tıkanıklık başlamadan önce tıkanıklığı önlenmeyi hedefler. AQM'ler; ağdaki veri akışının hızlanmasına, kuyruk gecikmelerinin önlenmesine ve paket kayıplarının azaltılmasına katkı sağlar. Bu durum internet ve gerçek zamanlı diğer ağlardaki uygulamaların performanslarının artmasını sağlar. AQM'lerin birçoğu ağdaki yoğun

trafikte kilitlenmeyi önlemek için gelen veri paketlerinde işaretleme veya düşürme işleminde rastsallığı kullanır. Ağda tıkanıklık başladığı zaman AQM'lerin çoğu gelen paketi rastgele olarak işaretler veya düşürür. Paketlerin işaretlenme veya düşürülme olasılığı tıkanıklığın derecesine bağlıdır. Kullanılan AQM yapısına ve çalışma prensibine göre ağdaki tıkanıklık çözümlenir. AQM'lerin tasarlanması ve geliştirilmesinde;

- tıkanıklığın derecesinin belirlenmesi,
- yoğun trafikte kilitlenmenin (küresel senkronizasyon) çözülmesi,
- paket kayıp oranlarının düşürülmesi,
- kuyruk gecikmesinin azaltılması,
- paket düşüşlerinin analiz edilmesi,
- tıkanıklığın erkenden algılanması,
- muhtemel tıkanıklığın sezilmesi ve
- istenilmeyen akışların önlenmesi göz önüne alınması gereken önemli konulardandır.

3.1. DROPTAIL KUYRUK YÖNETİM ALGORİTMASI

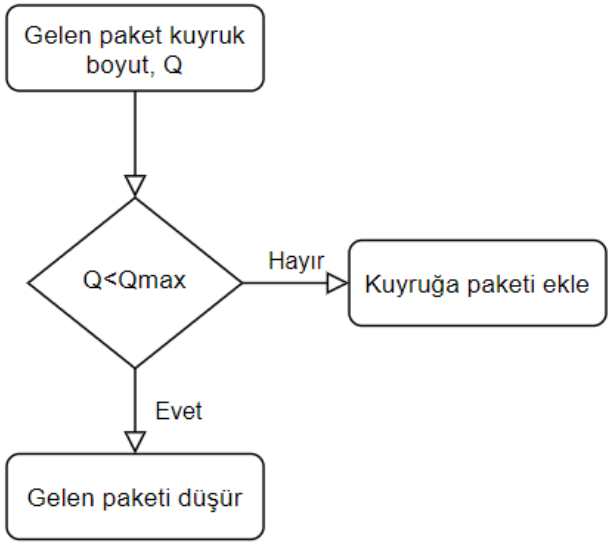
DropTail algoritmasında tampon bellek alanı maksimum sınırına ulaşıncaya kadar gelen paketler kuyruğa alınır. Maksimum kuyruk uzunluğuna erişildiğinde yeni gelen paketler kabul edilmez. Kuyrukta yeterli boşluk oluştuğunda, yeni gelen paketler kuyruğun en sonuna eklenir.



Şekil 3.1. DropTail paket düşürme yapısı.

Şekil 3.1'de DropTail'in paket düşürme yapısı gösterilmektedir. DropTail'deki temel yönetim şekli bellek dolduğu zaman kuyruktaki paketlerin düşürülmesi şeklindedir. Gelen her bir paket için kuyruk boyutu hesaplanır. Kuyruk boyutu bellekteki maksimum paket değerinden fazla ise paketler düşürülür. Eğer kuyruk boyutu maksimum değerden az ise paketler kuyruğa eklenir.

Şekil 3.2'de DropTail algoritmasının akış diyagramı gösterilmektedir. Gelen her paket için kuyruk boyutu hesaplanır (Q). Kuyruk boyutu eğer hesaplanan maksimum kuyruk boyutu (Qmax) fazla ise gelen paket düşürülür, küçük ise kuyruğa eklenir.



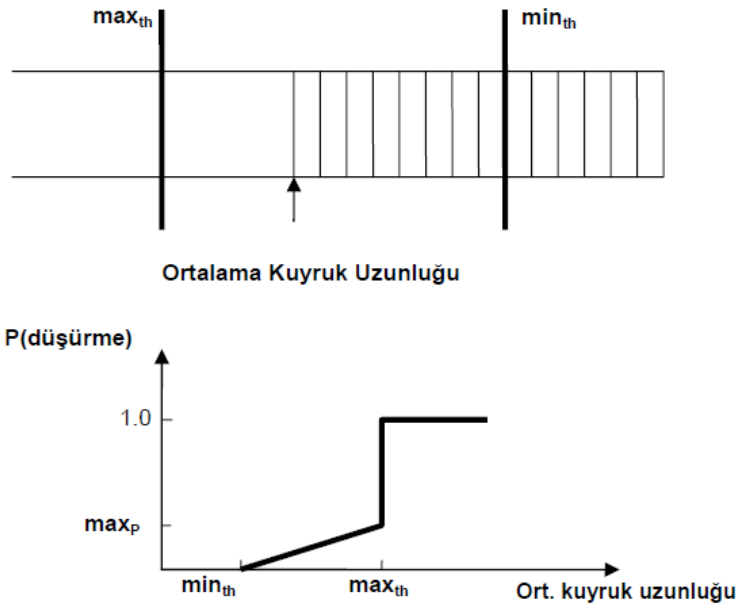
Şekil 3.2. DropTail kuyruk yönetim algoritması akış diyagramı.

3.2. RED KUYRUK YÖNETİM ALGORİTMASI

Rastgele Erken Algılama (RED), ağ yönlendiricileri üzerinde yaygın olarak kullanılan aktif kuyruk yönetim algoritmalarındandır. Yalnızca tampon dolduğunda paketleri düşüren geleneksel kuyruk yönetimi algoritmalarının aksine, RED algoritması gelen paketleri olasılıksal olarak düşürür. Düşme olasılığı, ortalama kuyruk boyutu büyüdükçe artar. RED, anlık bir kuyruk uzunluğuna değil, zamana dayalı bir kuyruk uzunluğuna göre çalışmaktadır [6].

RED üstel ağırlıklı hareketli ortalamayı kullanan alçak geçiren bir filtre ile ortalama kuyruk boyutunu hesaplar. Ortalama kuyruk boyutu minimum eşik (min_{th}) ve maksimum eşik (max_{th}) değerleri ile karşılaştırılarak hesaplanır. Ortalama kuyruk boyutu minimum eşiğin altında olduğunda, hiçbir paket işaretlenmez. Ortalama kuyruk boyutu maksimum eşikten büyük olduğunda, gelen her paket işaretlenir. Böylece ortalama kuyruk boyutunun maksimum eşiği aşmaması önlenmeye çalışılır.

Ortalama kuyruk boyutu minimum ve maksimum eşik arasında olduğunda her gelen paket, paket düşürme olasılık değeri (p_a) ile işaretlenir. p_a değeri ortalama kuyruk boyutunun bir fonksiyonudur. Bir paketin işaretlenme olasılığı, ağdaki bant genişliğinin miktarı ile orantılıdır. Şekil 3.3'de RED algoritmasının ortalama kuyruk uzunluğu değeri ile maksimum ve minimum eşik değerleri arasındaki bağlantı gösterilmektedir. RED algoritması minimum ve maksimum kuyruk boyutlarını referans alarak çalışır. RED algoritmasında ortalama kuyruk uzunluğu, maksimum eşik değerine ulaştıktan sonra gelen her bir paket belirli olasılıkla işaretlenir veya düşürülür. Ortalama kuyruk boyutu minimum değerin altındaysa paket kuyruğa eklenir.



Şekil 3.3. RED Ortalama kuyruk uzunluğu ve paket düşürme değeri.

RED, ortalama kuyruk uzunluğunu (avg) Denklem 3.1'de verilen formüle göre hesaplamaktadır.

$$avg = (1 - w_q)\, avg + w_q\, q \tag{3.1}$$

Denklemdeki q değeri, t anındaki kuyruk boyutunu göstermektedir. w_q değeri alçak geçiren filtre uygulanan kuyruk değerinin, ortalama değer üzerindeki ağırlığını göstermektedir. w_q değerinin artması *avg* değerini direkt olarak etkiler.

Ortalama kuyruk uzunluğu olan (avg), minimum ve maksimum eşik değerlerine göre hesaplanmaktadır. *avg*, minimum eşik değerinden küçük ise paketler işaretlenmez ve paket kuyruğa eklenir. *avg* maksimum eşik değerinden büyük ise paketler işaretlenir. Eğer *avg* minimum ve maksimum eşik değerinin arasında ise gelen paketler için p_a değeri p_b değerine göre hesaplanarak paketin işaretlenmesine veya düşürülmesine karar verilir. Kuyruktaki paketlerin işaretlenme olasılığı bağlantı bant genişliğinin kullanımı ile doğru orantılıdır. RED algoritmasının temel şeması aşağıdaki Çizelge 3.1'de gösterilmiştir.

Çizelge 3.1. RED kuyruk yönetimi temel şeması.

Algoritma 1:
Gelen her bir paket için;
Ortalama kuyruk uzunluğu hesapla, avg
Eğer $min_{th} <= avg < max_{th}$ *ise*
p_a *olasılığını* p_b *değeri ile hesapla ve*
gelen paketi işaretle
Eğer değilse $max_{th} < avg$
paketi işaretle

RED algoritmasında ilk ortalama kuyruk boyutunun hesaplaması boş bir kuyruğa bir paket ulaştığında gerçekleştirilir. Paket, boş bir kuyruğa ulaştıktan sonra, yönlendirici, hattın uygun olduğu süre boyunca iletilmiş olan paket sayısını hesaplar (m değeri). Hesaplama aşağıdaki Çizelge 3.2'deki gibidir.

Çizelge 3.2. RED avg değerinin hesaplanması.

avg değerinin hesaplanması:
Gelen her bir paket için;
avg ← (1 - wq)m avg; kuyruk boşken
avg ← (1 - wq)avg+ w_qq; kuyruk boş olmadığında
w_q: alçak geçiren filtre için zaman sabiti
m: kuyruğun boşta kalma süresi / iletim süresi

RED algoritmasında paketin işaretleme olasılığına karar vermek için min_{th} ve max_{th} değerleri kullanılır. Ortalama kuyruk değeri min_{th} değerinin altında ise hiçbir paket işaretlenmez ve düşürülmez. Ortalama kuyruk değeri min_{th} ile max_{th} değeri arasında ise tüm paketlerin işaretleneceği ortalama kuyruk boyutunu belirtir.

Ani bir trafik artışında min_{th}, bağlantı kullanımının kabul edilebilir bir şekilde yüksek bir seviyede tutulmasına izin vermek için büyük olmalıdır. max_{th} için elde edilmesi gereken optimum değer yönlendirici tarafından izin verilen ortalama gecikme değerine bağlıdır. $max_{th} >= 2 (min_{th})$ en iyiye yakın oranı göstermektedir.

Paket işaretleme olasılığı p_b, ortalama kuyruk boyutunun doğrusal bir işlevi olarak hesaplanır ve doğrusal olarak 0'dan p_{max}'e kadar değişir. p_b değeri Denklem 3.2'deki gibi hesaplanır.

$$P_a \leftarrow p_{max} (avg - min_{th}) / (max_{th} - min_{th}) \quad (3.2)$$

Paket işaretleme olasılığı olan p_a, işaretli paket sayısı arttıkça yavaşça artar. p_a değeri Denklem 3.3'deki gibi hesaplanır.

$$p_a \leftarrow p_b / (1– count*p_b) \quad (3.3)$$

Paket işaretleme olasılığını hesaplamak $min_{th} \leq avg < max_{th}$ olduğunda, her gelen paket için yeni bir rastgele sayı R hesaplanır. Hesaplama için $R = Random [0,1] [0,1]$ dağılımı kullanılır. Gelen paket değerleri Denklem 3.4'deki durumlara göre işaretlenir.

$$R < p_b / (1 - count * p_b) \quad (3.4)$$

Ağdaki ortalama kuyruk boyutu zamanla değiştiğinden R / p_b oranı yeniden hesaplanır. Çizelge 3.3'de RED algoritmasının işlem basamakları gösterilmektedir.

Çizelge 3.3. RED algoritması.

RED Algoritması Genel Akışı:
Gelen her bir paket için;
avg ← *0 ; count* ← *-1*
her paket ulaştığında;
ortalama kuyruk uzunluğunu hesapla, avg
Eğer *kuyruk boş değil* ***ise***
avg ← *(1- w_q)avg + w_qq ile hesaplanır*
boş ise
m ← *f (kuyruğun boşta kalma süresi / iletim süresi)*
avg ← *(1- w_q)^{m}avg*
Eğer *$min_{th} \leq avg < max_{th}$* ***ise***
count değerini artır
p_a olasılığını hesapla,
$p_b \leftarrow p_{max}(avg-min_{th})/ (max_{th}-min_{th})$
$p_a \leftarrow p_b /(1-count* p_b)$
p_a olasılığı ile gelen paketi işaretle
count ← *0*
Eğer *$max_{th} \leq avg$* ***ise***
Gelen her paketi işaretle
count ← *0*
Değilse
count ← *-1*

3.3. CoDel KUYRUK YÖNERİM ALGORİTMASI

CoDel, Van Jacobson ve Kathleen Nichols tarafından ağ programlaması için geliştirilmiştir. CoDel, arabellekte yığılma sorununu çözmeye çalışmaktadır [74].

CoDel, tamponlardan geçen paketlerin gecikme durumlarına sınırlar koyarak paketlerin yığılma sorunun çözmek için tasarlanmıştır.

CoDel geleneksel algoritmaların aksine kuyruk boyutu, kuyruk boyutu ortalamaları, kuyruk boyutu eşikleri, paket düşürme oranı ve kuyruk doluluk oranları değerleri ile ilgilenmez. CoDel'in arkasındaki teori, paket anahtarlamalı ağlarda paket davranışlarının takip edilmesine dayanmaktadır. CoDel tasarımındaki temel prensiplerden biri, gerçekten önemli olan tek bir parametrenin olmasıdır. Bu durum bir paketin kuyrukta ilerlemesi ve hedefine gönderilmesinin ne kadar süreceği ile ilgilidir. CoDel, belirlenen bir zaman aralığındaki minimum gecikme süresiyle ilgilenir. Bu minimum değer çok yüksekse, kuyruktaki paketlerin düşürülmediğini gösterir ve bu durumda arabelleğe paket alma işlemi devam eder. CoDel, her paket kuyruğa alınırken ve yerleştirilirken bir zaman damgası ekleyerek çalışır. Paket kuyruğun önüne ulaştığında, kuyrukta geçirilen süre hesaplanır [53].

CoDel'in amacı; gerektiğinde kuyrukların büyümesine izin vermek, ancak kararlı durumu makul bir seviyede tutmaya çalışmaktır. CoDel, sabit bir hedef kuyruğunun kabul edilebilir olduğunu ve arabellekte birden az bayt değeri (MTU olarak) olduğunda paketlerin düşürülmediğini varsayar. CoDel, yerel minimum kuyruk gecikme paketlerin, izleyerek gecikmeyi tanımlar. Kuyruk gecikmesi en az bir aralık değeri için hedefi aştığında bir paket düşürülür ve bir sonraki düşürme zamanını bir kontrol sistemi belirler. Bir sonraki paket düşürme süresi paket düşürme sayısının karekköküne ters orantılı olarak azaltılır. Gecikme hedefin altına düştüğünde, CoDEL kontrolörü paket düşürmeyi durdurur. Tampon MTU değerinden daha az bayt içeriyorsa, düşürme işlemi gerçekleştirilmez. CoDel algoritmasında paket ya düşme durumundadır ya da değildir. Paket bekleme süresi, belirli bir zaman aralığı boyunca hedefin üzerinde kalırsa, CoDel paket düşürme durumuna girer ve paketleri proaktif olarak düşürmeye/işaretlemeye başlar. CoDel algoritmasında paketler kuyruktan çıkartılırken değil kuyruğa eklenirken düşmektedir. İki proaktif paket düşüşü arasındaki süre Denklem 3.5'deki gibi hesaplanır.

gelen_düşüş_zamanı += zaman_aralığı/ Karekök(sayıcı) (3.5)

CoDel'in çalışma prensibi aşağıdaki Çizelge 3.4'de gösterilmiştir.

Çizelge 3.4. CoDel algoritmasının detaylı akış algoritması.

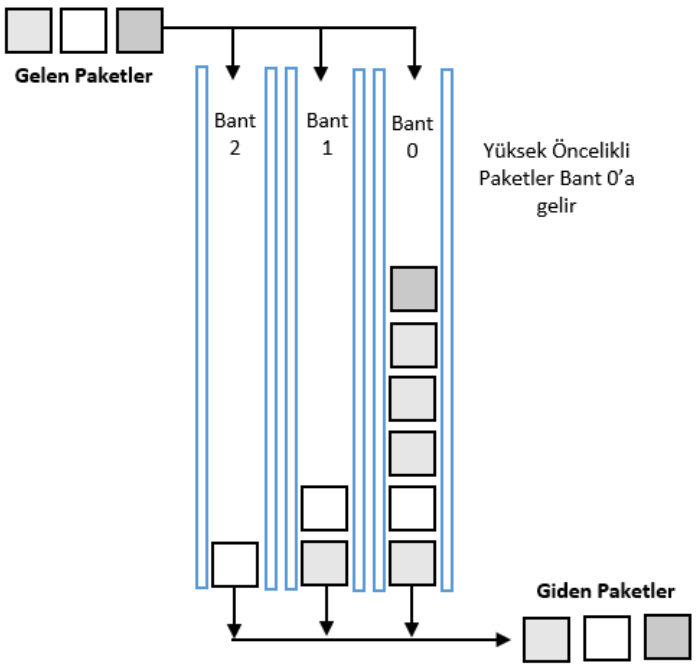
Algoritma
Kuyruğa gelen her bir paket için; ***Eğer*** *mevcut_kuyruk_boyutu < kuyruk_limit_degeri* ***ise*** *Paketi kuyruğa ekle* *Paket başlığına zaman etiketi ekle* ***Bitir*** ***Değilse*** *Paketi düşürülür* ***Bitir*** *Kuyruktan ayrılan her bir paket için;* *kuyruktan_ayrılma_zamanı = kuyruktan_ayrıldığındaki_zaman_etiketi* *geçici_kalış_süresi = kuyruktan_ayrılma_zamanı – kuyruğa_eklenme_zamanı* ***Eğer*** *paket düşme durumunda* ***ise*** ***Eğer*** *geçici_kalış_süresi < hedef_değer* ***veya*** *mevcut_kuyruk_boyutu <MTU* ***ise*** *Paketi düşürme* *Paketi düşürme durumundan çıkart* ***Bitir*** ***Değilse*** ***Her*** *kuyruktan_ayrılma_zamanı ≥ gelen_düşüş_zamanı* ***uygula*** *Paketi düşür* *Sayıcı = sayıcı+1* *gelen_düşüş_zamanı += zaman_aralığı/ Karekök(sayıcı)* ***Bitir*** ***Bitir*** ***Bitir*** ***Eğer*** *düşürme durumunun dışında ve ilk düşürülen paket* ***ise*** *Paket düşürme durumuna girer* ***Bitir***

Algoritma paketin kuyruğa alınması ve paketin kuyruktan çıkartılması şeklinde çalışır. Her paket geldiğinde, mevcut kuyruk boyutu kontrol edilir. Kuyruk limit değerinden küçükse paket kuyruğa alınır ve zaman etiketi başlığa eklenir. Bu zaman etiketi, kuyruğa alma süresini gösterir. Her paketin ayrılışında, zaman etiketi başlıktan çıkartılır. Kuyruktaki geçici kalış süresini hesaplamak için paketin kuyruğa eklenme zamanından paketin kuyruktan ayrılma zamanından çıkartılır.

3.4. pFIFO KUYRUK YÖNETİM ALGORİTMASI

Geleneksel bir İlk Giren İlk Çıkar (FIFO) algoritmasına dayanan pFIFO algoritması FIFO'ya göre bazı öncelikler sağlar. pFIFO ağ trafiğini ayırmak için üç farklı bant (ayrı FIFO) kullanmaktadır. Ağ trafiği bantlara öncelik değerlerine göre yerleştirilir. En yüksek öncelikli trafik (etkileşimli akışlar) 0 bandına yerleştirilir ve o banda her zaman en önce hizmet verilir.

pFIFO, yan yana üç FIFO kuyruğu gibidir. Burada her bir paket Hizmet Türü (ToS) bitlerine göre üç FIFO kuyruğundan birinde sıralanır. Üç bandın tümü aynı anda kuyruktan çıkarılmaz. Daha düşük numaralı bantlarda trafik akışı olduğu sürece daha yüksek numaralı bantlar hiçbir zaman kuyruktan çıkarılamaz. Belirli bir bant dolduğunda gelen ek paketler kuyruğa alınmayarak düşürülür [75].



Şekil 3.4. pFIFO algoritmasının paket akışı.

pFIFO ayrıştırılmış hizmet yapısını destekleyen diğer algoritmalar göre basit bir yönteme sahiptir. Hesaplama maliyetinin az olması ve gerçek zamanlı trafiği öncelikli olarak iletmesi avantajlı sağlamaktadır. pFIFO'daki en önemli problemlerden birisi çok sayıda yüksek öncelik akışının meydana geldiği durumlardır. Çok sayıda yüksek

öncelik trafik akışı oluşursa, düşük öncelikli trafik akışında gecikmeler yaşanmaktadır. Bu durum paket düşüşlerine neden olur [21].

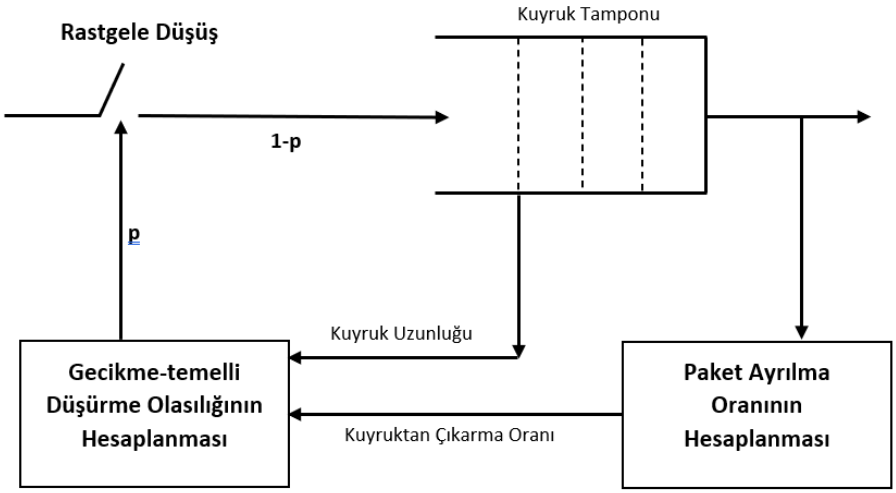
Şekil 3.4'de algoritmanın çalışma sistemi gösterilmektedir. Gelen paketler belirlenmiş kuyruk boyutundan fazla ise paketler düşürülür. Kuyruğa alınan paketler, önceliklerine göre kanallara yerleştirilir. Paketler daha sonra bant önceliğine göre kanallardan çıkartılır.

3.5. PIE KUYRUK YÖNETİM ALGORİTMASI

Geliştirilmiş Orantılı İntegral Denetleyici (PIE), kuyruktaki gecikmeyi etkili bir şekilde kontrol etmek için tasarlanmıştır. Ortalama kuyruk oranı kuyrukta bekleyen paketlere göre belirlenir. Hız değeri ağdaki mevcut gecikmeyi hesaplamak için kullanılır. Gecikme değeri periyodik olarak, düşme olasılığını hesaplamaktadır. Hesaplanan olasılık değerine göre paket düşürülür (veya işaretlenir). PIE, gecikmenin artıp azalmasına bağlı olarak olasılık değerini günceller.

α ve β değerleri, düşme olasılığının artmasını kontrol etmek için statik olarak seçilen parametrelerdir. α değeri mevcut ve hedef gecikme arasındaki sapmanın olasılığını hesaplamak için kullanılır. β değeri, gecikme eğilimini takip ederek ek ayarlamalar yapar.

Şekil 3.5'de gösterildiği gibi PIE kuyruğa alma esnasında rastgele düşürme, periyodik düşme olasılığı güncellemesi ve kuyruktan ayrılma oranının tahmini olarak üç basit bileşenden oluşmaktadır.



Şekil 3.5. PIE algoritması çalışma prensibi.

PIE, paketleri "düşme olasılığı hesaplaması" bileşeninden elde edilen bir düşme olasılığına (p) göre rastgele düşürür. Zaman damgası eklemek gibi ekstra bir adım gerekmez.

PIE algoritması, düşme olasılığını periyodik olarak günceller. Little yasasını kullanarak mevcut kuyruk gecikmesini (cur_{del}) Denklem 3.6'daki gibi tahmin eder.

$$cur_{del} = \frac{q_{len}}{avg_{drate}} \tag{3.6}$$

p düşme olasılığını Denklem 3.7'deki gibi hesaplanır.

$$p = p + \alpha * (cur_{del} - ref_{del}) + \beta * (cur_{del} - old_{del}) \tag{3.7}$$

Önceki gecikme değeri Denklem 3.8'deki gibi güncellenir.

$$cur_{del} = old_{del} \tag{3.8}$$

Kuyruğun ortalama boşaltma hızı (avg_{drate}), "ayrılma oranı tahmini" bloğundan elde edilir. Değişkenler, cur_{del} ve old_{del}, kuyruk gecikmesinin mevcut ve önceki tahminini

temsil eder. Referans gecikme değeri (ref_{del}) olarak ifade edilir. Güncelleme aralığı, T_{update} olarak belirtilir. α ve β parametreleri ölçeklendirme faktörleridir.

Düşme olasılığının hesaplanması kuyruktaki gecikmesinin tahmin edilmesine ek olarak gecikmenin uzayıp kısalmasına göre de değişmektedir. Bu değer cur_{del} ve old_{del} arasındaki fark olarak ölçülebilir. α parametresi, mevcut gecikmenin düşme olasılığını nasıl etkilediğini gösterir. β gecikmenin yukarı veya aşağı eğilimine bağlı olarak ek ayarlama değerlerini günceller. Düşme olasılığı $cur_{del} = old_{del}$ olduğunda sabitlenmektedir.

Ağdaki bir kuyruğun boşaltma hızı genellikle ya diğer kuyrukların aynı bağlantıyı paylaşması ya da bağlantı kapasitesinin dalgalanması nedeniyle değişir. Bu nedenle, PIE'de paket hareket oranını akışı Çizelge 3.5'de gösterilmiştir.

Çizelge 3.5. PIE kuyruk akışı.

Algoritma
Paket ayrıldıktan sonra; **1. Ölçüm döngüsüne kara verilir. Eğer** $q_{len} > dq_{threshold}$ **2. Yukarıdaki doğru ise ayrılma sayısını güncelle** dq_{count}*;* $dq_{count} = dq_{count} + dq_{pktsize}$*;* **3. Kalkış oranını** $dq_{count} > dq_{threshold}$ **ise sayıcıları resetle** dq_{int} *=now-start;* $dq_{rate} = \frac{dq_{count}}{dq_{int}}$*;* $avg_{drate} = (1-\varepsilon) * avg_{drate} + \varepsilon * dq_{rate}$ *start= now;* $dq_{count} = 0$

Boş kuyruklarda zaman zaman kısa ve kalıcı olmayan paket patlamaları meydana gelir, bu durum ölçümün doğruluğunu azaltır. Bu nedenle, arabellekte yeterli veri varsa kuyruk uzunluğu $dq_{threshold}$ değerinin üzerinde ise yalnızca dq_{rate} değeri hesaplanır.

PIE, kullanıcılara ani patlama durumları için kesin bir kontrol sağlar. Maksimum patlama parametresi varsayılan olarak 100 ms olarak ayarlanmıştır. Kullanıcılar

uygulama senaryolarına göre bu değeri değiştirebilirler. Patlama durumu, temel PIE tasarımına Çizelge 3.6'da gösterildiği gibi eklenmektedir.

Çizelge 3.6. PIE ani patlama durumu hesaplanması.

Algoritma
Paket ayrıldıktan sonra; **1. *Eğer*** $burst_{allow}>0$ *rastgele düşürmeyi atlayarak paketleri kuyruğa yerleştir* dq_{rate} *oranının güncellenmesi* **2. *Ani patlama değerinin güncellenmesi*** $burst_{allow} = burst_{allow} - dq_{int}$ **3. *Eğer*** $p=0$ ***ve*** cur_{del}, old_{del} ***değerinin her ikisi de*** $ref_{del}/2'den$ ***küçük ise*** $burst_{allow}$ *resetle* $burst_{allow} = max_{burst}$

PIE'de $burst_{allow}$ parametresi maksimum patlama işlemini başlatılır. $burst_{allow}$ sıfırın üzerinde olduğu sürece, gelen bir paket rastgele bırakma işlemini atlayarak kuyruğa alınacaktır. dq_{rate} oranı her güncellendiğinde, $burst_{allow}$ değeri dq_{int} tarafından azaltılır. Tanımlanan tıkanıklık ortadan kalktığında ve $p=0$ ise patlama maksimum patlama değerine (max_{burst}) eşitlenir.

3.6. KUYRUK YÖNETİM ALGORİTMALARININ KARŞILAŞTIRILMASI

Ağın yapısına uygun kuyruk yönetim algoritmasını seçmek, araştırmacılar için çok önemli bir görevdir. Yukarıda bahsedilen kuyruk yönetim algoritmaları bağlantı kullanım oranı, adalet değeri, karmaşıklık uçtan uca ortalama gecikme ve uçtan uca ortalama verim performansları açısından karşılaştırılmaktadır. Aşağıdaki çizelge 3.7'de karşılaştırma sonuçları gösterilmektedir. DropTail herhangi bir kontrol mekanizması içermemesi sebebiyle bütün başarım göstergelerinde diğer algoritmalara göre en düşük değere sahiptir. CoDel, belirlenen bir zaman aralığındaki minimum gecikme süresiyle ilgilendiği için paketleri düşüşlerini kontrol etmede DropTail, PIE ve pFIFO'ya göre daha iyi başarım değerleri elde etmektedir. pFIFO algoritması kullandığı çoklu bant yapısı ve paketleri önceliklerine göre bantlara yerleştirmesi sebebiyle gecikme sorunları yaşamaktadır. Gecikme değeri periyodik olarak, düşme olasılığını hesaplanan PIE algortiması pFIFO'ya göre uçtan uca verim değerinde daha

iyi sonuçlar elde etmektedir. Ortalama kuyruk boyutunu minimum ve maksimum kuyruk eşik değerleri arasında tutmaya çalışan RED algoritması CoDel ile beraber en iyi sonucu üreten algoritmadır. RED algoritması CoDel'e göre basit ve ağa kolay uygulanabilir bir yapıya sahiptir.

Çizelge 3.7. Kuyruk yönetim algoritmalarının karşılaştırılması.

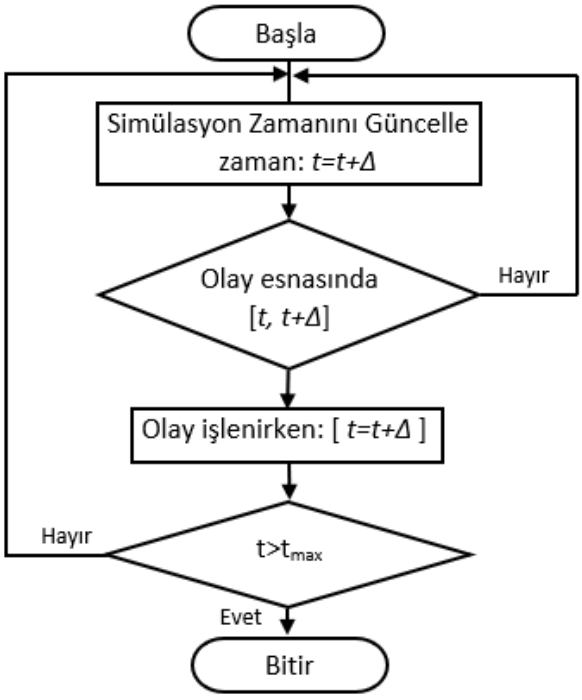
Özellik	**DropTail**	**RED**	**PIE**	**pFIFO**	**CoDel**
Bağlantı Kullanım Oranı	Düşük	Yüksek	Orta	Orta	Yüksek
Adalet İndeksi	Çok düşük	Orta	Yüksek	Orta	Yüksek
Karmaşıklık	Düşük	Düşük	Yüksek	Düşük	Yüksek
Uçtan Uca Ortalama Gecikme	Yüksek	Düşük	Orta	Orta	Düşük
Uçtan Uca Ortalama Verim	Düşük	Yüksek	Yüksek	Orta	Yüksek

Ağdaki ani paket düşüşlerinin önüne geçilmesi ve bağlantı bant genişliğinin ağdaki yükün durumuna göre kullanılması ağın performansına doğrudan etki etmektedir. Önerdiğimiz AVRED-r algoritması sanal kuyruk yapısı ile ortalama kuyruk değerini ağın durumuna göre ayarlayarak ani paket düşüşlerinin önüne geçer. Ayrıca ağdaki yükün durumuna göre ağdaki paket düşürme olasılığını kontrol ederek bağlantı bant genişliğinin tam olarak kullanılmasına katkı sağlamaktadır.

BÖLÜM 4

AĞ SİMÜLATÖRLERİ

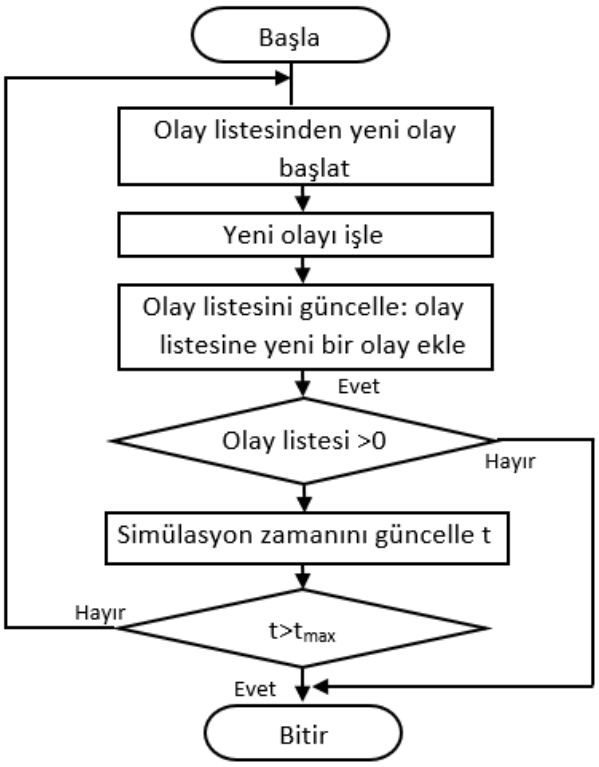
Ağ tasarım süreci kullanıcı performans beklentilerinin maliyet ve kapasite arasında dengelemesini gerektiren zorlu bir iştir. Ağdaki karmaşıklıkla başa çıkmak için açık yaklaşımlardan biri, modelleme ve simülasyon tekniklerinin kullanılmasıdır. Ağ Simülatörleri, değişen ağ koşulları altında ağ algoritmalarını ve yönlendirme protokollerini tasarlamak ve değerlendirmek için kullanılır [76]. Ağ simülasyonu yazılım programları ile ağdaki hub, anahtar, yönlendirici, erişim noktaları, bağlantılar vb. farklı ağ cihazları arasındaki ilişki hesaplanarak ağın davranışı analiz edilmektedir. Gerçekte donanımsal, maddi veya diğer kısıtlar nedeniyle oluşturulması imkânsız olabilecek ağ senaryoları ağ simülasyon araçları ile kolaylıkla oluşturulabilir. Ağ simülatörleri görselleştirme ve animasyon imkânı sağlar. Simülatörler ağı tasarlamak ve ağ performansını izlemek için maliyeti düşük bir yöntemdir.



Şekil 4.1. Zaman çizelgeli simülasyon akış diyagramı.

Ağ simülasyonları, zaman çizelgeli simülasyon ve ayrık olay simülasyonu olarak kategorize edilebilir. Zaman çizelgeli yapıda simülasyon, zaman aralıklarının yinelemeli ilerlemesi yoluyla hazırlanır. Yinelenen zaman dilimi içindeki olaylar, simülasyon ilerlerken yürütülür. Zaman çizelgeli simülasyonun akış çizelgesi Şekil 4.1'de gösterilmektedir.

Ayrık olay simülasyonunda simülasyon, planlanan sonraki olayın yürütülmesi ile ilerler. Simülasyon zamanı, bir sonraki planlanan olay yürütüldükten sonra güncellenir. Ayrık olay simülasyonunun akış şeması Şekil 4.2'de gösterilmektedir.

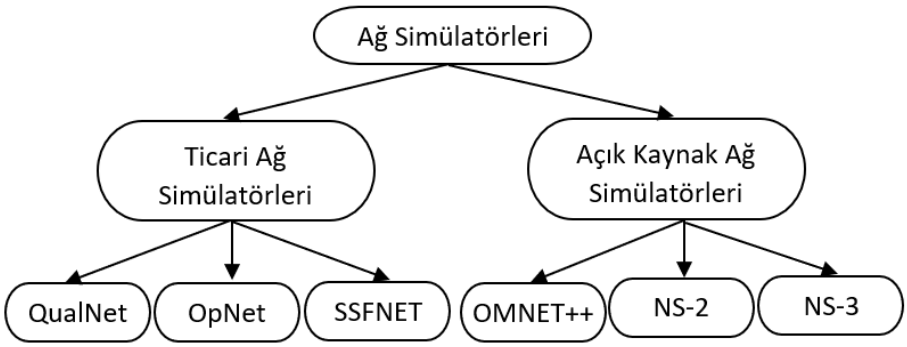


Şekil 4.2. Ayrık olay simülasyonu sınıflandırması.

Zamana dayalı simülasyondaki zaman dilimleri, gerçek dünyadaki saat zamanını ifade etmektedir. Simülasyon süresi, modelin çalıştırılmasında kullanılan süreyi tanımlar. Bu nedenle simülasyon süresi ve bir simülasyonun çalıştırmasında geçen süre iki farklı kavramdır. Ayrık olay simülasyonu ile karşılaştırıldığında, zaman çizelgeli simülasyon, belirli bir zaman aralığında olaylar olup olmadığına bakılmaksızın tüm zaman aralıklarını yineleyecektir. Ayrık olay simülasyonu, zaman çizelgeli

simülasyonda ortaya çıkan verimsizliği önlemek için yalnızca sıralı bir şekilde işlenmesi gereken planlanmış olayları yineler. Bu nedenle, çoğu modern simülatör, ayrık olay simülasyonu destekler.

Ticari ve açık kaynak olmak üzere iki tür ağ simülatörü mevcuttur. Açık kaynak simülasyon araçları, resmî web sitelerinden ücretsiz olarak indirilebilir. Ticari simülatörlerin lisansa ihtiyacı vardır ve tüm özelliklerinin kullanılması için satın alınması gerekmektedir. Şekil 4.3'de ağ simülatör örnekleri gösterilmektedir.



Şekil 4.3. Ağ simülatör çeşitleri.

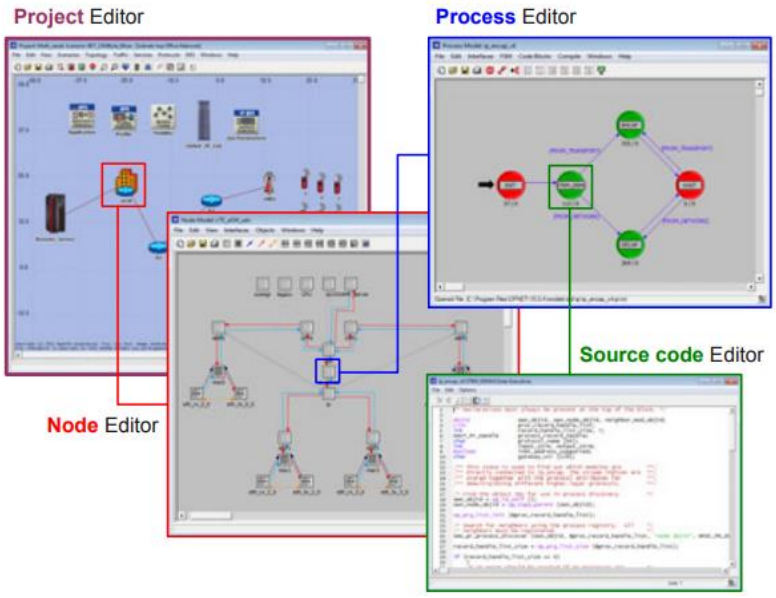
Bu tez çalışmasında Ns-2, Ns-3, OMNeT++, OPNET ve QualNet gibi çeşitli simülatörleri incelenmiştir. Yukarıda bahsedilen her bir simülatör için, kısa giriş, avantajlar, dezavantajlar, arka uç ortamı, destekleyici işletim sistemi, minimum donanım gereksinimi ve diğer simülatör ile karşılaştırma tartışılmaktadır. Dolayısıyla, yukarıda belirtilen bilgi ve analize dayalı olarak, ihtiyaçlara göre uygun bir ağ simülatörü seçimi yapılabilir.

4.1. OpNET AĞ SİMÜLATÖRÜ

OpNET (Optimize Edilmiş Ağ Mühendisliği Araçları), ilk olarak 1986 yılında MIT tarafından önerilen ve C ++ ile yazılmış bir ayrık olay ağ simülatörüdür [77]. OpNET, çeşitli ağ ve uygulama performans yönetimi çözümleri sağlayan ticari bir simülasyon aracıdır. Araştırmacılar, mühendisler, üniversite öğrencileri ve ABD ordusu tarafından yaygın olarak kullanılmaktadır. OpNET nesne yönelimli ve hiyerarşik modelleme, hata ayıklama ve analiz ile desteklenen grafik kullanıcı arayüzüne (GUI) sahip

dinamik bir ayrık olay simülatörüdür. Askeri amaç için geliştirilse de daha sonra önde gelen ticari ağ simülasyon araçlarından biri haline gelmiştir. Ticari amaç için pahalı olsa da eğitim amaçlı kullanım için ücretsiz lisanslar vermektedir. OpNET simülatörü model editörleri, model kitaplığı, analiz araçları ve animasyon görüntüleyici gibi çeşitli araçlara sahiptir. OpNET'de modelleme esas olarak ağ alanı, düğüm alanı ve işlem alanı olmak üzere üç seviyede yapılmaktadır.

Ayrık olay simülatörü OpNET hibrit, analitik, 32-bit ve 64-bit tam paralel simülasyonları desteklemektedir. Dağıtılmış simülasyonlar için grid hesaplama desteğine sahiptir. System-in-the-Loop arayüzü ile gerçek dünya verilerini ve bilgilerini simülasyon ortamına taşıyabilmektedir. Harici nesne dosyalarını, kitaplıkları ve diğer simülatörleri entegre etmek için açık bir arayüz sağlar. Geniş bir protokol ve teknoloji paketi içerir ve çok çeşitli ağ türlerinin ve teknolojilerinin modellenmesini sağlamak için bir geliştirme ortamı içerir. OpNET simülatöründe standart tabanlı yerel alan ağı (LAN), geniş alan ağı (WAN), hiyerarşik ağlar, mobil ağlar, sensör ağları, uydu ağları, protokollerin ön araştırması ve iletişim ağı mimarisi yapıları modellenebilmektedir. OpNET simülatörü LTE simülasyon modelini tasarlamak ve uygulamak içinde kullanılmaktadır. Şekil 4.4'de OpNET'in proje editörü, işlemci editörü, düğüm editörü ve kaynak kod editörü gösterilmektedir.



Şekil 4.4. OpNET simülatör arayüzü.

OpNET simülatörü kullanıcı dostu grafik arayüz sağlar. Dağıtılmış simülasyon için grid hesaplamaya imkân verir. Çok çeşitli iletişim ağlarının yüksek doğruluk modellemesini, simülasyonunu, analizini ve tasarımını destekler. Tüm heterojen ağların çeşitli protokollerle simülasyonuna izin verir. Yazılımın kullanımını kolaylaştırmak için ek modüller ve araçlar sağlar. OpNET'in dezavantajlı yönleri ise ticari amaç için pahalı olması, sınırlı kablosuz mobilite sağlaması, az sayıda protokolleri desteklemesi ve enerji modeli eksikliğidir.

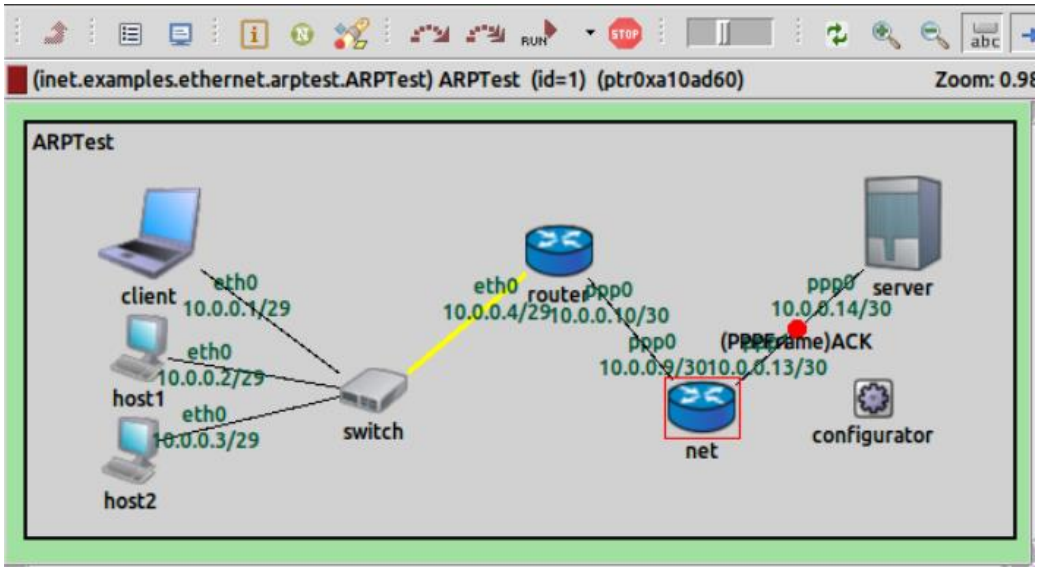
OpNET simülatörü ile ilgili yapılan literatür çalışmaları incelenmiştir. [78]'de OpNET ile LTE düğüm modeli oluşturulmuş ve MAC katmanındaki (schedular) algoritmalarının karşılaştırılması yapılmıştır. [79]'de oluşturulan LTE ağında GBR ve GBR olmayan veri akışları test edilmiştir. [80]'de kablosuz ağlarda performans karşılaştırılması yapılmıştır. [81]'de internet tabanlı kablosuz altyapı ağlarında devir teslimleri (handover) sistemi oluşturulmuştur. [82]'de kablosuz sensör ağı olan Zigbee için yönlendirme simülasyonu yapılmıştır. [83]'de OpNET modelleyici ile aktif kuyruk yönetimi algoritmalarının karşılaştırılması yapılmıştır.

4.2. OMNET++ AĞ SİMÜLATÖRÜ

OMNeT++; genişletilebilir, modüler, bileşen tabanlı ağ simülatörüdür. OMNeT++ simülasyon kitaplığı, entegre geliştirme ve grafik çalıştırma ortamı içerir [84]. "Açık Modüler Ağ Test Yatağı" anlamına gelen OMNeT++ ayrık olay simülatörlerinin iyi bir örneğidir. OMNeT++ Eclipse tabanlı bir Entegre Geliştirme Ortamı (IDE), model yapılandırması, veri analizi ve görselleştirme için ek araçlar sağlar.

OMNeT++ geleneksel TCP / IP ağı protokolleri, eşler arası paylaşımlı ağlar (ör. BitTorrent), hücresel ağlar, uydu ağları, Mobil Ad Hoc NETworks (MANET'ler), paralel / dağıtılmış depolama ağları vb. ağlar için simülasyon ortamı sağlar. OMNeT++ simülatörü gerçek zamanlı simülasyon, ağ emülasyonu, alternatif programlama dilleri desteği (Java, C #), veritabanı entegrasyonu, SystemC entegrasyonu vd. birçok işlev için uzantılara sahiptir. OMNeT++ iletişim ağlarının ve diğer dağıtılmış sistemlerin simülasyonu için tasarlanmıştır.

OMNeT++ simülasyon oluşturmak için gerekli temel yapı ve araçları içermesine rağmen herhangi bir bileşen sunmaz. Uygulama alanları INET Framework veya Castalia gibi çeşitli simülasyon modelleri ve çerçeveleri tarafından desteklenir. OMNeT++ 'de model çerçeveleri simülasyon çerçevesinden tamamen bağımsız olarak geliştirilir. Aşağıdaki Şekil 4.5'de OMNET++ arayüzü gösterilmektedir.



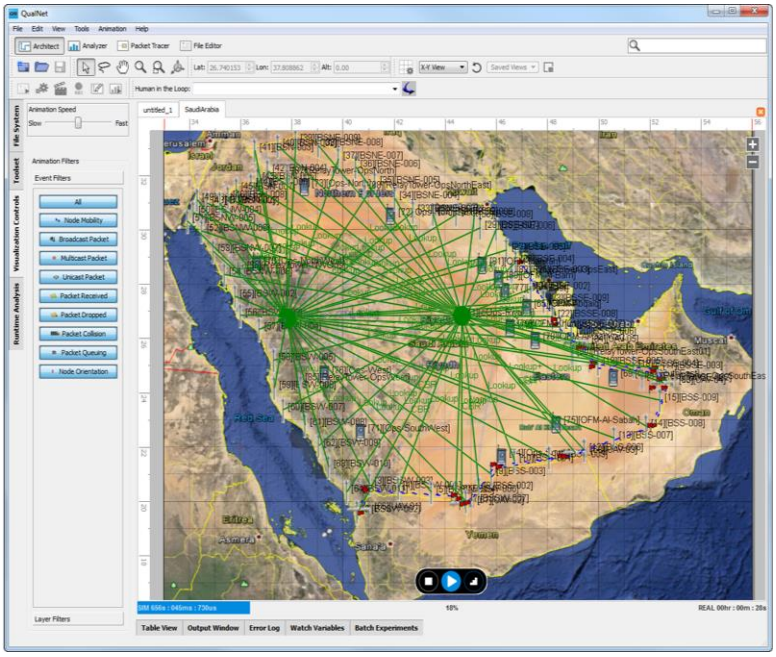
Şekil 4.5. OMNeT++ simülatör arayüzü.

Literatür incelemesinde OMNeT++ kullanılarak geliştirilen birçok akademik araştırma bulunmaktadır. [85]'da OMNeT++ ile IEEE 802.15.4'ün simülasyon modeli oluşturulmuştur. [86]'de SimuLATE isminde OMNeT++ tabanlı LTE / LTE-A ağları için modüler bir sistem geliştirilmiştir. [87]'de BitTorrent tabanlı yapı için modül geliştirilmiştir. [88]'de kablosuz ağlarda pil tüketimini simüle etmek için genişletilebilir bir ağ yapısı tasarlanmıştır. [89]'de OMNeT++ kullanarak DDoS saldırısının modellenmesi ve simülasyonu yapılmıştır. [90]'de nesnelerin internet (IoT) için hibrit bir yöntem geliştirilmiştir.

4.3. QualNeT AĞ SİMÜLATÖRÜ

QualNet ağ simülasyon yazılımı gerçek iletişim ağlarının davranışını taklit eden bir planlama, test etme ve eğitim aracıdır [91]. QualNet protokolleri tasarlamak, ağ senaryolarını oluşturmak ve ağ performanslarını analiz etmek için kapsamlı bir ortam

sağlar. QualNet ağ simülasyon aracı ticari, askeri ve devlet kurumları için simülasyon çözümleri üretmektedir [92]. QualNet, donanım ve paralel hesaplama tekniklerinden yararlanarak çok sayıda düğümü modelleyebilir. QualNet, yüksek doğrulukta büyük ağları modellemek için çok çekirdekli ve çok işlemcili sistemler üzerinde çalışabilir. Ağ modelinin değerini artırmak için gerçek ağlar ve üçüncü taraf görselleştirme yazılımı gibi diğer donanım ve yazılım uygulamalarına bağlanabilir [93]. Aşağıdaki Şekil 4.6'de QualNet arayüzü gösterilmektedir.



Şekil 4.6. QualNet simülatör arayüzü.

Tasarım Modu, sürükle ve bırak grafik kullanıcı arayüzü aracılığıyla sanal ağ modelleri oluşturmak için kullanılır. Çeşitli ağ öğelerini ve uç noktaları temsil eden düğümler bir tuval üzerine yerleştirilir. Her düğüm, kablolu veya kablosuz ağ arayüzleri ve iletişim özellikleri ile yapılandırılır. Görselleştirme Modu, kullanıcıya Tasarım Modunda oluşturulan bir ağ senaryosunun derinlemesine görselleştirilmesi ve analizi için imkân sağlamaktadır. Simülasyonlar çalışırken, kullanıcılar ağ üzerinden akan çeşitli katmanlardaki paketleri izleyebilir ve kritik performans ölçümlerinin dinamik grafiklerini görüntüleyebilir. QualNet platformu, Geliştirici Modeli Kitaplığı, Multimedya ve Kurumsal Model Kitaplığı, Kablosuz Model Kitaplığının dahil olduğu üç bileşen modeli kitaplığı kullanarak simülasyonları üretir [91]. Çok yönlü ve hızlı çalışmasına rağmen, QualNet, açık kaynaklı bir simülatör değildir. Bazı dosyaların

kaynak kodunu gizler. Bu nedenle, bu tür dosyalar için değişiklik yapılamamaktadır. QualNet, ticari olduğu için ücretli ve diğer ticari simülatörlere göre pahalıdır.

Literatür incelemesinde QualNet simülatörü kullanılarak geliştirilen bir çok akademik araştırma bulunmaktadır. [94]'de ağdaki Dağıtılmış Hizmet Engelleme (DDOS) saldırılarının tespitinde Qualnet 5.2 simülatörü kullanılmıştır. [95]'de Mobil Ad hoc NETwork (MANET) için çok yollu yönlendirme protokolleri, ölçeklenebilirlik, güvenlik (gizlilik ve bütünlük), ağların ömrü, kablosuz iletimlerin kararsızlığı ve bunların uygulamalara uyarlanması gerçekleştirilirmiştir. [96]'da kablosuz ağ yapısında kullanılan Çarpışmaya Dirençli Çoklu Erişim (CRMA) adı verilen yeni bir spektrum paylaşım protokolü çalışması test edilmiştir. [97]'da iç mekanda mobil nesne takibi için kullanılan RFID için MAC katmanında, biri Çerçeveli Yuvalı Aloha (rastgele erişim) ve diğeri ise temel algılama bileşeni olan Ağaç Yürüyüşü (sorgulama) olmak üzere geliştirilen iki protokolün tasarımı gerçekleştirilmiştir.

4.4. Ns-3 AĞ SİMÜLATÖRÜ

Ns-3, öncelikle araştırma ve eğitim amaçlı kullanım için hedeflenen, internet sistemleri için geliştirilen ayrık olay tabanlı bir ağ simülatörüdür. Ns-3 ücretsiz bir yazılımdır, GNU GPLv2 lisansı altında lisanslanmıştır ve araştırma, geliştirme ve kullanım için halka açıktır [98]. Ns-3 projesinin amacı, modern ağ araştırmalarının simülasyon ihtiyaçlarını karşılamak için herkese açık bir simülasyon ortamı geliştirmektir.

Ns-2 [99], uzun yıllar boyunca ağ protokolleri ve iletişim yöntemlerine yönelik akademik araştırmalar için kullanıldı. Ns-2 ortamı ile ilgili çok sayıda akademik araştırma, raporlama sonuçları ve yüzlerce yeni model yazıldı. Ns-3 simülatörü ns-2 simülatöründeki eksik yönleri tamamlamak için geliştirilen ns-2'den üretilmeyen ayrı bir simülatör aracıdır. Ns-2, OTcl ile yazılmışken, Ns-3 C++ ve Phyton programlama dilleri ile yazılmıştır. Ns-3'deki protokol ortamları gerçek dünyaya yakın bir yapıda tasarlanmıştır.

Ns-3 dokümantasyonu geniş, kullanımı ve hata ayıklaması kolay bir yapıya sahiptir. Ayrıca simülasyon yapılandırması, izleme, toplama ve analiz gibi tüm simülasyon iş akışının gereksinimlerini karşılayan simülasyon çekirdeğine sahiptir [100].

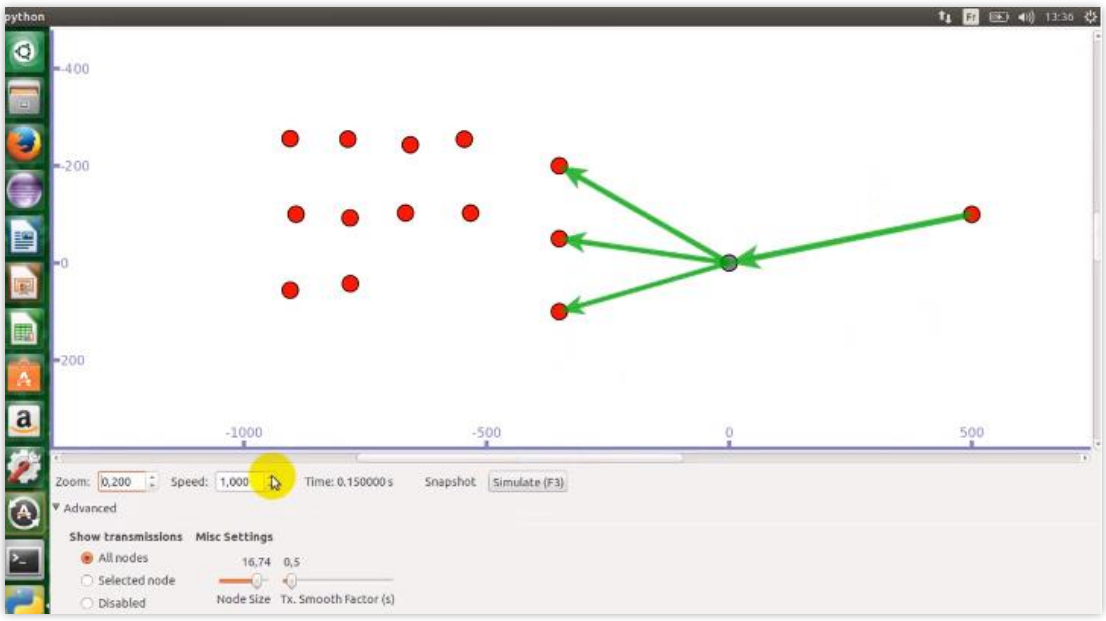
Ayrıca, Ns-3 yazılım altyapısı, ns-3'ün gerçek zamanlı bir ağ benzetim ortamı olarak kullanılmasına izin vererek gerçekçi simülasyon modellerinin geliştirilmesini teşvik etmektedir. Gerçek dünya ile birbirine bağlanan ve mevcut birçok gerçek dünya protokol uygulamasının geliştiriciler tarafından yeniden kullanılmasına izin vermektedir.

Ns-3 simülasyon çekirdeği hem IP hem de IP tabanlı olmayan ağlarda araştırmayı destekler. Ns-3 kullanıcılarının büyük çoğunluğu Wi-Fi, WiMAX, LTE, OLSR ve AODV gibi çeşitli statik veya dinamik yönlendirme protokollerini içeren simülasyonları gerçekleştirebilmektedir [101].

Ns-3 ayrıca gerçek sistemlerle etkileşim için "döngü içinde simülasyon" yapısını kullanan gerçek zamanlı bir simülatördür. Örneğin, kullanıcılar gerçek ağ cihazlarında Ns-3 tarafından üretilen paketleri gönderebilir ve alabilir ve ns-3, sanal makineler arasında bağlantı eklemek için bir ara bağlantı çerçevesi görevini gerçekleştirebilir.

Ns-3'de C tabanlı uygulama kodunu kolaylaştırdığı için programlama dili olarak C++ seçilmiştir. Ns-3 ayrıca Linux bilgisayarlar için tasarlanmış olup, dahili arayüzler (soketler) ve bilgisayarların çalışma mantığına iyi bir şekilde uyum sağlayan arayüzler ile tasarlanmıştır. Ns-3, simülasyondan deneye geçerken olası kesiklikleri azaltmak amacıyla gerçek aygıtlar ve uygulamalarla kullanılabilen benzetim yeteneklerine sahiptir.

LTE-EPC Network SimulAtor (LENA), Ns-3'te LTE ağ modeli oluşturabilmek için geliştirilmiş modül yapısıdır. Kullanıcı ekipmanlar (UE) ile eNodeB düğümlerinin tanımlamak için LTE Model kullanılırken, MME, SGW ve PGW ağ yapıları için EPC Modeli kullanılmaktadır. Şekil 4.7'de Ns-3 simülasyonuna ait bir ağ yapısının Phyton ile görselleştirmiş yapısı bulunmaktadır.



Şekil 4.7. Ns-3 simülatör arayüzü.

Ns-3 ile yapılan çok sayıda akademik çalışma bulunmaktadır. [102]'de Extreme Test Modülü kullanılarak ns3 simülatöründe IEEE 802.11 MAC modelinin doğrulanması sağlanmıştır. [103]'de IoT'ler için LTE ağ yapısında performans analizi yapılmıştır. [104]'de kablosuz sensör ağlarında QoS'a dayalı enerji verimli test edilmiştir. [105]'de Ns-3 simülatöründe LTE ağları üzerinden gerçek zamanlı video akışının performans göstergeleri incelenmiştir. [106]'de VANET ağında OLSR ve DSDV protokollerinin performansı Ns-3 simülatöründe değerlendirilmiştir. [107]'de P2P ağlarında DDOS saldırılarının simülasyonu Ns-3 üzerinden yapılmıştır. [108]'de OFDM hata oranı modelinin doğrulanması için bir simülasyon ortamı hazırlanmıştır. [109]'de Ns-3 simülatörü için yeni nesil TCP tasarımları yapılmıştır. [110]'de akıllı ortamlar (ev, araç) için kullanılan kablosuz ağlar için dinamik spektrum erişiminin simülasyonu yapılmıştır.

4.5. AĞ SİMÜLATÖRLERİNİN KARŞILAŞTIRILMASI VE SİMÜLATÖR SEÇİMİ

Uygun ağ simülatörünü seçmek, araştırmacılar için çok önemli bir görevdir. Ağ protokollerinin kesin performansını değerlendirmek için farklı ağ parametreleri [111] ile farklı simülatörlerde farklı yönlendirme protokollerini [76] test eden araştırmalar

yapılmıştır. Simülasyon yazılımları ağ modeli, performans, dokümantasyon vb. birçok parametreye göre karşılaştırılmaktadır. Aşağıdaki çizelge 4.1'de ilgili simülatörlerin performans karşılaştırılması yapılmıştır.

Çizelge 4.1. Ağ simülatörlerinin AODV ağı performansının karşılaştırılması.

Özellik	**Ns-2**	**Ns-3**	**OMNET++**	**OPNET**	**QualNet**
Hafıza Kullanımı	Yüksek	Düşük	Orta	Yüksek	Yüksek
CPU Kullanımı	Yüksek	Orta	Düşük	Düşük	Düşük
Hesaplama Zamanı	Orta	Çok düşük	Düşük	Düşük	Çok düşük

Ağ ortamında izlenen ortam ve özellikleri simülatörlerin çalışma yapılarına göre performansa direk olarak etki etmektedir. Bir ağ yapısının tüm detaylarını modellemek çoğu zaman pek mümkün olamamaktadır. Küçük bir detay parametresi ağ performansı üzerinde ciddi etkilere sebep olabilmektedir.

Çizelge 4.2'de ilgili simülatörlerin genel özelliklerinin karşılaştırılması yapılmıştır. Simülatörlerin detay yapıları performansı etkileyen önemli parametrelerdendir. Ticari bir yazılım olan QualNet en yüksek detay seviyesine sahiptir. OPNET'de yüksek detay seviyesine sahipken, Ns-3 ve OMNET++ daha düşük detay seviyelerine sahiptir.

İncelenen literatür ve performans göstergelerine göre QualNet ve OPNET iyi performans göstermelerine rağmen ticari ve endüstriyel ihtiyaçların karşılanmasına yöneliktir. OMNET++ güçlü grafik arayüzüne rağmen dış model oluşturma ve kullanıcı tabanının bulunmaması zayıf yanlarıdır.

Akademik araştırmalarda en çok tercih edilen Ns-3 simülatörü karmaşık mimari yapısıyla dezavantaja sahipken detay seviyede ağ yapısının oluşturulması, yönetilmesi ve görsel simülasyon sonuçlarının etkili bir şekilde alınmasıyla avantajlı hale gelmektedir. Ayrıca çok sayıda kullanıcı grubu ve geliştiriciye sahip olması sebebiyle karşılaşılan sorunlar için çok sayıda kaynak bulunmaktadır. Bu tez çalışmasında açık kaynak erişimi olması, geliştirmeye açık yapısı ve LTE ağı için özel bir modüle sahip olması sebebiyle Ns-3 simülatör aracı seçilmiştir.

Çizelge 4.2. Ağ simülatörlerinin karşılaştırılması.

Araçlar	**Ns-2**	**Ns-3**	**OMNET++**	**OPNET**	**QualNet**
Amaç	Araştırma ve eğitim	Araştırma ve eğitim	Araştırma ve eğitim	Ticari	Ticari
Arayüz	C++/OTcl	C++/Python	C++/NED	C/C++	Parsec/C temelli
Paralellik	Yok	Yok	MPI/PVM	Düşük	SMP/Beowulf
Lisans	Açık kaynak	Açık kaynak	Ticari ve akademik	Ticari	Ticari
Kullanım Kolaylığı	Kolay	Orta	Orta	Orta	Çok kolay
Platform	Linux, Windows (Cygwin)	Linux, Windows (Cygwin)	Linux, Windows, Macos	Windows	Linux
Ağ Protokolü	TCP/IP Kablolu ve Kablosuz ağlar	TCP/IP Kablolu ve Kablosuz ağlar, Geni alan ağları	Kablolu ve kablosuz ağlar	ATM, TCP, FDDI, IP, Ethernet, Frame Relay, 802.11 ve wireless	Kablolu ve kablosuz ağlar, Geniş alan ağları
Ölçeklenebilirlik	Sınırlı	Sınırlı	Geniş	Orta	Geniş

BÖLÜM 5

HÜCRESEL AĞ İÇİN AKTİF KUYRUK YÖNETİM ALGORİTMASININ TASARIMI

Hücresel LTE ağlarında yaşanan tıkanıklık ve darboğaz problemlerinin çözümü için RLC katmanında çalışan aktif kuyruk yönetim algoritmasının sistem verimliliğinin artırılması gerekmektedir. Ayrıca RLC tamponunun farklı yük değerlerinde dengeli çalışması sağlanmalıdır.

LTE ağlarında uçtan uca daha yüksek verim ve daha düşük gecikme değerlerinin sağlanması ve tıkanıklık sorunlarının çözümü için RED tabanlı yeni bir AQM olan Adaptive Virtual RED (AVRED-r) algoritmasını öneriyoruz. İki adımda gerçekleştirilen AVRED-r algoritması, ilk adımda kuyruk doluluğunu kontrol edilmesini sağlayan sanal kuyruk yapısını kullanılır. Sanal kuyruk, gerçek ortalama kuyruk boyutunu kontrol ederek ağın ani salınımlardan korunmasını sağlamaktadır. İkinci adımda ise eNodeB'deki kullanıcı kuyruklarında yaşanan kuyruk taşması problemini çözmek için paket düşürme olasılık değeri ağdaki yük durumuna göre adaptif şekilde yeniden hesaplamaktadır.

Sanal kuyruk kullanımındaki temel hedef, ağdaki tıkanıklığı önceden haber vermesidir. Her bir gerçek kuyruğa karşılık gelen bir sanal kuyruk değeri bulunmaktadır. Tampon doluluğunu kontrol ederek çalışan sanal kuyruk yapısı gerçek kuyruktaki ortalama kuyruk değerinin yeniden hesaplanmasını sağlayarak ani paket düşüşlerinin önüne geçer. Bölüm 5.1'de önerilen sanal kuyruk yapısının çalışması anlatılmaktadır.

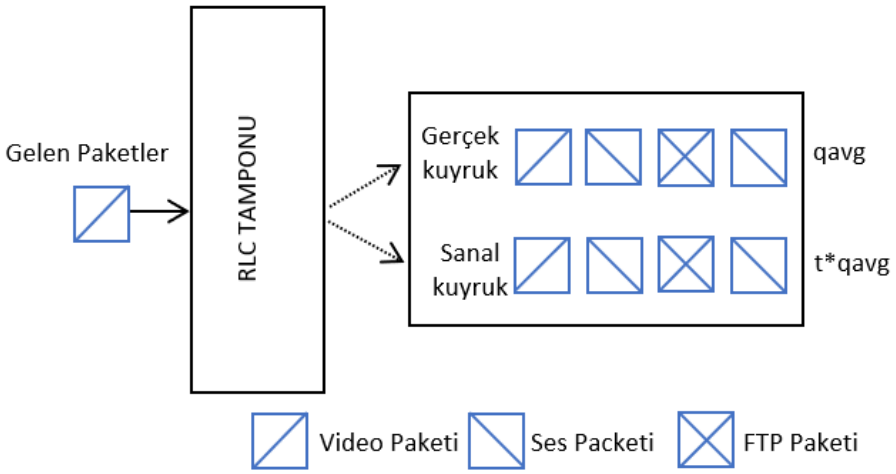
Bağlantı bant genişliğinin maksimum seviyede kullanılması için paket düşürme değerlerinin ağın durumuna göre ayarlanması gerekmektedir. Bu amaçla düşük, orta ve yüksek trafik hacmine göre paket düşürme değerini yeniden hesaplıyoruz. Böylece

ağın daha verimli kullanılması ve paket düşüşlerinin önü geçilmesi sağlanmaktadır. Bölüm 5.2'de önerilen yeni paket düşürme kontrol yapısı anlatılmaktadır.

5.1. SANAL KUYRUK DEĞERİNİN HESAPLANMASI

Önerilen sanal kuyruk yapısı ağdaki ani dalgalanma ve paketlerin kuyrukta yığılma sorunlarının çözümü için geliştirilmiştir. Sanal kuyruk yapısında tampondaki mevcut kuyruk ile maksimum kuyruk değerleri kıyaslanır. Elde edilen tampon kullanım değerine göre yeni bir kuyruk değeri hesaplanır.

Sanal kuyruk, gerçek kuyruğun kapasitesinden daha az kapasiteye sahip bir kuyruktur. Her bir gerçek kuyruğa karşılık gelen bir sanal kuyruk değeri bulunmaktadır. RCL tamponuna paketler geldikçe gerçek kuyruk değeri kullanılarak sanal kuyruk değeri hesaplanır. Gerçek ortalama kuyruk değeri qavg iken sanal kuyruk değeri *Vqavg=t*qavg* değeri ile hesaplanır. Sanal kuyruk kullanımındaki temel mantık Şekil 5.1'de gösterilmiştir.



Şekil 5.1. Gerçek ve sanal kuyruk yapısı.

Sanal kuyruk değeri, gerçek kuyruk tabanlı yönteme göre bir serbestlik *(t)* değerine sahiptir. RLC tamponuna paketler geldikçe gerçek kuyruk değeri kullanılarak sanal kuyruk değeri elde edilir. Sanal kuyruk değeri gerçek kuyruktan elde edilen ortalama kuyruk değerine göre *(0<t<=1,05)* hesaplanır. Sanal kuyruk değeri hesaplanırken kullanıcı kuyruğunun doluluk oranına göre işlem yapılır. Sanal kuyruk değerini

eNodeB'deki kullanıcı kuyruklarının mevcut kullandığı kuyruk uzunluğu ve maksimum kuyruk uzunluğu değerine göre hesaplarız. Mevcut kuyruğun kullanım oranı ($Qcur\_lenght$) kullanıcı kuyruğundaki şu anda birikmiş bayt (Qin_queue) sayısının kullanıcı kuyruğundaki maksimum bayt ($Qmax\_queue$) sayısına oranı ile elde edilir. Denklem 5.1 mevcut kuyruğun kullanım oranını hesaplanması gösterilmektedir.

$$Qcur\_lenght = (\frac{Qin\_queue * 100}{Qmax\_queue}) \tag{5.1}$$

Mevcut kuyruk değeri ile maksimum kuyruk değeri kıyaslanarak sanal kuyruk değerinin hesaplanması için kullanılacak *t* değerleri elde edilir. Eğer kullanıcı kuyruğundaki şuandaki kuyruk değeri maksimum kuyruk değerinin %60'ından küçük ise ağda trafik yükü düşük kabul edilir. Bu durumda *t* değeri 0,80 olarak hesaplanır. Böylece ortalama kuyruk boyutu daha küçük bir değer hesaplanır ve ağda muhtemel paket düşüşlerinin önüne geçilir. Eğer mevcut kuyruk değeri maksimum tampon değerinin %60 ile %80 arasında ise bu ağda trafik yükün ortalama bir değerde olduğunu gösterir. Bu durumda *t* değeri 0,90 olarak hesaplanır. Böylece ortalama kuyruk boyutu değeri kontrol altın tutulur. Kullanıcı kuyruğundaki kuyruk değeri maksimum tampon değerinin %80'nini geçtiğinde ise bu ağda trafik yükün yüksek bir değerde olduğunu gösterir. Bu durumda *t* değeri 1,05 olarak hesaplanır. Böylece ortalama kuyruk değeri yüksek bir değer hesaplanarak ağın yoğun trafiğe karşı paket gönderim hızını düşürmesi sağlanır. Denklem 5.2 *t* değerlerinin elde edilmesi gösterilmiştir.

$$(t) = \begin{cases} 0{,}80, & Qcur\_lenght < Qmax\_queue\ (\%60)\ ] \\ 0{,}90, & Qmax\_queue(\%60) \leq Qcur\_lenght < Qmax\_queue(\%80)] \\ 1{,}05 & Qmax\_queue(\%80) \leq Qcur\_lenght < Qmax\_queue] \end{cases} \tag{5.2}$$

5.2. PAKET DÜŞÜRME DEĞERİNİN HESAPLANMASI

AVRED-r algoritmasındaki ikinci aşama çözüm önerisi ise paket düşürme olasılığı değerinin farklı yük senaryolarına göre ayarlanmasıdır. Orijinal RED algoritmasında

ortalama kuyruk boyutu ve paket düşürme olasılığı doğrusal orantılıdır. Denklem 5.3'te orijinal RED algoritmasının düşürme olasılığı gösterilmektedir.

$$Pa = \begin{cases} pmax * \left(\frac{qavg - Minth}{Maxth - Minth}\right), qavg \in \text{Maxth})] \end{cases} \quad (5.3)$$

RED algoritması ağdaki düşük yük senaryosunda düşük bir ortalama gecikme değeri hesaplar. Bu durumda bağlantı bant genişliği tam olarak kullanılamamaktadır. Çözüm ise düşük yük gerçekleştiğinde bant genişliği kullanımını iyileştirmek için daha küçük bir paket düşürme olasılığı kullanmaktır. Eğer ağ durumu yüksek yük senaryosunda çalışıyor ise ağ bant genişliğinin tamamı kullanıyordur. Bu problemi çözmek ve ağdaki gecikmeyi azaltmak için daha büyük bir paket düşürme olasılığının kullanılması gerekmektedir. Yukarıdaki her iki durum da göz önüne ağdaki yük durumuna göre adaptif bir yapının kullanılması gerekir. Bu amaçla paket düşürme olasılık değerini ağdaki düşük orta ve yüksek yük durumlarına göre yeniden hesaplıyoruz. eNodeB'nin RLC katmanındaki kullanıcı kuyruklarında bekleyen paketlerin oranı *Minth* değeri ile *Maxth*'ın %60'ı arasında ise bu düşük yük durumunu gösterir. Bu durumda ağdaki bant genişliği tam olarak kullanılmıyordur. Ağ kapasitesinin daha iyi kullanılabilmesi için paket düşürme olasılığının daha da düşürülmesi gerekmektedir. Paket düşürme olasılığını trafik yüküne göre daha da düşürebilmek için *r* parametrelerin kullanılmasını öneriyoruz. Burada *r* değerlerinin artırılması (r=2,3,4,5) üstel olarak paket düşürme olasılık değerinin daha küçük hesaplanmasını sağlayacaktır. Düşük yük durumundaki paket düşürme olasılık hesabı Denklem 5.4'te gösterilmiştir.

$$Pa = \begin{cases} pmax * \left(\frac{Vqavg - Minth}{Maxth - Minth}\right)^r, Vqavg \in [\text{Minth} < Vqavg < \text{Maxth}(\%60)] \end{cases} \quad (5.4)$$

eNodeB'nin RLC katmanındaki kullanıcı kuyruklarında bekleyen paketlerin oranı *Maxth*'ın %60'ı ile %80'ı arasında ise bu orta yük durumunu gösterir. Bu durum için orijinal RED'in paket düşürme değerini değiştirmeden paket düşürme olasılığını hesaplıyoruz. Orta yük durumunda paket düşürme olasılık değeri hesaplanırken sanal kuyruk yapısı kullanıldığı için RED algoritmasına göre daha küçük bir değer elde

edilir. Orta yük durumundaki paket düşürme olasılık hesabı Denklem 5.5'te gösterilmiştir.

$$Pa = \begin{cases} pmax * \left(\frac{Vqavg - Minth}{Maxth - Minth}\right), Vqavg \in [\text{Maxth}(\%60) \leq \text{Vqavg} < \text{Maxth}(\%80)] \end{cases} \quad (5.5)$$

eNodeB'nin RLC katmanındaki kullanıcı kuyruklarında bekleyen paketlerin oranının *Maxth*'ın %80'ını geçmeye başladığı durumda ise ağ yüksek yük durumuna girmiştir. Bu durumda ağdaki bant genişliği tam olarak kullanılmadığı sonucu çıkmaktadır. Trafik hacminin yüksek olduğu durumlarda ani paket düşüşlerini önlemek ve gecikmeye engel olmak için daha yüksek bir paket düşürme olasılığının hesaplanması gerekmektedir. Yüksek trafik durumunda daha yüksek paket düşürme olasılık değerinin hesaplanması için *1/r* değerlerinin kullanılmasını öneriyoruz. Artırılan *r* değerleri ile (r=2,3,4,5) olması gerekenden daha yüksek paket düşürme olasılık değerleri elde edilecektir. Böylece ağda yaşanabilecek sıkışıklığın önceden kontrol edilmesi sağlanır. Yüksek yük durumundaki paket düşürme olasılık hesabı Denklem 5.6'de gösterilmiştir.

$$Pa = \begin{cases} pmax * \left(\frac{Vqavg - Minth}{Maxth - Minth}\right)^{\frac{1}{r}}, Vavg \in [\text{Maxth}(\%80) \leq \text{Vqavg} < \text{Maxth}] \end{cases} \quad (5.6)$$

Özetle r=2,3,4,5 şeklinde *r* değerinin artırılması, düşük trafik yükü durumunda daha düşük, yüksek trafik yükü durumunda daha yüksek paket düşürme olasılık değerinin elde edilmesini sağlamaktadır. Değişen *r* değerlerine göre AVRED'in farklı versiyonlarını AVRED-r olarak tanımlarız. AVRED-r'nin paket düşürme olasılık fonksiyonları özet olarak Denklem 5.7'de gösterilmiştir.

$$Pa = \begin{cases} 0, Vqavg \in [0, \text{Minth}] \\ pmax * \left(\frac{Vqavg - Minth}{Maxth - Minth}\right)^{r}, Vqavg \in [\text{Minth} < \text{Vqavg} < \text{Maxth}(\%60)] \\ pmax * \left(\frac{Vqavg - Minth}{Maxth - Minth}\right), Vqavg \in [\text{Maxth}(\%60) \leq \text{Vqavg} < \text{Maxth}(\%80)] \\ pmax * \left(\frac{Vqavg - Minth}{Maxth - Minth}\right)^{\frac{1}{r}}, Vqavg \in [\text{Maxth}(\%80) \leq \text{Vqavg} < \text{Maxth}] \\ 1, Vqavg \in [\text{Maxth}, +\infty] \end{cases} \quad (5.7)$$

AVRED-r algoritması için önerilen iki basamaklı çözümün algoritması Çizelge 5.1'de gösterilmektedir.

Çizelge 5.1. AVRED-r algoritması.

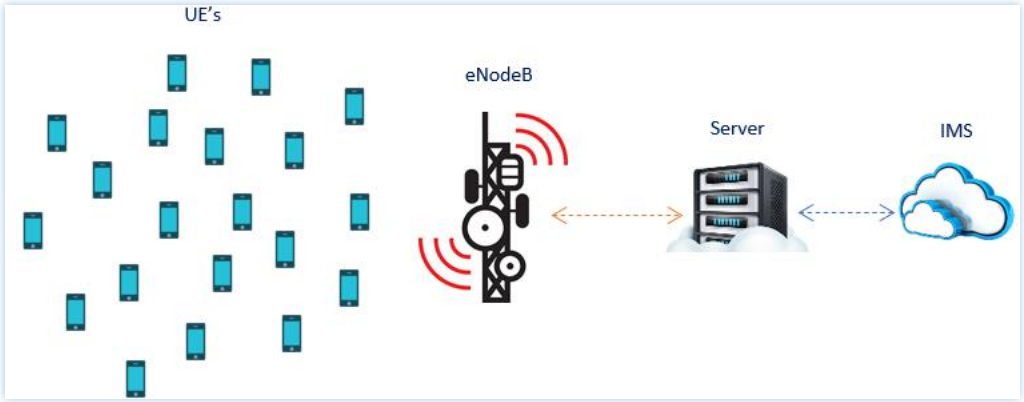
Algoritma Akışı
1: ***prosedür*** *eNodeB (parametreler ayarlanır)*
2: Değişken değerleri ayarlanır
3: Paketler RLC tamponuna gelir
4: qavg ve Vqavg değeri hesaplanır (Vqavg , qavg değerine göre hesaplanır)
5: Mevcut kuyruk değeri hesaplanır $Qcur\_lenght = (\frac{Qin\_queue * 100}{Qmax\_queue})$
6: ***Eğer (*** $Qcur\_lenght < Qmax\_queue$ (%60) ***) ise***
7: Vqavg sabit değeri t= 0.80 olarak ayarlanır
*8: Vqavg = qavg * t*
9: ***Eğer (*** $Qmax\_queue$ (%60) $\leq Qcur\_lenght < Qmax\_queue$ (%80)***) ise***
10: Vqavg sabit değeri t= 0.90 olarak ayarlanır
*11: Vqavg = qavg * t*
*12:****Değilse***
13: Vqavg sabit değeri t= 1.05 olarak ayarlanır
*14: Vqavg = qavg * t*
15: ***Eğer*** *(Vqavg<Minth)* ***ise***
16: paket kuyruğa eklenir
17: ***Eğer*** *(Minth ≤ Vqavg < Maxth (%60))* ***ise***
18: $Pa = Pmax * (\frac{Vqavg - Minth}{Maxth - Minth})^r$ *(Hesapla her bir r=2,3,4,5)*
19: ***Eğer*** *(Maxth (%60) ≤ Vqavg < Maxth (%80))* ***ise***
20: $Pa = Pmax * (\frac{Vqavg - Minth}{Maxth - Minth})$
21: ***Eğer*** *(Maxth (%80) ≤ qavg < Maxth)* ***ise***
22: $Pa = Pmax * (\frac{Vqavg - Minth}{Maxth - Minth})^{1/r}$ *(Hesapla her bir r=2,3,4,5)*
23: ***Değilse***
24: $Pa = 1$
25: Pa değerine göre paketi düşür

5.3. DENEYSEL ÇALIŞMA

Bu bölümde önce deneysel çalışma için ağ modeli oluşturuldu ve simülasyon parametreleri ayarlandı. Daha sonra simülasyon sonuçlarının değerlendirilmesi yapıldı. Performans değerlendirmesi için önce RED algoritması ile AVRED-r'nin farklı versiyonlarını karşılaştırıldı. Daha sonra ise DropTail, PIE , pFIFO ve CoDel algoritmaları ile AVRED-r karşılaştırıldı. Bölüm 5.3.1'de LTE ağ modeli ve sistem parametreleri gösterilmektedir. Bölüm 5.3.2'de simülasyon sonuçları açıklanmakta ve değerlendirilmektedir.

5.3.1 Ağ Modeli ve Simülasyon Ortamı

Bu bölümde simülasyon için kullanılan ağ modeli tanımlanmaktadır. Ağ yapısının parametreleri oluşturulur. Deneysel çerçeve için Şekil 5.2'deki tek hücreli ağ topolojisi oluşturulmuştur. Ağ simülasyonu ayrık olay tabanlı ağ simülatörü Ns3 (versiyon 3.29) ile gerçekleştirildi. Gerçek bir LTE yapısının oluşturulabilmesi için LTE/EPC Network Simulator ve Analysis (LENA) eklentisini kullanılmaktadır. LENA modülü, EPC ve E-UTRA dahil olmak üzere LTE yapısının tüm birimlerini barındırmaktadır. Ayrıca bu modül OFDMA ve ARQ gibi özellikleri kullanarak gerçek bir ağ ortamı sağlar.



Şekil 5.2. Tek hücreli Ns-3 simülasyon ortamı.

Şekil 5.2'de sağ tarafta gösterilen ana bilgisayar (server) kaynaktır. Sol tarafta tanımlanan UE'ler ise alıcıdır. Kaynak göndericinin TCP sıkışıklık algoritması TCP

New Reno tipindedir. TCP paket boyutu 1024 byte olacak şekilde ayarlanmıştır. UE'ler eNodeB'nin kapsama alanı boyunca düzgün bir şekilde yayılmaktadır. UE'lerin hareketlik yapısı için Random Walk 2D yapısı kullanılmıştır. Uzak ana makine için bağlantı kapasitesi 100 Gbps olarak ayarlanmıştır. Paket düşürme olasılığı için kullanılan eşik değerleri Minth = 20 paket ve Maxth = 80 paket olarak ayarlanmıştır. Simülasyon süresi 100 saniyedir. Önemli simülasyon parametreleri Çizelge 5.2'de gösterilmektedir.

Çizelge 5.2. Simülasyon parametreleri.

Simülasyon Parametresi	**Değer**
UE'lerin sayısı	10-100
eNodeB'lerin sayısı	1 Macro hücre
Paket Boyutu	1024 Bayt
Uygulama Veri Oranı	100 Mbps
Kuyruk Boyutu	1020-51200 bayt
Kablolu Ağ Veri Oranı	100 Gbps
Gecikme	10 ms
Kablolu Ağ Kapasitesi	100 Mbps
Uygulama Trafik Türü	TCP
eNodeB İletim Modu	MIMO
eNodeB Anten Tipi	Parabolic Anten
Downlink Paket Schedular	Priority Set Sheduler (PSS)
TCP Trafik Tipi	Tcp New Reno
Hareketlilik Tipi	Random Walk 2D
Toplam Simülasyon Süresi	100 s
Uygulama Başlama Zamanı	0.1 s'den
Uygulama Bitiş Zamanı	100 s'e kadar

5.3.2 Simülasyon Sonuçlarının Değerlendirilmesi

Bölüm 5.3.2.1'de AVRED-r algoritması ortalama verim, gecikme, fairness indeksi ve Paket Teslim Kesri (PDF) değerleri açısından RED, DropTail, CoDel, PIE ve pFIFO algoritmaları ile karşılaştırılmıştır.

5.3.2.1 Avred-r İle Diğer Kuyruk Yönetim Algoritmalarının Performans Karşılaştırılması

Bu bölümde önce RED ile AVRED-r'nin versiyonları daha sonra ise DropTail, PIE, pFIFO ve CoDel algoritmaları ile AVRED-r'nin versiyonlarının uçtan uca verim, gecikme, PDF ve fairness index değerleri açısından karşılaştırıyoruz.

Öncelikle LTE ağındaki kullanıcı sayısı değiştirilerek farklı yük senaryolarında sistem performansının değişimi incelendi. Kullanıcı sayısı 10 ile 100 arasında değiştirildi. Tüm kullanıcılar TCP uygulamasını kullanmaktadır. eNodeB'deki kullanıcı kuyruk boyutu 10240 bayt ile 51200 bayt arasında değişmektedir. Diğer simülasyon parametreleri Bölüm 5.3.1'de gösterilmektedir. Simülasyon yukarıdaki belirtilen Şekil 5.2'deki topolojiye göre kaynaktan hedef UE'lere tüm algoritmalar için simüle edildi.

Şekil 5.3 (a) RED ile AVRED-r'nin Şekil 5.3 (b) ise DropTail, PIE, pFIFO ve CoDel algoritmaları ile AVRED-r'nin versiyonlarının uçtan uca ortalama verim değerlerinin karşılaştırmasını göstermektedir. Her iki simülasyon için de kullanıcı sayısındaki değişim ile uçtan uca verim değeri arasındaki değişimin sisteme etkisi incelenmiştir.

Her iki simülasyon sonucuna göre elde edilen veriler analiz edildiğinde az sayıda UE'nin (10 UE ve 20 UE) eNodeB'den kaynak talebinde bulunması LTE ağında düşük bir trafik oluşturmuştur. Bu durumda AVRED-r yüksek r (r=5 ve r=4) değerlerinde daha yüksek verim değeri elde etmektedir. Bunun sebebi, trafik yükünün az olması durumunda yüksek (r=5 ve r=4) r değerlerinin paketlerin düşme olasılığını azaltması hatta bazı durumlarda paket düşüşünün olmamasıdır. Ayrıca düşük trafik yükünde RLC'nin kullanıcı kuyruklarında biriken paket sayısının az olması nedeniyle sanal

kuyruk değeri daha küçük elde edilmiştir. Bu durum daha küçük paket düşürme olasılık değerinin elde edilmesine katlı sağlamaktadır.

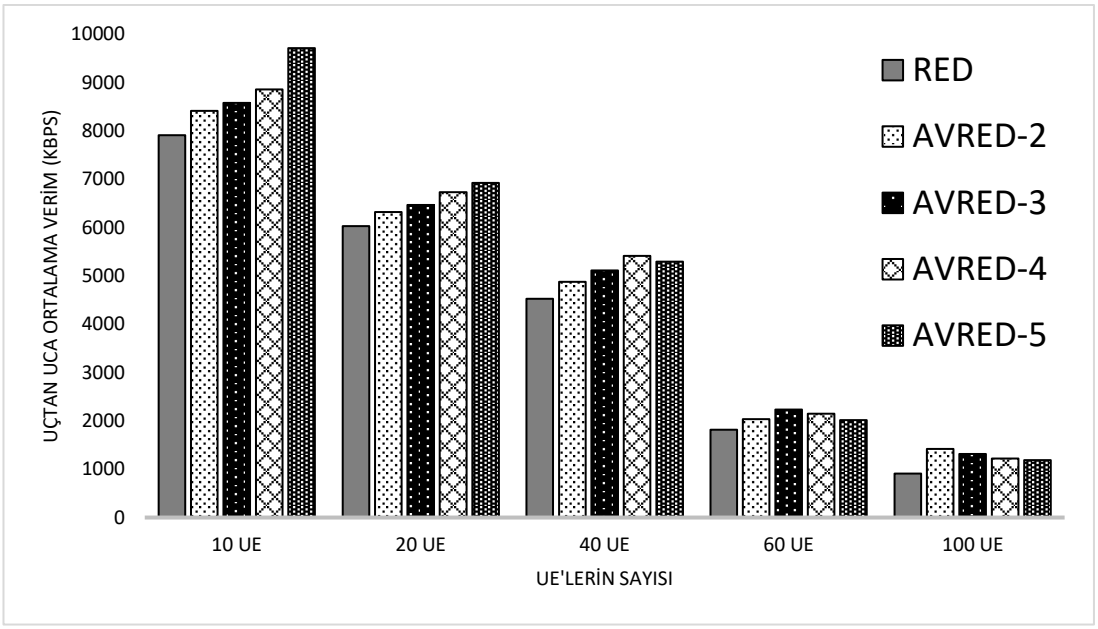
eNodeB'ye bağlanan kullanıcı sayısının artması (40 ve üzeri) LTE ağında yüksek trafiğe neden olmaktadır. Bu durumda AVRED-r'nin yüksek *r* (r=4 ve r=5) değerlerinde iyi sonuç üretememektedir. Bunun sebebi AVRED-r de kullanılan büyük *r* (r=4 ve r=5) değerlerinin yüksek trafik durumunda bile paket düşürme olasılık değerini küçük olarak hesaplamasıdır. AVRED-r yüksek *r* (r=5 ve r=4) değerlerinde TCP kaynağını olası tıkanıklık hakkında önceden bilgilendiremediği için kuyrukta paketler birikir. TCP paketlerinin düşürülmesi ile alıcı tarafından göndericiye tıkanıklık bilgisi iletilir. Göndericiden gelen bilgilendirme mesajı ile ağın tıkanıklık durumunda olduğu tespit edilir. Böylece TCP zaman aşımı gerçekleşir. Kaynak, TCP paketlerinin gönderim hızını yavaşlatır. Bu durum düşük işlem hacmi ve yüksek gecikmeye neden olur.

Yüksek trafik yükü durumunda TCP paketlerinin kuyruklardan erken düşürülmesini sağlamak için paket düşürme olasılığının artırılması gerekmektedir. Önerilen AVRED-r'nin düşük *r* değerleri (r=2 ve r=3) yüksek trafik durumunda paket düşürme değerlerini yüksek hesaplayarak ağdaki tıkanıklıkla ilgili önceden göndericiyi bilgilendirir ve paket gönderim hızının yavaşlatılmasını sağlar. Böylece kuyruklarda paketlerin yığılmasını önleyerek tıkanıklığın önüne geçer.

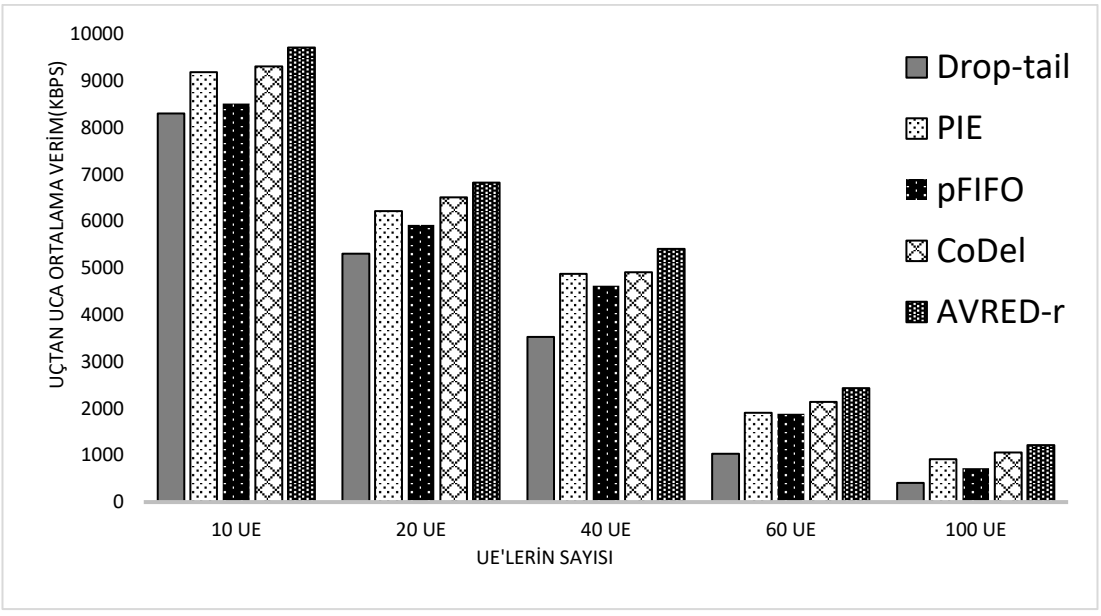
AVRED-r algoritması düşük ve orta trafik yoğunluğunda (10 UE, 20 UE ve 40 UE) yüksek *r* (r=5 ve r=4) değerleri ile daha iyi verim değerleri elde ederken ağdaki trafik yoğunluğu arttığında (60 UE ve 100 UE) düşük *r* (r=2 ve r=3) değerleri ile daha iyi verim değerlerini elde etmiştir. Bu durum AVRED-r algoritmasının farklı yük durumlarına göre adaptif çalıştığını göstermektedir.

Şekil 5.3 (b)'de DropTail algoritması kuyruğa gelen paketleri kuyruğun dolmasıyla kuyruğun önünden düşürmektedir. DropTail'deki aşırı paket düşüşleri ağın verimini olumsuz olarak etkilemektedir. Özellikle yoğun trafik durumunda DropTail tüm algoritmalar arasında aşırı paket düşürmesi ile en zayıf olanıdır. Paket gecikmesi değerini kullanan CoDel tüm trafik yüklerinde diğer algoritmalara göre (DropTail, PIE

ve pFIFO) en iyi sonucu üretmiştir. Zamana duyarlı olarak gecikme kontrolü yapan PIE, CoDel'den sonra en iyi sonucu üretmiştir. Trafik önceliklerine göre hesap yapan pFIFO yüksek öncelikli trafikleri belirlemede yaşadığı sorun sebebi ile PIE'den sonra en iyi performansı gösterebilmiştir.



(a)



(b)

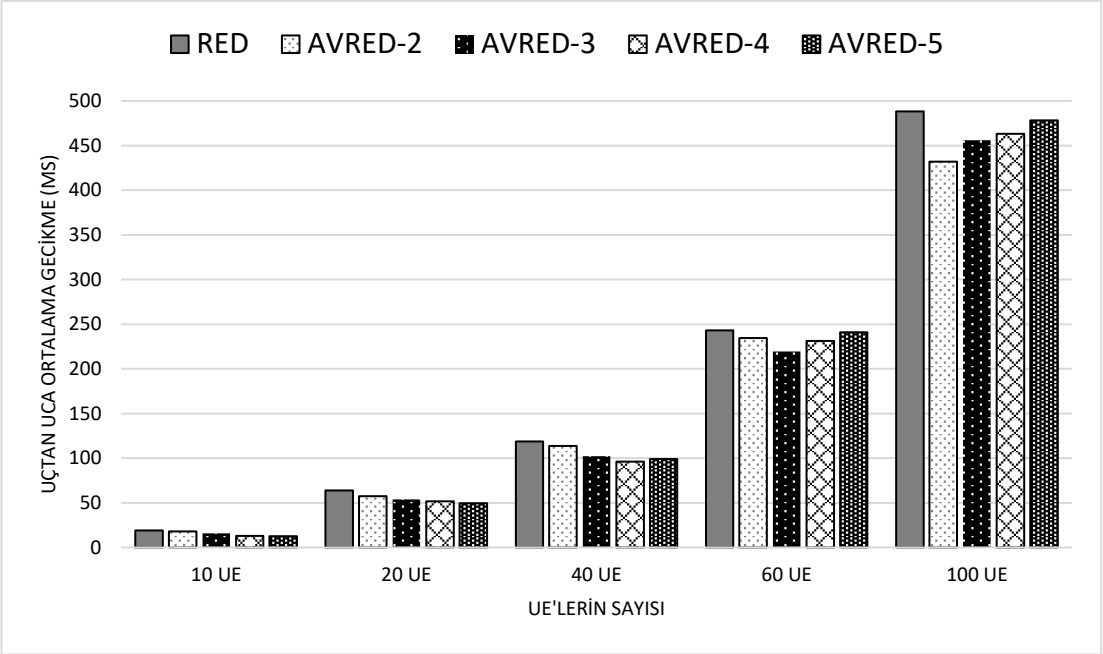
Şekil 5.3. Uçtan uca ortalama verim değerinin karşılaştırılması (a) AVRED-r ile RED algoritması arasındaki karşılaştırma (b) AVRED-r ile diğer algoritmalar arasındaki karşılaştırma.

Şekil 5.3 (a)'da AVRED-r'nin tüm versiyonları RED algoritmasından daha yüksek uçtan uca ortalama verim değeri üretmiştir. Bunun nedeni AVRED-r'nin tüm versiyonlarının sanal kuyruk değerini ağın durumuna göre hesaplaması ve her durumda RED algoritmasına göre daha düşük bir paket düşürme olasılığını elde etmesidir.

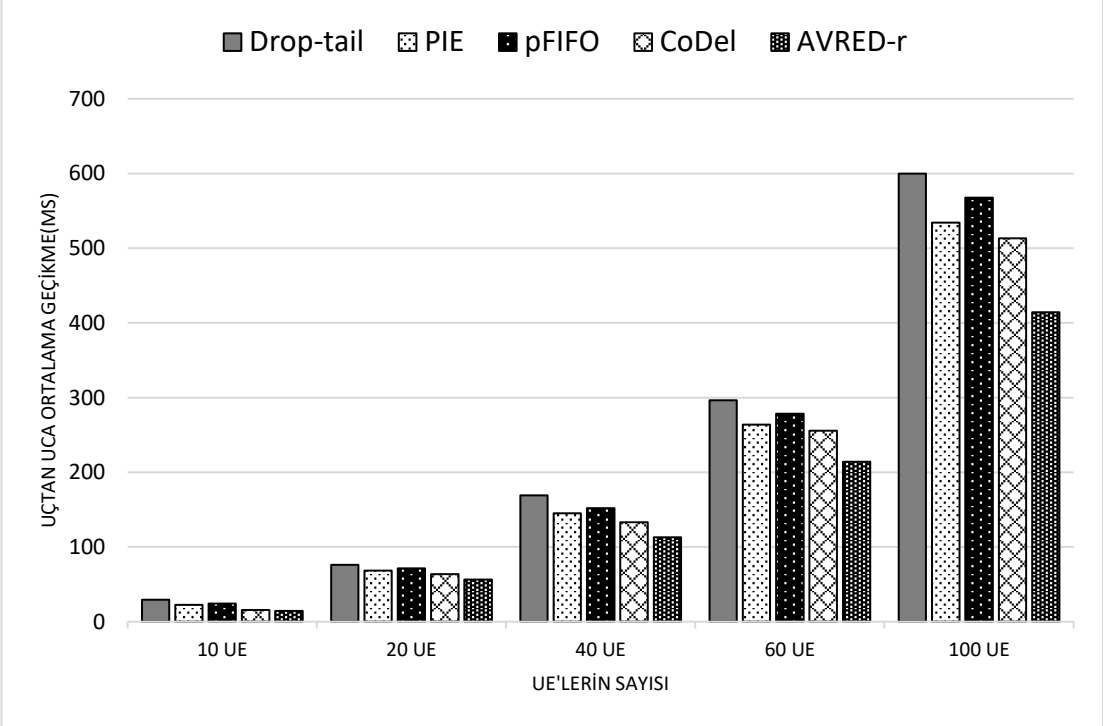
Şekil 5.4 (a) RED ile AVRED-r'nin Şekil 5.4 (b) ise DropTail, PIE, pFIFO ve CoDel algoritmaları ile AVRED-r'nin versiyonlarının uçtan uca ortalama gecikme değerlerinin karşılaştırmasını göstermektedir.

Her iki simülasyonda da eNodeB'nin kapsama alanındaki kullanıcı sayısının artmasıyla veri trafiğinin hızı ağın yürütme kapasitesine oranla hızla artmıştır. eNodeB'nin kapsama alanındaki aktif kullanıcı sayısı artıkça kullanıcı paketleri eNodeB'deki kuyrukta bekletilir. Bu durum LTE ağında darboğaz yaşanmasına neden olur. AVRED-r'nin yüksek *r* değerleri (r=4 ve r=5) trafik yükünün düşük olduğu durumlarda hiç paket düşürmeyerek ya da daha az paket düşürülmesini sağlayarak gecikmeyi azaltır. Trafik yükünün artmasıyla AVRED-r'nin düşük *r* değerleri (r=3 ve r=2) paket düşürme değerini artırarak olası tıkanıklığı önlemekte ve paketlerin kuyrukta yığılmalarının önüne geçmektedir. Böylece uçtan uca ortalama gecikme süresinin azaltılmasına yardımcı olmaktadır.

Şekil 5.4 (b) 'de DropTail kuyruğa gelen paketleri kuyruğun dolmasıyla kuyruğun önünden düşürür. Aşırı paket düşüşleri ağın uçtan uca ortalama gecikme değerini artırır. Özellikle yoğun trafik durumunda DropTail tüm algoritmalar arasında aşırı paket düşürmesi ile en fazla gecikmeye neden olan algoritmadır. CoDel, paket gecikmesi süresini kullanarak ağdaki tıkanıklığı tespit eder. Paket iletim süresi belirlenen hedef değeri aştığında CoDel tarafından tıkanıklık tespit edilir. Bu nedenle CoDel en düşük uçtan uca ortalama gecikme zamanına sahiptir. Zamana duyarlı olarak gecikme kontrolü yapan PIE, CoDel'den sonra en iyi sonucu üretmiştir. Yüksek trafik değerlerinde başarılı sonuç veren pFIFO düşük trafik yükü durumunda daha yüksek ortalama gecikme değeri elde etmiştir.



(a)



(b)

Şekil 5.4. Uçtan uca ortalama gecikme değerinin karşılaştırılması (a) AVRED-r ile RED algoritması arasındaki karşılaştırma (b) AVRED-r ile diğer algoritmalar arasındaki karşılaştırma.

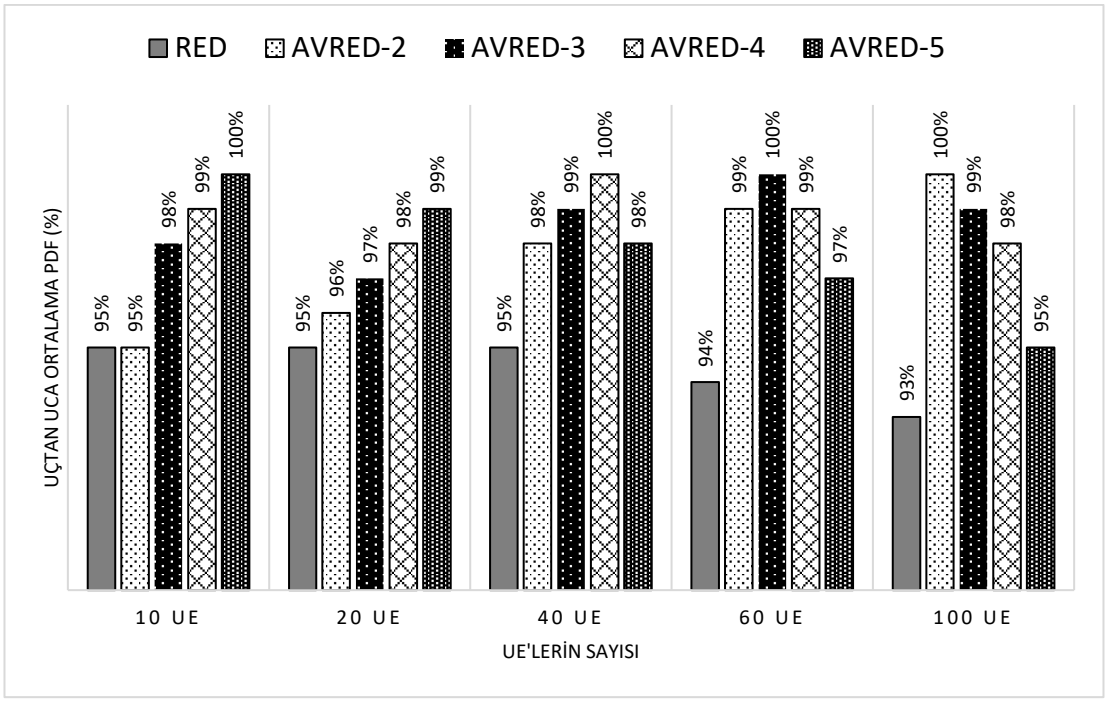
AVRED-r'nin tüm versiyonları RED algoritmasından daha küçük bir uçtan uca ortalama gecikme süresi üretmiştir. Bunun nedeni AVRED-r'nin tüm versiyonlarının

sanal kuyruk değerini ağın durumuna göre hesaplaması ve her durumda RED algoritmasına göre daha düşük bir paket düşürme değerinin hesaplanmasını sağlamasıdır.

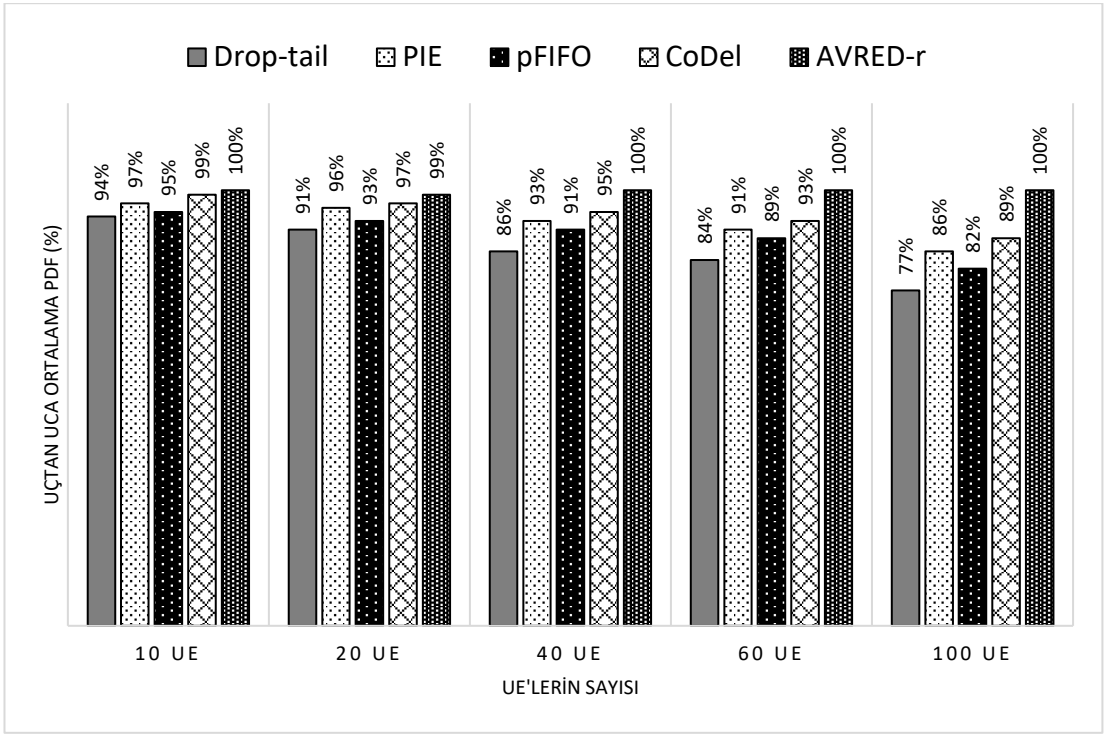
Şekil 5.5 (a) RED ile AVRED-r'nin Şekil 5.5 (b) ise DropTail, PIE, pFIFO ve CoDel algoritmaları ile AVRED-r'nin versiyonlarının PDF değerlerinin karşılaştırmasını göstermektedir.

Gönderilen toplam paketlerin toplam alınan paketlere oranı ile hesaplanan PDF, ağın performansını gösteren önemli bir değerlendirme kriteridir. Her iki simülasyonda da eNodeB'nin kapsama alanındaki kullanıcı sayısının artmasıyla talep edilen kaynak miktarında artış yaşanır. Kullanıcı paketleri eNodeB'deki kuyrukta bekletilir ve kaynak tahsisi için bekler. Bu durum eğer yeterli kaynak tahsisi yapılmazsa PDF değerinin azalmasına neden olur. AVRED-r'nin yüksek *r* değerleri (r=4 ve r=5) trafik yükünün düşük olduğu durumlarda kaynaktan hedefe gönderilen/alınan paketlere gecikme yaşanmadan cevap verir. Trafik yükünün artmasıyla AVRED-r'nin düşük *r* değerleri (r=3 ve r=2) paket düşürme değerini artırarak olası tıkanıklığı önlemekte ve paketlerin kuyrukta yığılmalarının önüne geçmektedir. AVRED-r olası tıkanıklığı önleyerek paketlerin tamponda bekletilmesinin önüne geçer. Bu durum diğer algoritmamalar göre daha iyi PDF değeri sağlar.

Şekil 5.5 (b)'ye göre yüksek trafik hacminde, DropTail tüm algoritmalar arasında en küçük PDF değerine sahiptir. Sırasıyla pFIFO, PIE ve CoDEl en iyi PDF değerlerini elde etmiştir.

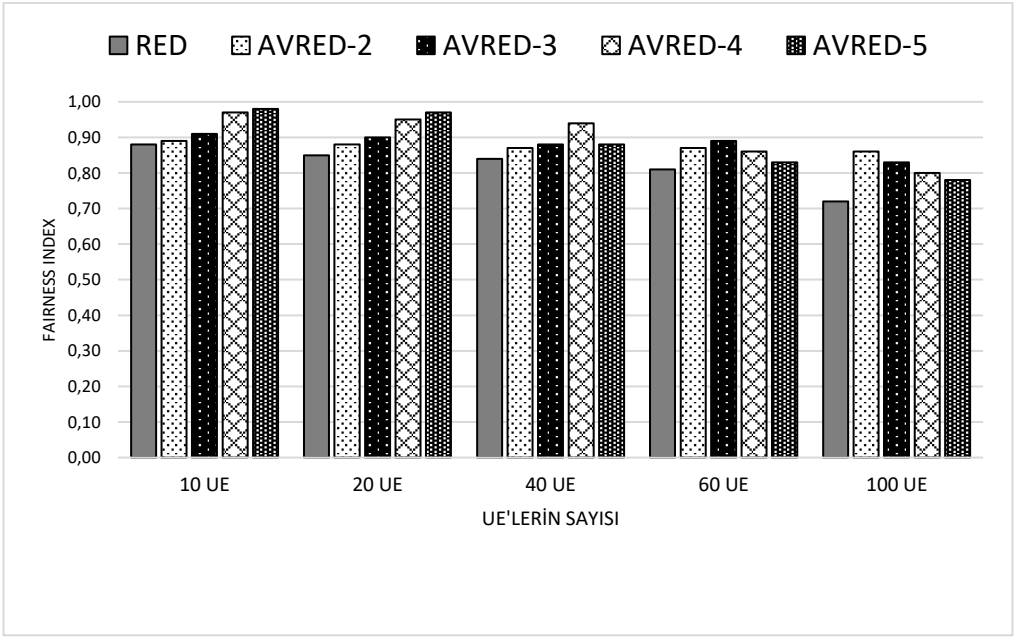


(a)

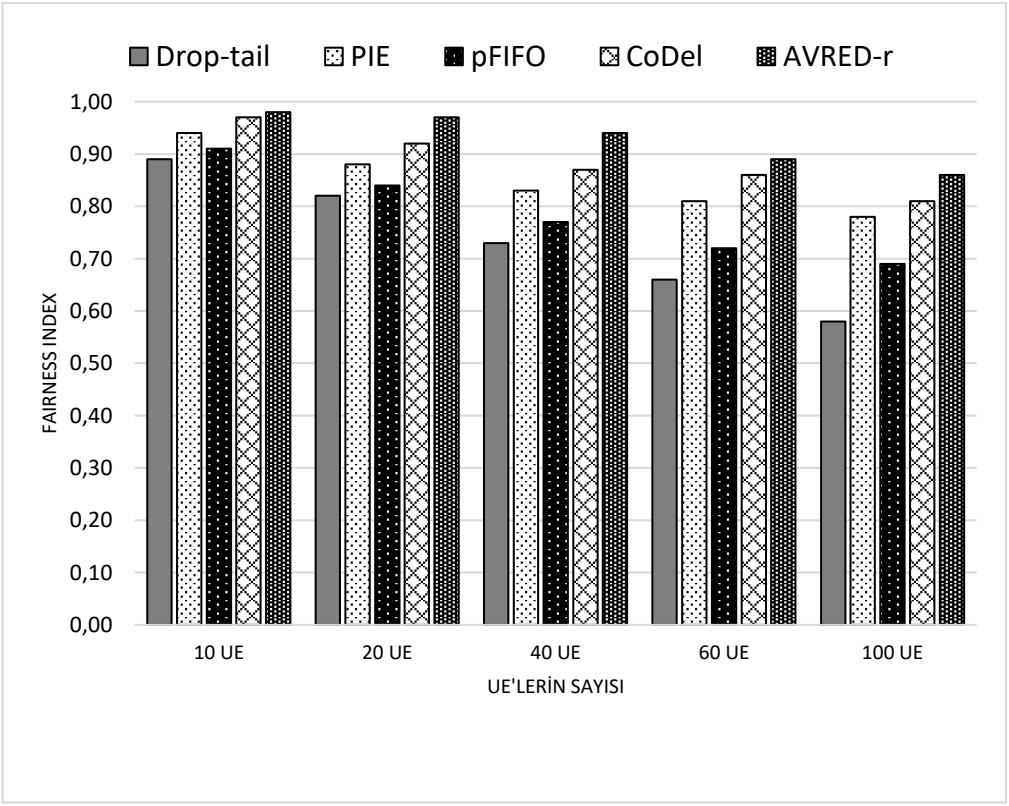


(b)

Şekil 5.5. Uçtan uca ortalama PDF değerinin karşılaştırılması (a) AVRED-r ile RED algoritması arasındaki karşılaştırma (b) AVRED-r ile diğer algoritmalar arasındaki karşılaştırma.



(a)



(b)

Şekil 5.6. Uçtan uca ortalama PDF değerinin karşılaştırılması (a) AVRED-r ile RED algoritması arasındaki karşılaştırma (b) AVRED-r ile diğer algoritmalar arasındaki karşılaştırma.

Şekil 5.6 (a) RED ile AVRED-r'nin Şekil 5.6 (b) ise DropTail, PIE, pFIFO ve CoDel algoritmaları ile AVRED-r'nin versiyonlarının fairness index değerlerinin karşılaştırmasını göstermektedir.

Her iki simülasyonda da eNodeB'ye bağlanan kullanıcı sayısının artmasıyla talep edilen kaynak miktarında artış yaşanır. Kullanıcılar kaynak tahsisi için bekler. Talep edilen kaynaklar için adil bir kaynak tahsisinin yapılması gerekmektedir. AVRED-r'nin yüksek *r* (r=4 ve r=5) değerleri trafik yoğunluğunun düşük olduğu durumlarda tüm talepleri cevaplar. Trafik yoğunluğunun artmasıyla AVRED-r'nin düşük *r* değerleri (r=2 ve r=3) paket düşürme olasılığını artırarak göndericinin paket gönderim hızını yavaşlatır. AVRED-r'ni versiyonları olası tıkanıklığı önleyerek eNodeB kaynaklarının adil dağıtılmasını sağlar.

Şekil 5.6 (b) 'de DropTail kuyruğa gelen paketleri kuyruğun dolmasıyla kuyruğun önünden düşürür. Aşırı paket düşüşleri ağın fairness index değerini azaltır. Özellikle yoğun trafik durumunda DropTail tüm algoritmalar arasında aşırı paket düşürmesi ile en fazla gecikmeye neden olan algoritmadır. Bu sebeple en fazla fairness index değeri azalan algoritmadır. Fairness index değeri açısından sırasıyla pFIFO, PIE ve CoDEl en iyi fairness index değerlerini elde etmiştir.

BÖLÜM 6

SONUÇLAR VE TARTIŞMA

Hücresel LTE ağında, eNodeB'ye gelen paketlerin tamponda birikmesi sebebiyle sistemde darboğaz yaşanmaktadır. LTE ağın yükünün artmasıyla uçtan uca verim, gecikme ve paket düşürme değerleri olumsuz yönde etkilenmektedir. LTE ağında kuyruk taşması ve paket gecikmesi sorunlarının çözümü zordur. Tıkanıklık ve paket gecikmesi için önerilen iyi bir AQM algoritması çok düşük hesaplama maliyeti gerektirmeli ve kolaylıkla ağa uygulanabilmelidir.

LTE ağında, veri iletiminde kapasiteyi aşan ani bir artış olduğunda eNodeB kullanıcı kuyruklarında aşırı paket yığılması oluşur. Eğer eNodeB'deki kullanıcı kuyruk boyutları yeterli değilse kuyruklardaki fazla paketler düşer. Bu tez çalışmasında önerilen LTE RLC katmanında çalışan RED'e dayalı adaptif sanal bir tıkanıklık kontrol algoritması olan AVRED-r, LTE ağında yaşanan düşük ve yüksek trafik senaryolarında tıkanıklık, gecikme ve bağlantı problemlerini çözmeye çalışmaktadır.

LTE ağında kullanıcı uygulama performansında kazanım elde etmek için RLC tamponunda dengeli bir tampon doluluğu yönetimi gerekmektedir. AVRED-r sanal kuyruk yapısı ile eNodeB tamponundaki doluluğu sürekli kontrol eder. Böylece tampondaki aşırı yığılmanın önüne geçilir.

Kuyruk yönetimindeki iyileştirme tarafında ise AVRED-r paket düşürme olasılık değerlerini ağın durumuna göre adaptif bir şekilde kontrol etmektedir. Böylece AVRED-r, RED'in dezavantajlı yönlerini iyileştirerek daha yüksek verim ve daha düşük gecikme süresi arasında denge sağlamaktadır. Ayrıca RED'in sadece yapısı sebebiyle AVRED-r'ye geçişte az bir değişikliğe ihtiyaç duyulur. Bu durum önerilen algoritmanın kolayca ağa uygulanmasını ve basit yönetim imkânı sağlamaktadır.

Önerilen AVRED-r, Ns-3 simlülatörü kullanılarak farklı LTE senaryoları altında uygulanır ve değerlendirilir. Simülasyon sonuçları AVRED-r'nin, DropTail, RED, CoDel, PIE ve pFIFO kuyruk yönetim algoritmaları ile karşılaştırıldığında kuyruk gecikmesi ve tampon taşması sorununu azaltmada daha verimli olduğunu göstermektedir. Ayrıca AVRED-r algoritması PDF'yi iyileştirir ve paket teslim süresini azaltır.

Gelecek çalışmalar için LTE MAC katmanında çalışan planlayıcı algoritmalarının performanslarının iyileştirilmesi sistem veriminin artırılmasına katkı sağlayacaktır.

KAYNAKLAR

1. Gupta, A. and Jha, R. K., "A Survey of 5G Network: Architecture and Emerging Technologies", ***IEEE Access***, 3: 1206–1232 (2015).
2. Stüber, G. L., "Principles of Mobile Communication", ***Principles of Mobile Communication***, (2012).
3. Kuo, Y. F. and Yen, S. N., "Towards an understanding of the behavioral intention to use 3G mobile value-added services", ***Computers In Human Behavior***, (2009).
4. Khan, A. H., Qadeer, M. A., Ansari, J. A., and Waheed, S., "4G as a next generation wireless network", ***International Conference on Future Computer and Communication***, (2009).
5. Rumney, M., "LTE Introduction", LTE and the Evolution to 4G Wireless, ***John Wiley & Sons, Ltd.***, West Sussex, United Kingdom, 1–10 (2013).
6. Note, A., "UMTS Long Term Evolution (LTE) Technology Introduction", ***Evolution***, (2008).
7. Cox, C., "An Introduction to LTE: LTE, LTE-Advanced, SAE and 4G Mobile Communications", ***John Wiley & Sons, Ltd*** (2012).
8. Salo, J., Nur-Alam, M., and Chang, K., "Practical Introduction to LTE Radio Planning", ***European Communications Engineering (ECE) Ltd, Espoo, Finland***, (2010).
9. Sauter, M., "From GSM to LTE: An Introduction to Mobile Networks and Mobile Broadband", ***John Wiley & Sons, Ltd,*** (2011).
10. Gómez, G., Pérez, Q., Lorca, J., and García, R., "Quality of service drivers in LTE and LTE-A networks", ***Wireless Personal Communications***, 75 (2): 1079–1097 (2014).
11. Lauridsen, M., Giménez, L. C., Rodriguez, I., Sørensen, T. B., and Mogensen, P., "From LTE to 5G for Connected Mobility", ***IEEE Communications Magazine***, (2017).

12. N.D., A. and A., R., "Avoiding queue overflow and reducing queuing delay at eNodeB in LTE networks using congestion feedback mechanism", ***Computer Communications***, 146 (May): 131–143 (2019).
13. Systems, A., Fuentes, M., Mi, D., Chen, H., Garro, E., Carcel, J. L., Vargas, D., Mouhouche, B., and Gomez-barquero, D., "Physical Layer Performance Evaluation of LTE-Advanced Pro Broadcast", ***IEEE Transactions on Broadcasting,*** 65 (3): 477–488 (2019).
14. Chaudhuri, S., Baig, I., and Das, D., "A novel QoS aware medium access control scheduler for LTE-advanced network", ***Computer Networks***, 135: 1–14 (2018).
15. Gupta, S. and Gupta, V., "Analytical modeling of RLC protocol of LTE using stochastic reward nets", ***International Journal Of Communication Systems***, 32 (6): e3903 (2019).
16. Szilágyi, P. and Vulkán, C., "Efficient LTE PDCP Buffer Management", ***2015 IEEE International Conference On Communications (ICC)***, 5928–5934 (2015).
17. Acharya, J., Gao, L., and Gaur, S., "Fundamentals of LTE", Heterogeneous Networks in LTE-Advanced, ***John Wiley & Sons, Ltd,*** (2014).
18. Rinne, M. and Tirkkonen, O., "LTE, the radio technology path towards 4G", ***Computer Communications***, 33 (16): 1894–1906 (2010).
19. Zheng, Y., Yu, Q., Meng, W., and Li, C., "A RED-based discard strategy for unacknowledged mode RLC in LTE", ***2013 IEEE Globecom Workshops, GC Wkshps 2013***, 2013-Janua: 4753–4758 (2013).
20. Asheralieva, A. and Miyanaga, Y., "Effective resource block allocation procedure for quality of service provisioning in a single-operator heterogeneous LTE-A network", ***Computer Networks***, 108: 1–14 (2016).
21. Adams, R., "Active queue management: A survey", ***IEEE Communications Surveys And Tutorials***, (2013).
22. Barrera, I. D., Arce, G. R., and Bohacek, S., "Statistical approach for congestion control in gateway routers", ***Computer Networks***, 55 (3): 572–582 (2011).
23. Thiruchelvi, G. and Raja, J., "A survey on active queue management mechanisms", ***IJCSNS International Journal Of Computer Science And Network Security***, 8 (12): 130–145 (2008).

24. Xu, Q. and Sun, J., "A simple active queue management based on the prediction of the packet arrival rate", ***Journal Of Network And Computer Applications***, 42: 12–20 (2014).
25. Floyd, S., "TCP and explicit congestion notification", ***ACM SIGCOMM Computer Communication Review***, 24 (5): 8–23 (1994).
26. Paul, A., Kawakami, H., Tachibana, A., and Hasegawa, T., "Effect of AQM-Based RLC Buffer Management on the eNB Scheduling Algorithm in LTE Network", ***Technologies***, 5 (3): 59 (2017).
27. Wang, Y. C. and Hsieh, S. Y., "Service-differentiated downlink flow scheduling to support QoS in long term evolution", ***Computer Networks***, 94 (2016): 344–359 (2016).
28. Qiu, Q. L., Jian, C., Ping, L. Di, and Pan, X. Z., "Hierarchy virtual queue based flow control in LTE/SAE", ***2nd International Conference On Future Networks, ICFN 2010***, 78–82 (2010).
29. Im, H., Joo, C., Lee, T., and Bahk, S., "Receiver-Side TCP Countermeasure to Bufferbloat in Wireless Access Networks", ***IEEE Transactions On Mobile Computing***, 15 (8): 2080–2093 (2016).
30. Abdullah, S. M., Younes, O., Moussa, H. M., and Abdelkader, H., "Dynamic window size of TCP for Long Term Evolution (LTE)", ***IEEE,*** (2016).
31. Feng, W., Shin, K. G., Kandlur, D. D., and Saha, D., "Active Queue Management Algorithms", ***IEEE Communications Surveys & Tutorials***, 10 (4): 513–528 (2002).
32. Floyd, S. and Fall, K., "Promoting the use of end-to-end congestion control in the Internet", ***IEEE/ACM Transactions On Networking***, 7 (4): 458–472 (1999).
33. Zhang, J., Xu, W., and Wang, L., "An improved adaptive active queue management algorithm based on nonlinear smoothing", ***Procedia Engineering***, 15: 2369–2373 (2011).
34. Park, E. C., Lim, H., Park, K. J., and Choi, C. H., "Analysis and design of the virtual rate control algorithm for stabilizing queues in TCP networks", ***Computer Networks***, 44 (1): 17–41 (2004).

35. Kshemkalyani, A. D. and Hu, L., "HRED: A Simple and Efficient Active Queue Management Algorithm", ***Computer Communications And Networks, 2004. ICCCN 2004. Proceedings. 13th International Conference On***, 00 (C): 387–393 (2004).
36. Wydrowski, B. and Zukerman, M., "GREEN: An active queue management algorithm for a self managed Internet", ***IEEE,*** (2002).
37. Feng, W., Kandlur, D. D., Saha, D., and Shin, K. G., "BLUE: A new class of active queue management algorithms", ***Ann Arbor***, (1999).
38. Feng, W., Kandlur, D. D., Saha, D., and Shin, K. G., "Stochastic fair blue: A queue management algorithm for enforcing fairness", ***IEEE,*** (2001).
39. Lin, D. and Morris, R., "Dynamics of Random Early Detection", ***ACM SIGCOMM Computer Communication Review***, 27 (4): 127–137 (1997).
40. Ott, T. J., Lakshman, T. V., and Wong, L. H., "SRED: stabilized RED", (1999).
41. Brakmo, L. S. and Peterson, L. L., "TCP Vegas: End to End Congestion Avoidance on a Global Internet", ***IEEE Journal On Selected Areas In Communications***, 13 (8): 1465–1480 (1995).
42. Iya, N., Kuhn, N., Verdicchio, F., and Fairhurst, G., "Analyzing the impact of bufferbloat on latency-sensitive applications", ***IEEE,*** (2015).
43. Jiang, H., Wang, Y., Lee, K., and Rhee, I., "DRWA: A Receiver-Centric Solution to Bufferbloat in Cellular Networks", ***IEEE Transactions On Mobile Computing***, 15 (11): 2719–2734 (2016).
44. Im, H., Joo, C., Lee, T., and Bahk, S., "Receiver-Side TCP Countermeasure to Bufferbloat in Wireless Access Networks", ***IEEE Transactions On Mobile Computing***, 15 (8): 2080–2093 (2016).
45. Argibay-Losada, P. J., Nozhnina, K., Suárez-González, A., López-García, C., and Fernández-Veiga, M., "Loss-based proportional fairness in multihop wireless networks", ***Wireless Networks***, 20 (5): 805–816 (2014).
46. Qureshi, B., Othman, M., Subramaniam, S., and Wati, N. A., "QTCP: Improving Throughput Performance Evaluation with High-Speed Networks", ***Arabian Journal For Science And Engineering***, 38 (10): 2663–2691 (2013).
47. Alfredo Grieco, L. and Mascolo, S., "End-to-End Bandwidth Estimation for Congestion Control in Packet Networks", ***IEEE/ACM Transactions on Networking,*** 645–658 (2003).

48. Abdelsalam, A., Luglio, M., Roseti, C., and Zampognaro, F., "TCP Wave: A new reliable transport approach for future internet", ***Computer Networks***, 112: 122–143 (2017).
49. Jinsheng Sun, Ko, K.-T., Guanrong Chen, Chan, S., and Zukerman, M., "PD-RED: to improve the performance of RED", ***IEEE Communications Letters***, 7 (8): 406–408 (2003).
50. Hollot, C. V., Misra, V., Towsley, D., and Wei-Bo Gong, "On designing improved controllers for AQM routers supporting TCP flows", ***EEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference***, (2001).
51. Singh, P. K. and Gupta, S. K., "Variable length virtual output queue based fuzzy congestion control at routers", ***IEEE 3rd International Conference on Communication Software and Networks,*** (2011).
52. Pan, R., Natarajan, P., Piglione, C., Prabhu, M. S., Subramanian, V., Baker, F., and VerSteeg, B., "PIE: A lightweight control scheme to address the bufferbloat problem", ***IEEE International Conference On High Performance Switching And Routing, HPSR***, 148–155 (2013).
53. Nichols, K. and Jacobson, V., "Controlling queue delay", ***Queue Com.***, 10 (5): 1–15 (2012).
54. Casoni, M., Grazia, C. A., Klapez, M., and Patriciello, N., "How to avoid TCP congestion without dropping packets: An effective AQM called PINK", ***Computer Communications***, 103: 49–60 (2017).
55. Universal, E. and Radio, T., "LTE Overall description", ***The UMTS Long Term Evolution: From Theory to Practice,*** 10-148 (2009).
56. Huang, J., Qian, F., Gerber, A., Mao, Z. M., Sen, S., and Spatscheck, O., "A close examination of performance and power characteristics of 4G LTE networks", ***International Conference on Mobile Systems, Applications, and Services,*** 89–97 (2012).
57. Khan, F., "LTE for 4G Mobile Broadband", ***Cambridge Publish,*** (2009).
58. Katsalis, K., Nikaein, N., Schiller, E., Ksentini, A., and Braun, T., "Network Slices toward 5G Communications: Slicing the LTE Network", ***IEEE Communications Magazine***, (2017).

59. Holma, H. and Toskala, A., "LTE for UMTS: Evolution to LTE-Advanced: Second Edition", ***Wiley,*** (2011).
60. Akyildiz, I. F., Gutierrez-Estevez, D. M., and Reyes, E. C., "The evolution to 4G cellular systems: LTE-Advanced", ***Physical Communication***, 3 (4): 217–244 (2010).
61. Clerckx, B., Lozano, A., Sesia, S., van Rensburg, C., and Papadias, C., "3GPP LTE and LTE-Advanced", ***EURASIP Journal On Wireless Communications And Networking***, 2009 (1): 472124 (2009).
62. Freescale, "Long Term Evolution Protocol Overview", ***White Paper***, (2008).
63. Cox, C., "An Introduction to LTE: LTE, LTE-Advanced, SAE, VoLTE and 4G Mobile Communications: Second Edition", ***John Wiley & Sons Ltd,*** (2014).
64. Myung, H., "Technical overview of 3GPP LTE", ***IEEE***, (2008).
65. M. Rumney, "LTE - The UMTS Long Term Evolution", ***John Wiley & Sons, Ltd,*** (2011).
66. Sesia, S., Toufik, I., and Baker, M., "LTE - The UMTS Long Term Evolution: From Theory to Practice: Second Edition", ***John Wiley & Sons,*** (2011).
67. Parkvall, S., Furuskär, A., and Dahlman, E., "Evolution of LTE toward IMT-advanced", ***IEEE Communications Magazine***, (2011).
68. Parkvall, S., Dahlman, E., Furuskär, A., Jading, Y., Olsson, M., Wänstedt, S., and Zangi, K., "LTE-Advanced - Evolving LTE towards IMT-Advanced", ***IEEE Vehicular Technology Conference***, (2008).
69. Ghosh, A. and Ratasuk, R., "Essentials of LTE and LTE-A", ***Cambridge University Press***, (2011).
70. Chmiel, M., Enescu, M., Holma, H., Koivisto, T., Lindholm, J., Lunttila, T., Pedersen, K., Skov, P., Roman, T., Toskala, A., and Yan, Y., "LTE-Advanced", LTE for UMTS: Evolution to LTE-Advanced: Second Edition, ***John Wiley & Sons, Ltd,*** (2011).
71. Ghosh, A., Ratasuk, R., Mondal, B., Mangalvedhe, N., and Thomas, T., "LTE-advanced: Next-generation wireless broadband technology", ***IEEE Wireless Communications***, (2010).
72. Astély, D., Dahlman, E., Furuskär, A., Jading, Y., Lindström, M., and Parkvall, S., "LTE: The evolution of mobile broadband", ***IEEE Communications Magazine***, 47 (4): 44–51 (2009).

73. Welzl, M., "Congestion control principles", Network Congestion Control, ***John Wiley & Sons, Ltd,*** (2006).
74. Nichols, K. and Jacobson, V., "Controlling queue delay", ***Communications Of The ACM***, 55 (7): 42–50 (2012).
75. Hamdi, M. M., Rashid, S. A., Ismail, M., Altahrawi, M. A., Mansor, M. F., and AbuFoul, M. K., "Performance Evaluation of Active Queue Management Algorithms in Large Network", ***IEEE Communications,*** (2018).
76. Siraj, S., Gupta, A. K., and Badgujar-Rinku, "Network Simulation Tools Survey", ***International Journal Of Advanced Research In Computer And Communication Engineering Vol. 1, Issue 4, June 2012***, (2012).
77. Lu, Z. and Yang, H., "Unlocking the Power of OPNET Modeler", Unlocking the Power of OPNET Modeler, ***Cambridge University Press,*** (2012).
78. Zaki, Y. and Zaki, Y., "LTE Network Simulator", ***Future Mobile Communications,*** (2013).
79. Zaki, Y., Zhao, L., Goerg, C., and Timm-Giel, A., "LTE mobile network virtualization", ***Mobile Networks And Applications***, (2011).
80. Suliman, I. A. and Abdalla, G. M. T., "Comparative Study of wireless LAN using OPNET and NS-2", ***Communications Of The ACM***, (2016).
81. Zhao, W. and Xie, J., "OPNET-based modeling and simulation study on handoffs in Internet-based infrastructure wireless mesh networks", ***Computer Networks***, (2011).
82. Kaoutar, E., Mohammed, P. M., and Bouchaib, P. N., "Zigbee Routing Opnet Simulation for a Wireless Sensors Network", ***International Journal Of Advanced Computer Science And Applications***, (2014).
83. Chengyu Zhu, Yang, O. W. W., Aweya, J., Ouellette, M., and Montuno, D. Y., "A comparison of active queue management algorithms using the OPNET Modeler", ***IEEE Communications Magazine***, 40 (6): 158–167 (2002).
84. Varga, A., "OMNeT++", Modeling and Tools for Network Simulation, (2010).
85. Chen, F. and Dressler, F., "A Simulation Model of IEEE 802.15.4 in OMNeT++", ***6. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze, Poster Session***, (2007).

86. Virdis, A., Stea, G., and Nardini, G., "SimuLTE - A modular system-level simulator for LTE/LTE-A networks based on OMNeT++", ***SCITEPRESS - Science and Technology Publications***, (2014).
87. Katsaros, K., Kemerlis, V. P., Stais, C., and Xylomenos, G., "A BitTorrent module for the OMNeT++ simulator", ***IEEE,*** (2009).
88. Feeney, L. M. and Willkomm, D., "Energy Framework: an extensible framework for simulating battery consumption in wireless networks", ***ICST,*** (2012).
89. Kaur, R., Sangal, A. L., and Kumar, K., "Modeling and simulation of DDoS attack using omnet++",, ***IEEE,*** (2014).
90. Fortino, G., Gravina, R., Russo, W., and Savaglio, C., "Modeling and Simulating Internet-of-Things Systems: A Hybrid Agent-Oriented Approach", ***Computing In Science And Engineering***, (2017).
91. Technologies, S. N., "QualNet 5.0 Distributed Reference Guide", ***Distribution***, (2009).
92. Internet: The QualNet network simulation software (QualNet), **https://www.scalable-networks.com/products/qualnet-network-simulation-software-tool/** (2020).
93. Maria, T., Samuel, N., "QualNet Guide", ***Simulators***, (2013).
94. Upadhyay, R., Bhatt, U. R., and Tripathi, H., "DDOS Attack Aware DSR Routing Protocol in WSN", ***Ad Hoc Networks***, (2016).
95. Yi, J., Adnane, A., David, S., and Parrein, B., "Multipath optimized link state routing for mobile ad hoc networks", ***Computer Networks***, (2011).
96. Li, T., Han, M. K., Bhartia, A., Qiu, L., Rozner, E., Zhang, Y., and Zarikoff, B., "CRMA: Collision-resistant multiple access", ***Wireless Communication,*** (2011).
97. Seol, S., Lee, E. K., and Kim, W., "Indoor mobile object tracking using RFID", ***Future Generation Computer Systems***, (2017).
98. İnternet: Açık kaynak kodlu Ns-3 simülatör yazılımı "Ns-3 Kullanımı", **https://www.nsnam.org/** (2020)
99. Issariyakul, T. and Hossain, E., "Introduction to Network Simulator NS2", Introduction to Network Simulator NS2, ***Springer US,*** (2012).
100. İnternet: ns-3 Consortium, "Ns-3", **https://github.com/nsnam,** (2020)

101. Riley, G. F. and Henderson, T. R., "The ns-3 network simulator", Modeling and Tools for Network Simulation, ***IEEE,*** (2010).
102. Baldo, N., Requena-Esteso, M., Núñez-Martínez, J., Portolès-Comeras, M., Nin-Guerrero, J., Dini, P., and Mangues-Bafalluy, J., "Validation of the IEEE 802.11 MAC model in the ns3 simulator using the EXTREME testbed", ***Computer Networks,*** (2010).
103. Costantino, L., Buonaccorsi, N., Cicconetti, C., and Mambrini, R., "Performance analysis of an LTE gateway for the IoT", ***IEEE,*** (2012).
104. Ali, S. M., Fatima, S. G., Fatima, S. K., and Sattar, S. A., "Energy efficient routing in wireless sensor networks based on QoS", ***International Journal Of Advanced Research In Engineering And Technology***, (2019).
105. Fouda, A., Ragab, A. N., Esswie, A., Marzban, M., Naser, A., Rehan, M., and Ibrahim, A. S., "Real-Time Video Streaming over NS3-based Emulated LTE Networks", ***International Journal Of Electronics Communication And Computer Technology***, (2014).
106. Spaho, E., Ikeda, M., Barolli, L., Xhafa, F., Younas, M., and Takizawa, M., "Performance of OLSR and DSDV protocols in a VANET scenario: Evaluation using CAVENET and NS3", ***Wireless Networks,*** (2012).
107. Qwasmi, N., Ahmed, F., and Liscano, R., "Simulation of DDOS attacks on P2P networks", ***Computer Communication,*** (2011).
108. Pei, G. and Henderson, T. R., "Validation of OFDM error rate model in ns-3", ***Boeing Research Technology***, (2010).
109. Casoni, M. and Patriciello, N., "Next-generation TCP for ns-3 simulator", ***Simulation Modelling Practice And Theory***, (2016).
110. Al-Ali, A. and Chowdhury, K., "Simulating dynamic spectrum access using ns-3 for wireless networks in smart environments", ***IEEE,*** (2014).
111. Weingärtner, E., Vom Lehn, H., and Wehrle, K., "A performance comparison of recent network simulators", ***Wiley,*** (2009).

ÖZGEÇMİŞ

Muhammet ÇAKMAK, 1985 yılında Giresun/Bulancak'ta doğdu. İlk orta ve lise öğrenimini Bulancak'ta tamamladı. Sakarya Üniversitesi Teknik Eğitim Fakültesi Bilgisayar Sistemleri Öğretmenliğinden 2008 yılında mezun oldu. 2008-2017 yılları arasında çeşitli meslek liselerinde bilgisayar öğretmeni olarak çalıştı. 2015 yılında Karabük üniversitesi Bilgisayar Mühendisliği bölümünde yüksek lisansını tamamladı. Halen Karabük Üniversitesi Uzaktan Eğitim Uygulama ve Araştırma Merkezi'nde çalışmaktadır.

ADRES BİLGİLERİ

Adres : Karabük Üniversitesi KBUZEM
Merkez/KARABÜK
Tel : 444 78 78 (Dahili: 8457)
E-posta : muhammetcakmak@karabuk.edu.tr
Web Sayfası : https://avesis.karabuk.edu.tr/muhammetcakmak

YAYINLAR

1. **Cakmak, M.**, Albayrak Z., Torun C., "A Performance Comparison Of Queue Management Algorithms In Lte Networks Using Ns-3 Simulator", ***Technical Gazette***, TV-20200411071703, 2020.
2. **Cakmak, M.**, Albayrak Z, "Performance Analysis of Queue Management Algorithms Between Remote-Host and PG-W in LTE Networks," **Acad. Platf. J. Eng. Sci.**, pp. 456–463, Sep. 2020.
3. **Cakmak, M.**, Albayrak Z, "A Review: Mobile Communication Past, Present and Future," in International Conference on Advanced Technologies, Computer Engineering and Science, 2018, pp. 141–145.

4. **Cakmak, M.**, Albayrak Z, “A Review: Active Queue Management Algorithms in Mobile Communication,” Int. Conf. Cyber Secur. Comput. Sci., pp. 180–184, 2018.
5. **Albayrak, Z.**, Hatem M, Çakmak M, “Performance Evaluation of WBANs MAC Protocols in Different dBm and OMNet++,” **Acad. Platf. J. Eng. Sci.**, pp. 1-7, Sep. 2021.