
GAZİ ÜNİVERSİTESİ
Gazi
1926

VERİ MERKEZLERİNDE DERİN PEKİŞTİRMELİ ÖĞRENME İLE AĞ TRAFİK OPTİMİZASYONU

Anuarbek AMANOV

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ŞUBAT 2023

ETİK BEYAN

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

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

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

Anuarbek AMANOV
23/02/2023

VERİ MERKEZLERİNDE DERİN PEKİŞTİRMELİ ÖĞRENME İLE AĞ TRAFİK OPTİMİZASYONU

(Doktora Tezi)

Anuarbek AMANOV

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Şubat 2023

ÖZET

Son zamanlarda iletişim hızlarındaki artışa paralel olarak ağ trafiğinde ve veri merkezlerindeki tıkanıklık problemlerinde artış meydana gelmektedir. Tıkanıklığı önlemede farklı tıkanıklık bildirim mekanizmaları önemli araştırma konularından birisidir. Bu problemlerin çözümünde genel bir yaklaşım olarak tek kuyruklu senaryolar için Açık Tıkanıklık Bildirimi (ATB) mekanizması tasarlanmaktadır. Ancak, Veri Merkezi Ağlarında (VMA) her anahtar için port başına birden fazla kuyruğa ihtiyaç vardır. Bu amaçla son yıllarda çok kuyruklu VMA'lar üzerine yapılan çalışmalar artmıştır. Ancak, bu amaçla veri merkezindeki anahtarlarda kullanılan paket işaretleme eşiği aşılırsa, aynı bağlantı portundaki tüm paketler ATB işaretini alabilir ve bu hatalı işaretleme işlemi veri aktarım kalitesini düşürebilir. Bu tezde, hatalı işaretleme problemini çözmek için bir derin pekiştirmeli öğrenme tabanlı öncelik ATB Haritalama yaklaşımı önerilmiştir. Önerilen yaklaşımda, arabellekten gelen akışlar sınıflayıcı tarafından fare ve fil akışları olarak sınıflandırılmakta ve devamında çıkış port arabelleği eşiğinde işaretlenerek öncelik verilmektedir. Ayrıca, çoklu-çift çıkış port arabelleğinde paketlerin hatalı işaretlenmesinden kaçınmak için derin pekiştirmeli öğrenme tekniklerinden yararlanmaktadır. Önerilen yaklaşımın etkinliği ve verimliliği büyük ölçekli bir NS-2 benzetim ortamında test edilmiştir. Çalışmada elde edilen sonuçlar detaylı bir şekilde incelenmiş ve sunulmuştur. Akış Tamamlama Süresi metriğine dayalı değerlendirmeler, önerilen yaklaşımın hedeflerine ulaşmada başarılı olduğunu ortaya koymaktadır.

Bilim Kodu	:	92407
Anahtar Kelimeler	:	Veri merkezi ağları, tıkanıklık kontrol mekanizması, derin pekiştirmeli öğrenme
Sayfa Adedi	:	73
Danışman	:	Prof. Dr. Aydın ÇETİN

NETWORK TRAFFIC OPTIMIZATION WITH DEEP REINFORCEMENT LEARNING IN DATA CENTERS

(Ph. D. Thesis)

Anuarbek AMANOV

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

February 2023

ABSTRACT

Recently, in parallel with the increase in communication speeds, there has been an increase in network traffic and congestion problems in data centers. Different congestion notification mechanisms have become one of the important research topics in preventing congestion. As a general approach to solving these problems, an Explicit Congestion Notification (ECN) mechanism is designed for one-tailed scenarios. However, in Data Center Networks (DCN) multiple queues are needed per port for each switch. In addition, if the packet marking threshold is exceeded in the switches in the data center, all packets on the same connection port may receive the ECN marking, and this false marking may reduce the data transfer quality in some cases. In this thesis, a deep reinforcement learning-based priority ECN-Mapping approach, is proposed to solve the erroneous marking problem. In the proposed approach, the flows coming from the buffer are classified as mice and elephant flows by the classifier and then priority is given by marking the flows at the output port buffer threshold according to their priority status. Additionally, it utilizes deep reinforcement learning techniques to avoid false marking of packets in the multi-queue output port buffer. The effectiveness and efficiency of the proposed approach has been tested in a large scale NS-2 simulation environment. The results obtained in the study were thoroughly examined and presented. Evaluations based on the Flow Completion Time metric reveals that the proposed approach is successful in achieving its objectives.

Science Code	:	92407
Key Words	:	Data center networks, congestion control mechanism, deep reinforcement learning
Page Number	:	73
Supervisor	:	Prof. Dr. Aydın ÇETİN

TEŞEKKÜR

Doktora eğitimim süresince, değerli bilgilerini benimle paylaşan, kendisine ne zaman danışsam bana kıymetli zamanını ayıran, sabır ve ilgiyle bana yardımcı olarak bugünlere gelmeme vesile olan çok değerli hocam ve danışmanım Prof. Dr. Aydın ÇETİN'e çok teşekkür ederim. Kıymetli zamanını benim hazırladığım teze ayırıp değerlendireceği için ve desteklerini esirgemeyen TİK üyesi olarak değerli yorumlarını ve düzeltmelerini esirgemeyerek, kaynak ve yöntem açısından bana yardımda bulunarak yol gösteren kıymetli hocalarım Prof. Dr. O. Ayhan ERDEM ve Prof. Dr. Ömer DEPERLİOĞLU'na teşekkürü borç biliyor ve şükranlarımı sunuyorum. Tez çalışmalarım süresince deneyimlerini paylaşarak katkı sağlayan Dr. Akbar MAJIDI arkadaşıma müteşekkirim.

Doktora eğitimimin ders döneminde ve ondan sonraki tüm süreçlerde daima maddi ve manevi destekleyen Doç. Dr. Hüseyin POLAT ve Doç. Dr. Cemal KOÇAK hocalarıma teşekkürlerimi sunarım. Ayrıca şahsıma Türkiye'de verdiği burs ile eğitim alma imkânı sağlayan Ahmet Yesevi Uluslararası Türk Kazak Üniversitesine, Ahmet Yesevi Üniversitesi Mütevelli Heyet Başkanlığına ve mensuplarına teşekkür borç bilirim.

Doktora eğitimim süresince beni büyük sabırsızlıkla bekleyen ve her zaman destekleyen sevgili eşim Meruert TASBOLATKIZI'na, canım kızlarım Zere NURSEİT ve Uljan NURSEİT'e, ayrıca sonsuz destek ve anlayışlarıyla bugünlere gelmemde büyük emeği olan canım aileme ve dostlarıma teşekkür ediyorum.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

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

Simgeler	Açıklamalar
Gbps	Gigabit/saniye
KB	Kilobyte
s	Saniye
ms	Milisaniye
µs	Mikrosaniye

Kısaltmalar	Açıklamalar
AB	Arabellek Boyutu (Buffer Size)
ANTB	Adil Nicemlenmiş Tıkanıklık Bildirimi (Fair Quantized Congestion Notification- FQCN)
AÖB	Açık Öncelik Bildirimi (Explicit Priority Notification-EPN)
ATB	Açık Tıkanıklık Bildirimi (Explicit Congestion Notification-ECN)
ATS	Akış Tamamlanma Süresi (Flow Completion Time-FCT)
ÇK	Çok Kuyruklu (Multi Queue-MQ)
ÇK-ATB	Çok Kuyruklu-Açık Tıkanıklık Bildirimi (Multi–Queue Explicit Congestion Notification)
ÇPA	Çıktı Port Arabelleği (Output Port Buffer)
DBG	Darboğaz Bant Genişliği (Bottleneck Bandwidth and RTT- BBR)
DEME	Kuyruğa Eklemede İşaretleme Ayırma (DEcouple Marking from Enqueuing)
Derin-PÖ	Derin Pekiştirmeli Ögrenme (Deep Reinforcement Learning-DRL)
FilK	Fil Kuyruk (Elephant Queue-EQ)
FareK	Fare Kuyruk (Mice Queue-MQ)
GDS	Gidiş-Dönüş Süresi (Round Trip Time-RTT)
GTB	Geriye doğru Tıkanıklık Bildirimi (Backward Congestion Notification- BCN)
KA	Kuyruk Arabelleği (Queue Buffer-QB)

Kısaltmalar	Açıklamalar
KZ	Kuyruk Zamanlama (Queue Scheduling-QS)
L4S	Ölçeklenebilir Verim ile Düşük Gecikme-Düşük Kayıp (Low Latency–Low Loss with Scalable throughput)
MÖ	Makine Öğrenmesi (Machine Learning- ML)
NTB	Nicemlenmiş Tıkanıklık Bildirimi (Quantized Congestion Notification-QCN)
SAPA	Seçici Anlayışsızlık ile Port Başına İşaretleme (Per–Port Marking with Selective Blindness-PMSB)
VMA	Veri Merkezi Ağları (Data Center Networks-DCN)
VM-ATB	Veri Merkezi-Açık Tıkanıklık Bildirimi (Data Center-Explicit Congestion Notification- DC-ECN)
VMTCP	Veri Merkezi TCP (Data Center Transport Control Protocol-DCTCP)
YK	Yükü kes (Cut Payload-CP)

1. GİRİŞ

Veri iletişimi, dünyada çok sayıda sunucularda ve veri merkezlerinde kritik rol oynamaktadır. Özellikle iletişimin olduğu her yerde, her noktada kullanılması nedeniyle günlük hayatın önemli bir parçası hâline gelmektedir. Veri iletişim uygulamaları, protokolleri, mekanizmaları temel bilgisayar ağları üzerinde çok çeşitli gereksinimlere sahiptir [1]. Veri artışı ve gerçek zamanlı akışa olan yüksek talep nedeniyle ortaya çıkan tıkanıklık, denetimi daha karmaşık hâle getirmektedir. Oluşan tıkanıklığın etkisini anlamak ve tahmin etmek için bu verilerden bilgi çıkarmak, ağın verimli şekilde yönetilmesini sağlamak giderek daha fazla önem kazanmıştır.

Tıkanıklık denetimi, ağ kapasitesini verimli şekilde kullanmak için trafik kaynaklarının veri aktarım hızlarını modüle eden temel ağ görevidir. Makine öğreniminin ortaya çıkmasıyla birlikte, onun yöntemlerinin biri olan Derin Pekiştirmeli Öğrenmeye (DPÖ) dayalı tıkanıklık denetimi yoğun ilgi görülmektedir, ağ trafiği modellerine ve performansına dayalı olarak dinamik, gerçek zamanlı karar almayı etkinleştirerek Veri Merkezi Ağlarında (VMA) tıkanıklık denetimini iyileştirme potansiyeline sahiptir.

Optimal ağ yapılandırmalarını sürdürmek için denetleyicinin çoklu akış problemini çözmesi ve sıkı zaman kısıtlamaları altında ağı bütünsel olarak güncellemesi gerekmektedir. Veri Merkezi Ağlarındaki bu merkezî en iyileme, yavaş yakınsama hızına ve çok az ölçeklenebilirliğe sahip birçok değişken ve kısıtlama içermektedir. Öncelikle gecikme açısından en çok etkilenen ve sonraki ağ güncellemesinde yapılandırılacak akışların belirlenmesi için güncel bir algoritmaya ihtiyaç duyulmaktadır.

Veri Merkezi Ağlarında Tıkanıklık Denetimi

Veri merkezlerinde tıkanıklık denetimi, aktif araştırma alanıdır. Çok Kuyruklu – Açık Tıkanıklık Bildirimi (ÇK-ATB) gibi Aktif Kuyruk Yönetimi (AKY) algoritmalarının yanı sıra akış denetimi, trafik şekillendirme ve zamanlamayı kullanan teknikler de dâhil olmak üzere veri merkezlerindeki ağ trafiğini yönetmek için önerilen farklı teknikler [2-4] bulunmaktadır. Bu tekniklerin amacı, ağ kaynaklarını verimli şekilde kullanarak tıkanıklığı önlemek, aynı zamanda düşük gecikme süresi ile yüksek verimi korumaktır.

Veri merkezi trafiği, genellikle veri akışları açısından karakterize edilmekte ve ölçülmektedir [5, 6]. Veri akışı, kaynakta bulunan uygulamadan hedef ana bilgisayardaki uygulamaya giden dizi pakete karşılık gelmekte ve kurallı beş bileşen – *kaynak IP, hedef IP, kaynak bağlantı noktası, hedef bağlantı noktası ve protokol* – ile benzersiz bir şekilde tanımlanmaktadır. Veri merkezi trafiği ölçüm çalışmaları, veri merkezi trafiğinin iki modlu olduğunu, gecikmeye duyarlı kısa akışlardan ve iş hacmine istekli uzun akışlardan oluştuğunu göstermektedir. Araştırmalarda [7-9], ağ içi anahtarlarda Açık Tıkanıklık Bildirimi (ATB) işaretleme eşiğini dinamik olarak uyarlayarak veri merkezlerindeki uzun akışlar için verimi iyileştirmeye yönelik mekanizmalar önerilmektedir. Veri Merkezleri için ATB destekleyen herhangi bir tıkanıklık denetimi yönteminin kullanması gerekmektedir. Örnek olarak, Veri Merkezi TCP (VMTCP) [8] ve ATB^{+} [9] çalışmalarında uzun akışlar için daha yüksek verim elde edilmektedir.

Veri merkezi ağlarındaki tıkanıklık, mesafe gecikmesi, kuyruk gecikmesi ve paket kaybı gibi sorunları beraberinde getirmektedir [6, 7]. TCP, Veri Merkezi Ağlarının (VMA) tüm gereksinimlerini karşılamakta yetersiz kalmaktadır. Bu nedenle VMA'lar, çeşitli uygulamalardan ve hizmetlerden kaynaklanan trafikleri barındırmak zorunda olmaktadır. Bazı hizmetler, grup video görüşmeleri gibi düşük gecikme süresi gerektiren uygulamalar olabilmektedir [8, 9]. Hizmetler yüksek çoğuşma (patlama) toleransına ihtiyaç duyabilmektedir [10] ve diğerleri Hadoop [11] gibi yüksek verim isteyen uygulamalar olabilmektedir. Ayrıca, sırasıyla gecikmeye ve verime duyarlı olan fare (kısa, küçük) ve fil (büyük) veri akışların aynı anda bulunması veri merkezi ağlarında (VMA) verimlilik çöküşü (TCP incast) ve bant genişliği dağılımındaki adaletsizlik (TCP outcast) gibi çeşitli sorunlara yol açabilmektedir [12]. VMTCP, veri merkezleri için özel olarak tasarlanmış ilk aktarım protokolüdür ve tıkanıklığın boyutunu tahmin etmek için [13] Açık Tıkanıklık Bildirimine (ATB) dayalı bir mekanizma kullanmaktadır. VMTCP, daha sonra bu tahminini tıkanıklık penceresini ölçeklendirmek için kullanmakta ve böylece geleneksel TCP'de olduğu gibi tıkanıklık penceresini yarıya indirerek tıkanıklığın varlığına mantıksızca tepki vermek yerine tıkanıklığın boyutuna tepki vermektedir. Bu şekilde VMTCP, kuyruk doluluk oranını düşük tutmaya yardımcı olmakta ve aynı zamanda tıkanıklığın varlığına değil boyutuna tepki vererek verimin etkilenmemesini sağlamaya çalışmaktadır.

ATB, [14] VMA'ların tüm gereksinimlerini karşılamak üzere son VMA'ları taşıma protokollerinde yaygın olarak kullanmaktadır. Aslında yapılan çalışmalar ATB'nin yüksek

performans elde etmek için güçlü bir araç olduğunu göstermektedir. İnternet Mühendisliği Görev Gücü (İMGG-IETF), kısa süre önce günümüzün VMA'larının genel performansını iyileştirmek amacıyla ATB ve Aktif Kuyruk Yönetim'i yaygın olarak dağıtmak üzere İnternet Taslak (İT, I – Ds), standartları ve RFC'lerle ilgili farklı bir ATB serisi önermiştir [15, 16]. VMTCP [2] ve VMATB [17] gibi modern taşıma mekanizmaları, basitlikleri nedeniyle endüstride ve VMA anhatarlarının üretiminde yaygın olarak kullanılmaktadır. Ancak ATB işaretleme şeması, tek kuyruk görünümünün varsayıldığı TCP için tasarlanmıştır. Bu nedenle endüstri, VMA anahtarlarında bağlantı noktası başına birden fazla kuyruk üretme eğiliminde olmuştur.

Çok hizmetli çok kuyruklu Açık Tıkanıklık Bildirimi

ÇK-ATB, veri merkezlerinde çoklu hizmet ve çoklu kuyruk ile Aktif Kuyruk Yönetimine izin veren teknolojidir. ÇK-ATB, arabellek taşmasını önlemek ve istikrarlı bir ağ sürdürmek için sıkışık kuyruklardaki paketleri seçici olarak geciktirerek veya bırakarak ağ trafiğinin verimli yönetimine izin vermektedir. Hizmetlerin etkin yönetimini sağlamak ve sınırlamaları en aza indirmek için bazı veri merkezlerinde uygulanmıştır [4]. ATB tek kuyruklu senaryolarda uygulandığı için bu eğilim beraberinde aşılması gereken yeni zorluklar getirmiştir. Örneğin, 4-8 kuyruk arabelleğinden gelen tüm paketler, port başına işaretleme yoluyla ATB işareti alabilmektedir. Bunun nedeni, aynı paylaşılan çıkış bağlantı noktasına ait diğer kuyrukların arabellek doluluğu ve sıkışmasıdır. Ancak, Çoklu Kuyruk (ÇK) arabelleklerinden bazıları tıkanıklık yaşamamaktadır. Sonuç olarak Çıkış Portu Arabelleği (ÇPA), daha sonra gelen tüm paketleri işaretleyecek olan ATB işaretleme eşiğine (*K*) kolayca ulaşabilmektedir. Bu durum, kuyruk arabelleğinde tıkanıklık yaşanmayan paketlerin hatalı işaretlenmesine ve dolayısıyla ağ performansının düşmesine neden olacaktır. Daha sonra taşıma protokolü hatalı davrandığında bu kuyrukların akış hızı tıkanıklıktan etkilenecektir. Bu nedenle; zamanlama, gecikme ve aktarım hızı gereksinimleri olumsuz etkilenmektedir.

ÇK-ATB, kuyruk uzunluğuna ve gidiş-dönüş süresine bağlı olarak "*K*" ağırlık faktörünün dinamik hesaplamasını kullanmaktadır. ÇK-ATB, ağ trafiğini yönetmek için birden çok kuyruk kullanan tekniktir ve paketlerin ne zaman işaretleneceğine veya bırakılacağına karar vermek için belirli algoritmalar kullanmaktadır. Bunlar kuyruk uzunluğu, gidiş dönüş süresi vb. farklı parametrelere dayalı olabilmektedir [4]. ÇK-ATB'de ek yük ve kesin olmayan

ölçüm önemlidir. Aynı zamanda bu model yalnızca çevrim tabanlı zamanlayıcılara uygulanabilmektedir. Bununla birlikte Veri Merkezlerindeki birden çok hizmet için Kuyruğa almadan DEcouple paketi işaretleme (DEME) ve genişletilmiş sürümü olan DemePro isimli etkin bir algoritma tanıtmışlardır. DemePro [11, 18] toplam kuyruk uzunluğunu ölçer ve en sıkışık kuyruktaki baş paketi işaretler ve taşma sorununu çözmeye yardımcı olabilir.

Veri aktarımında Düşük Gecikme-Düşük Kayıp (L4S) sağlamak için, sistem kısa Akış Tamamlama Süresine (ATS) gereksinim duymaktadır. Bu amaçla kısa ATS elde etmek için her anahtarda sırasıyla basit *K* işaretleme eşiği tanımlanması gerekmektedir. Ayrıca, düşük kayıp için mikro çoğuşma trafiğinin öğrenilmesi ve kuyruk sıkışmasının ortadan kaldırılması gerekir.

Makine Öğrenimi tabanlı tıkanıklık denetimi üzerine yapılan araştırmalar, yalnızca tıkanıklığı azaltmak için kullanılan denetim stratejisi tasarlamak üzere tek görev senaryolarına odaklanmaktadır. Spesifik olarak çoklu görev durumlarında, Derin Pekiştirmeli öğrenmeye dayalı tıkanıklık denetim modeli önerilmektedir [5]. Denetim modeli, esasında tıkanıklık denetimini ve bununla birlikte yük dengeleme görevini almaktadır. Model, tek görev yöntemiyle karşılaştırıldığında tıkanıklık özelliklerinin ve yük dengeleme özelliklerinin paylaşılan temsilini öğrenerek ağ ortamını daha iyi temsil edebilmektedir. Model ayrıca, ağ trafiği denetimini, tıkanıklık denetimini ve yük dengelemeyi içermektedir. Bu nedenle görevler arasındaki ortak noktalardan ve farklılıklardan yararlanırken birden çok görevi ortaklaşa öğrenmek görev koordinasyonunun maliyetini düşürmeye yardımcı olmaktadır.

Bu tezde, hatalı işaretleme problemini çözmek için bir derin pekiştirmeli öğrenme tabanlı öncelik ATB Haritalama yaklaşımı önerilmiştir. Önerilen yaklaşımda, arabellekten gelen akışlar sınıflayıcı tarafından fare ve fil akışları olarak sınıflandırılmakta ve devamında çıkış port arabelleği eşiğinde işaretlenerek öncelik verilmektedir. Bununla birlikte, çoklu-çift çıkış port arabelleğinde paketlerin hatalı işaretlenmesinden kaçınmak için derin pekiştirmeli öğrenme tekniklerinden yararlanmaktadır. Önerilen yaklaşımın etkinliği ve verimliliği büyük ölçekli bir NS-2 benzetim ortamında test edilmiştir. Elde edilen sonuçlar detaylı olarak incelenerek sonuçları sunulmuştur.

Tezin organizasyonu

Tezin diğer bölümleri şu şekilde organize edilmiştir. Tezin 2. bölümünde Veri Merkezlerinde ve Modern TCP ve farklı denetim mekanizmaları hakkında detaylı literatür incelemesi yapılmış, bu incelemeler çizelgelerle özetlenmiş ve çalışmaya dair arka plan bilgileri verilmiştir. 3. bölümde kullanılan matematiksel modeller ile birlikte tezde kullanılan algoritmalar verilmektedir. Ayrıca Veri Merkez Ağlarında en iyileme problemi çözümünde kullanılan Derin Pekiştirmeli Öğrenme ile ATB-haritalama yaklaşımı sunulmuştur. 4. bölümde kullanılan modeller ve yöntemlerin deneysel çalışmaları ve mevcut diğer mekanizmalar ile karşılaştırmalı sonuçları sunulmuştur. Son bölümde ise tez çalışmasının değeri, kapsamlarının belirtildiği sonuç ve öneriler bölümü yer almaktadır.

2. LİTERATÜR İNCELEMESİ VE ARKA PLAN

Veri Merkezlerindeki tıkanıklık denetimi amacıyla modern TCP mekanizmaları birçok kaynakta incelenmiştir. Araştırmacılar çalışmalarında farklı modeller önermişler, önerilen modellerin üstünlük ve eksikliklerini tartışmışlardır. Ağ trafiğini tanımlamaya yönelik çeşitli yaklaşımlar sunmuşlardır. Tıkanıklık denetiminde geleneksel olarak taşıma katmanı bağlantı nokta bilgisi ve paket yükünün doğrudan denetimi kullanılmaktadır. Bununla birlikte bölüm 2.1. ve bölüm 2.2'de verilen iki yaklaşımın çeşitli eksiklikleri, Yapay Zekâ algoritmaları, daha özel olarak Makine Öğrenimi (MÖ) algoritmaları literatürlerde ele alınmıştır. Bu çalışmalar, veri merkezi tıkanıklık denetiminde ATB ile ilgili mevcut çalışmaların özetlenmesini kapsamaktadır ve elde edilen bazı önemli bulguları içermektedir. ATB'nin veri merkezlerinde tıkanıklığı azaltmada ve ağ performansını iyileştirmede etkili olduğu gösterilmiştir. Örneğin; Wang ve diğerleri tarafından yapılan bir çalışmada [19] ATB'nin veri merkezi ağlarında verimi iyileştirdiğini ve paket kayıp oranını azalttığını ortaya koymuştur. ATB, ağ performansını daha da iyileştirmek için diğer tıkanıklık denetim mekanizmalarıyla birlikte kullanılabilmektedir. Alizadeh ve diğerleri veri merkezlerinde ATB'yi dinamik bir kaynak sağlama ile birleştirme ağın genel performansını iyileştirmiştir [2]. ATB, ağın trafik modellerindeki değişikliklere hızlı bir şekilde yanıt vermesine izin verdiği için özellikle yoğun trafik yaşayan veri merkezi ağlarında faydalı olabilmektedir. ATB'nin uygulanabileceği farklı yollar vardır. Örneğin VMTCP gibi bazı ATB algoritmalarının veri merkezi ağlarında diğerlerinden daha etkili olduğu görülmüştür. ATB'nin veri merkezlerindeki etkinliğinin belirli ağ ortamına ve trafik modellerine bağlı olacağı dikkat çekmektedir. Ayrıca, ATB'nin faydaları ile onu veri merkezinde uygulamanın zorlukları arasındaki ödünleşimi dikkate almakta yarar bulunmaktadır.

2.1. Veri Merkezlerinde Tıkanıklık Denetimi

Verilerin artması ve gerçek zamanlı akış için yüksek talep nedeniyle oluşan tıkanıklık denetimi, daha karmaşık hâle gelmektedir. Bu durum ağda mesafe gecikmesi, kuyruk gecikmesi ve paket kaybından oluşan gecikme veya gecikmeyi içeren üç ana soruna neden olmaktadır [6, 20, 21]. Paketlerin kaynaktan hedefe aktarılması için geçen süre mesafe gecikmesi olarak tanımlanmaktadır. Gecikme süresi ise yayılma gecikmesine neden olmaktadır. Veri Merkezi ağlarında bu yayılma gecikmesi 100 ms'ye kadar çıkabilmektedir [22, 23]. Onlarca anahtar/yönlendirici kuyruğunun arabelleğinden gelen kuyruk gecikmesi

normal bir olgudur. Genellikle yüzlerce milisaniye kadar artan bu süre farklı durumlarda ağdaki yoğun trafiği karşılayabilmek için gereklidir [24].

VMA ilk yıllarında, eski TCP tıkanıklık denetimi [25] ve VMA veri iletişimi için doğrudan İnternetten benimsenmiştir. Bu protokoller tipik olarak tıkanıklık geri bildirimi için paket kaybı sinyalini kullanmaktadır. Paket kayıplarının genellikle iki nedeni vardır. Bunlar: aktarım sırasında paket bozulması veya yetersiz arabellek kapasitesi nedeniyle ağ tıkanıklığıdır. Çoğu ağda paket bozulması son derece nadirdir. Diğer yandan, üçlü yinelenen ACK'ler alındığında paket kaybına ve yeniden sıralama gecikmesine neden olmaktadır. Bu nedenle, hızlı yeniden iletimi etkinleştirmek için bir zaman aşımı tetikleyicisi olabilmektedir. Her iki durumda da yeniden iletim gecikmesi meydana gelmekte ve bu süre 200 ms'den fazla sürmektedir [26]. Neticede performans ve gecikme süresindeki bozulma, ağ kaynaklarının yanı sıra enerjiyi de boşa harcamaktadır. Ayrıca, ağ kalitesi temel olarak tıkanıklık denetimine dayanmaktadır ve tıkanıklık denetimi geniş çapta çalışılan geri bildirimin doğruluğuna bağlıdır [27, 28]. Ancak, VMA'lerde Gidiş Dönüş Süresinin (GDS), 40 ms ve daha aşağısında olması gerekir. Bu nedenle, tıkanıklık denetimi için en popüler bildirim algoritması olan Açık Tıkanıklık Bildirimi [14] ve Yeni-ATB, geleneksel ATB geri bildiriminin [29] geliştirilmiş sürümü olarak ortaya çıkmıştır.

Görüntülü grup araması, internet aramaları gibi bazı uygulamalar ve ağlar düşük gecikme süresi gerektirmektedir [8]. Hadoop gibi bazı diğer hizmetler yüksek verime ihtiyaç duymaktadır [11, 18]. Ayrıca, tüm bu hizmetler iki tür hizmete dayanmaktadır. Bunlar arama motoru gibi kullanıcılar için minimum gecikme ile çevrimiçi hizmetler sunan Skype ve Google, kullanıcılara kaynaklar sağlayan Amazon bulutu ve Microsoft Azure'dir. Daha iyi Hizmet Kalitesi (QoS), yüksek performans ve düşük gecikme süresine sahip olmak için tüm trafik türlerinde tıkanıklık denetimi iyi yönetilmelidir [30, 31].

Kayba dayalı tıkanıklık denetimi, basitliği nedeniyle geniş çapta kabul edilmesine rağmen anahtar arabelleğini doldurma ve aşırı paket kayıplarına neden olma eğilimindedir. Bu nedenle, VMA'deki sert düşük gecikme gereksinimlerini karşılayamamaktadır. Bu gözlemden hareketle yeni ufuklar açan VMA aktarım tasarımı VMTCP [2], 2010 yılında önerilmiştir. VMTCP, Açık Tıkanıklık Bildirimi [10] sinyali ile ağ tıkanıklığını tespit eder ve ACK paketine dayalı olarak tıkanıklığın boyutuna göre ATB işaretleri ile tepki verir. 2010 yılından günümüze kadar geçen sürede, VMA'da yüksek performanslı veri iletişimi için çok

sayıda tıkanıklık denetim mekanizması önerilmiştir [32, 33]. IETF yakın zamanda İnternet taslakları (I-Ds–Internet drafts) ve RFC'ler [26] ile ilgili çeşitli ATB serisi önermiştir. IETF çalışanları, günümüzdeki ağın genel performansını iyileştirmek için ATB ve AKY'yi geniş çapta kullanır. Uygulamalarda RFC7567, ATB ve AKY'nin standardizasyonu, test edilmesi ve devreye alınması şiddetle tavsiye edilmektedir [15]. RFC 7928; ATB [16] için RFC/ID tarafından üretilen iş hacminde gecikmenin azaltılmasına ve maksimize edilmesine yardımcı olacak yönergelere sahiptir [16]. Ayrıca, dalgıç RFC'leri, İnternet taslakları, AKY anahtarları, yönlendiricileri ATB'yi iyileştirmek ve etkinleştirmek için çözüm önerileri sunmaktadır [21, 22].

Veri Merkezleri veri aktarımı hakkında yapılan araştırmalar, Veri Merkezi ağlarında uzun akış verimi ile kısa akış gecikmesi arasında ödünleşim olduğuna işaret etmektedir [2, 34]. VMTCP, kısa akışlar için düşük gecikme süresi sağlarken aynı zamanda uzun akışların istenen en iyilenmiş değerde olmasa da iş hacmine ulaşmasını sağlayarak bunlar arasında bir denge kurmaktadır. [2]'te sunulan sonuçlar, uzun akışların tek başına düşünülmesi ve işaretleme eşiğinin uzun akış verimini en üst düzeye çıkarmak amacıyla ayarlanması durumunda uzun akış veriminde %10 ila %15'lik bir iyileşme yapılabileceğini göstermektedir. Aynı durum, uzun akış verimi ve kısa akış gecikmesi arasında karşılık gelen değiş tokuşla bir dizi olası ATB işaretleme eşiğinin tanıtıldığı [34]'te de vurgulanmıştır. İşaretleme eşiği için yüksek olan değerler, uzun akışlar için daha yüksek verimle sonuçlanmaktadır. Ancak bu durum kısa akış gecikmesinde artışa yol açabilmektedir. [2, 34]'e dayalı olarak burada dikkat edilmesi gereken ana nokta, ATB işaretleme eşiği yalnızca uzun akışlar dikkate alınarak ayarlandığında uzun akış veriminde önemli bir gelişme için yer olduğudur. Önerilen mekanizma, darboğaz bağlantılarında ATB işaretleme eşiğini dinamik olarak uyarlamak için bir çerçevenin ana hatlarını çizmektedir. Şöyle ki; (1) kısa akışların sayısı uzun akışlardan fazla olduğunda, uzun akış verimini düşürmeden kısa akışlar için düşük gecikmeyi garanti eden bir alt sınır değerine ayarlanır ve (2) uzun akışlar baskın olduğunda uzun akış verimini daha da geliştirmeye yardımcı olacak bir üst sınır değerine ayarlanır.

Endüstri ve akademi tarafından önerilen çeşitli tıkanıklık denetim mekanizmalarının ele alınışı konu hakkında bazı fikirler sunabilmektedir.

Endüstride:

- RED ve PI gibi Aktif Kuyruk Yönetimi (AKY) algoritmaları, yönlendiricilerde ve anahtarlarda yaygın olarak kullanılmaktadır.
- TCP-dostu hız denetim (TCP-friendly rate control) protokolü, kablosuz ağlarda yaygın olarak kullanılır [25].
- Veri Merkezi TCP'si, veri merkezi ağlarında TCP trafiğinin performansını artırmak için kullanılır.

Akademide:

- Oransal Entegre Denetleyici Gelişmiş (Proportional Integral controller Enhanced-PIE) algoritması, veri merkezi ağları için tasarlanmıştır.
- Denetimli Gecikme (Controlled Delay) algoritması, kablosuz ağlar ve AKY'de kullanım için tasarlanmıştır.
- Ağ Kodlaması (Network Coding), kablosuz ağların ve gecikmeye dayanıklı ağların performansını artırmak için kullanılır.
- Düşük Ekstra Gecikmeli Arka Plan Aktarımı (Low Extra Delay Background Transport), BitTorrent benzeri dosya paylaşım sistemlerinde kullanılmak üzere tasarlanmıştır.

Veri merkezleri arası ağlar için veri iletişimini optimize etmeyi amaçlayan tıkanıklık denetim mekanizmaları da vardır. Örneğin, DBR [35] öncelikle kurumsal Geniş Alan Ağı (GAN: Wide Area Network-WAN) için önerilmiştir. Burada temel fikir, gelişmiş ağ algılama yaklaşımı (örneğin, kesin bant genişliği ve GDS tahmini) yardımıyla teorik olarak en uygun noktada [36] çalışmaktır. Gecikme gelişimini gözlemleyerek arabellek doldurucuları algılar ve gecikmeye duyarlı ve TCP-rekabetçi mod arasında geçiş yapılır. Bu geniş alan taşıma protokolleri, karmaşık WAN ortamında çalışmak için genellikle ağ anahtarlarından çok az yardım (örneğin ATB desteği olmaması durumu) alır.

Bu çalışmaların endüstri ve akademi tarafından önerilen birçok tıkanıklık denetim mekanizmasından sadece birkaç örnek olduğu ve yenilerinin önerilmeye ve geliştirilmeye devam ettiğini belirtmekte yarar bulunmaktadır. Ek olarak, bu algoritmaların uygunluğu, belirli ağ ortamına ve trafik modellerine bağlı olmaktadır. Endüstri ve akademi tarafından önerilen ve temele özelliklerine bazı tıkanıklık denetim mekanizmaları Çizelge 2.1'de listelenmiştir.

Çizelge 2.1. Tıkanıklık denetim mekanizmalarının kullanım alanları ve özellikleri

Şema	Akademi	Endüstri	TCP tabanlı	İşletim Katmanı	Tıkanıklığa tepkisi	Hata Toleransı	Değişiklik Gereksinimi		
							gönderici	alıcı	anahtar
VMTCP	+	+	+	K4	Etkin		+	+	ATB
CC-NQUIC	-	+	+	K3	Etkin		+	+	+
PIE	+	+	+	K3, K4	C/Y	-	+	+	ATB
RCP	+	+	+	K4	Etkin		+	+	ATB
VMCP	+	+	+	K4	Etkin		+	+	ATB
XCP	+	+	+	K4	Etkin		+	+	+
DeTail	-	+	-	K2, K3, K4	Etkin		+	+	ATB
pFabric	+	+	+	K4	Etkin	+			+
CUBIC-ECN	+	+	+	K3, K4	Etkin		+	+	+
XCP	+	+	C/Y	K3, K4	C/Y				ATB
RepFlow	+	-	+	K5	Etkin				
WFQ	+	-	+	K3,K4	Etkin	+			+
RPS	+	-	+	K3, K4	Etkin	+			+
XMP	+	-		k4	Etkin	+	+	+	+
CC-NQUIC	-	+		K4	Etkin		+	+	+
PCC	+	+		K4	Etkin		+	+	+
D^2TCP	+	+	+	K4	Etkin		+	+	ATB
L^2 VMT	-	+	+	k4	Etkin			+	
BBR-ECN:	+	-	C/Y	K3	Etkin		+		+
BCN	+	+	C/Y	K2	Etkin		+		+
QCN	+	-	C/Y	K2	Etkin		+	+	+
FATB	+	-	C/Y	K2	Etkin		+	+	+

Çizelge 2.2'de Protokol şemalarının performansında açıklık, süre bitimi, akış önceliği, çoğuşma ve TCP ile uyumluluk açısından karşılaştırılması ve Çizelge 2.3'te Protokol şemaların anahtar ve kaynak işlemleri verilmektedir.

Çizelge 2.2. Protokol şemalarının performans açısından karşılaştırılması

Şema	Açıklık	Süre bitimi	Akış Önceliği	Çoğuşma Toleransı	TCP ile Uyumluluk
VMTCP	Evet	Hayır	Hayır	Yüksek	Evet
PIE	Evet	Evet	Hayır	Düşük	Evet
CUBIC-ECN	Evet	Hayır	Hayır	Yüksek	Evet
L^2 VMT	Hayır	Hayır	Evet	Düşük	Evet
D^2TCP	Hayır	Evet	Evet	Düşük	Hayır
BBR-ECN:	Evet	Hayır	Hayır	Düşük	Evet

Çizelge 2.3. Protokol şemalarının anahtar ve kaynak işlemleri

Şema	Anahtar İşlemleri	Kaynak İşlemleri
VMTCP	ATB	GDS başına a güncellemesi yapar
PIE	ATB	Tıkanıklığı gidermede ortalama kuyruklama gecikmesinin hedeflenen değerde olması için Oransal İntegral denetleyiciyi ayarlar
CUBIC-ECN	ATB	CUBIC tıkanıklık bildirim denetim algoritmasından yararlanır
L^2 VMT	ATB	Her GDS'de akış akımı ağırlığını w_c hesaplar
D^2 TCP	ATB	Son teslim tarihine yakın akışlara öncelik vermek için yakınlık katsayısı d'yi hesaplar
BBR-ECN	ATB ile fantom kuyruklama	q-bayt iletimi düzenler

2.2. Modern TCP Mekanizmaları

Veri Merkezi TCP

Veri Merkezi TCP veri merkezi ağlarında TCP performansını iyileştirmeyi amaçlayan tıkanıklık denetim algoritmasıdır. Düşük gecikme süresi ve yüksek verim sağlamak için ATB ve AIMD'nin bir birleşimini kullanmaktadır. Çalışmalar, VMTCP'nin veri merkezi ortamlarında paket kayıp oranını azaltabildiğini ve ağ kaynaklarının kullanımını artırabildiğini göstermiştir. [2] Tıkanıklık Çıkarma (Congestion Exposure-ConEx) [37] gibi modern TCP mekanizmalarının daha iyi performansa sahip olması için önerilmiştir. VMTCP ve ConEx algoritması, her GDS'de [22] birden fazla işaretin alınması durumunda klasik ATB'den [38] çok daha doğru geri bildirime ihtiyaç duymaktadır. Örneğin; kaynak ConEx'i destekleyen standart TCP, tıkanıklık denetimini kullanıyorsa veya ConEx ile ConEx olmadan VMTCP tıkanıklık denetimini kullanıyorsa tüm bu senaryolar doğru tıkanıklık bilgilerini yeniden yankılamak için her GDSde fazladan bilgi kullandığından aynı sonucu alacaktır. Bununla birlikte, mevcut bant genişliğini tahmin etmek ve gönderme hızını buna göre ayarlamak için Darboğaz Bant Genişliği (Bottleneck Bandwidth) ve GDS kullanılmıştır. Çalışmalar, bu yaklaşımların paket kayıp oranını azaltabildiğini ve yüksek bant genişliğine sahip ağlarda TCP'nin genel performansını artırabildiğini göstermiştir [39].

Veri Merkezi TCP, geliştirilmiş ATB'ye ve en son teknoloji mekanizmalardan birine dayanmaktadır. VMTCP'de tıkanıklık, ACK bayrağına sahip paket alındığında hedefe

karşılık gelen ACK de ATB yankı bayrağını ayarlayacaktır. Tüm ACK'leri aldıktan sonra gönderen, tıkanıklığı bir "α" katsayısı ile tahmin etmek için bir tıkanıklık deneyimi bayrağına sahip olan ACK'leri hesaplar. "α" değeri 1'e ulaşırsa; tıkanıklık penceresi TCP ile aynı olacak, ağda düşük değerli bir tıkanıklık olduğunda "α" değeri sıfıra yakın olacak ve tıkanıklık penceresi fazla azalmayacaktır [2]. Ancak bu durumda VMTCP, paket kaybının oluşabileceği kablosuz ağlarda sorun yaşayabilir. Eğer ACK'ler kaybolursa bu durum doğruluk nedeniyle VMTCP'yi etkileyecek ve tıkanıklığı yeniden yankılamak için fazladan bitler kullanacaktır.

İleri Açık Tıkanıklık Bildirimi (İATB)

İleri Açık Tıkanıklık Bildirimi (İATB) [40]; kaynağa giden yoldaki trafiği her hedefin yardımıyla periyodik olarak araştıran, kapalı-çevrim ve uçtan-uca bir mekanizmadır. İleri yoldan gelen mevcut bant genişliği; araştırma hızının değerinden düşükse araştırmadaki oran ileri yolu değiştirecektir, aksi hâlde aynı kalacaktır [41]. Ayrıca, anahtar aynı oranı bildirmedikçe İATB'deki açıklık iyi tasarlanmış demektir.

Geriye doğru Tıkanıklık Bildirimi

Geriye doğru Tıkanıklık Bildirimi (GTB) [41], kuyruk örnekleme tabanlı bir tıkanıklık bildirimi algoritmasıdır. GTB'deki anahtar, yalnızca olumsuz bir geri bildirim göndermekle kalmaz, aynı zamanda ek yüke neden olan pozitif oran bilgisi de göndermektedir. Bu nedenle, Nicemlenmiş Tıkanıklık Bildirimi (NTB)'nin yalnızca negatif tıkanıklık geri bildirimi kullanan kuyruk örnekleme tıkanıklık bildirim algoritmasının kullanıldığı bir GTB önerilmektedir [42, 43]. GTB, uç nokta, gerçek zamanlı veya kritik veriler gibi belirli trafik türlerine daha az önemli trafiğe göre öncelik verebilir, trafiği ağın sıkışık alanlarından uzağa yeniden yönlendirmek için yönlendirme protokollerini ve belirli zamanda ağa erişebilen cihazların sayısını sınırlamak için erişim denetim mekanizmalarını kullanabilir.

Nicemlenmiş ve Adil Nicemlenmiş Tıkanıklık Bildirimi (NTB, ANTB)

Nicemlenmiş Tıkanıklık Bildirimi (NTB) her ikisinin de dinamik olduğu tıkanıklık noktası ve hız sınırlayıcısına sahip iki ana bölümden oluşmaktadır. Tıkanıklık noktasında fazla abone olunan bağlantıya eklenen bir arabellek, gelen paketleri örneklemektedir. Ayrıca hız

sınırlayıcısı, tıkanıklık noktasından alınan tıkanıklık noktasının mesajına göre hızı düşürmektedir. Aynı zamanda kaybolan bant genişliğini serbest olarak kurtarmak ve ağı mevcut ekstra bant genişliğini araştırmak için hızı artırmaktadır [43, 44]. Adil Nicemlenmiş Tıkanıklık Bildirimi (ANTB), ağdaki akışlar arasında adalet sağlamayı amaçlayan NTB mekanizmasının uzantısıdır. ANTB, veri ağlarında, özellikle Veri Merkezi ağlarında kullanılan tıkanıklık denetim mekanizmasıdır [45]. NTB'deki tüm paylaşım kaynakları arasında bağlantı kapasitesinin adil dağılımını geliştirmek için önerilen ANTB, darboğaz bağlantı kapasitesinin adil paylarının üzerindeki oranlarda paketler gönderen tüm akış kaynaklarına tıkanıklık bildirimi mesajı vermektedir. Sonuç olarak; ANTB, GTB, NTB ve ANTBnin tümü tıkanıklık bildirimine dayalı kuyruk örnekleme mekanizmalarıdır ve ANTB ile NTB arasındaki fark sadece tıkanıklık noktası mekanizmasındadır.

Çoklu-Hizmetli Çoklu-Kuyruklu Veri Merkezleri

Çok Hizmetli Çok Kuyruklu Veri Merkezlerinde (ÇK-ATB) ATB'nin etkinleştirilmesi [4], veri merkezlerinde çok kuyruklu senaryosun etkinleştiren ilk protokoldür. ÇK-ATB 'de her kuyruğun kendi ATB işaretleme eşiği vardır. Ayrıca ÇK-ATB, ağırlıklı adil paylaşım sağlamak için her kuyruktaki bağımsızlığın avantajını kullanmaktadır. Bu, statik adil paylaşım ağırlığına bağlı kalmak yerine ÇK-ATB'nin dinamik ağırlıklı adil paylaşım oranına göre her kuyruk için *K*'yi bağımsız olarak ayarlayabileceği anlamına gelmektedir. ÇK-ATB, ağırlık ve çevrim süresi ile belirtildiği gibi her kuyruk için dinamik bir *K* değeri hesaplamaktadır. Ancak VMA'lerin üretiminde, akış planlaması için mevcut tüm öncelik sırası anahtarları çok sınırlıdır (yalnızca 2 veya 3). Bu sınırlama pratikte ÇK-ATB yaklaşımının performansını ciddi şekilde azaltmaktadır. VMA trafiği çoğuşmalıdır ve bu da doğru çevrim zamanını ölçmeyi zorlaştırmaktadır. Ölçümler genellikle anahtarlar için ihmal edilemez ek yüke neden olmaktadır. Ayrıca, ÇK-ATB sadece çevrim tabanlı zamanlayıcı için çalışır ve diğerlerini desteklememektedir [4].

VMA’da paket işaretlemeyi birden çok hizmet için kuyruğa alma işleminden ayırma (Decouple packet marking from enqueuing for multiple services-DEME) amacıyla geliştirilmiş olan uygulamanın genişletilmiş sürümünde (DemePro) ÇK-ATB için hafif bir algoritma tanıtılmaktadır. DemePro, ÇK-ATB gibi sıranın çevrim süresini periyodik olarak ölçmek ve kuyruk başına *K* ayarlamak yerine, toplam kuyruk uzunluğunu ölçmekte ve en sıkışık kuyruktaki baş paketi işaretlemektedir. Diğer bir deyişle, paket arabelleğini işlemesi

beklenirken işaretleme kararı tek kuyruk uzunluğu yerine toplam kuyruk uzunluğuna bağlı olmaktadır. Ayrıca, tüm kuyruk uzunluğunun ortalaması *K*'den büyük olursa DemePro baş paketini sıkışık kuyrukta işaretlemeye başlamaktadır. Bu şekilde kuyruklar arasındaki gerekli açıklık garanti edilebilir [11, 18]. Ancak bu ölçüm, anahtarlar için ek yüke neden olmaktadır. Ayrıca, toplam kuyruk uzunluğunu ölçmek sıkışık kuyrukta taşmaya ve baş paketini işaretlemek de verimin düşmesine neden olabilmektedir.

Açık Öncelikli Bildirim (AÖB)

Açık Öncelikli Bildirim (AÖB) [46], yeni bir zamanlama mekanizmasıdır. AÖB'de ana fikir, bant genişliğinin tamamen en yüksek önceliğe sahip akışa tahsis edilmesidir. Bu nedenle, her bağlantı yalnızca daha yüksek ve daha düşük öncelik gerektirmektedir. En yüksek öncelikli akış; yüksek kuyruğa, kalan akışlar ise düşük kuyruğa atanmaktadır. Yüksek kuyruk akışları bittiğinde bir sonraki istenen en yüksek önceliğe sahip akışlar, yüksek kuyruğa yükseltilir ve bu işlem böyle devam ettirilir. Ancak bu sistem, fil kuyrukları olan düşük öncelikli akışlar için açıklık getirecektir. Ayrıca, verim tatmin edici olmamaktadır.

Seçici Anlayışsızlık ile Port Başına İşaretleme (SKPBİ)

Son zamanlarda, Seçici Körlük ile Port Başına İşaretleme (SKPBİ) [47] önerilmiştir. Araştırmacılar, VMA'lerde bağlantı noktası başına çoklu-kuyruğa ilişkin mevcut sorunları tartışmışlardır. Çalışmalar, ATB'nin aynı anda düşük gecikme ve yüksek verim elde edebilen tek kuyruk senaryolarında hizmet kalitesi için güçlü araç olduğunu göstermektedir. Ancak; bağlantı noktası başına işaretleme için geliştirilen ECN, bir endüstri trendi olduğu için çok kuyruklu senaryolara doğrudan uygulanamamaktadır. Çünkü gecikme, aktarım hızı ve zamanlama politikası arasında en azından tanımlanan metriklerin uygulanamamasına neden olmaktadır. SKPBİ'de, ÇPA'deki tıkanıklık nedeniyle paket işaretlenmesi durumunda işaretlenen paket kuyrukta tıkanıklık yaşamazken; SKPBİ, sıkışık olmayan kuyrukları korumak için paketin işaretini veya akışın geri çekilmesini iptal edebilir. SKPBİ'ye sahip olarak yalnızca her iki durum da tutuluyorsa anahtar ATB ile işaretlemektedir. Eğer, bağlantı noktası arabelleğinin uzunluğu, bağlantı noktası başına eşikten yani, bağlantı noktası işaretlemesinden daha büyükse ve kuyruk arabelleğinin uzunluğu da kuyruk başına filtre eşiğinden daha kısa değilse seçici körlük oluşmaktadır. Ancak, veri merkezindeki GDS'nin çok kısa olmasından ve GDS'yi ölçmek için gelişmiş donanıma ihtiyaç duyulduğundan ATB

eşiğini gerçek zamanlı olarak belirlemek GDS ile zor olmaktadır. Ayrıca, SAPA'daki ek yük ihmal edilebilir düzeyde değildir ve taşma olgusuna katkıda bulunabilmektedir.

Tıkanıklık denetim mekanizmalarını sınıflandırmanın amacı, ağ üzerinden veri akışını yönetmek için kullanılabilecek farklı yöntemleri anlamak ve organize etmek için yol sağlamaktır. Mekanizmaların detaylı olarak amaçlarına göre sınıflandırılması Çizelge 2.4'te gösterilmektedir. Ağ yöneticileri ve tasarımcılar, farklı mekanizma türlerini anlayarak, belirli bir ağ veya uygulama için hangi yaklaşımın en uygun olduğu konusunda bilinçli kararlar verebilirler.

Çizelge 2.4. Mekanizmaların amaçlarına göre sınıflandırılması

Şema	Kısa Akışlar için Ortalama (O) Kuyruk (K) ATS'yi Azaltma	Yok/Az Değişiklikle Değişiklik VMA Altyapısı	Doğruluk	Hedeflerine Uygun Akış Sayısını Maksimum Düzeye Çıkarma	Yüksek Çoğuşma Toleransı	Düşük/ Sıfır Arabellek Doluluğu	Yüksek Verim Sağlamak	TCP Incast Azaltma
VMTCP	O ve K				+		+	+
CC-NQUIC		+	+					
PIE	O				+		+	+
RCP	O	+						
VMCP	O ve K					+	+	
XCP	O ve K	+						+
DeTail	K							
pFabric							+	
CUBIC-ECN			+				+	
XCP	O ve K	+					+	
RepFlow	O ve K	+						
WFQ	O	+					+	
RPS							+	
XMP	O		+				+	
CC-NQUIC				+	+		+	
PCC	O			+				
D^2TCP		+		+				
L^2 VMT				+			+	
BBR-ECN:	O							+
BCN			+			+		
QCN		+	+					
FECN								

Her mekanizma türünün kendine özgü üstünlük ve sakıncaları vardır. Bir mekanizma belirli ağ türleri veya trafik modelleri için daha uygun olabilir. Örneğin, akış tabanlı mekanizmalar, düşük bant genişliğine sahip, yoğun trafiği yönetmek için çok uygun olabilirken, pencere

tabanlı mekanizmalar, yüksek bant genişliğine sahip, kararlı durum trafiğini yönetmek için daha etkili olabilir.

Tıkanıklık denetim mekanizmalarının sınıflandırılması, ağ yöneticilerinin ve tasarımcıların farklı senaryolarda kullanmak üzere uygun mekanizmayı seçmesini ve buna göre ağ performansını optimize etmesini kolaylaştırır. Ağda tüm tıkanıklık denetim mekanizmaları çalışma mekanizmalarına göre sırasıyla Çizelge 2.5'te listelenmektedir.

Çizelge 2.5. Tıkanıklık denetiminin çalışma mekanizmalarına göre sınıflandırılması

Şema	ATB Geri Bildirimi	Pencere Ayarı	Yük dengeleme	Önleyici Zamanlama	Kısa Akışlar için Önceliklendirme	Çoklu Yoldan Yararlanma	Çapraz Katman Tasarımı	Donanım Modifikasyonu
VMTCP	+	+(tpnc)						
CC-NQUIC	+		+					
PIE	+				+			
RCP	+	+(tpnc)			+			
VMCP	+							+(anahtar)
XCP					+			
DeTail	+		+			+	+	
pFabric						+		
CUBIC-ECN			+			+	+	+(anahtar)
XCP			+			+	+	
RepFlow						+		
WFQ			+			+	+	+(anahtar)
RPS			+			+		
XMP	+		+			+		
CC-NQUIC					+			
PCC		+(tpnc)		+	+			
D^2 TCP		+(tpnc)			+			
L^2 DCT	+	+(tpnc)			+			
BBR-ECN:		+(tpnc)					+	+(anahtar)
BCN								
QCN								+(anahtar)
FECN								

2.3. Açık Tıkanıklık Bildirimi genel sistem mimarisi

Bu bölümde tezin ön hazırlıkları ve hedefleri tartışılmış, mevcut sistem mimarisi ve çok-kuyruklu ATB protokolü kullanımında ortaya çıkan zorluklar ele alınmıştır. Veri merkezi tıkanıklığı denetim mekanizmalarını, veri merkezlerindeki ağ tıkanıklıklarını yönetmek ve hafifletmek için kullanılabilecek çeşitli tekniklerden bahsedilmektedir. Bunlardan bazıları aşağıdaki gibidir;

- *Akış denetimi:* Bu mekanizma, ağda gönderilen ve alınan veri miktarını izlemekte ve aşırı yüklemeyi önlemek için veri aktarım hızını ayarlamaktadır.
- *Trafik şekillendirme:* Bu mekanizma önceden belirlenmiş bir kalıba uyulmasını sağlamak için ağda gönderilen ve alınan trafiğin miktar ve oranlarını düzenlemektedir.
- *Kaynakların dinamik olarak sağlanması:* Bu mekanizma, mevcut ihtiyaçlara göre farklı uygulamalara ve hizmetlere tahsis edilen kaynak miktarını (bant genişliği ve işlem gücü gibi) ayarlamaktadır.
- *Tıkanıklıktan kaçınma:* Bu mekanizma olası ağ tıkanıklıklarını erken algılayabilmek ve önleyebilmek için gerekli olan Rastgele Erken Tespit (RED) vb. teknikler kullanmaktadır.
- *Hizmet Kalitesi*: Bu mekanizma farklı ağ trafiği türlerini, önemlerine göre önceliklendirerek, kritik uygulama ve hizmetlerin kaynaklarının gerektiği şekilde kullanılmasını sağlamaktadır.
- *Aktif kuyruk yönetimi:* Bu mekanizma kuyruk boyutunun yönetilmesi, tampon taşmasının önlenmesi için kuyruk bırakma ve Rastgele Erken Tespit (RED) gibi bazı teknikler kullanmaktadır.

Geleneksel ATB mimarisi aşağıdaki bileşenleri içermektedir;

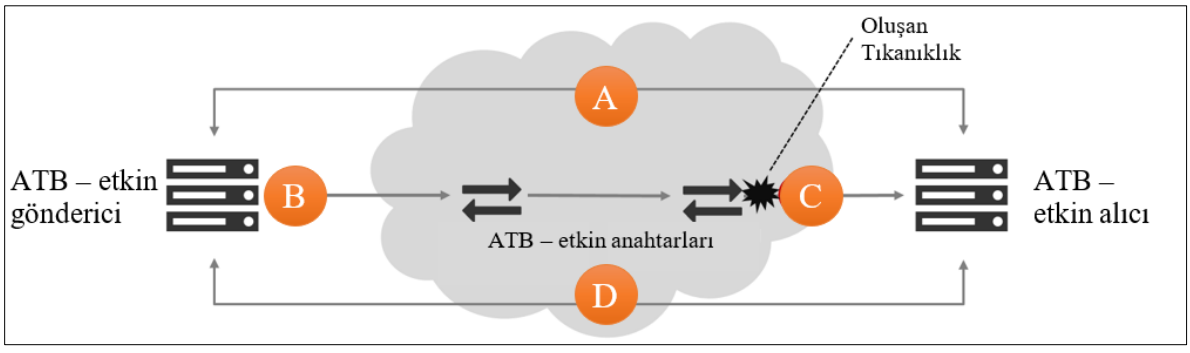
- ATB özellikli yönlendiriciler, ağ tıkanıklığı sebeplerinin tespiti ve paketlerin bir ATB (Açık Tıkanıklık Bildirimi) koduyla işaretlemesini sağlama özelliğine sahiptir.
- ATB özellikli uç noktalar, ağ bağlantısının ucundaki (bilgisayar veya sunucu gibi) ATB kodlarını yorumlayabilen cihazlardır.
- ATB özellikli taşıma protokolü, uç noktalar arasında veri iletmek için kullanılan ve ATB kodlarına yanıt verebilen bir protokoldür (TCP veya SCTP).

ATB özellikli bir uç noktadan, ATB özellikli yönlendiriciye paket gönderildiğinde, yönlendirici ağ tıkanıklığı algılarsa paketi ATB koduyla işaretler. Paket, ATB etkin bitiş

noktasına ulaştığında uç nokta, ATB koduna ağa paket gönderim hızını azaltarak yanıt verecektir. Bu, ağ tıkanıklığını azaltmaya ve genel ağ performansını iyileştirmeye yardımcı olmaktadır.

Arabellek yönetimini denetlemek için gerekli iki mekanizma bulunmaktadır. Bunlar, Aktif Kuyruk Yönetimi ve Kuyruk Zamanlama mekanizmalarıdır. Paketler arabelleğe ulaştığında, ağ denetleyicisinin olabilecek en hızlı ve en kolay çözümleri bulması için ön işlem yapması gerekmektedir. Veri merkezi anahtarları, kuyruk arabelleğinde AKY, ÇPA'da KZ ile çalışmaktadır. AKY kuyruğun sonunda çalışmakta, gelen paketlerin kesrini rastgele ve/veya gelen tüm paketleri işaretlemektedir. Arabellekteki ortalama ya da anlık kuyruk uzunluğu maksimum eşiği aştığında, bu durum tıkanıklık deneyimi olarak işaretlenmektedir. Ancak arabelleğin kuyruk uzunluğunun minimum eşikten daha az olması halinde bu durum yetersiz olarak işaretlenmektedir. Böylece, taşıma protokolü Tıkanıklık Penceresini (TP) azaltmakta veya artırmaktadır. KZ, süreçlerin özelliklerine göre planlanıp sınıflandırılabildiği veri merkezlerinde ÇPA'nın önünde çalışmaktadır.

Paket alıcı uç noktasına ulaştığında, Oluşan Tıkanıklık (OT) işareti, alıcıya ağ tıkanıklığı olduğunu söyler. Alıcı sonraki gönderene ağda tıkanıklık olduğunu belirten bir mesaj gönderir (yankılar). Gönderen, tıkanıklık bildirim mesajını kabul eder ve iletim hızını düşürür. Şekil 3.1, geleneksel ATB yapısının ağ tıkanıklığını azaltmak için nasıl çalıştığını özetlemektedir. Şekil 3.'de A) bağlantı kurulması sırasındaki ATB, B) EUT (ATB Uyumlu Taşıma) seti, C) OT seti ve D) alıcıda oluşan tıkanıklığı OT'ye yansıtır.



Şekil 2.1. Geleneksel ATB Yapısı

2.4. Çok-Kuyruklu ATB

ATB İşaretleme eşiğini (*K*) ayarlamak ve paketleri işaretlemeye karar vermek için kullanılan parametreler ve kullanımındaki zorluklar aşağıda özetlenmiştir.

- *Port-başına*, bağlantı noktası için mevcut herhangi bir *K*'yi ayarlar (Her kuyruktan bağımsız değil). Bu nedenle, bireysel kuyrukları ölçmek için bir eğilim yoktur. Ayrıca, doğruluk kolayca garanti edilememektedir.
- *Hizmet-havuzu başına,* paylaşılan çıkış bağlantı noktasındaki tıkanıklık nedeniyle paketleri işaretler (bağlantı noktaları VMA anahtarlarında paylaşılır), işaretlenen paketlerde kuyrukta tıkanıklık yaşanmamaktadır. Bu fenomen, paketlerin sıkışık olmayan kuyruklardan işaretlenmesine, performansta bir düşüşe neden olmaktadır.
- *Kuyruk başına Standart K*, üzerindeki bağlantı noktası için her kuyruk aynı standart eşiği kullanır. Böylece, standart *K*, herhangi bir kuyruğun uzunluğu her kuyruktaki bağımsız olarak *K* değerinden geçer geçmez paketleri işaretlemeye başlar. Ayrıca, doğru şekilde garanti edilebilir gibi görünmektedir, ancak sorun, birden fazla kuyruk paralel olarak aktif olduğunda ortaya çıkarabilmektedir. Örneğin, *N* kuyruk hep birlikte kullanılan bant genişliği meşgul edilirse, arabellek doluluğu kolayca *N* çarpı *K*'ye ulaşabilir. Bu, arabellekte büyük bir basınca neden olur ve bunu düşük performans ve gecikme izlemektedir. Ayrıca, arabellek kolayca taşabilir. Düşük gecikmeyi korurken standart K'yi hesaplamak için aşağıdaki formül önerilmiştir [47, 48].

$$\mathrm{K} = \mathrm{C} \times GDS \times \lambda \qquad (2.1)$$

  Eşitlik 2.1'de *K* eşik değeri, C, bağlantı kapasitesi (paket/saniye), GDS, gidiş-dönüş süresi ve λ , tıkanıklık denetim algoritmasıyla ilgili değişken parametresidir. Genel olarak VMTCP'de [2] $\lambda = 0.17$ ve ATB^{+} [3]'deki araştırmacılar $\lambda = 1$ kullanmışlardır.
- *Kuyruk başına Maksimum K değeri*, standart eşiği tüm kuyruk arasında ağırlıklarına göre bölmekte ve herhangi kuyruğun doluluk oranı bölünmüş *K*'yi geçer geçmez işaretlemeye başlamaktadır. Bu yaklaşımın üstünlük ve sakıncaları vardır. Örnek vermek gerekirse, maksimum K değerine sahip her kuyruk iyi verime sahiptir ve hiçbir alt akış olgusu yoktur. Ancak, bu model fare (mice) akışlarının gecikmesini artırmaktadır.
- *Kuyruk başına Minimum K değeri*, ise herhangi kuyruğun doluluğu belirli kuyruk için bölünmüş *K*'yi aştığı anda işaretlemeye başlamaktadır. Minimum *K*'ye sahip olmanın fare (mice) akışları için düşük bir gecikmeye sahip olduğu oldukça açıktır. Ancak; diğer tüm kuyruklar çoğunlukla boş ve etkin değilse, erken ve geniş işaretlemeye neden oluyorsa,

arabellek yetersiz akış olgusuyla karşı karşıya kalmaktadır. Ayrıca, bu model, bireysel kuyrukların ağırlıklarını ihmal ettiği için doğruluk açısından zayıf performans sergilemektedir.

- *Kuyruk başına oransal K değeri*, anahtarda küçük arabellek doluluğuna sahip olduğundan düşük gecikme süresi garanti etmektedir. Kuyruk başına ATB standart eşiğe sahipken standart eşik dikkate alındığında tüm kuyruklar arasında düşük gecikme süresi sağlayabilmektedir. Örneğin, her bir anahtar bağlantı noktasının N kuyruğu olduğunu ve kuyruk j ağırlığının w_j olduğunu varsayalım. Böylece K_j oransal eşiği, aşağıdaki gibi hesaplanan "oransal eşik ile kuyruk başına ATB" dikkate alınarak belirlenmektedir:

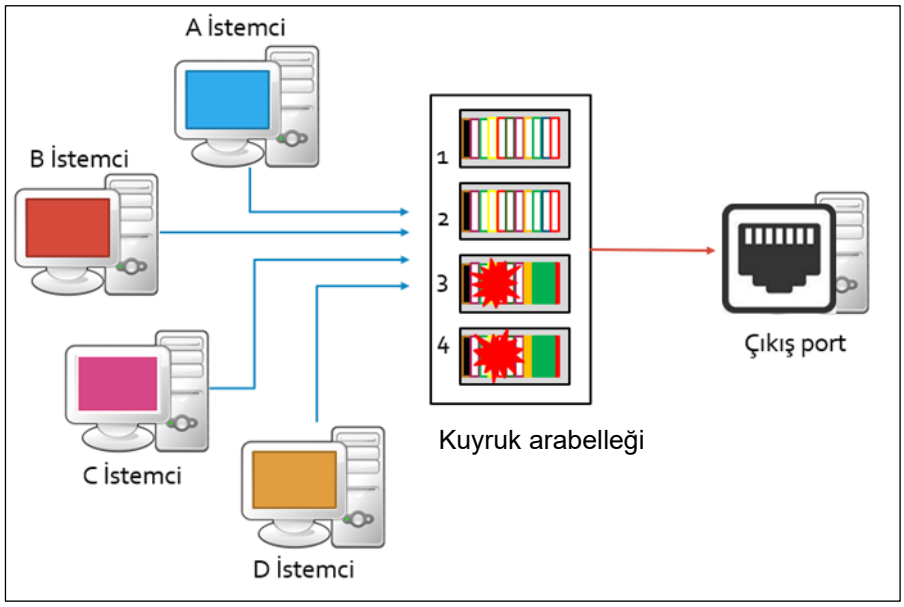
$$K_j = \frac{w_j}{\sum_{q=1}^{N_j} w_q} C_j \times RRT \times \lambda \tag{2.2}$$

Eşitlik 2.2'de N_j, her ikili çift arabelleğindeki her iki kuyruk için de aynı olan kuyruktaki akış sayısını gösterir. Burada $kuyruk_i$'nin ağırlığının w_i olduğunu ve K_i'nin sıranın oransal eşiği ve λ'nın ayarlanabilir parametre olduğu varsayılmaktadır.

Aktarım protokolleri, düşük kuyruk gecikme süresine sahip olmak için [10, 2] K değerini düşük tutmuştur. Bu nedenle arabellek boyutunun büyük kısmı, çoğuşma trafiğini sönümlemek için K değerine bağlı olarak kullanılabilmektedir [49]. Ayrıca, eşiğin ötesindeki arabellek boyutunun önemli bir kısmı, paketlerin çoğunu işaretleyebilmektedir. Bu da ciddi bir performans düşüşüne ve arabellekte yetersiz akış sorununa neden olmaktadır. Bununla birlikte K'yi artırmak, kuyruk uzunluğuna ve verimliliğine yardımcı olabilecek en basit yoldur. Ancak bu çözüm yolu kısa akışlarda kuyruğa girme gecikmesini artırmaktadır. Farklı bir K'nin, yukarıda bahsedilen basit ayarlarını daha fazla analizle ve tüm sorunlarıyla birlikte ele almak için VMA'lardaki ATB işaretlemesinde kapsamlı bir tasarıma ihtiyaç duyulmaktadır. Bu amaçla önerilen sistem tasarımı ve kullanılan model geliştirilen yaklaşımlar ile birlikte bir sonraki bölümde yöntem ve araçlar başlığı altında açıklanmaktadır.

3. YÖNTEM VE ARAÇLAR

Bu bölümde, önerilen modeller ve modellerin temelini oluşturan, çıkış portu arabelleklerindeki paketlerin ATB'de hatalı işaretleme sorununun çözümü ele alınmıştır. Şekil 3.1'de kuyruk arabelleğindeki trafik yoğunluğunun çıkış portu arabelleğini nasıl etkilediğini gösterilmektedir. Şekil 3.1'e yakından bakacak olursak 3 ve 4 numaralı kuyruk arabelleklerinin *K*'yi aştığı görülmektedir, bu da tıkanıklığa sebep olmaktadır. Dolayısıyla bu kuyruklardan gelen tüm paketler, kuyruk arabelleğinde işaretleneceklerdir. Ancak, 1 ve 2 numaralı arabelleklerin kuyruk uzunlukları *K*'ye ulaşamamıştır, bu durum trafiğin düşük olduğunu göstermektedir. 3 ve 4 numaralı arabelleklerdeki sıkışmadan dolayı, çıkış port arabelleği *K*'ye erişebilmektedir. Sonrasında tüm kuyruk arabelleklerinden gelen diğer tüm paketler çıkış port arabelleğinde işaretlenebilirken, 1 ve 2 numaralı kuyruklarda herhangi bir tıkanıklık yaşanmamaktadır.



Şekil 3.1. Çıkış port arabelleğinde paketlerin hatalı işaretlenmesi

Üstelik işaretlenmemiş paketlerin gerçek uzunlukları, hala *K*'den az olmasına rağmen KA'dan gelen işaretli paketler ÇPA kuyruk uzunluğunun *K*'yi çok daha hızlı aşmasına, bunun sonucu olarak da Tıkanıklık Penceresini düşürmeye neden olabilir. Ek olarak, sanal çıkarma işlemi çok büyük kuyruk boyutundan ötürü kuyruktaki birikimleri önlemekle kalmaz, gecikmenin azalmasına ve iş verimin artmasına da yardımcı olur. Bunun nedeni, hatalı ATB işaretlemesinin gecikmeyi veya iş hacmini düzeltmemesidir.

Çıkış portu arabelleğindeki, hatalı ATB'de paketleri işaretleme problemi, çözümde çok temel bir soru ortaya çıkartmaktadır. Bu soru, *hatalı ATB işaretlemeden nasıl kaçınılabilir ve bu politika veri merkezlerindeki gecikme, iş hacmi ve zamanlama gereksinimlerini nasıl karşılayabilir sorusudur?* Düşük gecikme, düşük kayıp elde etme ve ölçeklenebilir verimi garanti etmek için ATB hatalı işaretleme sinyallerinden kaçınan çok kuyruklu VMA'lar için yeni bir politika tasarlanması amaçlanmaktadır. Hatalı işaretleme sorununu çözmek amacıyla hazırlanan bu tezde, iki farklı yaklaşım sunulmakta ve VM-ATB'de tasarımın nasıl kullanılacağı, özellikleriyle birlikte açıklanmaktadır. Bu yaklaşımlar temelde Derin Pekiştirmeli Öğrenmeden yararlanmaktadır.

3.1. Derin Pekiştirmeli Öğrenme

Derin Pekiştirmeli Öğrenme son zamanlarda dikkate değer başarılar sağlamaktadır. Aslında Makine Öğrenimi, Derin Öğrenme ve Pekiştirmeli Öğrenme geçmişiyle başlamaktadır. ATB'deki hatalı işaretlenme sinyallerini önlemek için aşağıda cevaplanması gereken iki temel soru bulunmaktadır.

- *ATB'deki hatalı işaretleme sinyallerini önlemek ÇK arabelleklerinde hatalı olarak işaretlenen Çoklu Kuyruklu arabelleklerdeki toplam paketlerden nasıl çıkarabiliriz?*
- *ÇPA'daki toplam paketlerden nasıl ayrıştırabilir ve bu çözüm veri merkezlerindeki gecikmeyi, iş hacmi ve zamanlama gereksinimlerini nasıl karşılayabilir?*

ATB'de DPÖ kullanılmasının temel prensibi, kuyruk arabelleğindeki herhangi bir paket işaretlendiğinde ÇPA'daki işaretleme kararı için, işaretlenmiş bu paketleri tekrar dikkate almamasıdır. Özetle DPÖ, paketlerin KA'da işaretlenmiş olan kısmını, ÇPA'da işaretlenmemiş olanlardan ayırmaktadır. Bunu, kuyruk arabelleğinde işaretli paketlerin atlanan oranının teorik analiziyle ve paylaşılan ÇPA'deki tüm paketlerden çıkararak yapmaktadır.

Çok kuyruklu ve çok hizmetli VMA senaryolarındaki hatalı işaretleme sorunlarını çözmek için port başına (per-port) Derin Pekiştirmeli Öğrenme ile işaretleme kararı verilmelidir. Bu bildirim bir DPÖ sorunu olarak ifade edilir ve mümkün olan en iyi aracı politikası için Derin Sinir Ağı yöntemi kullanılır. VMA'da matematiksel modellere odaklanmış ya da Makine Öğrenme yöntemini kullanmış geçmiş araştırmalardan farklı olarak DPÖ, sürekli eylem alanı ile eşiğin gerçek değerini optimize eden, özgün DPÖ tabanlı bir yöntemdir.

VMA'ların çok kuyruklu çoklu hizmet senaryolarındaki hatalı işaretleme sorunlarını çözmek için bağlantı noktası başına Derin Pekiştirmeli Öğrenme ile işaretleme kararı kullanılmaktadır. İfadeyi DPÖ sorunu olarak formüle etmekte ve en iyi politikanı elde etmek için ek olarak Derin Sinir Ağı (DSA) kullanmaktadır. Bu şekilde, kuyruk arabelleklerinde işaretlenen paketlerde herhangi bir sorun olmadığında, çıkış bağlantı noktasındaki optimum eşiği elde etmek için karmaşık VMA'ları modellenebilmektedir. VMA'da matematiksel modellere odaklanan veya Makine Öğrenimini kullanan önceki araştırmaların aksine DPÖ, sürekli eylem alanıyla eşiğin gerçek değerini optimize eden yeni bir yöntemdir ve bu tez çalışmasını önceki araştırmalarla karşılaştırılamaz kılmaktadır. Sonuçlarda, DPÖ'nün arabellek kapasitesini tam olarak %30 oranında kullandığı ve optimuma yakın akış tamamlama süresine ulaşıldığı gösterilmektedir.

DPÖ'nün ana fikri, kuyruk arabelleğinde bir paketin işaretlenmesi durumunda, ÇPA'daki işaretleme kararı için aynı paketin tekrar dikkate alınmayacak olmasıdır. Yani DPÖ, paketlerin KA'da işaretlenmiş kısmını ÇPA'da işaretlenmemiş olanlardan keserek kuyruk arabelleğinde işaretli paketlerden atlanan oranının teorik analizi ve paylaşılan ÇPA'daki tüm paketlerden çıkararak yapmaktadır [55].

Tez çalışmasında ayrıca DPÖ formülasyonları ve çözümleri de kullanılmaktadır ve Çıkış Portu Arabelleğindeki eşiği dinamik olarak optimize eden yeni DPÖ şeması önerilmektedir. Hedeflenen amaç, kuyruk arabelleğinden (KA) gelen işaretli paketleri dikkate almadan, ÇPA için ideal eşiği hesaplamak için DPÖ uygulamaktır. Önceki tüm araştırmalarda eşiği optimize etmek için matematiksel modellere odaklanılmıştır ancak gerçek zamanlı ve karmaşık VMA'larda matematiksel modellerin gecikmeye neden olduğu bilinmektedir. DPÖ, sürekli işlem alanından seçilmesi gereken eşiği optimize etmeyi amaçlar. Problemi DPÖ problemi olarak formüle etmek, karmaşık VMA'ları modellemek, KA'dan gelen işaretli paketlerin dikkate alınması gerekmeyen durumlarda ÇPA eşiğini hesaplamak için DSA kullanmalıdır. Ayrıca, analizde de açıkça görüldüğü gibi, DPÖ yalnızca tıkanıklığı önlemekle kalmaz aynı zamanda iletim durumuna da etkileyebilir, çünkü hesaplanan eşik her zaman sırasıyla Q_{min} ve Q_{max} ile gösterilen minimum ve maksimum kuyruk arasında hesaplanır.

ÇPA'nın paket sayısı 90’dan az olana kadar (işaretli + işaretsiz paketler) işaretli paketlerin bir kısmını ÇPA'dan sanal olarak çıkarttığını belirtmekte fayda vardır. 60 paket olan standart

(K) eşik değeri kullanılmaktadır. Bu, *K*'nin arabellek boyutunun neredeyse %90'ına kadar genişletildiği ve arabellek kapasitesinin tam olarak %30'da kullanıldığı anlamına gelmektedir. Aksi takdirde, KA'dan gelen işaretli paketler, çıkış bağlantı noktası kuyruk uzunluğunun *K*'yi çok daha hızlı aşmasına ve sonuç olarak tıkanıklık penceresini düşürmesine neden olabilirken, gerçekte gelen işaretlenmemiş paketlerin uzunluğu hala *K*'den azdır. Ayrıca sanal çıkarma, büyük kuyruk boyutları nedeniyle oluşan kuyruk birikimlerini önleyebilir. Ancak gecikmeye ve verimliliğe de etkisi olabilmektedir. Bunun nedeni hatalı ATB işaretlemesinin gecikmeye veya verime yardımcı olmamasıdır. Ayrıca tekrar denetlemek için kapsamlı bir benzetim çalıştırmaktadır. DPÖ'nün, farklı akış boyutlarında kararlı durum analizi ve farklı akış türleri ile yapılan kapsamlı deneyler yoluyla optimuma yakın Akış Tamamlanma Süresine ulaştığı gösterilmektedir. DPÖ'deki ana zorluk, Veri Merkezi TCP'lerin (VMTCP) SIGCOMM 2010'da [2] uyguladığı gibi, KA'da işaretli paketlerin kesrinin (α) nasıl hesaplanacağıdır. Ancak, DPÖ ek kayıtlara ihtiyaç duymamakta ve kuyruğu, VMTCP'ye benzer karmaşıklık ölçeğinde düşük tutmaktadır. Daha da kötüsü bir paket arabellek kuyruğundaki yetersiz akış nedeniyle işaretlendiğinde, çıkış bağlantı noktası arabelleğinde yaşanan tıkanıklık nedeniyle olumsuz işaretlenebilir. Sonuç olarak taşıma protokolü, yeterli metaya sahip olmayan belirli bir kuyrukta bulunan işaretli paketlerin akış hızını azaltacaktır ancak çıkış bağlantı noktası arabelleğindeki paylaşılan sistem nedeniyle sorun tersine çevrildiğinde bu durum geçerli olmayacaktır. Daha açık şekilde belirtmek gerekirse paylaşılan çıkış bağlantı noktası arabelleği, 4-8 kuyruk arabelleğinin verilerini, aynı paylaşılan çıkış bağlantı noktası arabelleğine gönderdiği alt akış olgusuna sahip olamaz. Bunu tekrar denetlemek için küçük bir benzetim da çalıştırmaktadır. *2*

3.2. Derin Pekiştirmeli Öğrenme Tasarımı

Çıkış portu arabelleğinde, kuyrukların minimum ve maksimum değerlerinin ayrıştırılması için Derin Pekiştirmeli Öğrenmenin kararlı durum analizinin kullanılması zorunludur. Çok-kuyruk arabellekte işaretli paketlerin toplanması için DPÖ algoritması Çizelge 3.1'de gösterilmektedir.

Çizelge 3.1. DPÖ ile arabellekte işaretli paketlerin toplanma algoritması

	Giriş: *kuyruk_uzunluk_j;ATB_eşik; j_ağırlık; ağırlık_toplamı; KA_toplamı; j kuyruk_toplamı*
	Çıktı: *KA_m_pak_toplamı*
1	$kuyruk\_eşik\_j \leftarrow \frac{ağırlık\_J}{ağırlık\_toplamı} \text{ x } ATB\_eşik$
2	for *j* = 1; 2; …, *arabellek_sayısı* do
3	if *kuyruk_uzunluk_j* ≥ *i_kuyruk_eşik* then
4	*işaret* ← *true*
5	end
6	if *işaret* = = *true then*
7	*KA_işaret_pak_sayısı* ← *KA_işaret_pak_sayısı + 1 işaret* ← *false*
8	end
9	*işaret* ← *false*
10	end
11	*işaret*← *false*
12	return *KA_işaret_pak_sayısı*

Kararlı durum analizi için ilk olarak, ATB hatalı işaretlemesini önlemek için işaretli paketlerin sayısı (kuyruk arabelleğindeki işareti alan) ÇPA'daki tüm paketlerin dışında bırakılmalıdır. Ardından, bu işlemin iş hacminden ödün vermeden düşük gecikme elde etmek için kuyruk doluluğunu ve *K*'yi nasıl daha düşük tuttuğu gösterilebilir. Bu, Çoklu Kuyruğa ve ÇPA'da minimum ve maksimum kuyruk salınımlarının ve işaretli paket oranlarının teorik analizi ile yapılabilir. DPÖ analizi sırasında matematiksel denklemlerdeki kullanılan simgelerin açıklaması Çizelge 3.2'de detaylı gösterilmektedir.

Çizelge 3.2. Kullanılan değişkenler listesi

Semboller	Açıklama
A	OPB'deki akış genliklerinin toplamı
B	Tam arabellek boyutu
C	Darboğaz bağlantısının kapasitesi
C_e	*e* bağlantısının kapasitesi
D	Tek bir akış için genlik salınımı
d	*f* akış talebi
E	Bağlantı seti (*u,v*)
E^T	Bağlantı kümesi ($u(t_i)$, $v(t_j)$)
F	Tüm akışların kümesi *f*
f_e	*e* bağlantısındaki akış
$K_{max/min}$	Maksimum/Minimum ATB eşiği
K	ATB işaretleme eşiği
L	Ortalama paket boyutu
M	KA'de işaretlenen paket sayısı

Çizelge 3.2. (devam) Kullanılan değişkenler listesi

Semboller	Açıklama
$Q_{min/max}$	ÇPA'deki minimum/maksimum kuyruk uzunluğu
T	Zaman noktası kümesi T$\{t_0, t_1, \ldots, t_n\}$
Th	Fil kuyruğundaki ATB eşiği
W	Tıkanıklık penceresini temsil eden tek akışın boyutu
W^*	Pencere boyutu (kuyruk K'yi aştığında)
$W(t)$	Tıkanıklık penceresi boyutu (her GDS'deki)
x_e	e üzerindeki akış boyutu
γ	Anlık fare kuyruğu uzunluğu
γ'	Bir sonraki GDS'deki anlık fare kuyruğu uzunluğu
δ	Anlık fil kuyruğu uzunluğu
β	İşaretli akışların oranı
α	Kuyruğun eşiği aşan kısmı
α'	Bir sonraki GDS'deki α değeri
σ_e	e bağlantısı için iletim gecikmesi
Δ_f	f akışının talep değişimi
τ_e	Bağlantıdaki bir paketin kuyruk gecikmesi e

Optimum K değeri ve verimden ödün vermeden düşük gecikme süresi elde etmek için ÇPA'deki işaretli paketlerin kesrini çıkarmak için minimum ve maksimum kuyruk değerinin analiz edilmesi gerekmektedir [55]. Eşitlik 3.1'deki $S(W_1, W_2)$ iki farklı pencere boyutu arasında gönderici tarafından gönderilen paket sayısını gösterirse ve $W_2 > W_1$ ise eşitlik aşağıdaki gibi tanımlanmaktadır:

$$S(W_1, W_1) = (W_2^2 - W_1^2)/2 \tag{3.1}$$

Kuyruk boyutu K'ye ulaştığında, pencere boyutunu tıkanıklık penceresi olarak kabul eder ve $W^* = (C \times GDS + K)/N$ olarak gösterilir. Kuyruk Arabelleğinde zaten işaretlenmiş olan paketlerden kaçınırken, ÇPA'deki işaretli paketlerin kesrini hesaplamak gerekir.

$$S_j\left(W_j^* + 1, W_j^*\right) = \frac{\left(W_j^* + 1\right)^2 - \left(W_j^*\right)^2}{2} = \frac{2W_j^* + 1}{2}. \tag{3.2}$$

Eşitlik 3.2'*de* S_j işlevi, j'inci kuyruğundaki işaretli paketlerin sayısını gösterir. $j = 1, 2, \ldots, p$ arabellek olduğu varsayılmakta ve toplam işaretli paket sayısı Eşitlik 3.3'teki gibi hesaplanmaktadır:

$$M = \sum_{j=1}^{p} S_j\left(W_j^* + 1, W_j^*\right), \tag{3.3}$$

Eşitlik 3.4'te de varsayıldığı gibi, her kuyruğun kendi trafiği vardır:

$$W_j^* = C_j RTT + K_j. \tag{3.4}$$

Arabellekteki tüm kuyrukların, diğer arabelleklerden farklı olarak aynı GDS'ye ve kapasiteye sahip olduğunu varsayarak Eşitlik 3.5 elde edilmektedir,

$$\begin{aligned} M &= \sum_{j=1}^{p} S_j\left(W_j^* + 1, W_j^*\right) \\ &= \sum_{j=1}^{p} \frac{2W_j^* + j}{2} \\ &= \sum_{j=1}^{p} \frac{2\left(C_j \times RTT + K_j\right)/N_j + 1}{2}. \end{aligned} \tag{3.5}$$

Eşitlik 3.5'te N_j, her arabellekteki kuyruklar için aynı oranda kuyruktaki veri akış sayısını gösterir. Eşitlik 3.6'da K_i, hesaplanan kesirli eşik (fractional threshold) ile kuyruk başına ATB dikkate alınarak belirlenmektedir.

$$K_i = \frac{w_i}{\sum_{q=1}^{N_i} w_q} C_i \times RRT \times \lambda. \tag{3.6}$$

Çizelge 3.1'deki algoritmada görüldüğü üzere, *kuyruk* $_i = w_i$ ve K_i 'nin ağırlığının *kuyruk* $_i$ 'nin kesirli eşiği olduğu ve λ'nın ayarlanabilir parametre olduğu varsayılmaktadır. Böylece eşitlik 3.7 K_i 'nin yerini almakta ve sonra aşağıdaki eşitlik ortaya çıkmaktadır:

$$M = \sum_{j=1}^{p} \left(\frac{\left(C_j \times RTT \left(1 + \frac{w_j}{\sum_{q=1}^{N_j} w_q} \lambda \right) \right)}{N_j} + \frac{1}{2} \right). \tag{3.7}$$

Çıkış portu arabelleğindeki işaretli paketleri çıkarmak için, toplam işaretli paket sayısına sahip olarak algoritma devam etmektedir.

Eşitlik 3.8'de Çıkış Port Arabelleğinde bulunan işaretli paketlerin bölümünü bulmak hedeflenmektedir. Ardından, kuyruk arabelleğinde bulunan (önerilen yaklaşıma dayalı olarak) işaretli paketlerden kaçınarak ÇPA'nın maksimum uzunluğu elde edilebilmektedir. Eşitlik 3.8'den de anlaşılacağı gibi α, GDS'deki işaretli paket sayısının (W^*) ve (W^*+1) arasında, tüm süreç boyunca kaynak tarafından gönderilen paketlerin toplam sayısına bölümüdür. Bu amaca ulaşmak için, Eşitlik 3.7'de elde edilen KA'daki işaretli paket sayısını *(M)* Eşitlik 3.8'in paydası tarafından belirlenen ÇPA'daki toplam paketlerden çıkarılması gerekir. Bu nedenle, önerilen algoritmaya dayalı olarak α'yı ÇPA'ya uyarlamak için, denklemi açıklamak ve ispatlamak üzere Eşitlik 3.9'un kullanması gerekmektedir.

$$\alpha = \frac{S(W^*, W^*+1)}{S\left((W^*+1)\left(1-\frac{\alpha}{2}\right), (W^*+1)\right)}, \tag{3.8}$$

$$\alpha = \frac{S(W^*, W^*+1)}{S\left((W^*+1)\left(1-\frac{\alpha}{2}\right), (W^*+1)\right) - M} \tag{3.9}$$

S fonksiyonu işaretli paketlerin sayısını hesaplamak amacıyla kullanılmaktadır. İşaretli paketlerin sayısı hesaplanması işleminde *M*, KA'daki işaretli paketlerin sayısını ifade etmektedir. Eşitlik 3.1-3.8 uygulanarak eşitlik 3.10 elde edilebilir.

$$\frac{1}{\alpha} \simeq \frac{W^*}{2}\left(\alpha - \frac{\alpha}{4}\right)^2 - \frac{M}{2W^*+1} \Rightarrow \frac{1}{\alpha} = \frac{W^*(2W^*+1)}{2}\left(\alpha - \frac{\alpha}{4}\right)^2 - 2M \tag{3.10}$$

α'yı elde etmek için yukarıdaki ifadeyi basitleştirerek aşağıdaki eşitlik 3.11 elde edilmektedir;

$$\frac{M}{W^*} \simeq \alpha - \frac{\alpha^2}{4} - \frac{1}{\alpha}. \tag{3.11}$$

α'yı bulmak için yukarıda belirtildiği gibi bir özel koşula ihtiyaç vardır. Bir diğer deyişle, işaretli paket sayısı $\sqrt{(5W^*)^5}$'den büyük olmalıdır. Böylece. $M>\sqrt{(5W^*)^5}$ ise basitleştirilmiş α aşağıdaki eşitlik 3.12 gibidir:

$$\frac{3\sqrt{W^*M}}{2W^*+1}. \tag{3.12}$$

Bundan sonra, D ve dolayısıyla A genliği kolayca hesaplanabilmektedir.

$$D=(W^*+1)-(W^*+1)\left(1-\frac{\alpha}{2}\right). \tag{3.13}$$

D'nin tek akış için pencere boyutundaki salınım genliği olduğu ve A'nın ÇPA'daki kuyruğun genliği olduğu unutulmamalıdır. Gösterildiği gibi, α, A alanını etkileyebilir. Ayrıca, N akışı vardır ve $A=ND=N(W^*+1)\frac{\alpha}{2}$ burada $\alpha=C\times GDS+K$ varsayalım ki $a=C\times GDS+K$, o zaman yukarıdaki denklem eşitlik 3.14 de olduğu gibi yeniden yazılmaktadır:

$$\begin{aligned} A &= \frac{3N}{2}\left(\frac{C\times RTT+K}{N}+1\right)\left(\frac{\sqrt{\frac{N(C\times RTT+K)M}{N}}}{2N(C+K)}\right) \\ &= \frac{\left(\frac{3}{2}(C\times RTT+K)+\frac{3}{2}N\right)}{2(C\times RTT+K)+N}\sqrt{N(C\times RTT+K)M}. \end{aligned} \tag{3.14}$$

VMTCP'den bildindiği gibi, Q_{max} aşağıdaki gibi tanımlanabilir:

$$Q_{\max}=N+K. \tag{3.15}$$

Daha sonra ÇPA'da kuyruğun maksimum uzunluğu (Q_{max}) elde edildiğinden, ÇPA'da da kuyruğun minimum uzunluğuna (Q_{min}) ulaşılabilmektedir.

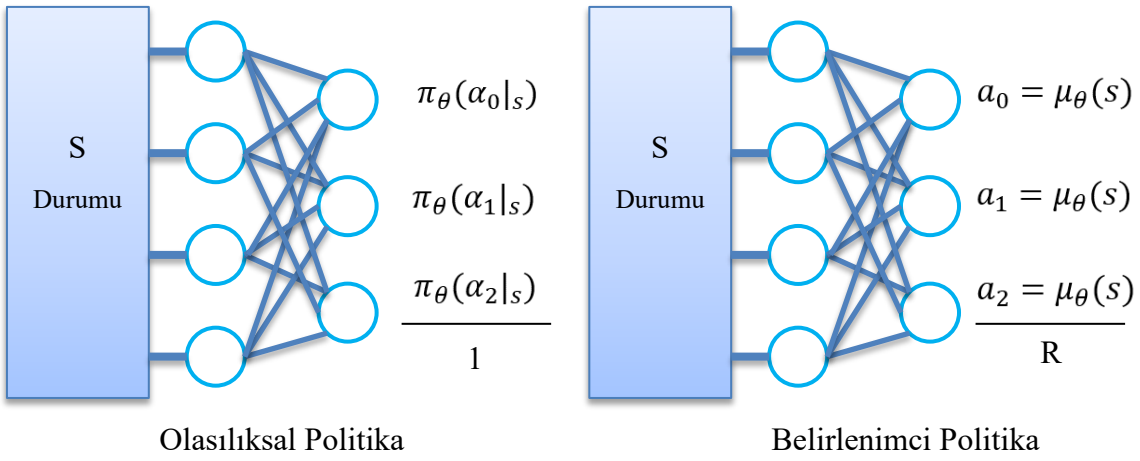
$$Q_{\min}=N+K-A=N+K-\frac{\frac{3}{2}(a+N)}{2a+N}\sqrt{aMN}. \tag{3.16}$$

Önceki tüm matematiksel çalışmalar gibi K'yi optimize edebilmek için optimizasyon problemine çözüm bulmak gerekmektedir. Ayrıca gerçek zamanlı ve karmaşık veri merkezi,

ağlarda *K*'yi optimize etmeye yönelik tüm matematiksel modeller gecikmeye neden olmuştur. Bu sebeple *K*'yı matematiksel olarak optimize edilememektedir. Bunun yerine, *K*'yi optimize etmek için Derin Pekiştirmeli Öğrenim kullanımının daha verimli olacaktır.

Önceki araştırmalarda, eşiği optimize etmek için matematiksel modellere odaklanılmış veya VMA tıkanıklık sorunlarını çözmek için makine öğrenimi kullanmıştır. Ancak matematiksel modeller, gecikme gereksinimi nedeniyle karmaşık veri merkezi ağlarda, gerçek zamanlı karar verme karmaşıklığını karşılayamamaktadır. Makine öğrenmesinden farklı olarak, sürekli eylem alanından seçilmesi gereken *K* değerini optimize etmeyi amaçlamaktadır. Bu işlem DPÖ problemi olarak formüle edilmekte ve aracı için mümkün olan en iyi politikayı elde etmek amacına ek olarak Derin Sinir Ağına ihtiyaç duymaktadır. Bu şekilde, kuyruk arabelleğinden gelen işaretli paketler önemsiz olduğunda ÇPA'da optimum eşiği elde etmek için karmaşık VMA'ları modellemek mümkündür. Bu gerçek, modeli daha önce önerilen yaklaşımlarla karşılaştırılamaz kılmaktadır.

Bu tezde, Politika Gradyanı (Policy Gradient-PG) olan temel DPÖ algoritmasından faydalanılmaktadır. Aracı, PG'e dayalı olarak, θ vektörü tarafından parametre olarak belirlenmiş $\Pi_\theta(a|s)$ politikası kullanır. Daha fazla deneyim kazanmak temsilcinin performansını iyileştirmesine de yardımcı olmaktadır.
Politika gradyanı algoritması olasılıksal politikalara odaklanır, $\Pi_\theta(a|s) = P[a|s; \theta]$. Bu, verilen küme üzerindeki olasılık dağılımına dayalı olarak eylem kümesi *A*'nın verilmesi anlamına gelir. PG algoritmaları, *S* durumunda *A* eylemi gerçekleştirmeye karar vermektedir (politikanın θ ile parametresine dikkat edilmektedir). Bu, PG'nin optimizasyon için uygulanamayacağını göstermektedir. Bu nedenle, belirli durumlarda eşik değerinin optimal değerine yaklaşmak için Belirlenimci Politika Gradyanı (Deterministic Policy Gradient) [57, 58] olarak bilinen başka PG biçimine ihtiyaç duyulmaktadır ve bu da $i = 0, \ldots, n.$ için $a_i = \mu_\theta(s)$ ile sonuçlanır.



Şekil 3.2. Derin belirlenimci ve derin olasılıksal karşılaştırma

Olasılıksal ve Belirlenimci politikaları kısaca karşılaştırması Şekil 3.2'de gösterilmektedir. Ayrıca, belirlenimci politikalar alanında, DPG aktör-kritik [61] algoritması olarak bilinmektedir. Başka bir deyişle, DPG aktör işlevi μ (mevcut politika) ve kritik sinir ağı $Q(s,a)$ demektir. Burada, Q-öğrenmesine [59] benzer şekilde, $Q(s,a)$'yı güncellemek için Bellman denklemi kullanılır.

$$\theta^{k+1} \leftarrow \theta^k + \alpha E_{s\sim\rho^{\mu^k}}\left[\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu^k}(s,a)\Big|_{a=\mu_\theta(s)}\right], \tag{3.17}$$

$$J(\mu_\theta) = \int_S \rho^\mu(s) r(s,\mu_\theta(s)) ds = E_{s\sim\rho^\mu}[r(s,\mu_\theta(s))], \tag{3.18}$$

$$\nabla_\theta J(\mu_\theta) = \int_S \rho^\mu(s) \nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu^k}(s,a)\Big|_{a=\mu_\theta(s)} ds \tag{3.19}$$

Eşitlik 3.18'de μ_θ 3.19 yerine koyulduğunda eşitlik 3.20 elde edilir,

$$\nabla_\theta J(\mu_\theta) = E_{s\sim\rho^\mu}\left[\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu^k}(s,a)\Big|_{a=\mu_\theta(s)}\right]. \tag{3.20}$$

Belirlenimci Politika Gradyanı (BPG) algoritmasının bir başka biçimi de Derin Öğrenme Yöntemi'nden [59] yararlanan Derin Belirlenimci Politika Gradyanı'dır (DBPG)(Deep Deterministic Policy Gradient-DDPG) [60]. Problemi optimize etmek için AuTo [57] gibi kaynaklarda DBPG'nin avantajını kullanılmakta. DBPG aynı zamanda aktör-kritik(actor-

critic) [61] algoritması olarak bilinir ve dört DSA kullanmaktadır. Hem kritik $Q_{\theta Q}(s,a)$ hem de aktör $\mu_{\theta\mu}(s)$ DSA kullanılır. Aktör ve kritik θ_Q ve θ_μ ağırlıklarıyla eğitim, N büyüklüğünde örneklenmiş mini partiler üzerinden yapılır. Temsilci, çevre ile etkileşime girerken rastgele örnekler ve deneyimli geçiş, çok-ögeli (s_i, a_i, r_i, s_i+1) arabellekte saklanmaktadır. Saklanan rasgele örnekler, ilişkili durumlardan kaçınmak maksadıyla DSA'ları eğitmek için kullanılır ve bu kullanım DSA'ların çeşitli hale getirir [59]. Aktör ve kritik ağların sorunsuz güncellenmelerinin, diğer iki DSA, hedef aktör $\mu'_{\theta Q}$ ve hedef kritik $Q'_{\theta Q}$ (s,a) kullanarak yönetildiği görülmekte, bu durum Çizelge 3.2'de gösterilmektedir. Güncelleme adımları, aktör-kritik ağlarının test edilmesini dengeler ve sürekli uzay eylemleri üzerinde son teknoloji ürünü sonuçlar elde edilmesini sağlar [59]. Ayrıca DPÖ 'ye dayalı olarak DPÖ, en iyi akış programlama kararlarını elde etmek için eşik değerini optimize etmek üzere DBPG'yi uygulamaktadır.

DPÖ formülasyonunda, [63]'ün yardımıyla ÇPA'ya uygun optimal eşik ayarlayarak Q_{min}'i en aza indirilmiştir. Problemin DPÖ algoritmasına çevirilmesi için durum alanını, aksiyon alanını ve ödülleri aşağıdaki gibi tanımlaması gerekmektedir:

Durum alanı: Durumlar, çıkış bağlantı noktası arabelleğindeki tüm etkin akışların kümesi olarak tanımlanmıştır. Her akış için 6 özellik vardır; *kaynak IP, hedef IP, kaynak port numarası, hedef port numarası, taşıma protokolü ve akışın ÇPA'ya getirdiği işaretli paket sayısı*. *Eylem alanı:* Eylemler, temsilci tarafından her zaman adımında belirlenen eşiklerdir. *Ödüller:* Ödüller aracı kararlarına geri bildirim olabilir ve paketlerin onayı alındığında hesaplanır. Ayrıca ödüller, aracının önceki zaman adımında iyi kararla paketleri işaretlemedeki performansını temsil eder. Ödülü $r_1 = ln(\frac{Ack\_time_{t-1}}{Ack\_time_t})$ iki ardışık zaman adımında modellemiştir. Ödüller, önceki eylemler daha uzun teslimat süresiyle sonuçlandıysa negatif puanlarla veya karar iyi sonuca ulaşırsa pozitif puanlarla temsilci kararını etkilemektedir.

Çizelge 3.3'de aracı kararlarının nasıl güncelleme sözde kodu gösterilmektedir. Derin Sınır Ağları, Q_{min}'in hesaplanmasından sorumludur. Her yeni durum için, temsilci kararlar alır ve öğrenmeden sonraki adım için arabellekte sırasıyla s_t, a_t, r_t, s_t+1 *mevcut durum, mevcut eylem, mevcut eyleme verilen ödül ve sonraki durum* DSA'de saklanır. Sonraki güncellemeyi alana kadar r_t ve sonraki s_t+1 durumunun bilinmediğine dikkat edilmesi gerekir. Bu durumda, gerekli tüm veriler alınana kadar s_t ve a_t saklanır. Farklılığı önlemek ve öğrenmeyi

stabilize etmek için parametrelerin güncellemeleri rasgele gruplar halinde gerçekleştirilmiştir. Ana bilgisayarda, ödül r_t'yi belirlemek, paketleri iki zamanlı adımlar arasında göndermek ve tüketilen süreyi hesaplamak için adım *t* ile önceki adım arasında karşılaştırma yapılmaktadır. Karşılaştırma sonucuna göre ödül (negatif veya pozitif) sonraki adımda eylemi güncellemek üzere aracıya iletmektedir.

Çizelge 3.3. DPÖ için DDPG aktör-kritik için güncelleme sözde kodu

Arabellek Seti $y_i = r_i + \gamma Q'_{\theta Q'}\left(s_{i+1}, \mu'_{\theta \mu'}(s_{i+1})\right)$'den
N rasgele geçişler (s_i, a_i, r_i , s_i+*1)* için mini-yığın paterni,
for epizod =1, M do
Güncelleme kritik tarafından kaybın en aza indirilmesi:
$L = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q_{\theta Q}(s_i, a_i)\right)^2$
for i=1, do
aktör politikasını güncellemek için politika gradyanını kullanımı:
$\nabla_{\theta\mu} J \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_{\theta\mu}(s_i)\mu_{\theta Q}(s_i)\nabla_{a_i} Q_{\theta Q}(s_i, a_i)\Big|_{a_i=\mu_{\theta Q}(s_i)}$
kararlı öğrenme için değerler olan λ ve τ kullanılarak ağ hedeflerinin güncellenmesi:
$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$
$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$
end for
end for

ÇK-ATB'de bağlantı noktası başına ATB işaretleme kararları düşük gecikme süresi ve yüksek verim sağlamakla birlikte zamanlama sorunu vardır ve aynı zamanda ağırlıklı uygun paylaşım oranını ihlal etmektedir. Bağlantı noktasındaki kuyruklar arasında arabellek yalıtımı sağlanmadığı için kuyruktaki paketler, paylaşılan çıkış bağlantı noktasının diğer kuyrukları tarafından paylaşılan arabellek doluluğu nedeniyle işaretlenebilmektedir. Sorunu çözmek için, port başına ATB işaretleme kararı verilirken, paylaşılan aynı porta ait farklı kuyruk arasında müdahaleden veya müdahale edilme sürecinden kaçınılmalıdır. DPÖ, hatalı işaretlemeyi önlemek ve optimum *K*'yi bulmak için tıkanıklık sinyali olarak yalnızca ATB'yi almak yerine VMTCP'nin avantajın, ATB ile birleştiren port başına ATB işaretlemesine dayanmaktadır.

3.3. Öncelik – Açık Tıkanıklık Bildirimi yaklaşımı

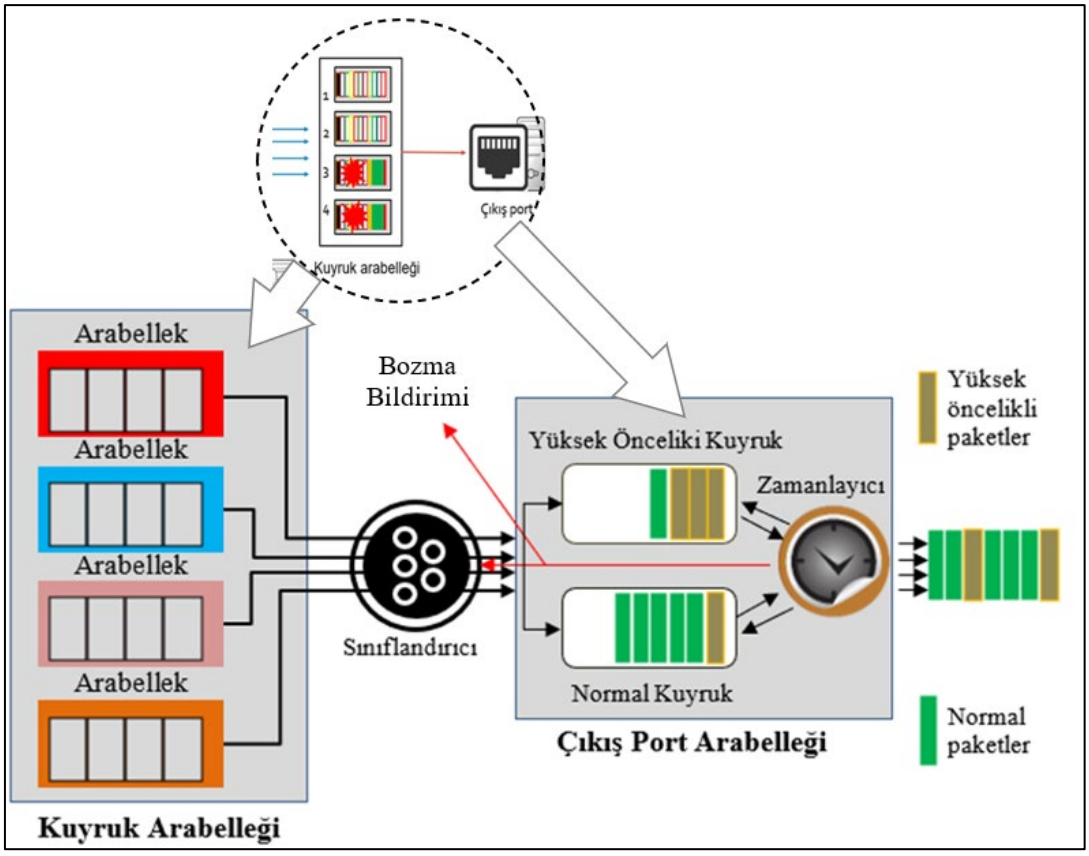
Çok kuyruklu ve çok hizmetli Veri Merkezi Ağlarında oluşan tıkanıklıkların hatalı işaretlenme sorununun çözümü için "Çok kuyruklu Veri Merkezi Ağlarında bağlantı noktası başına işaretleme yoluyla ATB desteği için öncelik sağlanabilir mi? Ayrıca, bu politika Veri Merkezlerinde gecikme, aktarım hızı ve zamanlama gereksinimlerini nasıl karşılayabilir?" temel sorularının sorulması gerekir. Bu sorulara çözüm arayarak, işaretli paketlere öncelik vermek ve çıkış bağlantı noktası arabelleğinde (Öncelik-ATB) hatalı işaretlenme sorununu önlemek için Veri Merkezi Ağlarında bağlantı noktası başına çok-kuyruklu ATB işaretlemesinde öncelik yaklaşımı sunulmaktadır.

Öncelik-ATB işaretlenen herhangi bir paketin iletim sırasında erken aşamada ayarlanmasına yardımcı olmak, ağların mevcut durumlarını bildirmek ve rotayı diğer paketlerden daha yüksek hızda geçirmek amacıyla yüksek önceliğe sahip olması gerekmektedir. Herhangi bir paketin kuyruk arabelleğinde işaretlenmesi halinde, işaretlenmiş bu paketin çıkış bağlantı noktası arabelleğinde tekrar yeniden işaretlenmemesini sağlamaktır. Aksi takdirde, çıkış bağlantı noktası arabelleği maksimum ATB eşiğine (K_{max}) kolaylıkla ulaşacak ve ardından çıkış bağlantı portu arabelleğindeki tüm gelen paketler işaretlenecektir. Bu durum, tıkanıklığa sahip olmayan paketler için de hata işaretlemesine neden olarak, verimin düşmesine ve yetersiz akış olgusuna yol açacaktır. K_{max}'a ulaşılması durumunda, gecikmeden ve yüksek verim hedefinden ödün vermeden tercih edilecek en dolaysız yol K_{max}'ı artırmak gibi görünse de bu gecikme gereksinimini ihlal etmektedir. Ayrıca, bu durumlarda Yükü kes [54] gibi benzer yaklaşım kullanmaktadır. YK, kuyruk eşiğe ulaşıp paket başlığına ulaşmadığında paketlerin yüklerini düşürür. Bu, veri bilgileri için kayıpsız ancak trafik yükleri için verimsiz bir ağ sağlar. Böylece, alıcıya gelen trafikle ilgili tam görüş sağlanır. VMA'lardaki küçük mesafelerle, bu tür yeniden iletimler 1 ms gibi çok hızlı bir sürede hedefe ulaşabilir. Bununla birlikte, göndericinin yeniden iletim zaman aşımları sona erdiğinde, eksik paketler hızla yeniden gönderilir. Bu, küçük kuyruklarda 400 ms sürmekte ve maksimum GDS sağlamaktadır [56].

3.3.1. Öncelik – ATB tasarımı

Algoritma işaretli paketler için öncelik atanabilmesi en ideal durumdur. Algoritma işaretli paketler için öncelik atanması halinde tüm işaretli paketler rotayı hızla geçebilir ve ağın

durumu ile ilgili bilgi taşıyan işaretli paketler de kaybolmaz. Çok-Hizmetli ve Çok-Kuyruklu Veri Merkezleri Düşük Gecikme – Düşük Kayıp ve Ölçeklenebilir Verim (L4S) sağlayan VMA'lar için yeni bir planlama ve politika tasarlamayı hedeflemektedir. Öncelik – ATB'nin temel tasarımı dört ana bileşen içermekte ve Şekil 3.3'te sistemin genel tasarımı ve bileşenleri gösterilmektedir. Bunlar, kuyruk arabelleği, sınıflandırıcı, çıkış bağlantı noktası arabelleği için ikili-çift kuyruğu ve zamanlayıcıdır.



Şekil 3.3. Sistemin genel tasarımı

3.3.2. Kuyruk arabelleği ve öncelik-ATB sınıflandırıcısı

Öncelik-ATB'nin kuyruk arabelleği, paketleri rasgele ve anlık/ortalama kuyruk uzunluğunun minimum eşikten (K_{min}) az olup olmadığını veya kuyruk uzunluğunun maksimum eşiği (K_{max}) aştığını işaretler. Anahtarda küçük arabellek doluluğu nedeniyle düşük gecikmeyi garanti etmek için kesirli K ile ATB kuyruk kullanılmıştır. Öncelik-ATB'nin sınıflandırıcısı, paketleri sınıflandırarak, ayrı ayrı istenen ikili-çift arabelleğine gönderecektir. Bu konu hakkında çok fazla çalışma yapılmış olmasına karşın sınıflandırıcı üzerinden makine öğrenmesi yaklaşımlarını kullanma konusunda henüz yeterli düzeyde çalışma yoktur.

Akışların önceden bilgisi olan veya olmayan paketlerin sınıflandırılması [46, 50]. Senaryodaki paketlerin veya akışların ayrılması sorunu gerçek zamanlı bir veri akışı tespitidir. Dolayısıyla, matematiksel modeller bunu kaldıramaz durumdadır. Bu nedenle, hedefe ulaşmak için veri madenciliği ve makine öğrenmesinin kullanılması önerilmektedir. Her akışın boyutu bağlantının başlarında tespit edilebilmektedir. Kilit nokta akış boyutunun çıkan ilk paketlerin özellikleri kullanılarak tahmin edilmesidir. Bu görevde makine öğrenimini kullanmanın avantajı, geliştirilmiş algoritmanın denetleyiciye kolayca uyarlanabilmesidir.

Öncelik – ATB'de İkili-Kuyruk Durumu

Sınıflandırıcı ağ trafiğinden gelen paketleri, işaretli paketler için ayrı kuyruk ve işaretlenmemiş paketler için ayrı kuyruk olmak üzere iki farklı kuyruğa ayırır. Çift bağlantılı arabellek kullanılmasının öncelikli nedeni, farklı birkaç önceliğe de sahip olan Akış Tamamlama Süresinin en iyilenmesi oldukça fazla sayıda öncelik gerektirmektedir. Çünkü VMA'larda her akış benzersiz bir önceliğe gerek duymaktadır. Örneğin, pFabric [51], her akış için benzersiz bir öncelik atanması durumunda en iyilenmiş değere yakın ATS'ye ulaşılabileceğini göstermiştir. Böylece, bağlantı bant genişliği, kesinlikle önceliklerine göre belirtildiği gibi akışlara atanabilir. Ancak VMA'da milyonlarca akış bulunmaktadır ve bu nedenle aynı sayıda önceliğe ihtiyaç duyulmaktadır. Veri merkezlerinin üretiminde, mevcut tüm anahtar kuyrukları için yalnızca iki ya da dört öncelik programlaması desteklenebilmektedir. *BCM 5324M* tabanlı anahtarlar, 4 öncelik kuyruğunu desteklemektedir. Bununla birlikte birkaç modern anahtar, sekiz adede kadar daha fazla öncelik kuyruğunu destekleyebilir. Ancak bu kuyruklar hâlihazırda başka amaçlar için ayrılmıştır.

Bu pratik sınırlama, son yaklaşımların performansını ciddi oranda azaltmaktadır [46]. Diğer bir ifadeyle, en iyilenmiş bir değere yakın bir zamanlama uygulaması için gereken çok sayıda ayrıntılı öncelik kuyruğunu desteklemek için çok sayıda anahtara sahip olmak gerekir. Böylece her akış, boyutuna göre öncelik kuyruğuna atanabilir. Ama pratikte, anahtar donanımındaki öncelik kuyruğunun sayısı minimum düzeydedir.

3.3.3. Öncelik–ATB'de zamanlayıcı

Kuyruk uzunluğunu ortadan kaldırmak için çıkış bağlantı noktasının her ikili-çift arabelleğinin girişi ve çıkışı izlenmektedir. Bu özellik, L4S'ye ulaşmada önemli bir rol oynamaktadır. Programlayıcı, K_{max}'ı ikili-çift kuyruklardaki tıkanıklık basıncını çıkartarak, işaretlenmemiş kuyruğu dinamik olarak ayarlar ve K_{max}'ı artırma veya azaltma kararı verir. Kesin olarak belirtmek gerekirse, eğer işaretli kuyruğun uzunluğu K_{max}'a ulaşırsa, gereksiz olan işaretli paketleri yeniden işaretlemek veya önemli olan işaretli paketleri bırakmak yerine sınıflandırıcıya programlayıcı İhlal Bildirimi (Violate Notification) gönderilir. Bu durum algoritma değiştirilmesi ve işaretli paket oranı Standart Durum Bildirimi (Back to Standard State Notification) alana kadar işaretlenmemiş kuyruğa gönderilmesi için bilgilendirme sürdürülür. İşaretlenmemiş kuyrukları kullanacak işaretli paketlerin oranı, işaretli kuyruktaki K_{max}'ı aşacak paketlerdir. İşaretli paketlerin bir kısmı işaretsiz kuyruğa gönderildikten sonra, yüksek öncelik sebebiyle rotayı hızla geçerek, mümkün olan en kısa gecikme süresine sahip olacak ve çıkış bağlantı noktası arabelleğinde işaretleme kararına katkıda bulunmayacaktır [52].

Bununla birlikte, kuyruklar temelde ilk giren ilk çıkar şeklinde işlemektedir ve önerilen şemalar, gerçekleşebilecek en kötü durumda işaretlenmemiş akışları boş bırakacaktır. Bu problemin çözülmesi için gereken programlayıcıda, Karuna [53] ile tamamlayıcı bir katkı sağlanmalıdır. Esasen öncelik kullanmak belirli akışları boş bırakabilir ve yoğunluk durumunda, işaretli paketlerin öncelikleri için tüm bant genişliğinin kullanılması gerekse de işaretlenmemiş akışlar boş kalacaktır. Böyle bir durumda taşıma mekanizmasında alternatif bir seçenek olmayacağı için ağ kapasitesinin arttırılması düşünülebilir.

İşaretli paketlerin öncelikleri nedeniyle tüm bant genişliğini kullanması halinde işaretlenmemiş paketler yetersiz kalabilir. Bu problemin çözümünde yetersiz kalan akışların önceliğini yükseltmek için akış zamanlama süresi (gecikme) kullanmaktadır. Zaman aşımı olaylarını gözlemleyerek son ana bilgisayarlarda yetersiz akışları belirlemektedir. Örneğin, akış TCP zaman aşımına uğradığında, Öncelik-ATB bu akışı daha yüksek önceliğe yükseltir. K_{max}'a ulaşan her iki çift-bağlı arabellek kuyruğunun yetersiz durumunda, en basit yol K_{max}'ı arttırmak olmaktadır, ancak bu, kuyruk uzunluğuna etkileyebilir ve kuyruk gecikmesini artıracaktır. Ayrıca, gecikmeden ödün vermeden yüksek iş hacmine ulaşmak için Yükü kes gibi yaklaşım kullanılabilir [54]. Kuyruğun eşiğe ulaştığı ancak paketin başlığa ulaşmadığı

durumlarda YK paket yüklerini azaltır. Bu işlem veri bilgileri için kayıpsız ancak trafik yükleri için ideal olmayan bir ağ sağlamaktadır. Yeniden sıralamaya rağmen, kayıpsız meta veriler, alıcıya gelen trafikle ilgili tam görüş sağlar. Böylece çok düşük gecikme oranları ile en iyilenmiş değere yakın bir ATS elde etmek için, işaretlenmemiş kuyrukta radikal ve yeni şema uygulaması amacıyla YK'nin sağladığı bu üstünlükten yararlanılır.

Sonuç olarak işaretli paketlerin işaretsiz kuyruğa gönderildikten sonra rotayı hızla geçmeleri için işaretli paketlere öncelik verilmesi gerekmektedir ve bu işaretlenmemiş kuyruk K_{max}'ı aştığı durumda gerçekleşmektedir. Öncellik-ATB, yükleri azaltarak işaretlenmemiş kuyruktan paketleri keser. Böylece başlıklar, paketleri hemen almak için en yüksek önceliğe sahip kaynağa gönderilir. VMA'lardaki küçük mesafelerle bu tür yeniden iletimler 1 µs. gibi çok yüksek bir hıza sahip olabilmektedir. Bununla birlikte göndericilerin yeniden iletim zaman aşımları sona erdiğinde, eksik paketler hızla yeniden gönderilir ve bu küçük kuyruklarda 400 µs. sürerek maksimum GDS sağlar [56].

3.4. Veri Merkezinde ATB-Haritalama Yaklaşımı

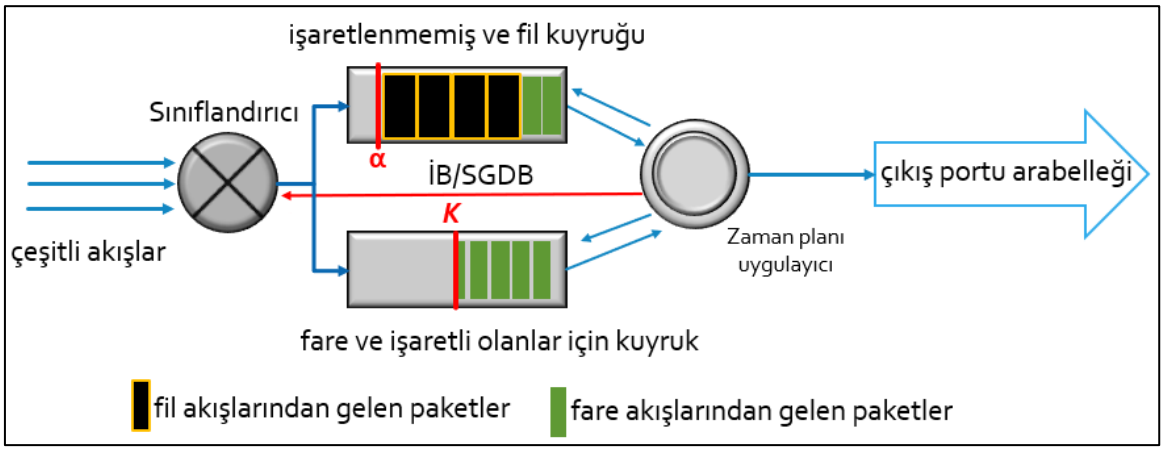
Ölçeklenebilir verim ile düşük gecikme ve düşük kayıp sağlanmaktır. Bilindiği gibi düşük gecikme süresi ve yüksek verime sahip olmak için kısa akış tamamlama süresine ihtiyaç vardır. Kısa ATS'ye sahip olmak için her bir anahtar arabelleğinde sırasıyla sığ ve derin ATB işaretleme eşiğine (K) ihtiyaç duyulmaktadır. Ayrıca, sırasıyla mikro çoğuşma trafiğinin sönümlenmesi ve düşük kayıp için kuyruklardaki tıkanıklığın giderilmesi gerekmektedir. VM-ATB ile her ikili çift arabelleğinde farklı minimum ve maksimum ATB eşikleriyle bu sorun çözüme kavuşturulmaktadır [62].

Bu yaklaşımı önermeden önce "Ölçeklenebilir aktarım hızı (L4S) ile birlikte mevcut emtia anahtar donanımı tarafından desteklenen özelliklerle sahip olarak Düşük Gecikme Süresi-Düşük Kayıp elde edebilir mi?" sorusu cevaplanmalıdır.

Fare ve fil akışları için ATS esas olarak, sırasıyla gecikme ve verim ile belirlenmektedir. Ayrıca, kısa kuyruk boyu gecikmeyi azaltmaya yardımcı olurken, uzun kuyruk boyu verim sağlamaya yardımcı olmaktadır. Böylece VM-ATB, kısa ve büyük akışlar için ATS'yi bağımsız olarak kısa ve büyük eşiklerle azaltabilmektedir.

3.4.1. VM ATB-Haritalama tasarım hedefleri

Bu, fare akışlarını fil akışlarından MÖ kullanarak ayıran ve onları en düşük ATS'ye sahip olmak için sırasıyla K_{min} ve K_{max} ile gösterilen her ikili çift kuyruk için istenen ve bağımsız minimum ve maksimum ATB işaretleme eşiğine sahip ikili çift arabelleğine koyan yeni şema olarak kullanılmaktadır. Şekil 3.4, VM-ATB'de Çoklu-Çift Port Arabellek tasarımı görülmektedir. VM-ATB, her iki ikili çift kuyruğunda K_{min} ve K_{max}'ı dinamik olarak ayarlar sonrasında VM-ATB, her ikili çift kuyruğunun uzunluğunu ölçerek, düşük gecikme elde etmek adına fare kuyruk arabelleği K_{max}'ı aşarsa, fare akışlarının kısmını fil arabelleğine gönderebilir. Daha sonra, fil kuyruğundaki K_{max}'ın anında ayarlanmasıyla, VM-ATB eksikliği önler ve mikro çoğuşma trafiğini öğrenmenin yanı sıra düşük kayba sahip olmayı başarmaktadır.

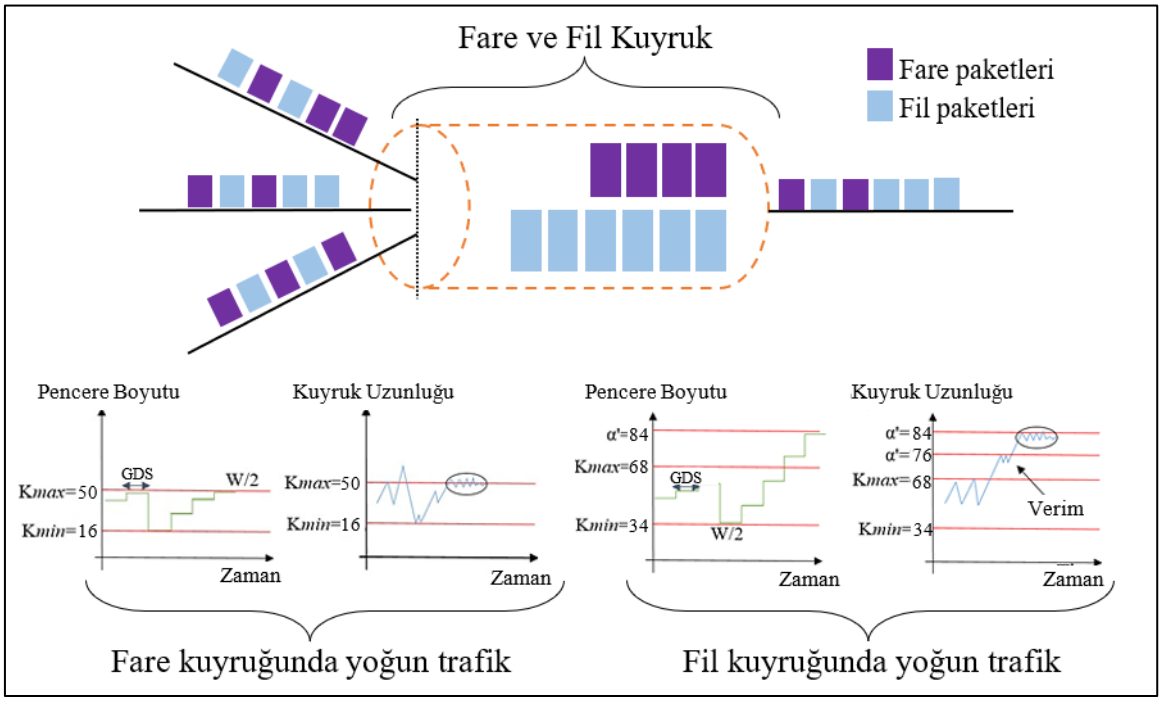


Şekil 3.4. Çoklu-Çift Port Arabellek tasarımı

Şekil 3.4'te VM-ATB ile fare ve fil kuyruğu için arabellek gösterimi görülmektedir. VM-ATB'nin tüm uygulamalar için ana ilgilerinden biri gecikmedir. Bu nedenle, tıkanıklık veya yoğun trafik durumunda, fare kuyrukları K_{max}'a ulaşırsa ve aynı anda fil kuyruğunda boş alan varsa, zamanlayıcı sınıflandırıcıya kuralları ihlal ettiğine dair bildirim gönderecektir. K_{max}'ı aşan fare, ikinci bildirimi alana kadar fil kuyruğuna yollanmakta yani normal kurala geri dönmektedir. Ayrıca, fare akışlarından gelen paketler daha yüksek önceliğe sahiptir. Böylece düşük gecikme süresi elde etmek için mümkün olan en yüksek hızda rotadan geçebileceklerdir. Bununla birlikte kuyruklar tipik olarak FIFO'dur ve önerilen planlar, en kötü senaryoda fil akışlarını boş bırakacaktır. Bu sorun 2016 yılında Karuna tarafından

çözülmüş, programlayıcılardaki bu sorunun çözümünde Karuna ile tamamlayıcı bir katkı gerçekleşmiştir [64].

VM-ATB'deki her iki çift bağlı arabelleğin K_{max}'a ulaştığı talihsiz durumda, en basit yol K_{max}'ı artırmak olacaktır ancak bu kuyruk uzunluğuna katkıda bulunabilmektedir. Bu çözüm, fare akışlarında kuyruk oluşumu nedeniyle kuyruk gecikme süresini artıracaktır. Bu durumda sınıflandırıcı, zamanlayıcıdan bildirim alarak tekrar fil kuyruğuna fare paketi göndermeye başlar. Ardından, fare akışlarının yüksek önceliği nedeniyle, tekrar düşük gecikme süresi elde etmek için hızlı şekilde rotadan geçeceklerdir.

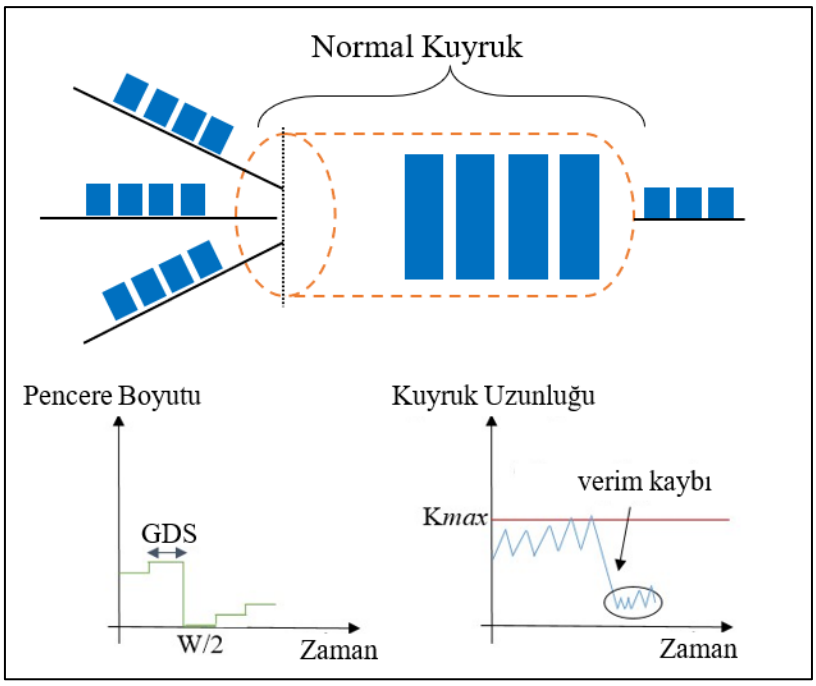


Şekil 3.5. VM-ATB ile fare ve fil kuyruğu için arabellek gösterimi

Ancak bu senaryoda, fil akışlarından gelen paketler daha fazla işaretlenmelidir. Bu nedenle, sadece fil kuyruğunda K_{max}'ı dinamik olarak artırıp azaltarak, VM-ATB, tıkanıklık oranına bağlı olarak K_{max}'tan (α ile gösterilir) sonra tampon alanı kullanır, bu, tıkanıklığın oranına dayanan katsayı aracılığıyla gerçekleşir ve fil kuyruğundaki akışlar (sonraki GDS'deki α değeri: α') ve işaretli akış oranıdır (β). Böylece VM-ATB, gecikmeden ödün vermeden daha fazla yoğun trafiği karşıladığını kanıtlar.

Açıklık ve TCP dış yayılım probleminin önlenmesi için VM-ATB'deki programlayıcı, kuyruk uzunluğunun ölçüm kesrine dayalı karar vermekte ve her kuyruk için hız limitini

önceliklendirmektedir. Ayrıca VM-ATB, adil paylaşımı sağlamak için her kuyruktaki bağımsızlığın avantajını kullanmıştır. Bu, statik adil paylaşıma bağlı kalmak yerine, VM-ATB'nin adil paylaşım oranına sahip olmak için kuyruk uzunluğunun dinamik ölçümüne dayalı olarak her kuyruk için talep edilen K_{max}'ın bağımsız olarak ayarlanabildiği anlamına gelir. Ayrıca VM-ATB, TCP outcast problemini de çözmektedir. Şekil 3.6'da Mikro-çoğuşma trafiğine sahip normal kuyruk için arabellek gösterimi görülmektedir.



Şekil 3.6. Mikro-çoğuşma trafiğine sahip normal kuyruk için arabellek gösterimi

Akışların ayrılması aynı zamanda akış son tarihleri konusunda da fayda sağlar ve genel son teslim tarihine de yardımcı olabilir. Örneğin, arama sorguları gecikmeye duyarlıdır ve kritik son tarihleri vardır (örneğin, vardıktan sonra 300 ms). VM-ATB sayesinde yoğun trafikte bile fil akışlarının arkasında durmadan hızlı şekilde rota geçilebilmektedir. İki tür çoğuşma trafiği olduğunu belirtmektedir. Bunlardan biri, fare akışlarına ait arama istekleri gibi akışların sayısını aniden artıran ve diğeri ise, fil akışlarına ait bireysel akışların TP'sini arttıran trafiktir.

Not: Dinamik artış/azalma sadece fil kuyruklarında olacak ve fare kuyruklarındaki gecikmeyi etkileyecek ve fil kuyruğundaki çıktıya yardımcı olacaktır. Paketlerin ne zaman işaretlendiği bilindiği için, K_{max}'tan sonra hala yeterli boş arabellek alanı bulunmaktadır [20].

Böylece sadece fil kuyruğunda K_{max}'tan sonra arabellek kapasitesini kullanmaya çalışmışlardır.

3.4.2. VM ATB-Haritalama algoritması

Bu bölüm önce tasarım hedefleri ve ardından VMA'da (VM ATB-Haritalama) ATB işaretlemesi için Makine Öğrenimine Dayalı dinamik eşik denetim şeması ayrıntılı olarak sunulmaktadır. VM-ATB, bağlantı noktası başına birden fazla kuyruğa sahip olma eğiliminde olan VM Ağlarındaki üreticiler tarafından motive edilmektedir. Çok Hizmetli Çok Kuyruklu Veri Merkezlerinde ATB'yi Etkinleştirme (ÇK-ATB) [15], en son teknoloji algoritmayı öne çıkararak veri merkezleri için çoklu sırayı etkinleştirmiştir. Ancak bu olgu bazı yeni zorlukları da beraberinde getirdi. Sonuç olarak, Düşük Gecikme Süresi-Düşük Kayıp ve Ölçeklenebilir verim (L4S) elde etmek için çoklu kuyrukların olmaması avantaja dönüşmüştür. Çizelge 3.3'te geleneksel VM-ATB'nin sözde kodu da gösterilmektedir. Bu bölüm, *K*'nin ayarlanmasıyla motive edilmiş, VMA için,çok hizmetli çok sıralı üretimde L4S'ye ulaşan yeni bir şema tasarlanması hedeflenmiştir.

- *Düşük gecikme süresi ve ölçeklenebilir verim:* Tüm bu hedeflerin gerçekleştirilmesi için öncelikle VM-ATB, en düşük gecikme süresine ve en yüksek verime ulaşmayı hedefler ancak her ikisine de aynı anda sahip olmak zordur. Düşük gecikme süresine sahip olmak için, düşük kuyruk gecikmesi sağlamak üzere sığ aktarım hızı (K) sürdürülmelidir. Ancak, yüksek verim elde etmek ve bağ bant genişliğini tam olarak kullanmak için oldukça fazla verime ihtiyaç duyulur.
- *Düşük kayıp:* Paketler işaretlendiğinde, maksimum ATB eşiğinden (K_{max}) sonra hala yeterli arabellek alanı olduğu açıktır. Trafik tıkanıklığı veya mikro çoğuşma durumunda, gelen tüm paketleri işaretlemek veya kaybetmek yerine, algoritma sadece fil kuyruğunda K_{max}'tan sonra arabellek kapasitesini kullanarak anında daha fazla trafik çekebilmelidir. Yani, fil akışları arabelleğindeki K_{max}'ı genişletmek, fare akışları kuyruğundaki gecikmeyi etkilemekle kalmaz, aynı zamanda gecikme ve düşük kayıp sorunlarına da yardımcı olmaktadır.

VM-ATB'nin çekirdek tasarımı üç ana bileşen içerir. Bunlar, sınıflandırıcı, çift-çift kuyruk ve zamanlayıcıdır. İlk olarak, sınıflandırıcı akış boyutu daha büyükse MÖ kullanarak, fil akışları için paketi diğer tüm akışlardan ayırır ve önce kısa akışları (İlk En Erken Süre Sonu

ve En Kısa Kalan İşlem Süresi) kuyruğa koymaktadır. Böylece fare akışları kuyruğa önce alınır. Buradaki kilit nokta iletişimin başlangıcındaki akış boyutu, ilk paketlerden MÖ'nün çıkarılan özelliklerine göre kullanılarak tahmin edilecektir.

3.4.3. VM ATB-Haritalama sınıflandırıcısı

Sınıflandırıcı akışları, akışların boyutlarına ve İlk En Erken Süre Sonu (Earliest Deadline First- EDF) ve ayrıca Kalan En Kısa İşlem Süresine (KEKİS) (Shortest Remaining Process Time-SRPT) göre sınıflandırır. Ardından akışları ayrı ayrı istenen çift-çift arabelleğe gönderir. Senaryodaki akışların ayrılma sorunu, gerçek zamanlı akış tespitidir. Veri Madenciliği ve Makine Öğrenimi kullanmak, hedefe ulaşmanın en iyi yöntemleridir. Her akışın boyutu başlangıçta tespit edilebilir ve sonuç olarak fil akışları fare akışlarından ayrılır. Püf nokta, akış boyutunun ilk paketlerin ayıklanan özelliklerinin kullanılarak tahmin edilmesidir. Bu görevde Makine Öğrenimini kullanmanın avantajı, algoritmanın denetleyiciye kolayca uyarlanabilmesidir.

Çıkarılan özellikler yani, kaynak IP, hedef IP, kaynak bağlantı portu, hedef bağlantı portu, protokol, sunucuya karşı istemci ve yalnızca herhangi akış için kullanılabilen üç paketin boyutu algoritmayı besler. Sunucu haricindeki tüm özelliklere, istemci ilk paketin başlığından erişilebilir. Sunucuya karşı istemci özelliğine yalnızca protokol TCP olduğunda erişilebilir ve UDP bu özelliğe sahip değildir. Ayrıca Veri Merkezi Ağında trafiğin %99,99'unun kendisine ait olduğu dikkat çekmektedir [46, 65, 66, 67].

ATB-Haritalama yaklaşımında DPÖ ile birlikte üç yöntem, Sinir Ağı (SA), Gauss Süreç Regresyonu (GSR) ve Çevrimiçi Bayes Moment Eşleştirme (çBME), ele alınmıştır.

SA, fil akışlarını sınıflandırabilen ilk algoritmadır. Önerilen NN, 106 düğümden oluşan girdi katmanına sahiptir. Sırasıyla bu katmanlardan ilki 60 hiperbolik teğet düğüme, diğeri 40 hiperbolik teğete sahip iki gizli katmana bağlanmaktadır. Bu sınıflandırmadaki yanıt değişkeni -1 ile 1 arasında değer almıştır. Yanıt değişkeninin negatif değeri fare, pozitif olanı ise fil akışını göstermektedir. 106 giriş düğümü özelliklerinin her biri -1 veya +1 değerine dönüştürülür. Modeli eğitmek, sinir ağının ağırlıklarını optimize etmek için geri yayılım algoritması (gradyan iniş) gereklidir. Bu en iyileme, akış kümesine ve boyutlarına göre uygulanır. Eğitim setinde k katmandan oluşan ağ ile n akış varsa, $O(kn)$, her aşamada geri

yayılım algoritmasının gradyanını belirlenmektedir [65].

GSR, önerilen ikinci yöntem olarak, özellikler kümesi göz önüne alındığında, bu parametrik olmayan Bayes modeli, akış boyutlarındaki dağılımı bulabilir. Yeni akış olduğunda, GSR, ortalama ve varyans σ^2 [68] ile akış boyutu üzerinden Gauss dağılımını, *N (μ, σ2)* hesaplayabilir. Yeni akışın boyutu tahmini ortalama olarak kabul edilir ve eşitlik 3.21 ile belirlenir:

$$\hat{s} = \mu = K(\hat{f}, F)[K(F, F) + \sigma^2 I]^{-1} s. \tag{3.21}$$

Eşitlik 3.21'de $\hat{s}$, yeni akış olan $f\hat{}$'nin tahmin edilen boyutunu, μ, ağdaki boyutların ağırlıklı toplamını ifade etmektedir. I, birim matrisidir ve belirli öznitelik kümesinde, σ^2, boyutların varyansını göstermektedir. *s*, *F* ile gösterilen eğitim verilerindeki *n* akış kümesine karşılık gelen akış boyutlarının *n*×1 boyutunda sütun vektörüdür. *K(f^, F)* ile gösterilen *1*×*n* satır vektörü, *F* ve *f^*'deki herhangi akış arasındaki benzerliği gösterir. Diğer yandan, *K (F, F)* *n*×*n* matrisi oluşturur bu da *F'*deki her iki satır arasındaki benzerliği gösterir. Akış sayısı eğitim süresini etkileyebilir. Akış sayısı ne kadar büyükse, işleme süresi o kadar büyük ve yönlendirmedeki gecikmeler o kadar ağır olur. Bu sorunu çözmek için, bu durumda azaltılmış kuyruk yaklaşımı şiddetle tavsiye edilmektedir. Bu, eğitimde ve ayrıca tahminde işleme süresini azaltmaya yardımcı olur. Öte yandan, regresyon tekniklerinin alt grubu *K(F, F)'*yi eşitlik 3.22 ile tahmin edilmektedir:

$$\hat{K}(F, F) = K(F, F_m) K(F_m, F_m)^{-1} K(F_m, F) \tag{3.22}$$

Eşitlik 3.22'de F_m, F tarafından adlandırılan redresörlerin alt kümesinin *m* akışlarının alt kümesini gösterir. Önceki aşamada *n*'den çok daha düşük olan *K^* kuyruğunun *m* olduğundan bahsetmek dikkat çekicidir. Eğitim aşamasında,

$$\hat{\alpha}(F, s) = \left[\hat{K}(F, F) + \sigma^2 I\right]^{-1} s = Q(F) K(F_m, F) s \tag{3.23}$$

ile ilişkilendirerek,

$$Q(F) = [K(F_m, F) K(F, F_m) + \sigma^2 K(F_m, F_m)]^{-1}. \tag{3.24}$$

elde edilir. Böylece, tahmin için *s^ = K(f^, Fm)α^(F, s)* ifadesi kullanılabilir.

çBME, fil akışlarını tahmin etmek için önerilen üçüncü algoritmadır ve Gauss Karışım Modellerini (GKM) kullanır [69]. GKM'nin her sınıfın özelliğini ürettiği varsayılmaktadır. Bu durumda,

$$x \mid C_e \sim \sum_{i=1}^{N_e} w_i^e N\left(\mu_i^e, \sum_i^e\right)$$
$$x \mid C_m \sim \sum_{i=1}^{N_m} w_i^m N\left(\mu_i^m, \sum_i^m\right) \quad (3.25)$$

elde edilir. Eşitlik 3.25'de, C_e fil akışları sınıfını ve C_m, fare akışları sınıfını gösterir. Buradaki Çevrimiçi Bayes Moment Eşleştirmesi, $(w_i^j, \mu_i^j, \sum_i^j)$ değerini verir. Burada $j = e$, fil akışları sınıfını ve $j = m$, fare akışları sınıfını belirtir.

Karma modellerin öğrenilmesi ve moment yöntemleri olarak bilinen popülasyon parametrelerinin, tahmin yöntemlerinin uygulanması durumunda çBME, çevrimiçi Bayesian öğrenme yöntemleri arasında esnek algoritmadır. Gözlenen veri noktaları, üstel oranda sonraki terimlerin sayısında artışa işaret eder ve bu nedenle çevrimiçi Bayesian öğrenmede karma modellerini esnek olmayan hale getirir. Alternatif dağılım gruplandırması yoluyla sonraki terimlerin tahmini, çBME'yi bu konuda daha gürbüz kılmaktadır. Bu yaklaşık dağılım, sonraki terim dağılımın yeterli momentleri eşleştirilmesiyle elde edilir. Eğer $(C_e|x) < P(C_m|x)$ ise, her ikisi de Bayes Teoremi tarafından üretilen $P(C_e|x) > P(C_m|x)$ eşitsizliği, özellik vektörü x'ten C_e ve C_m sınıflarına bir akışın belirlemesine yardımcı olmaktadır.

Akışların sınıflandırılması hakkında sonuca varmak için Çizelge 3.4'te, sırasıyla hem fil hem de fare akışları için Akış Tamamlama Süresini 99. yüzdelik dilim cinsinden karşılaştırılmıştır [65].

Çizelge 3.4. Fil ve fare ATS 99. yüzdelik dilim açısından karşılaştırılması

Özellikleri	Algoritma	Fil	Fare
Başlık	GSR	% 6,6	% -1,93
Paket	GSR	% 13,66	% -5,25
Başlık ve 3Paketler	GSR	% 14,17	% 0,59
3Paket	çBME	% 13,5	% -1,93

ATS'nin 99. yüzdelik dilimi, akışların % 99'unun tamamlandığı süreyi gösterir; burada kalan %1'lik kısım aykırı değer olarak değerlendirilirken üst sınır olarak düşünülebilir. Çizelge 3.4'deki karşılaştırma sonucu, tüm özelliklere sahip (Başlık ve 3 Paket) GSR'nin fil akışları için en iyi iyileştirmelere ulaştığını ve fare akışlarıyla hafifçe zorla temasa girdiğini göstermektedir [65]. Bu nedenle, akışları sınıflandırmak için modelde GSR kullanılmaktadır.

Veri Merkezi – ATB'de İkili-Kuyruk

Veri Merkezi – ATB'de İkili-Kuyruk hakkında kapsamlı açıklamalar kısım 3.3.2. Kuyruk arabelleği ve öncelik-ATB sınıflandırıcısı altında ele alınmıştı. İkili kuyruğun en ilginç kısmı, fare için en düşük sıraya girme gecikmesine ve fil için en yüksek verime ulaşılması için her kuyrukta farklı minimum ve maksimum ATB eşiğinin (sırasıyla K_{min} ve K_{max} ile gösterilir) bağımsız olarak ayarlanmasıdır. Ayrıca, sığ veya derin eşiğin ayarlanması, taşma veya taşma olgusuna katkıda bulunacaktır. Fare ve fil kuyrukları sırasıyla γ ve δ simgeleriyle tanımlamıştır. γ ve δ aslında anlık kuyruk uzunluğu olup fare ve fil kuyruklarının bir sonraki GDS'deki anlık uzunlukları sırasıyla γ' ve δ' simgeleriyle ifade edilmektedir. Bu durumda,

$$\gamma' \times B \leq \frac{1}{3}\gamma \times B + 0.1667 \times B \tag{3.26}$$

Olarak Eşitlik 3.26 elde edildiğinde, fare akışlarının arabelleği ne eksik kalma ne de taşma problemlerine sahip olamaz. Örneğin, γ 0 olduğunda, γ' fare kuyruğunda minimum kuyruk uzunluğunu K_{min} için sınırlama olan tam arabellek boyutunun %16,67'si yapacaktır. Öte yandan, γ tam arabellek boyutunun %100'ü olursa, γ' maksimum kuyruk uzunluğunu tüm arabellek boyutunun %50'si yapacaktır. Bu durum aslında tam olarak farekuyruğundaki K_{max} için tanımlanan değerdir. Böylece fare kuyruğundaki K_{min} ve K_{max} sırasıyla 16 ve 50 paket olacaktır. Ayrıca, benzer şekilde minimum ve maksimum eşik değerleri, fil kuyrukları için eşitlik 3.27 ile elde edilebilir. Bu durumda fil kuyruğundaki K_{min} ve K_{max} değerleri sırasıyla 34 ve 68 paket olacaktır.

$$\delta' \times B \leq \frac{1}{3}\delta \times B + 0.3333 \times B \tag{3.27}$$

3.4.4. VM ATB-Haritalama zamanlayıcısı

VM-ATB Zamanlayıcısı, Öncelik-ATB yaklaşımında açıklanmıştır ancak önemi dolayısıyla burada yeniden hatırlatma ve bazı özelliklerini ortaya çıkartmak gerekir. ATB-Haritalama zamanlayıcı, ikili-çift kuyruk uzunluğunu ortadan kaldırmak için her ikili-çift arabelleğin giriş ve çıkışını birden çok anahtarda izler. Bu özellik, L4S'ye ulaşmada önemli bir rol oynamaktadır. Daha sonra programlayıcı, ikili çift kuyruklarındaki tıkanıklık basıncını azaltarak K_{max}'ı dinamik olarak fil kuyruğunda ayarlar ve K_{max}'ı artırmaya/azaltmaya karar verir. Kesin olarak belirtmek gerekirse, fare kuyruğunun uzunluğu K_{max}'a ulaşırsa, fare paketlerini işaretlemek veya bırakmak yerine zamanlayıcı, sınıflandırıcıya İhlal Bildirimi (İB) (Violate Notification-VN) gönderir ve algoritmayı değiştirmesi ve küçük oranını göndermesi için onu bilgilendirir. Paketler, Standart Duruma Geri Dön Bildirimi (SDGB) (Back to Standard State Notification-BSN) alana kadar fil kuyruğuna gönderilir. Fil kuyruklarının belirli bir kısmını kullanacak fare paketlerinin oranı, fare kuyruğunda K_{max}'ı aşacak paketlerdir. Fare akışlarının bir kısmı fil kuyruğuna iletildikten sonra, fare akışlarının yüksek önceliği nedeniyle tekrar en kısa gecikmeye sahip olmak için rotayı olabilecek en yüksek hızda geçeceklerdir.

Bununla birlikte, kuyruklar tipik olarak FIFO'dur ve önerilen planlar en kötü durumda fil akışlarını boş bırakacaktır. Bu açlık sorununu çözmek için, zamanlayıcıda Karuna [47] tarafından sağlanan yaklaşım kullanılabilir. Daha net açıklamak gerekirse, öncelik belirlemede katı kurallar kullanmak belirli akışları boş bırakabilir. Fazla yoğunluk durumunda farenin KEKİS veya son tarih akışlarının öncelikleri için tüm bant genişliğini alması gerekirse, son tarihi olmayan fil akışları boş kalacaktır. Böyle bir durumda ulaşım mekanizmasının yapabileceği pek bir şey yoktur ve operatörler şebeke kapasitesini artırmayı düşünmelidir. Diğer senaryoda fare, KEKİS veya son tarih akışları, öncelikleri nedeniyle tüm bant genişliğini alırsa, fil kuyruğundaki son tarihi olmayan büyük akışları boş bırakabilirler. Bu problemin çözümü için boş bırakılan akışların önceliğini yükseltmek için akış yaşlanmasını kullanılmıştır. Zaman aşımı olaylarını gözlemleyerek son ana bilgisayarlarda eksik akışlar belirlenir. Örneğin akış TCP zaman aşımlarıyla karşılaştığında, VM-ATB bu akışı daha yüksek önceliğe yükseltir.

3.4.5. Düşük gecikme ve yüksek verim

Daha önce de ifade edildiği gibi, bu bölümün asıl amacı, IETF tarafından şiddetle tavsiye edilen L4S'yi sağlamaktır:

1) Düşük gecikme süresi için kısa ATS'ye ihtiyaç vardır. Kısa ATS'ye sahip olmak için her anahtar arabelleğinde ATB için düşük *K* değerine ihtiyaç duyulur. Ancak düşük *K* değeri akış verimini düşürebilmektedir.

2) Yüksek verim elde etmek için anahtarın büyük *K* değerine e sahip olması gerekir. Ancak bu durumda da kuyruk gecikmesi artmaktadır.

3) Bu problemlerle başa çıkmak için, birbirinden ödün vermeden aynı anda hem düşük gecikme hem de yüksek verim elde etmek için literatürde kuyrukları ikiye ayıran bir yaklaşım olan VM-ATB önerilmektedir.

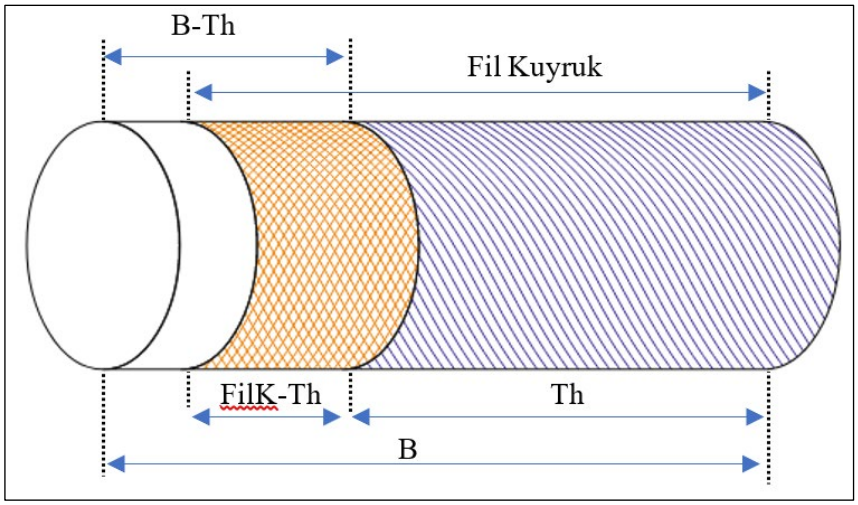
Her iki çift bağlı arabellek kuyruğu için K eşik değeri en düşük ve en yüksek değer aralığında seçilmektedir. Ancak, eşik değerinin K_{max}'a ulaştığı durumda kuyruk uzunluğu ve dolayısıyla kuyruk gecikmesi artabilmektedir. Fakat gecikmeden ödün vermeden yüksek verime ulaşmak için algoritmanın çok daha fazla ani trafiği emmesi gerekir. Bilindiği kadarıyla, mikro çoğuşma trafiği meydana geldiğinde ve paketler işaretlendiğinde veya bırakıldığında, K_{max}*'tan* sonra hala yeterli boş tampon alanı kalmaktadır [70].

Bu nedenle K eşik değerinin dinamik olarak ayarlanması gerekir. Dinamik Eşik (DE: Dynamic Threshold-DT) algoritması, anahtar üretici firmalar tarafından yaygın olarak kullanılmaktadır. DE algoritmasında, kuyruğun uzunluğu, arabellekteki boş kapasite miktarıyla orantılı olarak tüm çıkış portları tarafından paylaşılır. Ancak, DE'nin yeni aşırı yüklenen bağlantı noktalarının boş kalmaması için arabellek parçası ayırması gerekir. Böylece mikro çoğuşma trafiğinden gelen paketler, arabellekte boş arabellek alanı olduğunda işaretlenecek veya bırakılacaktır. Bu nedenle, gecikmeden ödün vermeden mikro çoğuşma trafiğini Sönümlemek için fil kuyruğundaki K_{max}'tan sonra arabellek kapasitesini kullanmak önemlidir.

Ayrıca, her iki ikili arabellek kuyruğunun K_{max}'a ulaştığı durumda sınıflandırıcı, zamanlayıcıdan bildirim alarak fil kuyruğuna fare paketi göndermeye başlar. Ardından fare akışları, yüksek öncelikleri sayesinde, düşük gecikme süresi elde etmek için hızlı şekilde

tekrar yoldan gidecektir. Ancak bu senaryoda, fil akışlarından gelen paketler daha fazla işaretlenmelidir. VM-ATB, fil kuyruklarında K_{max}'tan sonra arabellek kapasitesini tam olarak kullanarak gecikmeden ödün vermeden mikro çoğuşma trafiğini en yüksek verime sahip olacak şekilde sönümlemek mümkündür.

Şekil 3.7, VM-ATB'deki fil akışları için arabellek tasarımını göstermektedir. Burada arabellek kapasitesi ve fil kuyruğundaki K_{max}'tan sonraki kuyruk uzunluğu, sırasıyla *B-Th* ve *EQ-Th* ile gösterilir.



Şekil 3.7. Fil akışları için arabellek gösterimi

Kuyruğun eşiği aşan kısmı (α) eşitlik 3.28 ile ifade edilebilir.

$$\alpha = \frac{EQ - Th}{B - Th}, \tag{3.28}$$

Böylece, sonraki GDS'deki α değeri α' Eşitlik 3.28'e benzer şekilde hesaplanabilir.

$$\alpha' = \frac{EQ' - Th}{B - Th}, \tag{3.29}$$

Bilindiği gibi *EQ*, akış sayısı (*N*) ve tek akışın tıkanıklık penceresi (*W*) ve ayrıca bağlantının kapasitesi (*C*) ile ilgilidir.

$$EQ = NW - C \times GDS. \tag{3.30}$$

EQ', N ve *W* ile ilgili sonraki GDS'deki kuyruk uzunluğudur. Böylece, eşitlik 3.31. ile *EQ'* çıkarımı yapılabilir.

$$EQ' = \beta N\left(\frac{W}{2}+1\right) + (1-\beta)N(W+1) - C \times RTT. \tag{3.31}$$

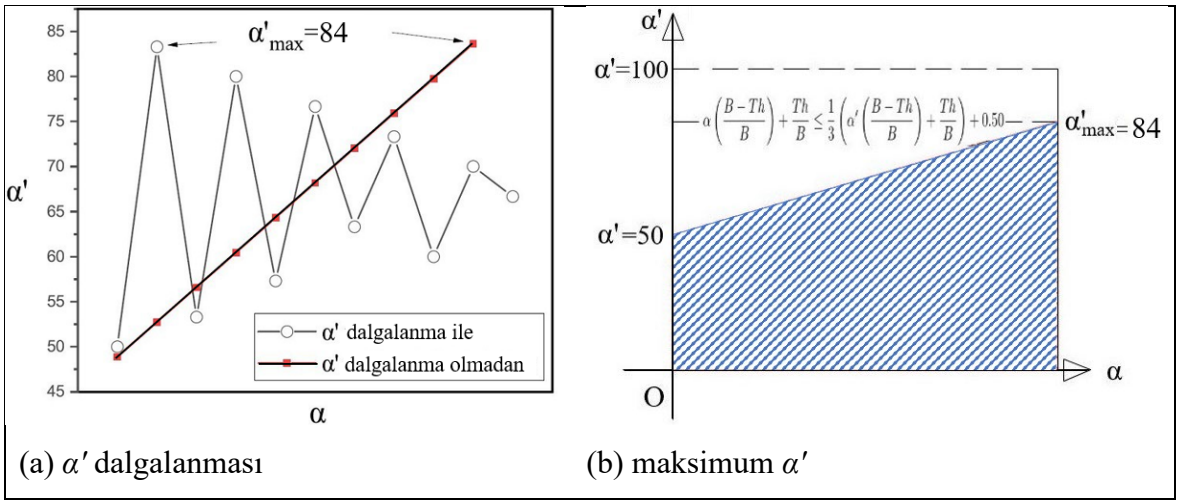
Ayrıca, α' için eşitlik 3.32'deki gibi bir sınırlama belirlenebilir.

$$\alpha\left(\frac{B-Th}{B}\right) + \frac{Th}{B} \leq \frac{1}{3}\left(\alpha'\left(\frac{B-Th}{B}\right) + \frac{Th}{B}\right) + 0.50 \tag{3.32}$$

Çizelge 3.5. VM-ATB Geleneksel algoritması

	Giriş: *Akış_i, 7_özellikler (kaynak_IP, hedef_IP, kaynak_port, hedef_port, protocol, sunucu_ve_istemci), FareK_K$_{max}$,FilK_K$_{max}$*
	Çıktı: İşaretenen Paketler
1	*Sınıflandırıcı* ← *Akış_i_ve 7_özellik*
2	if *akıl?sınıf == 1* then
3	│ *fil* ← *true*
4	else
5	│ *fil False*
6	end
7	*işarete_dönüş*
8	if *fil == True* then
9	│ *fil_kuyruk_j* ← *Akış_i*
10	else
11	│ *fare_kuyruk_j* ← *Akış_i*
12	end
13	*fare_kuyruk_j_Eş* ← $\frac{\text{ağırlık}_j}{\text{ağırlık\_nom}}$ x $ATB\_Eş$
14	if *fare_kuyruk_uzunluk_j => fare_kuyruk_j_th* then
15	│ *fil_kuyruk _j* ← *i_fare_akışı_işaretlendi* ← *False*
16	end
17	return *fare_işaretlendi*
18	if *fil_kuyruk _j => fil_kuyruk _Th* then
19	│ $MP \leftarrow \frac{filQ - 0{,}67 \times filB}{0{,}16 \times filB}$
20	│ PakNum ← paket + PakNum
21	end
22	while $\frac{num}{toplam\ paket} < MP$ do
23	│ *fil_işaretlendi*← *True*
24	│ *PakNum* ← *paket* + *PakNum*
25	end

Çizelge 3.5 ve Şekil 3.8(a) ve Şekil 3.8(b), VM-ATB'nin α' için sınırlamayı nasıl belirlediğini ve işaretlemek için akışların oranını nasıl seçtiğini göstermektedir. Ancak, talihsiz bir trafik kesilmesi durumunda Şekil 3.8(a)'deki gibi dalgalanmalar meydana gelecektir.



(a) α' dalgalanması (b) maksimum α'

Şekil 3.8. α ve α' dalgalanmaları ve sınırlamaları

α' için sınırlama konmasının nedeni, taşma olgusunun yanı sıra kuyruk taşma olgusunu da önlemektir. Bunun anlamı, eğer α – 100 paket olursa, α' taşmayı önlemek için sonraki GDS'de %84,44'e (84 paket) düşecektir. Ancak α küçülürse α' alt akışı önlemek için bir sonraki GDS'de yükseltilecektir.

Daha kesin belirtmek gerekirse, α eğer 50 paketten daha yüksek iken α', α'dan daha küçük olacaktır. Aksi durumda, α eğer 50 paketten küçükse, α' α'dan daha büyük olacaktır. , bu α' nün dinamik olarak α tarafından belirlendiği anlamına gelir. Daha net bir şekilde ifade etmek gerekirse, α oldukça küçük olduğunda, (örneğin, α = 0) yaklaşım, alt aşımı önlemek için α' = %50 olacaktır. Öte yandan, α çok büyük olduğunda, örneğin α = 100 paket olduğunda, α'dan yapılan çıkarımla α' = 84 paket olacak ve böylece tıkanıklık oranını azaltacak ve taşmayı önleyecektir.

Şekil 3.8(b), bu durumu görsel olarak ifade etmektedir. Şekil 3.8(b)'den görüleceği üzere taralı bölge α' VM-ATB'si tam olarak kullanabilir. Böylece, ortalama olarak α', yaklaşık 61 paket olacak ve gecikmeden ödün vermeden çıktıya yardımcı olmak için fil kuyruğunda maksimum sınır 84 paket olacaktır. Ancak, önceki çalışmalarda aynı alandaki tüm paketler işaretlenmektedir. VM-ATB'de işaretli akışların oranını belirlenmesi gerekmektedir. Bu amaç için Şekil 3.8(b)'de görülen gölgeli bölgeden yararlanılabilir. β, α' ve EQ' bağlı olarak

işaretli akışların oranını gösteri. Dolayısıyla, *α'* ve *EQ'*, *β* ve *α* arasında doğrudan bir ilişki bulunmaktadır. İlk olarak, eşitlik 3.28 ve 3.29'dan

$$= \frac{EQ \ - \ Th}{B \ - \ Th}, \tag{3.33}$$

elde edilir. Eşitlik 3.30 ve eşitlik 3.31 ile *EQ*'-*EQ* elde edilir.

$$= N(1 - \frac{\beta}{2}W, \tag{3.34}$$

Devamında, Eşitlik 3.33 ve Eşitlik 3.34'u bir araya getirilerek.

$$B(1 - Th)(\alpha' - \alpha) = N\left(1 - \frac{\beta}{2}W\right) \tag{3.35}$$

β, α ve α' cinsinden eşitlik 3.36 ile ifade edilebilir.

$$\beta = \frac{2(N - B - Th)(\alpha' - \alpha)}{NW}. \tag{3.36}$$

$$\beta \geq \frac{2(N - B - Th)\left(0.5 - \frac{2}{3}\alpha\right)}{NW}. \tag{3.37}$$

Eşitlik 3.32'yi kullanarak Eşitsizlik 3.37'da *NW*' yerine $(EQ + C \times GDS)$ ve *α* yerine $\frac{Q - Th}{B - Th}$ yazarak, eşitsizlik 3.38 elde edilir.

$$\beta \geq \frac{2(N - B - Th)\left(0.5 - \frac{2}{3}\frac{EQ - Th}{B - Th}\right)}{EQ + C \times GDS} \tag{3.38}$$

Devamında, *β* eşitsizlik 3.39'da olduğu gibi kolaylıkla ayarlanabilir.

$$\beta \geq \frac{2N - B + \frac{4}{3}EQ - \frac{1}{3}Th}{EQ + C \times RTT}, \tag{3.39}$$

4. BULGULAR VE DEĞERLENDİRME

Önerilen yaklaşım, benzer yaklaşımlarla karşılaştırarak, benzetim sonuçları sunulmuştur. Her yöntem için kullanılan araç ve benzetim ortamı farklı deneysel senaryolar kullanarak gösterilmiştir. Bulgular ve değerlendirmede ilk olarak Öncelik-ATB benzetim sonuçları ele alınmış devamında Derin-PÖ ile ATB-Haritalama Benzetim Sonuçları irdelenmiştir.

4.1. Öncelik-ATB benzetim sonuçları

Öncelik-ATB'nin performansı büyük ölçekli NS-2 ağ ortamında gerçekleştirilen benzetimlerle gösterilmiş ve değerlendirilmiştir. Arabellek boyutu her KA için 300 KB olarak belirlenmiştir. Paylaşılan her bir port başına ilk gelen ilk hizmeti alır şeklinde 8 adet KA ayrılmıştır. Bunların her birisi 1,5 KB olacak şekilde eşit olarak paylaştırılmıştır. Tüm bağlantıların bant genişliği 10 Gbps'dir ve GDS eşiği 39 mikro saniyeye kadar ayarlanmaktadır. Çizelge 4.1'te benzetimde kullanılan parametreler görülmektedir. Performans değerlendirmesinde temel ölçüt olarak ATS ele alınmıştır. Bu nedenle, tüm akışların genel ortalama ATS'si ve farklı akış boyutlarındaki (küçük, orta ve büyük) ortalama ATS dikkate alınmıştır. Ayrıca, varsayılan aktarım protokolü olarak VMTCP kullanılmış, pencere boyutu 16 paketi akış boyutu 50000 olarak belirlenmiştir.

Çizelge 4.1. Öncelik-ATB benzetim parametreleri

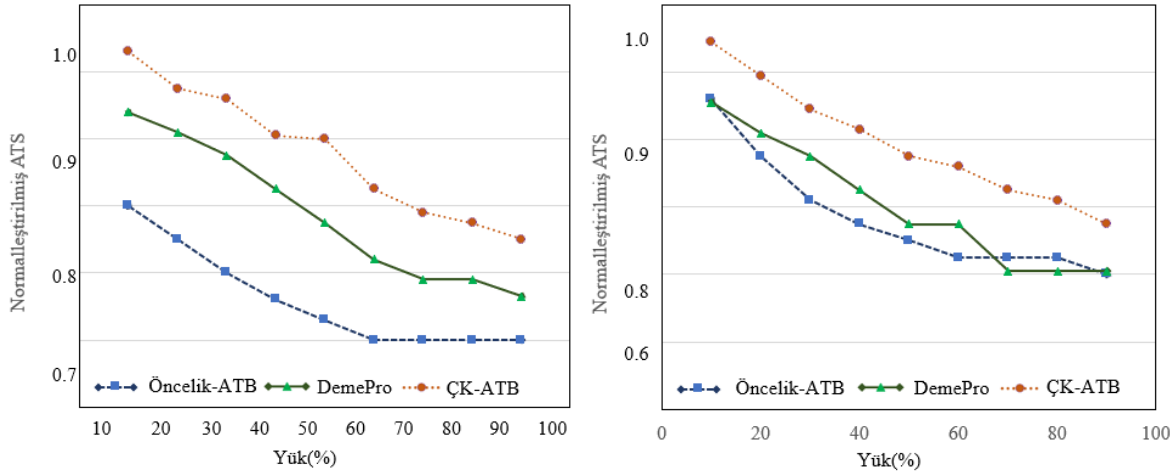
Parametreler	Değerler
Bağlantı kapasitesi	10 Gbps
Gidiş dönüş süreleri	15 − 35 ms
FareK'ta minimum ATB eşiği	34 paket
FilK'ta maksimum ATB eşiği	68 paket
Fare kuyrukta minimum ATB eşiği	16 paket
Fare kuyrukta maksimum ATB eşiği	50 paket
İletim protokolü	VMTCP
Kuyruk arabelleği boyutu	300 KB
Topoloji	yaprak-omurga (leaf-spine)
Host	144

ÇK-ATB için farklı zamanlayıcılar ele alınmıştır. En iyi performansı elde etmek için Açık Ağırlıklı Adil Kuyruklama (Deficit Weighted Fair Queuing-DWFQ), Kesin Öncelik (Strict Priority-SP), Açık Sıra (Deficit Round Robin-DRR) ve Ağırlıklı Adil Kuyruk (Weighted Fair Queuing- WFQ) gibi farklı zamanlayıcılar kullanılmıştır. Zamanlayıcılar arasında en iyi ATS WFQ kullanılarak elde edilmiştir. Buna dayanarak, aynı zamanda DWFQ kullanılarak üretilen benzetim sonuçları elde edilmiştir. Öncelik-ATB için parametrelerin belirlenmesi için kuyruk arabelleklerindeki eşik değerleri ve çift-çıkış port arabellekleri Eşitlik 2.2'e göre yapılandırılmıştır.

ÇK-ATB ve DemePro açık kaynak kodları sağlamaktadır. Bu nedenle, çalışmada karşılaştırmalar için bu kodlardan yararlanılmıştır. Topoloji, 12 yaprak (ToR) anahtarı ve 12 omurga (Core) anahtarı ile birlikte 144 ana yaprak-omurga topolojisi içeren ÇK-ATB şeklinde ayarlanmıştır. Her yaprak anahtarı, ana bilgisayarlar ve omurga için sırasıyla 12 ve 10 Gb/sn aşağı ve yukarı bağlantılara sahiptir. Omurga boyunca temel GDS süresi (4 atlama) 39 mikro saniyedir. Yük dengeleme için ECMP kullanılmıştır. Önbellek, İnternet arama, Hadoop ve Veri madenciliği sırasıyla 909 KB, 1669 KB, 4150 KB ve 7501 KB gibi ortalama akış boyutuna sahip iş yükleri için 4 farklı karma akış dağılımı kullanılmıştır.

Ayrıca, akış boyutu dağılımlarının toplam paket sayıları Hadoop için 5829, önbellek için 26505, veri madenciliği için 3213 ve internet aramaları için 14553tür. Hadoop dağılımı büyük, orta ve küçük akış boyutları için sırasıyla 4131, 1523 ve 175'tir. Önbellek akış boyutu, büyük 16034, orta 10253 ve küçük akış boyutu için 218dir. Veri madenciliği, büyük akış boyutu için 2638'lik dağılıma sahipken, orta ve küçük değerler sırasıyla 420 ve 155'tir. İnternet araması için dağıtımlar, 7881 büyük 6180 orta ve 392 küçük akış boyutlarındadır.

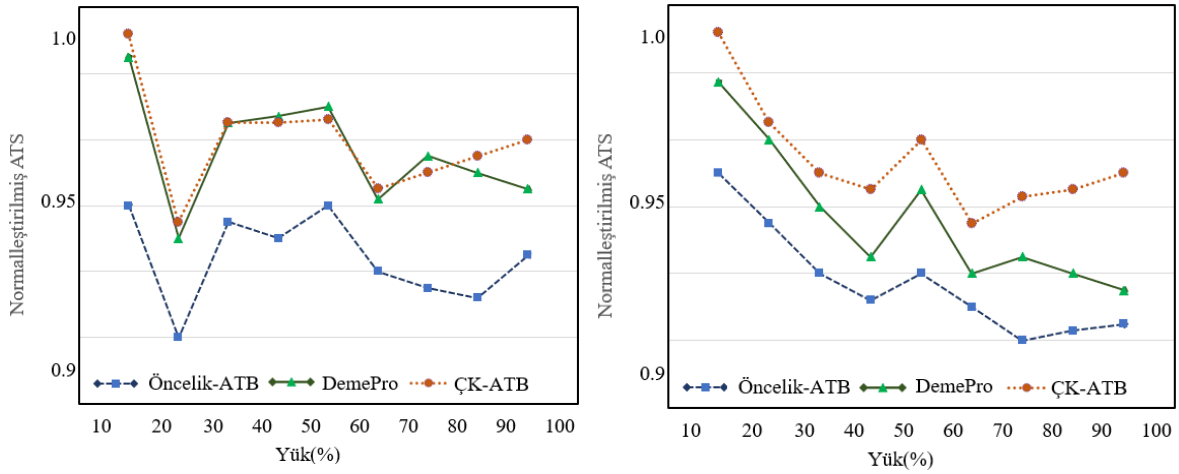
Tüm akışlar için genel ATS', kısa akışların ATS'leri ve her sınıftaki % 99'luk dilimdeki tüm akışların ortalama ATS'si karşılaştırılmıştır. Daha önce de belirtildiği gibi, ATS esas olarak sırasıyla fare ve fil akışları için gecikme ve verim ile belirlenmektedir.



(a) Kısa akışlı iş yükü (b) Büyük akışlı iş yükü

Şekil 4.1. İş akış türleri için ATS karşılaştırılması

Öncelik-ATB'nin performansını değerlendirmek için kapsamlı deneysel çalışmalar gerçekleştirilmiştir. Öncelik-ATB hem kısa hem de büyük akışlarda daha iyi performansa sahiptir. Şekil 4.1(a) ve Şekil 4.1 (b), de iş akış türleri için benzetim sonuçları incelendiğinde Öncelik-ATB'nin tüm yüklerde iyi performans gösterdiği görülmektedir. Önbellek, İnternet araması, Hadoop ve Veri madenciliği gibi tüm farklı iş yükleri için benzetim gerçekleştirilmiştir. Şeki 4.2'de Veri madenciliği ve Hadoop iş yükü için ATS karşılaştırması görülmektedir. Şekil 4.2'den de görüleceği üzere Öncelik ATB DemePro ve ÇK-ATB yaklaşımlarına göre daha iyi performans sergilemiştir.



(a) Veri madenciliği (b) Hadoop

Şekil 4.2. Veri madenciliği ve Hadoop iş yükü için ATS karşılaştırılması

4.2. DPÖ ile Veri Merkezinde ATB-Haritalama yaklaşım benzetim sonuçları

Bu kısımda VM ATB-Haritalama yaklaşımının test ortamı ve benzetim sonuçları verilmektedir. Tezde, büyük ölçekli NS-2 ağ benzetimi üzerinde Derin-karar (Deep-decision) uygulanmıştır. Derin-karar işlemlerinde Python 3.6 ve TensorFlow'dan yararlanılmıştır. Uygulamada derin-karar işlemleri ve ağ benzetimi için donanım olarak Intel(R) Core (TM) i9-7900X CPU@ 3.30 GHz işlemci) kullanılmıştır. Yapay Sinir Ağını oluştururken ileri ve geri yayılım şekillendirilmiş ve devamında her biri sırasıyla teker teker işlemci üzerinde çalıştırılmıştır.

Benzetimde hesaplamaları işlemci üzerinde yürütmek zaman alıcı olduğundan, işlemciden çok daha hızlı çalışan Grafik İşlemci Ünitesi (GİÜ: Graphical Procecessing Unit-GPU) tercih edilmesi gerekir. Hatta yoğun hesaplama nedeniyle işlem yükünü azaltarak gecikme süresini azaltmak için gerekli zamandan kazanmak için Chen ve arkadaşları [73] CPU-GİÜ hibrit eğitim yaklaşımını önermektedirler. Bu şekilde, GİÜ modeli öğrenirken aynı zamanda işlemci veri toplama işlemelerini gerçekleştirmekte ve bu şekilde benzetim sonuçları daha kısa sürede elde edilmektedir.

Benzetim için performans ölçütü olarak ATS kullanılmıştır. Bu amaçla, çeşitli boyutlardaki (küçük, orta, büyük) veri akışlar için ortalama toplam *veri akışı* ve ortalama toplam *ATS* hesaplanmıştır. Ek olarak, taşıma protokolü olarak 16 paket pencere boyutunda VMTCP protokolü kullanılmıştır ve benzetimde akış boyutu başına 50.000 akışla çalıştırılmıştır. WFQ zamanlayıcı olarak kullanılması en iyi ATS elde etmek için yardımcı olmaktadır, zamanlayıcılar hakkında bölüm 4.1'de verilmiştir.

ÇK-ATB ve DemePro açık kaynak kodları sağlamaktadır. Bu nedenle, çalışmada karşılaştırmalar için bu kodlardan yararlanılmıştır. Bunun için 144 ana yaprak-omurga topolojisi ile 12 yaprak (ToR) anahtarı ve 12 omurga (orta) anahtardan oluşan ÇK-ATB benzer topoloji kullanılmaktadır. Her yaprak anahtarı, ana bilgisayara ve omurgaya 12 ve 10 Gbit/sn'lik yükleme ve ileri bağlantıların sağlar. Tüm omurga için temel GDS (4 atlama) 39 mμ ve yük denetimi için Eşit Maliyetli Çoklu Yol (Equal Cost Multipath-ECMP) yönlendirme stratejisi kullanılmaktadır. Önbelleğe alma, İnternet araması, Hadoop ve Veri madenciliği gibi iş yükleri için 4 çeşitli karma-akış kullanılmıştır. Belirli medyan akışlar 914 KB, 1671 KB, 4149 KB ve 7495 Kb boyutunda verilmektedir. Çizelge 4.2'de ATB-

Haritalama benzetiminde önbelleğe alma, Hadoop, internet araması ve Veri madenciliği akışlarının dağılımı verilmiştir.

Çizelge 4.2. ATB-Haritalama benzetiminde akışların dağılımı

Ortam / Akış	Küçük	Orta	Büyük	Toplamı
Önbelleğe alma	218	10253	16034	26505
Hadoop	175	1523	4131	5829
İnternet araması	392	6180	7881	14453
Veri madenciliği	155	420	2638	3213

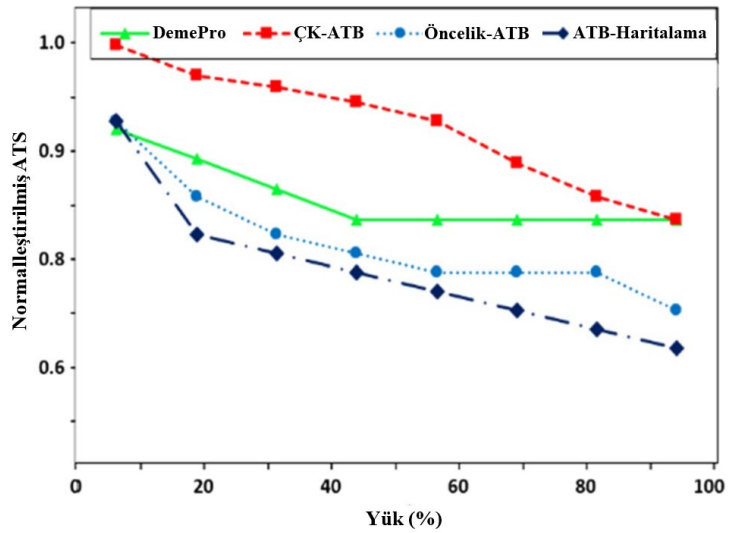
ATB-Haritalama yaklaşımında parametreleri belirlemek ve DPÖ tabanlı algoritmanın yanı sıra Eşitlik 2.1 ve 2.2 ‘ye bağlı olarak kuyruk arabelleği ve ikili-çıkış bağlantı noktası arabelleği için eşik değerlerinin belirlenmesi gerekmektedir. Bu amaçla ATB-Haritalama benzetiminde kullanılan parametreler Çizelge 4.3’de sunulmaktadır.

Çizelge 4.3. ATB-Haritalama benzetim parametreleri

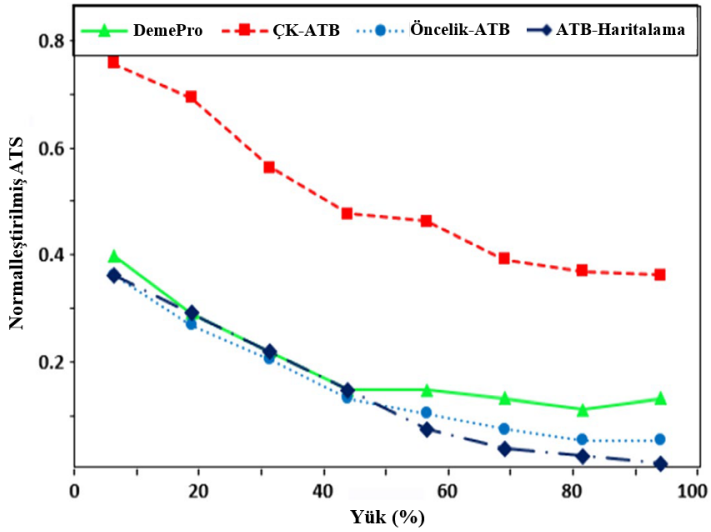
Parametreler	Açıklamalar
Benzetimler	NS-2
Bağlantı kapasitesi	10 Gbps
Gidiş-dönüş süreleri	55 − 39 mikro saniye
Taşıma protokolü	VMTCP
ÇPA'de ATB işaretleme eşiği	-
KA'de ATB işaretleme eşiği	Kesirli K ile kuyruk başına ATB
Arabellek ve bağlantı noktasında *K* değeri	Eşitlik 2.2’ye göre hesaplanır
Kuyruk arabelleğinin boyutu	300KB
Kuyruk arabelleği sayısı	8
Topoloji	Yaprak-omurga(leaf-spine)
Ana makine	144

Kısım 4.1’de olduğu gibi tüm akışlar için genel ATS', kısa akışların ATS'leri ve her sınıftaki % 99'luk dilimdeki tüm akışların ortalama ATS'si ile karşılaştırılmıştır. Daha önce de belirtildiği gibi, ATS esas olarak sırasıyla fare ve fil akışları için gecikme ve verim ile belirlenmektedir.

Derin pekiştirmeli öğrenme ile gerçekleştirilen ATB haritalama yaklaşımı ÇK-ATB, DemoPro ve Öncelik ATB yaklaşımlarıyla karşılaştırılmıştır. Şekil 4.3'te kısa akışlar için ATS'nin genel performansı ve Şekil 4.4'da ise tüm akışlar için ATS'nin 99. yüzdelik dilimdeki karşılaştırması görülmektedir. Şekil 4.4'dan görüleceği üzere ATB-haritalama en düşük gecikmeye 99. yüzdelik dilimde ulamaktadır.



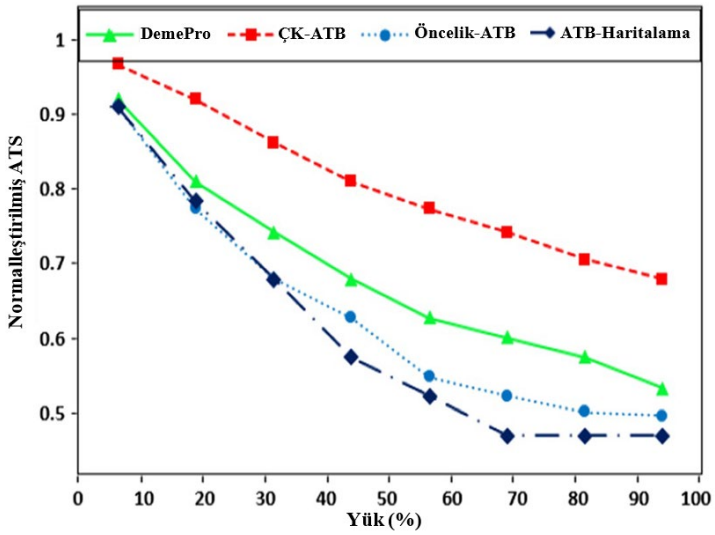
Şekil 4.3. Kısa akışlar için ATS'nin genel performansı



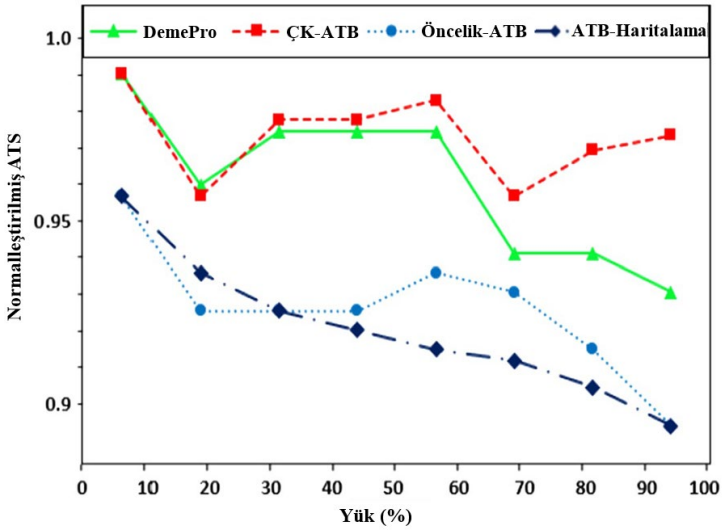
Şekil 4.4. Tüm akışlar için ATS'nin 99. yüzdelik dilimdeki karşılaştırılması

Şekil 4.5'de kısa ve Şekil 4.6'de ise büyük akışlı iş yükleri için ATS'lerin karşılaştırması görülmektedir. Şekil 4.57 ve Şekil 4.6'den de görüleceği gibi ATB-haritalama yaklaşımı hem kısa hem de büyük veri akışlarında diğer yaklaşımlara göre daha iyi performans sergilemiştir. Diğer bir deyişle, ATB-haritalama yaklaşımında performansı artışı görülmüştür. Performans

artışının temel nedenleri olarak ATB-Haritalama, her ikili-çift kuyruğu için K_{min} ve K_{max}'ı eşiği dinamik olarak ayarlaması verilebilir. Ayrıca, ATB-Haritalamada kuyruk arabelleği yüksek verim ve düşük kayıp nedeniyle K_{max}'ı aştığında, kısa akışların bir bölümü çift kuyruğunun uzunluğunu ölçmek suretiyle büyük akışlar için ayrılan arabelleğe gönderilebilmektedir [74].



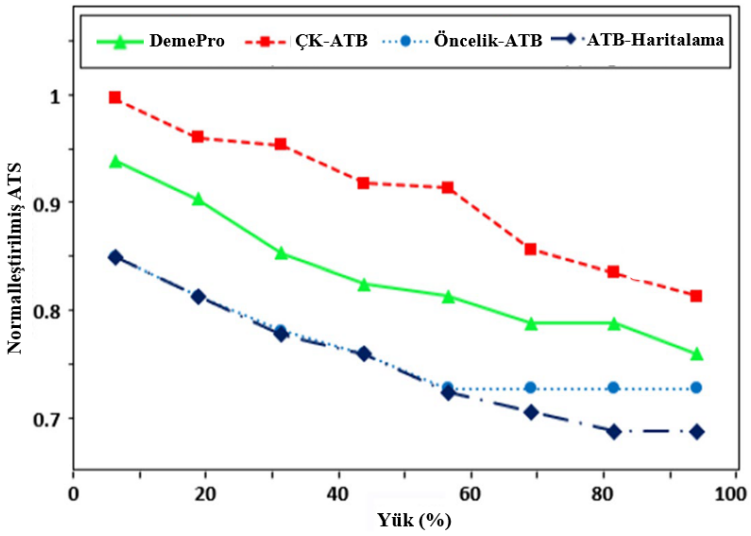
Şekil 4.5. Kısa akışlı iş yükü için ATS'yi karşılaştırılması



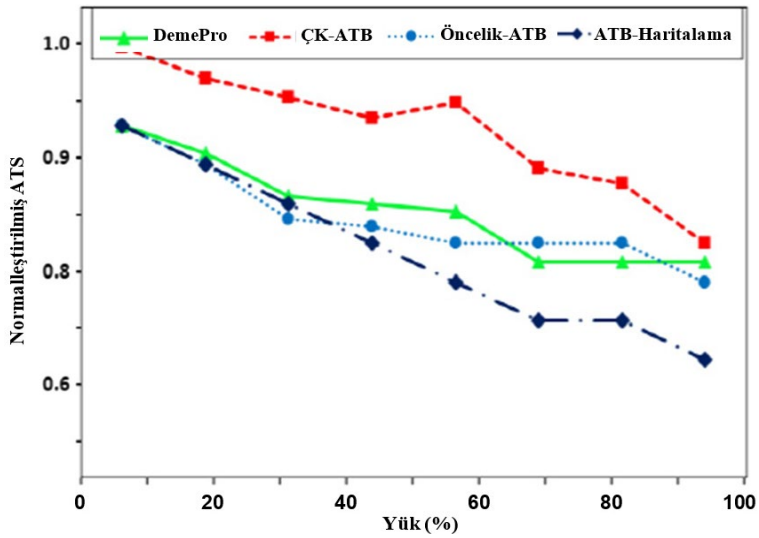
Şekil 4.6. Büyük akışlı iş yükü için ATS karşılaştırılması

Diğer yandan, ATB-Haritalama, bir yaşlanma algoritması kullanarak veri açlığını (starvation) önleyebilmekte ve çok az bir kayıpla mikro çoğuşma trafiğini sönümleyebilmektedir (absorb). ATB-Haritalama yaklaşımının kısım 4.1'deki yükler altında nasıl bir davranış sergilediği gözlemek amacıyla daha kapsamlı bir değerlendirme çalışması

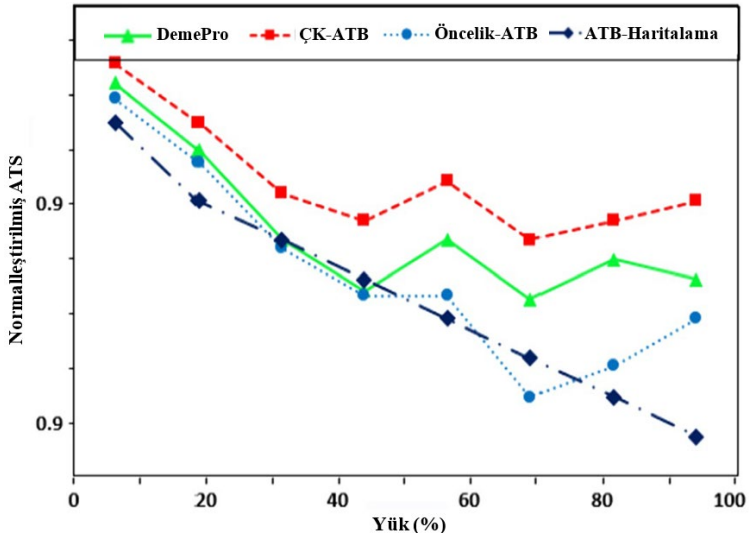
yapılmıştır. Bu amaçla Önbellek, Web araması, Hadoop ve Veri Madenciliği iş yükleri altında yaklaşım test edilmiştir. Şekil 4.7-4.10'de ATS'nin tüm akışlarda sırasıyla, önbellek, web araması, Hadoop ve veri madenciliği iş yükleri için karşılaştırılması görülmektedir.



Şekil 4.7. ATS'nin tüm akışlarda önbellek iş yükü için karşılaştırılması

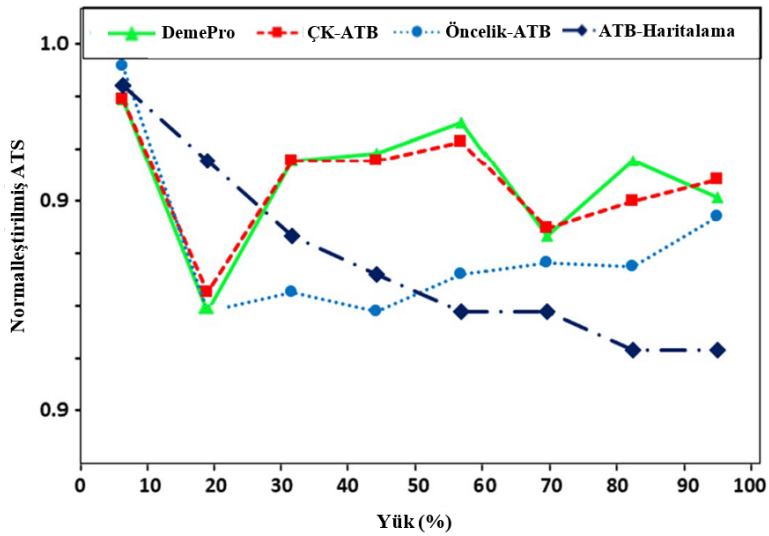


Şekil 4.8. ATS'nin tüm akışlarda web arama iş yükü için karşılaştırılması



Şekil 4.9. ATS'nin tüm akışlarda Hadoop iş yükü için karşılaştırılması

Şekil 4.10'de görüldüğü gibi ATB-Haritalama yaklaşımı başlangıçta diğer yaklaşımlarla aynı performansı göstermektedir. Benzer durum Şekil 4.7-10'de verilen diğer yüklerde de gözlenmektedir. Ancak, deneylerde benzetim süresi ilerledikçe ATB-Haritalama yaklaşımı tüm yüklerde en iyi performansı sergilemektedir. Bunun nedeni öğrenme algoritmasının yakınsama için belirli bir zamana ihtiyaç duymasındandır. Bu durum, ATB-Haritalama yaklaşımı büyük akışlar için daha iyi performans göstermekle birlikte kısa akışlarda düşük performans olarak şeklinde ortaya çıkmaktadır çünkü hesaplamada ihtiyaç duyulan süre genellikle kısa akışların için karar alınmadan önce bitmektedir.



Şekil 4.10. ATS'nin tüm akışlarda veri madenciliği iş yükü için karşılaştırılması

Bu aslında bir ölçeklenebilirlik problemi olarak ele alınabilir. AuTo[73], ölçeklenebilirlik problemini çözmek için Periferik ve Merkezi Sinir Sistemlerini taklit eden iki seviyeli bir DPÖ sistemi geliştirmiştir. Ancak, bu işlem sorunun ortaya çıkmayacağı son ana bilgisayarda gerçekleştirilmektedir. Oysa bu sorunun anahtarların içinde çözülmesi gerekir. Ayrıca, kısa akışın karşılaştırmasına yakından bakılacak olursa, benzetim süresi uzadıkça yaklaşım sonuçlarının iyileştiği görülmektedir. Son olarak, en düşük gecikmeye sahip anahtarların içindeki kısa akışların hesaplanmasının hala çözülmesi gereken bir sorun olduğu görülmektedir.

5. SONUÇ VE ÖNERİLER

Bu tezde, hatalı işaretleme problemini çözmek için bir derin pekiştirmeli öğrenme tabanlı öncelik ATB Haritalama yaklaşımı önerilmiştir. Önerilen yaklaşımda, arabellekten gelen akışlar sınıflayıcı tarafından fare ve fil akışları olarak sınıflandırılmakta ve devamında çıkış port arabelleği eşiğinde işaretlenerek öncelik verilmektedir. Ayrıca, çoklu-çift çıkış port arabelleğinde paketlerin hatalı işaretlenmesinden kaçınmak için derin pekiştirmeli öğrenme tekniklerinden yararlanmaktadır. Önerilen yaklaşımın etkinliği ve verimliliği büyük ölçekli bir NS-2 benzetim ortamında test edilmiştir. Çalışmada elde edilen sonuçlar detaylı bir şekilde incelenmiş ve sunulmuştur. Akış Tamamlama Süresi metriğine dayalı değerlendirmeler, önerilen yaklaşımın hedeflerine ulaşmada başarılı olduğunu ortaya koymaktadır. Önerilen yaklaşımın iletim ağının performansını arttırmadaki gereksinimleri karşılamak için sağladığı katkılar şu şekilde özetlenebilir:

- Diğer çalışmalara benzer şekilde küçük akışlar fare akışları, büyük akışlar ise fil akışları olarak tanımlanmıştır. Fare akışları, Gaussian Süreç Regresyonunu (GSR) kullanarak fil akışlarından ayrılmıştır. Özellikler kümesi göz önüne alındığında, bu parametrik olmayan Bayes öğrenme modeli akış boyutlarındaki dağılımı belirleyebilmektedir. Yeni bir akış olduğunda, GSR Gauss dağılımını hesaplayarak GSR özellikleri sayesinde en iyilenmiş akış değerine ulaşılabilmektedir.
- Ayrılmış fare ve içindeki fil akışlarını göndermek için ikili kuyruklu arabellek önerilmiştir. Çift-bağlı bu kuyruk yapısı, gelen akışları fare ve fil akışları olmak üzere iki farklı kuyrukta düzenler. Bu sayede, ATS akış en iyilemesi için farklı önceliklere uyum sağlayabilmektedir.
- Mikro çoğuşma trafik işlevleri ve paketleri işaretlendiğinde veya bırakıldığında, ATB işaretleme eşiğinden sonra hala yeterli boş arabellek alanı bulunur. Mikro çoğuşma trafiği durumunda fare paketlerinin bir kısmını fil kuyruğuna gönderilerek ve ayrıca fil arabelleğindeki ATB işaretleme eşiği dinamik olarak arttırılarak VM-ATB fare akışlarını asla yüksek öncelik nedeniyle işaretlenmemekte ve mikro-çoğuşma soğurulmaktadır.
- Önerilen ATB-Haritalama yaklaşımı, mevcut benzer yaklaşımlar ile kıyaslandığında ÇK-ATB'ye oranla daha iyi performans sergilemiş olup bu yönüyle çalışma literatüre katkı sağlamıştır.

KAYNAKLAR

1. Kandula, S., Sengupta, S. Greenberg, A., Patel, P., and Chaiken, R. (2009). *The Nature of Data Center Traffic: Measurements & Analysis.* Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference, 202-208.

2. Alizadeh, M., Greenberg, A., Maltz, D. A. Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and Sridharan, M. (2011). *Data center TCP (DCTCP)*, Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 41(4), 63-74.

3. Wu, H., Ju, J., Lu, G., Guo, C., Xiong, Y., and Zhang, Y. (2012). *Tuning ECN for data center networks.* ACM International Conference on emerging Networking Experiments and Technologies (CoNEXT), 25-36.

4. Bai, W., L. Chen, L., Chen, K., and Wu, H. (2016). *Enabling ECN in multi-service multi queue data centers.* USENIX Associations Symposium on Networked Systems Design and Implementation (NSDI), 537-549.

5. Chen, L., Lingys, J., Chen, K., and Liao, X. (Editörler). (2021). *Datacenter Traffic Optimization with Deep Reinforcement Learning.* Wiley: John Wiley ve Sons, 223-259

6. Choudhury, A., and Hahne, E. (1998). Dynamic queue length thresholds for shared memory packet switches. *IEEE/ACM Transactions on Networking (ToN)*, 6(2), 130-140.

7. Lu, Y., Fan, X., and Qian, L. (2018). Dynamic ECN marking threshold algorithm for TCP congestion control in data center networks. *Computer Communications (COMCOM)*, 129, 197-208.

8. Gao, X., Chen T., Chen, Z., and Chen, G. (2018). NEMO: Novel and efficient multicast routing schemes for hybrid data center networks. *Computer Networks (CN)*, 138, 149-163.

9. Wang, T., Wang, L., and Hamdi., M. (2018). A cost-effective low-latency overlaid torus based data center network architecture. *Computer Communications (COMCOM)*, 129, 89-100.

10. Shan, D., and Ren, F. (2017). *Improving ECN marking scheme with micro-burst traffic in Data Center Networks.* IEEE International Conference on Computer Communications (INFOCOM), 1-9.

11. Gao, C., Lee, V. C., and Li, K. (2017). DemePro: Decouple packet marking from enqueuing for multiple services with proactive congestion control. *IEEE Transactions on Cloud Computing (TCC)*, 22(3), 970-981.

12. Chkirbene, Z., Gouissem, A., Hadjidj, R., Foufou, S., and Hamila, R. (2018). Efficient techniques for energy saving in data center networks. *Computer Communications (COMCOM)*, 129, 111-124.

13. Chkirbene, Z., Gouissem, A., Hadjidj, R., Foufou, S., and Hamila, R. (2018). Efficient techniques for energy saving in data center networks. *Computer Communications (COMCOM)*, 129, 111-124.

14. Ramakrishnan, K., and Floyd, S. (1998). A proposal to add explicit congestion notification (ECN) to IP. *AT&T Labs Research.*1-25.

15. Baker, F., and Fairhurst, G. (2015). IETF recommendations regarding active queue management. *RFC – BCP Report.* Zürih, 1-31.

16. Kuhn, N., Natarajan, P., Khademi, N., and Ros, D. (2016). Characterization guidelines for active queue management (AQM). *SRL AS Report*, Oslo, 14-26.

17. Zhu, Y., Eran, H., Firestone, D., Guo, C., Lipshteyn, M., Liron, Y., Padhye, J., Raindel, S., Yahia, M. H., and Zhang, M. (2015). *Congestion control for large-scale rdma deployments.* Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 45(4), 523-536.

18. Gao, C., and Lee, V. C. (2020). *Deme: Decouple packet marking from enqueuing for multiple services in data center networks.* IEEE International Conference on Network Protocols (ICNP), 1-2.

19. Fu, F. Zhou, X., Dreibholz, T., Wang, K., Zhou, F., and Gan, Q. (2015). *Performance comparison of congestion control strategies for multi-path TCP in the NORNET testbed.* IEEE/CIC International Conference on Communications in China (ICCC), 1-6,

20. Zhang, T., Wang, J., Huang, J., Huang, Y., Chen, J., and Pan, Y. (2016). Adaptive marking threshold method for delay-sensitive tcp in data center network. *Journal of Network and Computer Applications (JNCA)*, 61, 222-234.

21. Bagnulo, M., and Briscoe, B. (2017). ECN++: Adding explicit congestion notification (ECN) to TCP control packets. *AI-D Report.* Madrid, 8-20

22. Kuehlewind, M., Scheffenegger, R., and Briscoe, B. (2015). Problem statement and requirements for increased accuracy in explicit congestion notification ECN feedback. *RFC - Informational.* Web: https://rfc-editor.org/rfc/rfc7560.txt. Son Erişim Tarihi: 14.10.2022.

23. Lv, C., Wang, Q., Yan, W., and Shen, Y. (2016). Energy-balanced compressive data gathering in wireless sensor networks. *Journal of Network and Computer Applications (JNCA)*, 61, 102-114.

24. Kushwaha, V., and Gupta, R. (2014). Congestion control for high-speed wired network: A systematic literature review. *Journal of Network and Computer Applications (JNCA)*, 45, 62-78.

25. Ha, S., Rhee, I., and Xu, L. (2008). Cubic: A new TCP-Friendly high-speed TCP variant. *ACM SIGOPS Operating Systems Review*. 42 (5), 64-74

26. Luo, J., Jin, J., and Shan, F. (2017). Standardization of low-latency TCP with explicit congestion notification: A survey. *IEEE Internet Computing*. 21 (1), 48-55.

27. Lee, C., Park, C., Jang, K., Moon, S., and Han, D. (2017). DX: Latency-based congestion control for datacenters. *IEEE/ACM Transactions on Networking (TON)*, 25 (1), 335-348.

28. Wang J., Wen, J., Li, C., Xiong, Z., and Han, Y. (2015). VM-Vegas: a delay-based TCP congestion control algorithm for datacenter applications. *Journal of Network and Computer Applications (JNCA)*, 53, 103-114.

29. Hamann, T., Walrand, J. (2000). *A new fair window algorithm for ECN capable TCP (New-ECN)*. IEEE International Conference on Computer Communications (INFOCOM), 1528-1536.

30. Alizadeh, M., Kabbani, A., Edsall, T., Prabhakar, B., Vahdat, A., and Yasuda, M. (2012, 25 Nisan). *Less is more: trading a little bandwidth for ultra-low latency in the data center*. USENIX Associations Symposium on Networked Systems Design and Implementation (NSDI), 253-266.

31. Yan, F., Lee, T. T., and Hu, W. (2017). Congestion-aware embedding of heterogeneous bandwidth virtual data centers with hose model abstraction. *IEEE/ACM Transactions on Networking (TON)*, 25 (2), 806-819.

32. Munir, A., Baig, G., Irteza, S. M., Qazi, I. A., Alex X Liu, Dogar, F.R. (2014). Not Foes: synthesizing existing transport strategies for data center networks. *ACM SIGCOMM*, 491–502.

33. Munir, A., Qazi, I. A., Uzmi, Z. A., Mushtaq, A., Ismail, S. N., Ikaal, M. S., Khan, B. (2013). *Minimizing flow completion times in data centers.* IEEE INFOCOM, 2157-2165.

34. Wu, H., Ju, J., Lu, G., Guo, C., Xiong, Y., and Zhang, Y. (2012, 10 Aralık). *Tuning ECN for Data Center Networks.* Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies, 25-36.

35. Cardwell, N., Cheng, Y., Gunn, C. S., Yeganeh, S. H., Jacobson, V. (2016). BBR: Congestion-based congestion control. *In ACM Queue.* 14 (5), 20-53.

36. Kleinrock, L. (1979, 10-14 Haziran). *Power and deterministic rules of thumb for probabilistic problems in computer communications.* ICC, Boston, 43-53.

37. Briscoe, B., Woundy, R., and Cooper, A. (2012). Congestion exposure (Conex) concepts and use cases. *RFC – Informational*. Philadelphia, 1-17.

38. Grossman, D. (2002). New terminology and clarifications for diffserv. *MInc Report.* İllinois*,* 50-60.

39. Allman, M., Paxson, V., and Blanton, E. (2020). TCP congestion control. *RFC 5681 Raport*. West Lafayette, 10-28.

40. Jiang, J., Jain, R., and So-In, C. (2008). *An explicit rate control framework for lossless ethernet operation.* IEEE International Conference on Communications (ICC), 5914-5918.

41. Zhang, J., Yu, F. R., Wang, S., Huang, T., Liu, Z., and Liu, Y. (2018). Load balancing in data center networks: A survey. *IEEE Communications Surveys & Tutorials*, 20 (3), 2324-2352.

42. Bergamasco, D. (2005, 12 Mayıs). *Data center ethernet congestion management: Backward congestion notification.* IEEE 802.1 Interim Meeting, 1-25.

43. Alizadeh, M., Kabbani, A., Atikoglu, B., and Prabhakar, B. (2011). *Stability analysis of QCN: the averaging principle.* ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), 49-60.

44. Cheung, C.M., Leung, K.C. (2018). DFFR: A flow-based approach for distributed load balancing in data center networks. *Computer Communications (COMCOM)*, 116, 1-8.

45. Zhang Y., Ansari N. (2011). *On mitigating TCP incast in data center networks.* IEEE International Conference on Computer Communications (INFOCOM), 51-55.

46. Lu, Y., Chen, G., Luo, L., Tan, K., Xiong, Y., Wang, X., and Chen E. (2017). *One more queue is enough: Minimizing flow completion time with explicit priority notification*. IEEE International Conference on Computer Communications (INFOCOM), 1-9.

47. Pan, Y., Tian, C., Zheng, J., Zhang, G., Susanto, H., Bai, and B., Chen, G. (2018). *Support ECN in multi-queue datacenter networks via per-port marking with selective blindness.* IEEE International Conference on Distributed Computing Systems (ICVMS), 33-42.

48. Alizadeh, M., Javanmard, A., and Prabhakar, B. (2011). *Analysis of DCTCP: stability, convergence, and fairness.* ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), 73-84.

49. Shan, D., Ren, F., Cheng, P., Shu, R., and Guo, C. (2018). *Micro-burst in data centers: Observations, analysis, and mitigations.* IEEE International Conference on Distributed Computing Systems (ICVMS), 88-98.

50. Poupart, P., Chen, Z., Jaini P., Fung, F., Susanto, H., Geng, Y., Chen, L., Chen, K., and Jin, H. (2016). *Online flow size prediction for improved network routing.* IEEE International Conference on Network Protocols (ICNP*)*, 1-6.

51. Alizadeh, M., Yang, S., Sharif, M., Katti, S., McKeown, N., Prabhakar, B., and Shenker, S. (2013). *PFABRIC: Minimal near-optimal datacenter transport.* Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 43 (4), 435-446.

52. Majidi, A., Gao,X., Jahanbakhsh, N., Zheng, J., and Chen, G. (2020). *Priority policy in multi-queue data center networks via per-port ECN marking.* IEEE International Conference on Ubiquitous Information Management and Communication (IMCOM), 1-9.

53. Chen, L., Chen, K., Bai, W., and Alizadeh, M. (2016). *Scheduling mix-flows in commodity Datacenters with karuna*. Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 174-187.

54. Cheng, P., Ren, F., Shu, R., and Lin, C. (2014). *Catch the whole lot in an action: Rapid precise packet loss notification in data center.* USENIX Associations Symposium on Networked Systems Design and Implementation (NSDI), 17-28.

55. Majidi, A., Gao, X., Jahanbakhsh, N., Jamali, S., Zheng J., and Chen, G. (2019). Deep-RL: Deep reinforcement learning for marking-aware via per-port in Data Centers. *IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, 1-5.

56. Handley, M., Raiciu, C., Agache, A., Voinescu, A., A. Moore, W., Antichi, G., and Wojcik, M. (2017). *Re-architecting datacenter networks and stacks for low latency and high performance.* Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 29-42.

57. Chen, L., Lingys, J., Chen, K., and Liu, F. (2015). *Auto: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization.* Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 191-205.

58. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. *Journal of Machine Learning Research (JMLR)*, 11 (14), 39-50.

59. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *ArXiv Kornel arXiv: 1312.5602.* 1-9.

60. Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2019). Continuous control with deep reinforcement learning. *ArXiv Kornel arXiv*. Kornel, 60-74.

61. Katta, N. P., Rexford, J., and Walker, D. (2013). *Incremental consistent updates.* Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 49–54.

62. Majidi, A., Jahanbakhsh, N., Gao, X., Zheng, J., and Chen, G. (2020). DC-ECN: A machine-learning based dynamic threshold control scheme for ECN marking in DCN. *Computer Communications (COMCOM)*, 150, 334-345.

63. Farrington, N., Porter G., Radhakrishnan, S., Bazzaz, H. H., Subramanya, V., Fainman, Y., Papen, G., and Vahdat, A. (2011). *Helios: A hybrid electrical/optical switch*

architecture for modular data centers. Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 41(4), 339-350.

64. Chen, L., Chen, K., Bai, W., and Alizadeh, M. (2016). *Scheduling mix-flows in commodity datacenters with karuna.* Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 174-187.

65. Poupart, P., Chen Z., Jaini, P., Fung, F., Susanto, H., Geng, Y., Chen, L., Chen, K., and Jin, H. (2016). *Online flow size prediction for improved network routing*. IEEE International Conference on Network Protocols (ICNP*)*, 1-6, 2016.

66. Lu, Y., Chen, G., Luo, L., Tan, K., Xiong, Y., Wang, X., and Chen, E. (2017). *One more queue is enough: Minimizing flow completion time with explicit priority notification.* IEEE International Conference on Computer Communications (INFOCOM), 1-9.

67. Bai, W., Chen, L., Chen, K., Han, D., Tian, C., and Wang, H. (2017). PIAS: Practical information-agnostic flow scheduling for commodity data centers, *IEEE/ACM Transactions on Networking (ToN)*, 25 (4), 1954–1967.

68. Williams, C. K., and Rasmussen, C. E. (2006). *Gaussian processes for machine learning.* ABD: MIT Press, 2 (3), 41-50.

69. Omar, F. (2016). *Online bayesian learning in probabilistic graphical models using moment matching with applications.* Yayınlanmış doktora tezi, University of Waterloo publications, Kanada, 31-38.

70. Shan, D., Jiang, W., and Ren, F. (2017). Analyzing and enhancing dynamic threshold policy of data center switches. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 28 (9), 2454-2470.

71. Zhu, K., and Ying, L. (2016). Information source detection in the sir model: A sample path-based approach. *IEEE/ACM Transactions on Networking (TON)*, 24(1), 408-421.

72. Neely, M. J. (2010). Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3, 1-211.

73. Chen, L., Lingys, J., Chen, K., Liu, F. (2015). *Auto: scaling deep reinforcement learning for datacenter scale automatic traffic optimization*. SIGCOMM ’18: Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM), 191-205.

74. Amanov, A., Majidi, A., Jahnabakhsh, N., Çetin, A. (2022). Adjusting ECN marking threshold in multi-queue DCNs with deep learning. The Journal of Supercomputing. 5443–5468.

Gazili olmak ayrıcalıktır