Gazi Üniv. Müh. Mim. Fak. Der.
Cilt 24, No 3, 397-407, 2009

J. Fac. Eng. Arch. Gazi Univ.
Vol 24, No 3, 397-407, 2009

WAP TRAFİĞİNDE TIKANIKLIK DENETİMİ VE ULAŞIM KATMANI PROTOKOLLERİ

Sinan TOKLU ve M. Ali AKCAYOL
Bilgisayar Mühendisliği Bölümü, Mühendislik Mimarlık Fakültesi, Gazi Üniversitesi, Maltepe, Ankara.
s.toklu@gazi.edu.tr, akcayol@gazi.edu.tr

(Geliş/Received: 30.01.2008 ; Kabul/Accepted: 02.07.2009)

ÖZET

Bu çalışmada WAP trafiğindeki tıkanıklık problemini çözmek için kuyruk yapıları kullanılarak bir protokol geliştirilmiştir. Geliştirilen protokol, WAP trafiğinde kullanılan kuyruk yapısını yeniden oluşturmakta ve bu sayede önceliği olan TCP paketlerinin atılmasını önleyerek tıkanıklığı önemli ölçüde azaltmaktadır. Geliştirilen protokol ns-2(network simulator-2) benzetim aracı kullanılarak test edilmiştir. Benzetimde farklı yük yoğunluğundaki ağlar için drop-tail kuyruk yapısı, red kuyruk yapısı ve geliştirilen protokolün sonuçları karşılaştırılmıştır. Elde edilen sonuçlar geliştirilen protokolün daha başarılı olduğunu göstermiştir.

Anahtar Kelimeler: WAP, kuyruk yönetimi, tıkanıklık, protokol.

CONGESTION CONTROL IN WAP TRAFFIC AND TRANSPORT LAYER PROTOCOLS

ABSTRACT

In this study, a congestion prevention protocol has been developed for WAP traffic with queue management. Developed protocol prevents congestion by creating new queue management for reducing droped TCP packets which has priority. The protocol has been tested using ns-2 (network Simulator-2) simulation tool. In simulation drop-tail, red queue and developed protocol have been implemented in different networks and results have been compared. Obtained results showed that the developed protocol is succesfull for congestion prevention in wireless networks.

Keywords: WAP, queue management, congestion, protocol.

1. GİRİŞ (INTRODUCTION)

Günümüzde yaygın olarak kullanılan mobil cihazlar ve bu cihazların sunduğu hizmetler hızla artmaktadır. Bunlardan biriside WAP(Wireless Application Protocol-Kablosuz Uygulama Protokolü) protokol kümesidir. WAP mobil cihazların dar bant, küçük pencere, düşük hızlı CPU(Central Processing Unit-Merkezi İşlem Birimi) gibi özelliklerinden dolayı veri iletişiminin daha hızlı ve güvenli yapılabilmesi için kullanılan protokoller kümesidir [1]. Bu kısıtlamalardan dolayı mobil kullanıcıya hizmet verilirken, ağ trafiğinin yoğunluğundaki artış ile mobil cihaz, hizmet sağlayıcı ve kullanıcı arasında tıkanıklık meydana gelebilmektedir.

Bu çalışmada WAP trafiğinde aşırı yoğunlukta meydana gelen tıkanıklığı çözmek için WAP sunucu ve WAP Gateway arasında Ulaşım katmanı protokolleri ve kuyruk yapıları kullanılarak yeni bir protokol geliştirilmiştir.

WAP iletişim trafiğinde aşırı yoğunluk sonucu meydana gelebilecek tıkanıklık problemini çözmek için WAP gateway üzerinde bulunan kuyruk yapısında önceliklendirme yapılmaktadır. Bu protokolde gelen paketler önem derecesine göre sınıflandırılmıştır. Kuyruk doluluk oranı %80'in altındayken gelen paketlerin önem derecesi dikkate alınmamış ve paketlerin direk WAP sunucuya gönderilme işlevi gerçekleştirilmiştir. Kuyruk doluluk oranı %80'e ulaştığında önem derecesi yüksek olan paketin kuyruğa dahil olmasına izin verilmiştir. Önem derecesi düşük

olan paket kuyruğa alınmamıştır. Kuyruk doluluk oranı %80'in altına düştüğü zaman önem dereceleri dikkate alınmadan gelen paketlerin kuyruğa girmesine izin verilmektedir.

Geliştirilen protokol kullanılarak trafik yoğunluğunun yüksek olduğu durumlarda önem derecesi yüksek olan paketlerin atılması önlenmiştir. Deneysel çalışmalardan elde edilen sonuçlar geliştirilen protokolün başarılı olduğunu göstermiştir.

2. WAP KATMANLARI VE TIKANIKLIK DENETİMİ (WAP PROTOCOLS AND CONGESTION CONTROL)

2.1. İlgili Çalışmalar (Related Works)

Meydana gelen tıkanıklığı çözmek ve iletişimin çok daha hızlı ve rahat yapılabilmesi için birçok çalışma yapılmıştır. Kannel projesi Wapit Ltd. tarafından 1999 yılında geliştirilmiştir [2]. WAP'ta tıkanıklığı çözmek için WAP gateway'lerin yapıldığı tarihten itibaren var olan yük dengelemenin açık kod olması bu projenin genel özelliğidir. Kannel Gateway'ler kümeleme mimarisinin temelini oluşturmaktadırlar [3].

WAP'ta tıkanıklığı çözmek için diğer bir donanımsal seçenek olan LVS(Linux Virtual Server-Linux Sanal Sunucu) gerçek sunucuların oluşturduğu kümede kurulumdan itibaren olan ve yüksek derecede ölçeklenebilir özelliklere sahip olan bir sunucudur. Aynı zamanda bu sistemde yük dengeleyicisi Linux işletim sisteminde çalışmaktadır [3]. 2004 yılında Te-Hsin Lin, Kuochen Wang, Ae-Yun Liu SWG (Scalable WAP Gateway-Ölçeklenebilir WAP Gateway) adında bir çalışma yapmışlardır. SWG bir grup gerçek gateway içermekte ve bunlar birbirine hızlı bir ağda bağlı bulunmaktadırlar. Bu kannel-level front-end distributör'ü bütünleştirmekte ve WAP dispatcher olarak adlandırılan yeni bir mekanizma sunmaktadır [3].

Robert Shorten, Chris King, FabianWirth, Douglas Leith 2006 yılında TCP paketlerinde meydana gelen tıkanıklığın kontrolü için drop-tail kuyruk yapısını kullanarak geliştirdikleri yeni yaklaşımı yayınladılar [4]. Kyeong Hur, Doo-Seop Eom, Yeon-Woo Lee, Jae-Ho Lee, Seokjoong Kang 2007 yılında mobil IP ağlarında kullanılan hareketli düğümlerle yönlendirme yaparak TCP performansını geliştirmesi için yeni bir metod önermişlerdir. Bu yaklaşımda kuyruk yapısı olarak Red kuyruk yapısı kullanılmıştır [5].

Jinsheng Sun ve Moshe Zukerman 2007 yılında kuyruk uzunluğu baz alınarak yeni bir aktif kuyruk yönetiminin oluşturulmasını önermişlerdir.Böylece kuyruk denetiminin aktif şekilde gerçekleştirilmesini sağlamışlardır [6]. Shan Chen ve Brahim Bensaou 2006 yılında yüksek hızlı ağlarda drop-tail kuyruk yapısı kullanılabilirliliği hakkında bir çalışma yapmışlardır [7]. Songtao Guo, Xiaofeng Liao, Chuandong Li, Degang Yang 2006 yılında heterojen gecikmeler ile yeni üssel red kuyruk tipinin sağlamlık analizini araştırmışlardır [8].

Mehmet Şimşek tarafından 2006 yılında kablosuz ağlarda yönlendirme protokolleriyle tıkanıklığın önlenmesi için bir çalışma yapılmıştır. Yapılan çalışmada, kablosuz ağlarda aşırı talep durumunda meydana gelen tıkanıklığın yük dengeleme ve yönlendirme işlemleri yapılarak önlenmesi için yeni bir yaklaşım sunulmuştur [9].

2.2. WAP Katmanları (WAP Layers)

WAP esnek bir yapıya sahiptir ve temel olarak beş bölüme ayrılmıştır. Bunlar;

- WAE (Wireless Application Environment-Kablosuz Uygulama Katmanı): WML(Wireless Markup Language-Kablosuz İşaret Dili) ve WML Script bu katmanda kullanılmaktadır. HTML(Hyper Text Markup Language-Hareketli Metin İşaretleme Dili) ile aynı işlevi görmektedir. OSI(Open System Interconnection)'de sunum katmanına denk gelmektedir [10].

- WSP (Wireless Session Protocol-Kablosuz Oturum Katmanı): Güvenli oturum açma, yönlendirme yeteneği, ayrıca bağlantı kopsa da askıya alınsa da tekrar kaldığı yerden devam etme yeteneği ile itme özellikleri bu katmanda kullanılmaktadır. HTTP(Hyper Text Transfer Protocol-Hareketli Metin Transfer Protokolü) ile aynı işlevi görmektedir. OSI'de oturum katmanına denk gelmektedir [11].

- WTP (Wireless Transport Protocol-Kablosuz Ulaşım Katmanı): İşlemlerin tutulması ve yönetilmesi, yeniden iletim, birden fazla aynı ACK(Acknowledgement) paketin silinmesi, birleştirme ve bölme özellikleri bu katmanda kullanılmaktadır. HTTP ile aynı işlevi görmektedir. OSI'de ulaşım katmanına denk gelmektedir [1].

- WTLS (Wireless Transport Layer Security-Kablosuz Ulaşım Katmanı Güvenliği): Sıkıştırma, şifreleme ve kimlik doğrulama özellikleri bu katmanda kullanılmaktadır. TLS (Transport Layer Security-Ulaşım Katmanı Güvenliği) / SSL (Secure Socket Layer-Güvenli Soket Katmanı) ile aynı işlevi görmektedir. OSI'de ağ katmanına denk gelmektedir [12].

- WDP (Wireless Datagram Protocol-Kablosuz Datagram Protokolü): Port numarasına göre adresleme, parçalarına ayırma ve parçaları tekrar birleştirme, hataları yakalama gibi özellikler bu katmanda kullanılmaktadır. TCP/IP(Transmission Control Protocol/Internet Protocol) ile aynı işlevi

görmektedir. OSI'de veri iletim katmanına denk gelmektedir [11].

WAP protokollerini kullanan mobil cihazların kısıtlamalarından dolayı HTML bazlı sayfalar görüntülenememektedir. Bu problem WAP gateway'lerin gelen HTML sayfalarını WML'e çevirmelerinden dolayı mobil cihazlar ile sunucu veya normal HTML bazlı sayfalarının iletişimi olanaklı hale gelmiştir. WML ikili dil olarak adlandırılmakta ve mobil cihazların anlayacağı ve görüntüleyebileceği bir dildir.

Paket kayıpları veya gecikmeler gibi olaylardan meydana gelebilecek tıkanıklık mobil cihazlarda daha fazla görülmektedir. Hem maliyet hem de WAP sayfasının ekrana yavaş ve küçük bölümler halinde gelmesi kullanıcıları sıkmakta ve birden fazla istekte bulunulduğunda tıkanıklık meydana gelebilmektedir.

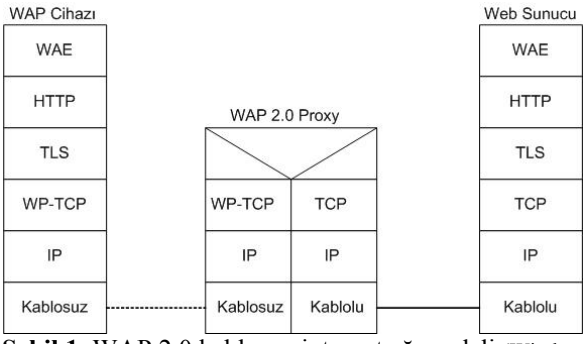
Paketlerin hepsi bir kuyruk yapısına dahil olmaktadır. Genelde kullanılan kuyruk yapısı drop-tail kuyruk yapısıdır. Drop-tail kuyruk yapısı ilk giren ilk çıkar mantığıyla işlem yapmaktadır. Kullanılan bir diğer kuyruk yapısı ise red'tir. Red kuyruk yapısında gelen paketlerden rasgele seçim yapılır ve buna göre bu seçilen paket kuyruğa dahil edilmez veya atılır [13].

Günümüzde, TCP en yaygın iletişim protokol kümesidir. Bu çalışmada TCP ve diğer bağlantılar kullanılarak, benzetim ortamında drop-tail ve red kuyruk tipleri ile geliştirilen protokolden elde edilen sonuçlar karşılaştırılmıştır.

3. GELİŞTİRİLEN PROTOKOLÜN MATEMATİKSEL MODELİ (MATHEMATICAL MODEL OF THE DEVELOPED PROTOCOL)

Şekil 1'de WAP 2.0 kablosuz internet ağ modeli görülmektedir.

WAP 2.0 aktarma gecikmesini analiz etmek için, WAP 2.0 mimarisi üç parçada incelenebilir. Bunlar uygulama, ulaşım ve ağ katmanlarıdır. Bu çalışmada ulaşım katmanında TCP ve UDP paketleri arasında öncelik tanımlanarak TCP paketlerinin atılma oranı azaltılmıştır. Herhangi bir paketin kaybolma durumu en kötü ve en iyi olarak iki durumla belirlenmiştir. En kötü durumda atılma olasılığı 1 en iyi durumda ise 0 olarak alınmıştır. Geçiş olasılıkları denklem 1'de verilen matrisle ifade edilebilir;



Şekil 1. WAP 2.0 kablosuz internet ağ modeli (Wireless Internet network model of WAP 2.0)

$$\mathbf{P} = \begin{pmatrix} p_{\mathrm{kk}} & p_{\mathrm{ki}} \\ p_{\mathrm{ik}} & p_{\mathrm{ii}} \end{pmatrix} \tag{1}$$

burada p_{xy} bir durumdan diğer duruma geçiş olasılığını gösterir ve $x \in \{\mathrm{i}, \mathrm{k}\}$ ve $\mathrm{y} \in \{\mathrm{i}, \mathrm{k}\}$ olur. Burada i iyi durumu k ise kötü durumu gösterir. Belirli bir e hata oranı için kötü durumdan iyi duruma geçiş olasılığı burst data uzunluğuyla (BDU) ters orantılıdır ve denklem 2'deki gibi ifade edilir.

$$p_{\mathrm{ki}} = \frac{1}{BDU} \tag{2}$$

İyi durumdan kötü duruma geçiş olasılığı ise denklem 3'teki gibi gösterilmektedir [14].

$$p_{\mathrm{ik}} = \frac{1}{BDU}\left(\frac{e}{1-e}\right) \tag{3}$$

Bu durumda $p_{\mathrm{kk}} = 1 - p_{\mathrm{ki}}$ ve $p_{\mathrm{ii}} = 1 - p_{\mathrm{ik}}$ şeklinde ifade edilir.

3.1. Performans Analizi (Performance Analysis)

Geliştirilen modelde bağlantı kurulumu ve veri aktarımı gerçekleştirilmiştir. Bağlantı kesilmesi aşaması bağlantı kurulumuyla benzer şekildedir. Ortalama veri aktarım süresi $T = T_{\mathrm{bs}} + T_{\mathrm{va}}$ şeklinde ifade edilebilir. Burada T ortalama aktarım süresi, T_{bs} bağlantı kurulum süresi ve T_{va} ise veri aktarım süresidir.

Bağlantı kurulumu

Bağlantı kurulumu "üç yollu anlaşma" şeklinde yapılmaktadır [15]. Hem uplink hemde downlink durumunda paket kayıpları olabilmektedir. Bağlantı kurulumu için sınırsız tekrar deneme yapıldığı ve ortalama hatanın 0.5 olduğu durumda ortalama bağlantı kurulumu süresi denklem 4'teki gibi ifade edilir [16].

$$T_{\mathrm{cs}} = 2l + T_s\left(\frac{2e}{1-2e}\right) \tag{4}$$

Burada l hat gecikmesidir ve bir paketin mobil birimle baz istasyonu arasındaki yolu alması için geçen süredir. Başlangıçta kullanılan SYN paketleri ve ACK paketleri için geçen süre ihmal edildiğinde l kanal gecikmesidir ve $2l$ ise round-trip süresidir.

Veri aktarım modeli

Veri aktarımı aşaması, bağlantının kurulmasından son paket için ACK gelmesine kadar devam eder. Veri aktarım modeli Şekil 2'deki gibidir [17].

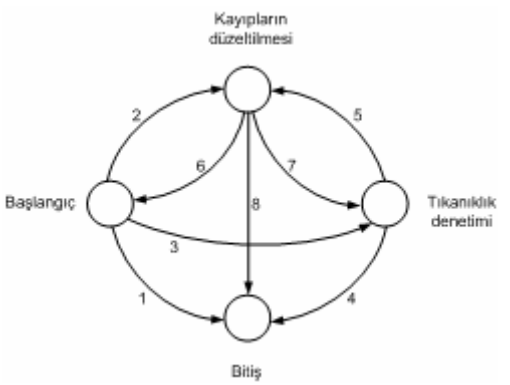
Gönderen başlangıç aşamasıyla başlar ve 1 geçişiyle veri aktarımı sonlanır. Kayıpların oluştuğu kötü duruma geçiş 2 ile gerçekleşir ve tıkanıklık penceresi dolduğunda 3 geçişiyle tıkanıklık denetimi durumuna geçilir. Tıkanıklık denetiminde 4 ile veri aktarım durumu sonlandırılır. Eğer kötü duruma geçiş yapılacaksa 5 ile yapılır. Kayıpların düzeltilmesi durumunda ACK paketleri veya timeout süresine göre hareket edilir. Eğer timeout süresi kullanılırsa gönderen 6 ile başla durumuna geçer ve retransmission yapılır. Negatif ACK geldiğinde tıkanıklık denetimi durumuna geçilir.

Veri aktarım süreci

$S(t) = (c(t-1), \omega(t), \omega_{\text{th}}(t), k(t))$ şeklinde ifade edilir. Burada $c(t) \in \{\text{i}, \text{k}\}$ kanal durumu, $\omega(t) \in \{1, 2, ..., W_{max}\}$ tıkanıklık penceresi, $\omega_{\text{th}}(t) \in \{2, 3, ..., \lceil W_{max}/2 \rceil, W_{max}\}$ olmak üzere pencerenin ikinci yarısını gösterir, $k(t) \in \{0, 1, ..., N\}$ ise başarıyla gönderilen paket sayısını ifade eder. W_{max} maksimum pencere boyutunu N ise toplam paket sayısını göstermektedir. N dosya boyutu paket olarak $N = \lceil F/\rho \rceil$ şeklinde gösterilir. Burada F dosya boyutunu, ρ paket boyutunu gösterir. Tüm durumların durum uzay gösterimi [18];

$$W_s = \{(c,\ \omega,\ \omega_{\text{th}},\ k) : c \in \{\text{i}, \text{k}\},\ \omega \in \{1, 2, ...,\ W_{\max}\},\ \omega_{\text{th}} \in \{2, 3, ...,\ \lceil W_{\max}/2 \rceil,\ W_{\max}\},\ k \in \{0, 1, ..., N\}\}$$

şeklindedir. Geçiş olasılık matrisi $\Phi = [\phi_{AB}]$ olarak ifade edilir. ϕ_{AB}, $A \in W_s$ durumundan $B \in W_s$ durumuna geçiş olasılığını gösterir. $\mathbf{D} = [\text{d}_{\text{AB}}]$ ise



Şekil 2. Veri aktarım modeli (Model of the data transfer)

$A \in W_s$ durumundan $B \in W_s$ durumuna geçiş süresini gösterir. Tüm kendi kendisine geçişler kaldırıldığında $(X,\ X,\ X,\ N)$ şekilde süper durum gösterimi elde edilir [18]. Beklenen gecikme $\mathbf{T} = [T_A]$ şeklinde ifade edilir. Burada $A \in W_s$ olarak başlangıç durumudur ve denklem 5'te gösterildiği gibi hesaplanır.

$$\mathbf{T} = \left[\mathbf{I} - \mathbf{\Phi}''\right]^{-1} \overline{\mathbf{D}} \tag{5}$$

Burada $\mathbf{I}$ birim matrisi, $\mathbf{\Phi}''$ süper durum matrisi ($\mathbf{\Phi}$ içinden satır ve sütunların silinmesiyle elde edilir) ve $\overline{\mathbf{D}}$ ise denklem 6'da gösterildiği gibi hesaplanır.

$$\overline{\mathbf{D}} = diag(\mathbf{\Phi}'[\mathbf{D}']^t) \tag{6}$$

$\mathbf{\Phi}'$ ve $\mathbf{D}'$ ise $\mathbf{\Phi}$ ve $\mathbf{D}$ matrislerinde 1 satır silinerek elde edilir [18]. Veri aktarım süresi ise denklem 7'de gösterildiği gibi hesaplanır.

$$T_{\text{dt}} = \pi_{\text{i}} T_{(\text{i},\ W_{\text{iw}},\ W_{\max},\ 0)} + \pi_{\text{k}} T_{(\text{k},\ W_{\text{iw}},\ W_{\max},\ 0)} \tag{7}$$

π_{i} ve π_{k} iyi ve kötü durumlar için kararlı durum olasılıklarıdır ve geçiş olasılık matrisi $\mathbf{P}$ ile hesaplanır [19].

Başlangıçta veri aktarım süresi başla durumuyla veya kayıp paket düzeltmesinden sonra başlar. Başlangıç durumları;

$$A = \{(\text{i},\ W_{\text{iw}},\ W_{\max},\ 0) \cup [(\text{i},\ 1,\ \omega_{\text{th}},\ k):\ \omega_{th} \in \{2,3,...,\lceil W_{max}/2 \rceil\}, k \in \{1, 2, ..., N-1\}]\}$$

olarak ifade edilir. Eğer $s \in \{1, 2, ..., N-k\}$ adet paketin başarıyla gönderildiği düşünülürse tıkanıklık penceresi ve gecikme modellemesi Kurose ve Ross tarafından yapılmıştır [15].

Başlangıç durumu $(i,\ \omega,\ \omega_{\text{th}},\ k)$ için olası sonuç durumları denklem 8'de gösterildiği gibi hesaplanabilir;

$$B = \begin{cases} (X,\ X,\ X,\ N), & s \le 2\omega_{\text{th}} - \omega,\ s = N - k \\ (\text{k},\ W_B^{\text{SS}}(s),\ \omega_{\text{th}},\ k+s), & s < 2\omega_{\text{th}} - \omega,\ s < N - k \\ (\text{i},\ W_B^{\text{SS}}(s),\ \omega_{\text{th}},\ k+s), & s = 2\omega_{\text{th}} - \omega,\ s < N - k \end{cases} \tag{8}$$

burada $W_B^{\text{SS}}(s) = \lfloor (\omega + s)/2 \rfloor$, s paket gönderdikten sonraki tıkanıklık penceresidir.

Birinci, ikinci ve üçüncü durumlar Şekil 2'deki 1, 2 ve 3 nolu geçişlerle gösterilmektedir. Geçiş süresi denklem 9'daki gibi gösterilir.

$$d_{AB} = \begin{cases} s, & ,\mathrm{Q} < 1 \\ s + (2l+1)Q - (2^{Q} - 1)\omega & ,\text{diger} \end{cases} \quad (9)$$

Burada,

$Q = \lfloor \min\{1 + \log_2(2l/\omega + 1/\omega), \log_2(1 + s/\omega - 1/\omega)\} \rfloor$ olarak ifade edilir. Burada l kanal gecikmesidir. Geçiş olasılığı ise denklem 10'daki gibi ifade edilir.

$$\phi_{AB} = \begin{cases} p_{\mathrm{ii}}^{s-1}, & s = \min\{N-k,\ 2\omega_{\mathrm{th}} - \omega\} \\ p_{\mathrm{ii}}^{s-1} p_{\mathrm{ik}}, & s < \min\{N-k,\ 2\omega_{\mathrm{th}} - \omega\} \\ 0, & \text{diğer} \end{cases} \quad (10)$$

Tıkanıklıktan kaçınmak için paket kayıplarının veya ACK bilgilerinin gözönüne alınması gerekir. Tıkanıklık penceresinin ikinci yarısında belirlenen limiti geçtiğinde bu duruma geçiş oluşur. $A = \{(\mathrm{i}, \omega_{\mathrm{th}}, \omega_{\mathrm{th}}, k) : \omega_{\mathrm{th}} \in \{2, 3, \ldots, \lceil W_{max}/2 \rceil, W_{max}\},\ k \in \{1, 2, \ldots, N-1\}\}$ başlangıç durumları kümesi olsun. $s \in \{1, 2, \ldots, \mathrm{N} - k\}$ geçişten önce başarıyla gönderilen paketler olsun. Başlangıç durumu $(\mathrm{i},\ \omega_{\mathrm{th}},\ \omega_{\mathrm{th}},\ k)$ için olası sonuç durumları denklem 11'de gösterildiği gibi elde edilebilir [15].

$$B = \begin{cases} (X,\ X,\ X,\ N), & s = N - k \\ (\mathrm{b},\ W_A^{\mathrm{CA}}(s),\ \omega_{\mathrm{th}},\ k+s), & s < N - k \end{cases} \quad (11)$$

Burada

$W_B^{\mathrm{CA}}(s) = \min\{-1/2 + \sqrt{(\omega_{\mathrm{th}} - 1/2)^2 + 2s},\ W_{\max}\}$

şeklinde hesaplanır. Geçiş süresi;

$d_{AB} = s + (2l + 2 - \omega_{\mathrm{th}})Q - (Q+1)Q/2$ olur. Burada, $Q = \min\{2l - \omega_{\mathrm{th}} + 2,\ 1/2 - \omega_{\mathrm{th}} + \sqrt{(\omega_{\mathrm{th}} - 1/2)^2 + 2s}\}$ olur. Geçiş olasılığı ise denklem 12'deki gibidir.

$$\phi_{AB} = \begin{cases} p_{\mathrm{ii}}^{\mathrm{s-1}}, & s = N - k \\ p_{\mathrm{ii}}^{\mathrm{s-1}} p_{\mathrm{ik}}, & s < N - k \\ 0, & \text{diğer} \end{cases} \quad (12)$$

Kayıpların tekrar alınması durumuna ilk kayıp paketle geçilir. Başlangıç durumları kümesi;

$$A = \{(\mathrm{i}, \omega, \omega_{\mathrm{th}}, k) : \omega \in \{1, 2, \ldots, W_{\max}\},\ \omega_{\mathrm{th}} \in \{2, 3, \ldots, \lceil W_{\max}/2 \rceil, W_{\max}\},\ k \in \{0, 1, \ldots, N-1\}\}$$

olarak tanımlansın. $\alpha_c(s, \omega),\ s \in \{1, 2, \ldots, \omega - 1\}$ olmak üzere ω tane paketin başarıyla gönderilme olasılığı olsun. $c \in \{\mathrm{i}, \mathrm{k}\}$ olduğu için α_k ve α_i denklem 13 ve 14'te gösterildiği gibi ifade edilir.

$$\alpha_k(s,\omega) = \begin{cases} p_{\mathrm{kk}}^{\omega}, & \mathrm{s} = 0 \\ \sum_{j=1}^{\min\{s, \omega - s\}} \binom{\omega - s}{j} \binom{s-1}{j-1} p_{\mathrm{kk}}^{\omega - s - j} & \\ \quad \cdot p_{\mathrm{ki}}^{j} p_{\mathrm{ik}}^{j} p_{\mathrm{ii}}^{s-j}, & \mathrm{s} = 1, \ldots, \mathrm{w}\ \text{-}1 \\ 0, & \mathrm{s} \geq \omega \end{cases} \quad (13)$$

$$\alpha_i(s,\omega) = \begin{cases} \sum_{j=1}^{\min\{s+1, \omega - s\}} \binom{\omega - s - 1}{j - 1} \binom{s}{j-1} p_{\mathrm{kk}}^{\omega - s - j} & \\ \quad \cdot p_{\mathrm{ki}}^{j} p_{\mathrm{ik}}^{j-1} p_{\mathrm{ii}}^{s-j+1}, & \mathrm{s} = 0, \ldots, \omega - 1 \\ 0, & \mathrm{s} \geq \omega \end{cases} \quad (14)$$

Kayıpların timeout ile tekrar gönderilmesinde ACK alınmadan geçen süre esas alınır. $(k, \omega, \omega_{\mathrm{th}}, n)$ başlangıç durumu için olası sonuç durumları $B = (c,\ 1,\ \lceil \omega/2 \rceil,\ n+s)$ olur [20]. Burada $c \in \{\mathrm{i}, \mathrm{k}\}$ olur. Geçiş süresi;

$$d_{AB} = \begin{cases} T_0, & s < Z \\ T_0, & Z \leq s,\ s < \omega/2 \end{cases}$$

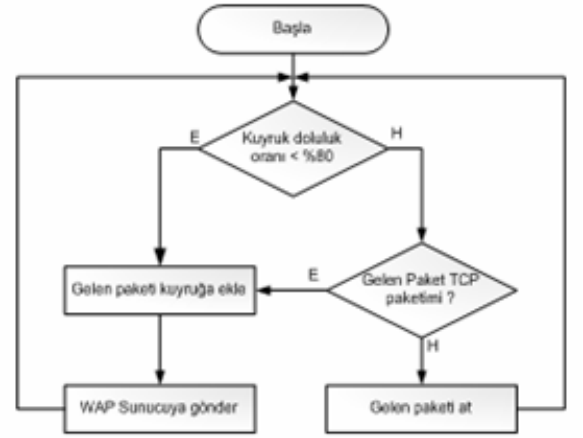
olur. Burada $T_0 = 2(2l+1)$, l kanal gecikmesi, $2l$ round-trip süresidir. Z, ACK sayısı eşik değeri, s başarıyla gönderilen paket sayısı, ω gönderilen paket sayısıdır. Geçiş olasılığı ise denklem 15'teki gibi ifade edilir.

$$\phi_{AB} = \begin{cases} [\alpha_{\mathrm{i}}(s,\omega) + \alpha_{\mathrm{k}}(s,\omega)]\, p_{\mathrm{kc}}(T_0 - 1), & s < Z \\ [\alpha_{\mathrm{i}}(s,\omega) + \alpha_{\mathrm{k}}(s,\omega)]\, p_{\mathrm{kk}}(\omega - s + Z + 2l) & \\ \quad \cdot p_{\mathrm{kc}}(T_0 - 1), & Z \leq \mathrm{s},\ \mathrm{s} < \frac{\omega}{2} \\ 0, & \text{diğ} \end{cases} \quad (15)$$

4. GELİŞTİRİLEN PROTOKOL VE BENZETİM SENARYOLARI (DEVELOPED PROTOCOL AND SIMULATION SCENARIOS)

Geliştirilen protokolde ulaşım katmanı protokolleri ve kuyruk yapıları çok önemli rol oynamıştır. Herhangi bir düğümden WAP sunucuya erişimin yapılabilmesi için gelen isteklerin ilk önce WAP gateway'den geçmesi öngörülmektedir. WAP gateway üzerinde gelen isteklerin öncelikleri değerlendirilip bu işlem bittikten sonra WAP sunucuya gönderilmesi amaçlanmıştır. Bu sayede WAP sunucuda tıkanma problemi önemli ölçüde azaltılmıştır. Birçok düğümün aynı anda WAP sunucu ile iletişime geçmesi durumunda öncelik güvenli paketlerin iletişimini sağlayan TCP protokolüne verilmiştir. Benzetimde tıkanıklığa kuyruğun doluluk oranına göre karar verilmiştir. Kuyruk doluluk oranı %100 seviyesine geldiğinde paket kayıpları ve gecikmeler artmaktadır. Bu da tıkanıklığı oluşturmaktadır.

Geliştirilen protokolde önemli verileri transfer eden ve güvenli olmasını sağlayan TCP paketlerinin geçişine öncelik verilmiştir. Gelen diğer paketler ise atılmıştır. Burada kuyruk doluluk oranını kuyruk uzunluğunun %80'i olarak belirlenmiştir. Dolayısıyla WAP gateway üzerinde bulunan kuyruk doluluk oranı %80'e ulaşınca önceliklendirme işlemi devreye girmektedir. Bunun sebebi ise TCP paketlerinin %20'lik kısma girebilmesini sağlamak ve bu sayede atılan TCP paket sayısını azaltmaktır. Şekil 3'te geliştirilen protokolün akış şeması gösterilmektedir.

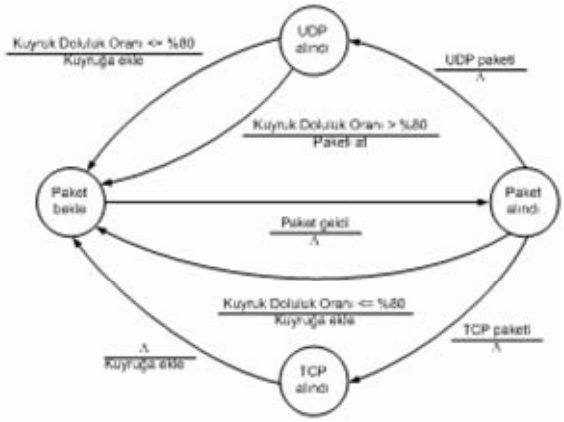


Şekil 3. Geliştirilen protokolün akış şeması (Flow chart of the developed protocol)

Şekil 3'de gösterildiği gibi herhangi bir düğümden gelen paketler ilk önce WAP gateway görevi üstlenen düğüme gelmektedir. Bu düğümde var olan kuyruk yapısına paketlerin girmesi önceliklendirmeye göre yapılmaktadır. Buradaki kuyruk yapısının kuyruk doluluk oranı %80'den küçükse önceliklendirme devre dışı kalmakta ve gelen her paket türü ne olursa olsun kuyruğa dahil edilmektedir. Burada ilk giren ilk çıkar mantığı uygulanmaktadır. Kuyruğun doluluk oranı %80'den az olduğu için tüm paketlerin geçişine ve WAP sunucu görevi üstlenen düğüme gitmesine izin verilmektedir. Kuyruk doluluk oranı %80'e eşit veya fazla olduğu durumlarda önceliklendirme devreye girmekte ve gelen paketlerin türlerine bakılmaktadır. İletişimin önemli bir parçasını gerçekleştiren TCP paketleri önceliklendirme kuralı gereği direkt kuyruğa alınmaktadır. Gelen diğer paketler TCP paketlerinin atılmaması ve iletişimin devam etmesi için atılmaktadır. Kuyruk doluluk oranının %20 seviyesi aynı anda gelen birçok istek durumunda öncelikli olan paketin atılmasını kuyruk doluluk oranında bırakılan bu kısım sayesinde önlemektedir.

Durum Geçiş Diyagramı: Şekil 4'te gösterilen, geliştirilen protokolün durum geçiş diyagramı otomatlar kullanılarak oluşturulmuştur.

Şekil 4'te gösterildiği gibi geliştirilen protokolün durum geçiş diyagramında WAP gateway görevi gören düğüm 2'nin çalışma prensibi otomatlar kullanılarak oluşturulmuştur. Burada ilk önce gelen paketin olup olmadığına bakılmaktadır. Gelen bir paket olduğunda

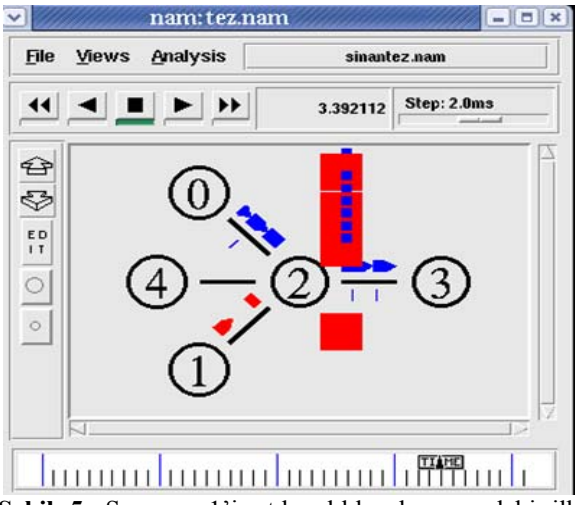


Şekil 4. Geliştirilen protokolün durum geçiş diyagramı (State diagram of the developed protocol)

hemen durum değişmekte ve kuyruk durumuna geçilmektedir. Gelen paketin tipine göre UDP veya TCP olduğuna karar verilmektedir. Paket tipi UDP ise bir sonraki duruma yani UDP durumuna geçilmekte ve kuyruk doluluk oranı incelenmektedir. Kuyruk doluluk oranı kuyruk uzunluğunun %80'inden az ise gelen paket kuyruğa alınmakta ve tekrar paket bekleme durumuna dönülmektedir. Eğer ki kuyruk doluluk oranı kuyruk uzunluğunun %80'inden büyük ise gelen UDP paketi atılmakta ve tekrar paket bekleme durumuna dönülmektedir. Kuyruk durumunda paketin tipi TCP olarak belirlenirse hemen TCP durumuna geçilmekte ve kuyruk doluluk oranı kuyruk uzunluğundan büyük veya küçük ise gelen paketi kuyruğa dahil ederek hemen paket bekleme durumuna dönmektedir. Şekilde gösterilen Λ işareti ise herhangi bir işlem gerçekleşmeden diğer duruma geçmesi işlevini gerçekleştirmektedir.

Bu çalışmada iki senaryo hazırlanmış ve bu senaryoların benzetim işlemi sonucu elde edilen sonuçlar karşılaştırılmıştır.

Senaryo 1: Şekil 5'te senaryo 1 için önerilen tı-



Şekil 5. Senaryo 1'in tıkanıklık oluşumundaki ilk benzetim çıktısı (Output of simulation at creation of congestion in scenerio 1)

kanıklık sisteminin nasıl çalıştığı gösterilmiştir. Burada 0 ve 4 numaralı düğümler TCP bağlantısıyla WAP sunucu görevi üstlenen düğüm 3 ile iletişim kurmuşlardır. 1 numaralı düğüm ise 3 numaralı düğüm ile UDP bağlantısı kurmuştur. 2 numaralı düğüm ise WAP gateway görevini üstlenmiştir. Paketlerin durumuna göre önceliklendirme yapan 2 numaralı düğümde bulunan kuyruk yapısında drop-tail, red ve geliştirilen kuyruk tipi uygulanmış ve sonuçlar karşılaştırılmıştır. Benzetim ns-2 ile yapılmıştır. Kuyruk tiplerinin ve ağların ns-2 ile tanımlanması test sürecinin daha hızlı olmasını sağlamıştır.

Birinci senaryoda 5 düğüm kullanılmıştır. Bunlardan iki tanesi TCP bağlantısı yaparken, bir tanesi UDP bağlantısı yapmış, bir tanesi WAP gateway görevini üstlenmiş ve bir tanesi de WAP sunucu görevi üstlenmiştir. İlk senaryodaki düğümleri oluşturan OTCL kodu aşağıdaki gibidir:

```
set ns [new simulator]
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
$ns duplex-link $no $n2 2Mb 2ms DropTail
$ns duplex-link $n1 $n2 2Mb 2ms DropTail
$ns duplex-link $n2 $n3 2Mb 2ms DropTail
```

Burada 5 tane düğüm oluşturulmakta ve hepsi birbirine duplex-link'ler ile bağlanmaktadır. Her bir bağlantının bant genişliği ve kullanacağı kuyruk tipide belirlenmektedir. Bant genişliği birinci senaryoda 2Mbps ve benzetim süresi ise 2 sn olarak belirlenmiştir. İkinci senaryoda bant genişliği 10Mbps ve benzetim süresi ise 50 sn olarak belirlenmiştir.

```
#UDP bağlantısının oluşturulması

set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2

#UDP bağlantısı üzerinde CBR'nin
oluşturulması

set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mbu
```

Yukarıdaki kod ile bazı düğümlerin WAP sunucu görevi üstlenen 3 numaralı düğüm ile UDP bağlantısı kurması için gerekli olan OTCL kodu verilmiştir. Aşağıda ise TCP bağlantısı için gerekli olan OTCL kodu verilmiştir.

```
#TCP bağlantısının oluşturulması

set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#TCP bağlantısı üzerinde FTP'nin
oluşturulması

set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
```

Aşağıda yazılan C++ kod parçacığı ile 2 numaralı düğüme gelen paketler için ilk önce IP paket kütüphanesinden her bağlantı için kurulan ayrı akış numaraları alınmaktadır. Daha sonra `enque` fonksiyonu `if(q_->length()<8)` koşulunun içine yazılmaktadır. Eğer ki kuyruk doluluk oranı %80'den fazla ise diğer if koşulu çalışmaktadır. `q_->length()` ile gelen paketlerin dahil edildiği kuyruğun uzunluğu elde edilmektedir. Ayrıca, aşağıdaki kod parçacığı 2 numaralı düğüme gelen paketlerin toplam sayılarının bulunması işlemini gerçekleştirmektedir.

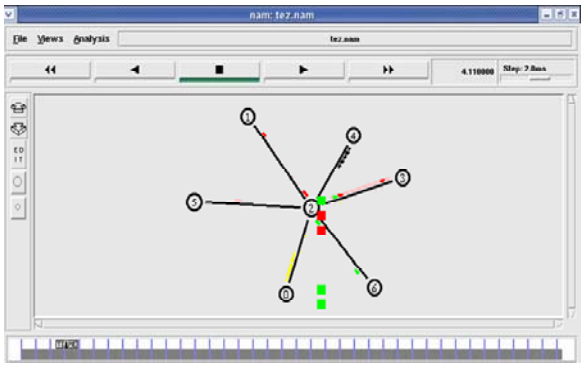
```
hdr_ip *rh=hdr_ip::access(p);
int flowid=rh->flowid();

if(q_->length()<q_->length*80/100)

if(flowid!=1)
 {tcp=tcp+1;
 printf("tcp: %i\n",tcp);
 }
else
 {udp=udp+1;
 printf("udp: %i\n",udp);
 }
drop(p);
```

Senaryo 2: Bu senaryoda ilk senaryoya ek olarak 2 tane yeni düğüm (Düğümlerden biri TCP bağlantısı kurmakta diğeri ise UDP bağlantısı kurmaktadır) daha eklenmiş ve bağlantıların bant genişliği 2 Mbps'tan 10 Mbps'a çıkarılmıştır. Ayrıca bütün TCP ve UDP paketlerinin gönderilme işlemi aynı anda olacak şekilde ayarlanmıştır. Zaman aralıkları ise 4 s'den 50 s'ye çıkarılmıştır. Şekil 6'da oluşturulan yeni senaryonun benzetim çıktısı görülmektedir.

Şekil 6'da gösterildiği gibi kuyruk doluluk oranının eşik değeri %80'i aştığı durumlarda düğüm 1 ve düğüm 6'dan gelen UDP paketleri atılmaya başlanmaktadır. Burada TCP paketleri kuyruk doluluk oranı %100'e ulaştığında atılmaya başlanmaktadır. Birden fazla TCP bağlantısı olduğu için ve hepsi aynı anda iletişime geçtiği için TCP paketleri de atılabilmektedir. Ancak kuyruk doluluk oranının %80'i aştığı durumlarda atılmaya başlanan UDP paketleri sayesinde TCP paketlerinin atılma oranı düşürülmüştür.



Şekil 6. Senaryo 2'nin tıkanıklık oluşumundaki ilk benzetim çıktısı (Output of simulation at creation of congestion in scenario 2)

İkinci senaryodaki düğümleri oluşturan OTCL kodu aşağıdaki gibidir:

```
set ns [new simulator]
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
$ns duplex-link $no $n2 2Mb 2ms DropTail
$ns duplex-link $n1 $n2 2Mb 2ms DropTail
$ns duplex-link $n4 $n2 2Mb 2ms DropTail
$ns duplex-link $n5 $n2 2Mb 2ms DropTail
$ns duplex-link $n6 $n2 2Mb 2ms DropTail
```

Düğümlerin bağlantı kodları birinci senaryoyla aynıdır.

Aşağıda yazılan C++ kod parçacığı 2 numaralı düğüme gelen paketlerin toplam sayılarının bulunması işlemini gerçekleştirmektedir.

```
hdr_ip *rh=hdr_ip::access(p);
int flowid=rh->flowid();

if(q_->length()<q_->length*80/100)

if(flowid!=4 && flowid!=6)
 {tcp=tcp+1;
 printf("tcp: %i\n",tcp);
 }
else
 {udp=udp+1;
 printf("udp: %i\n",udp);
 }
drop(p);
```

Benzetimde ulaştırma katmanında TCP ve UDP kullanılmıştır. Belirli zaman aralıklarında TCP ve UDP paketlerinin gönderiminin aynı anda birçok düğüm ile yapılması ve bant genişliğinin düşük tutulması nedeniyle ağda tıkanıklık meydana gelmiştir. Elde edilen sonuçlarda öncelikli gönderim hakkına sahip olan TCP paketlerinde geliştirilen protokol ile çok başarılı sonuçlar elde edilmiştir.

5. DENEYSEL SONUÇLAR (EXPERIMENTAL RESULTS)

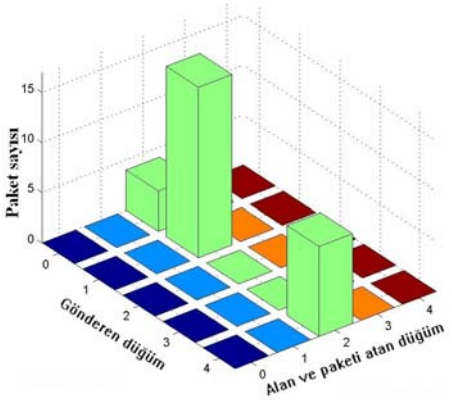
Geliştirilen protokol ile, drop-tail ve red kuyruk tiplerinin uygulandığı ağda, bant genişliği arttırılarak değişik şekillerde yapılan benzetimlerde, karşılaştırma yapılarak sonuçlar elde edilmiştir. Burada kuyruk tipleri ve geliştirilen protokol ile öncelikli olan bağlantıların değişik bant genişliğinde paket kayıp oranları elde edilmiştir. Paket kayıplarının incelenmesinin nedeni bu kayıpların büyük bölümünün tıkanıklık sonucu meydana gelmesindendir.

Senaryo1: Şekil 7'de sırasıyla drop-tail, red kuyruk tipleri ile geliştirilen protokolün ilk senaryo için benzetim işlemi sırasında elde edilen paket kayıplarının sayıları gösterilmektedir.

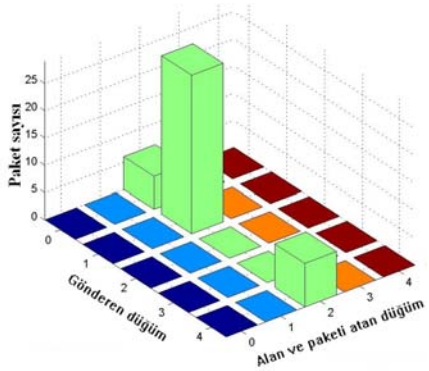
Şekil 7'de gösterildiği gibi geliştirilen protokolde kuyruk doluluk oranının %80'i aştığı durumlarda devreye giren önceliklendirme sayesinde WAP gateway görevi gören düğümde atılan paket sayısının diğer benzetim sonuçlarına göre çok daha iyi olduğu görülmektedir. Üç protokolün ilk senaryoya göre yapılan benzetim işlemi ve bu benzetim işleminin 10 kere tekrar edilmesi ile elde edilen toplam atılan paket sayılarının ortalaması Şekil 8'de verilmiştir.

Şekil 8(b)'de gösterildiği gibi drop-tail kuyruk tipinin uygulandığı ilk senaryonun 10 kere tekrarlanan benzetiminde, düğüm 2'ye gelen ortalama toplam paket sayısı 1358 olarak tespit edilmiştir. Atılan TCP paket sayısı 15 ve UDP paket sayısı 20 olarak bulunmuştur. Şekil 8(c)'de gösterildiği gibi red kuyruk tipinde ise düğüm 2'ye gelen ortalama paket sayısı 1364 olarak tespit edilmiştir. Atılan TCP paket sayısı 17 ve UDP paket sayısı da 31 olarak bulunmuştur. Şekil 8(a)'da gösterildiği gibi geliştirilen protokolde ise düğüm 2'ye ortalama 1715 tane paket gelmiştir ve bu gelen paketlerden TCP paketlerinde 2 tane kayıp olmuştur. Ancak UDP paketlerinden 255 tanesi atılmıştır.

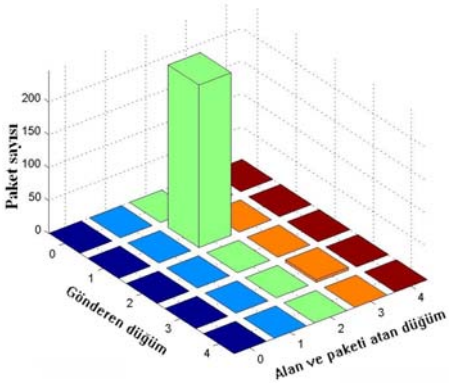
Geliştirilen protokolde WAP gateway görevi gören düğüm 2'ye gelen ortalama toplam paket sayısı diğer benzetimlerden daha fazladır. Elde edilen sonuçlardan da görüleceği gibi atılan ortalama TCP paket sayısı diğer kuyruk tiplerinden elde edilen sonuçlar ile karşılaştırıldığında çok daha azdır. Burada geliştirilen protokolün öncelik hakkına sahip TCP paketlerinde çok başarılı olduğu görülmektedir. Atılan UDP paketlerinin sayısının fazla olması ise kuyruk doluluk oranının %80'e eriştiği durumlarda önceliklendirme işlevinin devreye girmesiyle oluşmaktadır. Ancak tüm benzetim işlemi boyunca düğüm 2'ye gelen ortalama toplam paketlerin diğer kuyruk tiplerinin uygulandığı benzetimlerden daha fazla olduğu görülmektedir. Dolayısıyla atılan paket sayılarının toplam paket sayılarına doğrusal oranını gerçekleştirirsek atılan UDP paket sayılarının da çok fazla olmadığı görülmektedir. Böylece elde edilen tüm sonuçlar karşılaştırıldığında



(a)



(b)



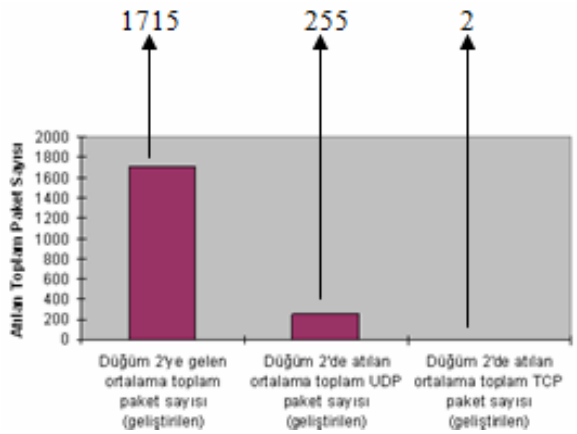
(c)

Şekil 7. (a) drop-tail (b) red ve (c) geliştirilen protokol için ilk senaryoda atılan paket sayıları (Number of dropped packet for the first scenario in (a) drop-tail, (b) red and (c) developed protocol)

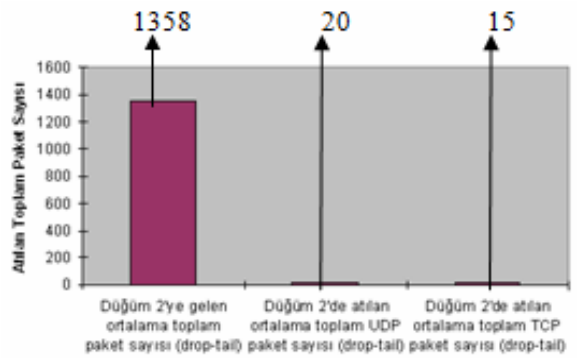
geliştirilen protokolün daha başarılı olduğu görülmektedir.

Senaryo 2: Şekil 9'da sırasıyla drop-tail, red kuyruk tipleri ile geliştirilen protokolün ikinci senaryo için benzetim işlemi sırasında elde edilen paket kayıplarının sayıları gösterilmektedir.

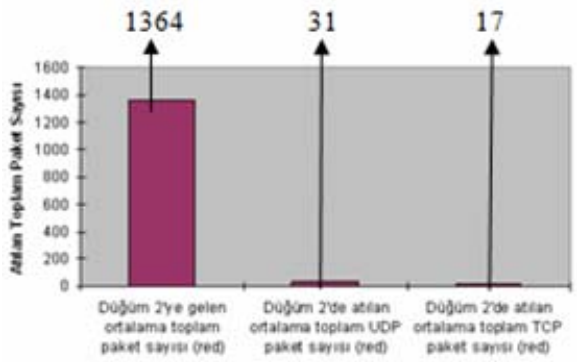
Şekil 9'da gösterildiği gibi geliştirilen protokolde kuyruk doluluk oranının %80'i aştığı durumlarda devreye giren önceliklendirme sayesinde WAP gateway görevi gören düğümde atılan paket sayısının diğer benzetim sonuçlarına göre çok daha iyi olduğu görül-



(a)



(b)

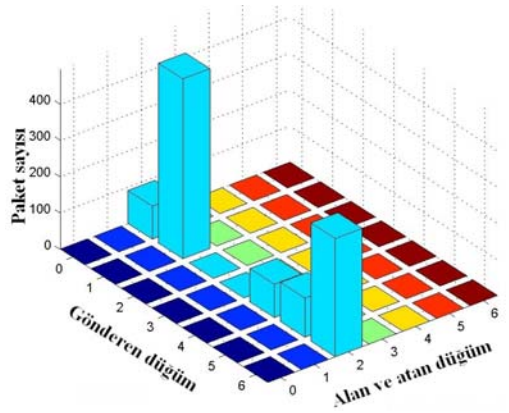


(c)

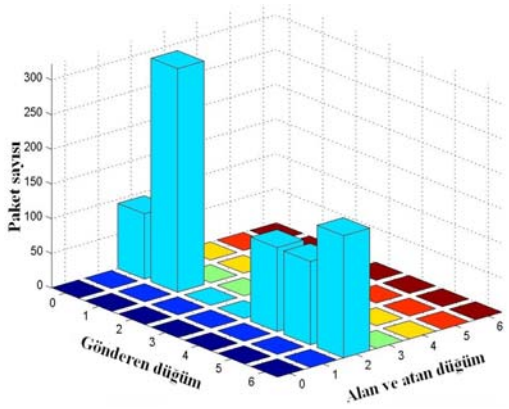
Şekil 8. İlk senaryoda toplam atılan paket sayıları (a) geliştirilen protokol, (b) drop-tail, (c) red (Total number of dropped packet for the first scenario (a)developed protokol, (b) drop-tail, (c) red)

mektedir. Üç protokolün ikinci senaryoya göre yapılan benzetim işleminin 10 kere tekrar edilmesi ile elde edilen toplam atılan paket sayılarının ortalaması Şekil 10'da gösterilmiştir.

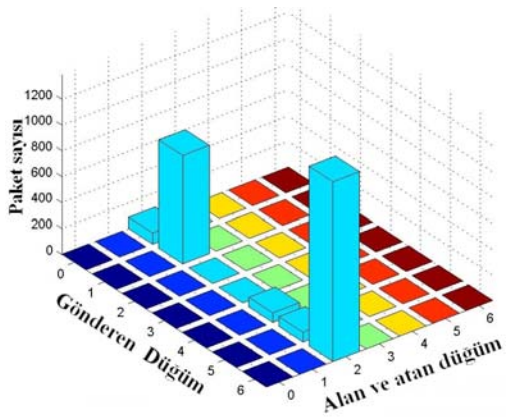
Şekil 10(a)'da gösterildiği gibi geliştirilen protokolde düğüm 2'de atılan ortalama toplam paket sayısının 2478, atılan TCP paket sayısı 465, UDP paket sayısı ise 2013 olarak elde edilmiştir. Şekil 10(b)'de gösterildiği gibi drop-tail kuyruk tipinde düğüm 2'de atılan ortalama toplam paket sayısı 1143'tür. Atılan ortalama TCP paket sayısı 752, atılan ortalama UDP paket sayısı ise 391'dir. Şekil 10(c)'de gösterildiği gibi red



(a)



(b)

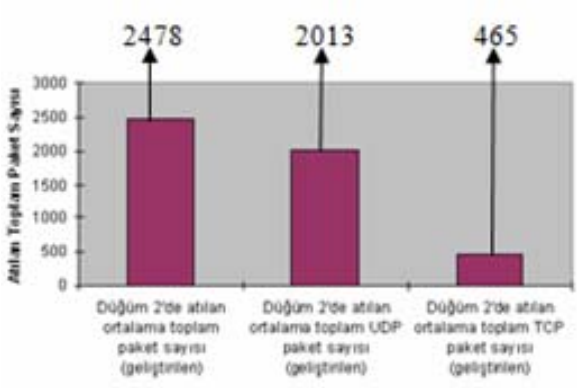


(c)

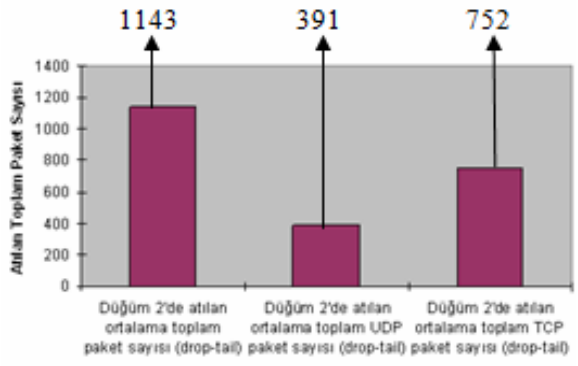
Şekil 9. (a) drop-tail (b) red ve (c) geliştirilen protokol için ikinci senaryoda atılan paket sayıları (Number of dropped packet for the second scenario in (a) drop-tail, (b) red and (c) developed protocol)

kuyruk tipinde ise düğüm 2'de atılan ortalama toplam paket sayısı 842, atılan ortalama TCP paket sayısı 623, UDP paket sayısı ise 219 olarak elde edilmiştir.

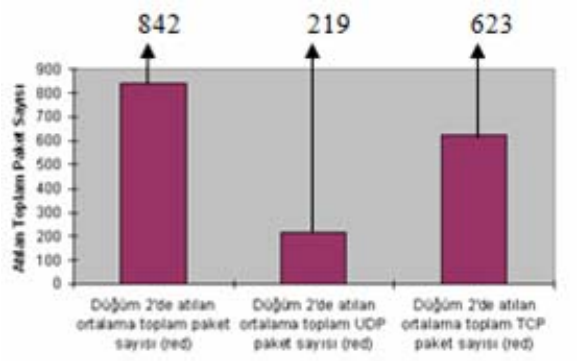
İkinci senaryoda elde edilen sonuçlardan sadece atılan paket sayıları üzerinde durulmuştur. Şekil 10'da gösterildiği gibi burada atılan toplam paket sayılarında geliştirilen protokolde daha fazla paketin atıldığı görülmektedir. Ancak, öncelik hakkına sahip olan TCP paketlerinin ortalama atılma sayısının diğer kuyruk tiplerinin kullanıldığı benzetimlere göre daha iyi



(a)



(b)



(c)

Şekil 10. İkinci senaryoda toplam atılan paket sayıları (a) geliştirilen protokol, (b) drop-tail, (c) red (Total number of dropped packet for the second scenario (a)developed protokol, (b) drop-tail, (c) red)

sonuç verdiği görülmektedir. UDP paket sayısının fazla olması ise kuyruk doluluk oranının %80 değerine ulaştığında önceliklendirmenin devreye girmesiyle olmaktadır. Geliştirilen protokolde düğüm 2'ye gelen toplam paket sayıları geliştirilen protokolde ortalama 35848, drop-tail kuyruk tipinde ortalama 34752 ve red kuyruk tipinde ise ortalama 33943 olarak elde edilmiştir. Atılan paket sayılarıyla düğüm 2'ye gelen paket sayılarının orantısı yapılırsa geliştirilen protokülün kullanıldığı benzetimde elde edilen sonuçların diğer benzetimlerde elde edilen sonuçlardan UDP paket atımı açısından çok fazla fark olmadığı görülmektedir.

Burada benzetim süresinin birinci senaryoya göre daha fazla olması ve kullanılan düğümlerin artması nedeniyle atılan paket sayılarındaki artış fazla olmaktadır. Birinci senaryodaki benzetim süresi 4 sn iken ikinci senaryoda 50 sn olarak alınmıştır. İkinci senaryoda aynı anda düğüm 2'ye gönderilen paket sayısının fazla olması sebebiyle atılan paket sayıları daha fazla olmaktadır.

Bütün elde edilen sonuçlar karşılaştırıldığında iki senaryoda da geliştirilen protokolün öncelik hakkına sahip paket türünde çok başarılı olduğu görülmektedir. Öncelik hakkı olmayan paket türünde ise çok fazla bir fark olmadığı görülmektedir.

6. SONUÇLAR (CONCLUSIONS)

Bu çalışmada WAP trafiğinde meydana gelen tıkanıklığın çözümü için bir protokol geliştirilmiştir. Geliştirilen protokol ns-2 benzetim aracı kullanılarak test edilmiştir. Kuyruk tipleri incelenerek tıkanıklığın kuyruk yapısında değişiklikler ile çözümü amaçlanmış ve başarılı sonuçlar alınmıştır.

Bugüne kadar yapılan çalışmalar genelde donanım üzerinde geliştirilmiş, cihazların kümelenmesi ve yük dağılımı esas alınarak tıkanıklık probleminin çözülmesi amaçlanmıştır. Ancak bu sistem hem pahalı hem de uygulanması zor olan bir yöntemdir. Genele yönelik önceliklendirme parametreleri arttırılarak geliştirilecek bir yöntemin hem uygulanması hem de maliyeti daha uygun olacaktır. Bu çalışmada geliştirilen protokol WAP gateway ile WAP sunucu arasındaki trafiği kontrol etmektedir.

Bu çalışmada WAP sunucu ile kullanıcı arasındaki iletişim sırasında oluşabilecek tıkanıklığı çözmek için kurulan bağlantılar üzerinde önceliklendirme yapılmıştır. Bu çalışmada TCP ve UDP arasında bir önceliklendirme yapılarak gelen paketlerin öncelik sırasına göre kuyruğa alınmasını sağlanmıştır. Böylece önem sırası büyük olan bir isteğin kuyruğa ilk girmesi amaçlanmıştır. Bu sayede genelde önemli iletişimlerin yapıldığı TCP bağlantısında paket kaybının ve gecikmelerin azaltılması sağlanmış ve buna bağlı olarak tıkanıklığın çözümünde başarılı olunmuştur. Bu çalışma sonucunda kuyruk tiplerinde yapılacak değişikliklerle veya yeni kuyruk tiplerinin oluşturulmasıyla daha verimli iletişim yapılabileceği görülmüştür.

KAYNAKLAR (REFERENCES)

1. Toksoy Y., "Kablosuz Uygulama Protokolü Mimarisi", Yüksek Lisans Tezi, **İTU Fen Bilimleri Enstitüsü,** İstanbul, 1-45 (2002).
2. Shen K., Yang T., Chu L., "Cluster Load Balancing For Fine-Grain Network Services", **Proceedings of IPDPS,** 493-500(2002).
3. Lin T.H., Wang K., Liu A.Y., "An efficient load balancing strategy for scalable WAP gateways", **Computer Communications,** 28(9): 1028-1037 (2005).
4. Shorten R., King C., Wirth F., Leith D., "Modelling TCP congestion control dynamics in drop-tail environments", **Automatica,** 43(3): 441-449 (2007).
5. Hur K., Eom Seop D.-, Lee Woo Y.-, Lee Ho J., Kang S., "Priority forwarding for improving the TCP performance in mobile IP based networks with packet buffering", **Computer Communications,** 30(6): 1337-1349 (2007).
6. Sun J., Zukerman M., "RaQ: A robust active queue management scheme based on rate and queue length", **Computer Communications,** 30(8): 1731-1741 (2007).
7. Chen S., Bensaou B., "Can high-speed networks survive with DropTail queues management?", **Computer Networks,** 51(7): 1763-1776 (2007)
8. Guo S., Liao X., Li C., Yang D., "Stability analysis of a novel exponential-RED model with heterogeneous delays", **Computer Communications,** 30(5): 1058-1074 (2007).
9. Şimşek M., "Kablosuz Ağlarda Yönlendirme Protokolü ve Tıkanıklık Denetimi", Yüksek Lisans, **Gazi Üniversitesi Fen Bilimleri Enstitüsü,** Ankara, 1-10 (2006).
10. Lindberg T., Uneman J., "WAP Research", MSc., **Chalmers University of Technology,** Göteborg, 1-69 (2001).
11. Arehart C., Chidambaram N., "Professional WAP", Wrox Pres, NewYork, 1861004044 (2000).
12. İnceoğlu M.M., Kılınç E., "WAP Ağ Geçidinde Ortaya Çıkan Boşlukların Kapatılmasına İlişkin Öneriler", **İnet-tr'03**, İzmir, 1-6 (2003).
13. Feng W., Shin Kang G., Kandlur Dilip D., Debanjan S., "The BLUE Active Queue Management Algorithms", **IEEE/ACM Transactions on Networking (TON),** 10(4): 1-18 (2002).
14. Zorzi, M., Rao R. R., and Milstein L. B., "On the accuracy of a first-order Markov model for data block transmission on fading channels," **in Proc.IEEE ICUPC**, Tokyo, Japan, Nov. 1995, pp. 211–215.
15. Kurose J.F., Ross K.W., Computer Networking: **A Top-Down Approach Featuring the Internet.** Reading, MA: Addison-Wesley, 2000.
16. Cardwell N., Savage S., Anderson T., "Modeling TCP latency," **Proc. IEEE INFOCOM**, Tel Aviv, Israel, 1742–1751, 2000.
17. Rutagemwa H., Shen X., Mark J. W., "Transfer Delay Analysis of WAP 2.0 fo Short-Lived Flows", **IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY**, 56(3), 1418-1426, 2007.
18. Howard R.A., **Dynamic Probabilistic Systems**, New York: Wiley, 1971.
19. Ross S.M., **Introduction to Probability Models**, 7th ed. Orlando, FL: Harcourt Brace Jovanovich, 2000.
20. Padhye J., Firoiu V., Towsley D. F., and Kurose J.F., "Modeling TCP Reno performance: A simple model and its empirical validation," **IEEE/ACM Trans. Netw.**, vol. 8, no. 2, pp. 133–145, Apr. 2000.