PAMUKKALE ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ
PAMUKKALE UNIVERSITY ENGINEERING COLLEGE
MÜHENDİSLİK BİLİMLERİ DERGİSİ
JOURNAL OF ENGINEERING SCIENCES
YIL : 2004
CİLT : 10
SAYI : Özel Sayı
SAYFA : 7-12

BİLGİSAYAR AĞLARINDA KULLANILAN DELİK KOVA TIKANIKLIK KONTROL ALGORİTMASI İÇİN SİMULATOR

Birim BALCI
Maltepe Üniversitesi, Müh.-Mim. Fakültesi, Bilgisayar Müh. Bölümü, 34857-Başıbüyük- Maltepe/İstanbul

ÖZET

Bilgisayar ağlarında tıkanıklık önemli bir sorundur. Bu soruna çözüm getirmek için çeşitli tıkanıklık kontrol algoritmaları geliştirilmiştir. Bu çalışmada, yönlendirici tıkanıklık kontrol algoritmalarından Delik Kova tıkanıklık kontrol algoritmasının eğitim amaçlı simulatörü tasarlanmıştır. Bilgisayar ağları dersinde ağ tıkanıklığı ile ilgili temel prensipler verilmektedir. Ancak laboratuarlarda ağ tasarımı yapılamamakta ve öğrencilere uygulamalarla tıkanıklık kontrol algoritmaları anlatılamamaktadır. Bu çalışmada tasarlanan simulatör sayesinde, delik kova tıkanıklık kontrol algoritmasının işleyiş mantığının adım adım gözlenmesi mümkün olacaktır. Yönlendiricilerin kendilerine gelen paketleri alması ve göndermesi ile ilgili kurallar gerçektekine uygun olarak belirlenmiştir.

Anahtar Kelimeler : Tıkanıklık kontrolü, Delik kova algoritması, Bilgisayar ağları, Simulatör

SIMULATION FOR LEAKY BUCKET CONGESTION CONTROL ALGORITHM USED IN COMPUTER NETWORKS

ABSTRACT

Congestion control in computer networks is an important problem. To bring a solution to this problem, various control algorithms are improved. In this study, simulator having the purpose of education has been designed for the Leaky Bucket Congestion Control Algorithm. The basic principles of the network congestion are given in the computer network lessons. However, in order to teach the congestion control algorithms by practices, a network design can not be done in the laboratories. By the simulator designed in this study, it will be possible to see the working way of the leaky bucket congestion control algorithm step by step. The rules about taking and sending the packets of the routers were determined suitable to the real working rules.

Key Words : Congestion control, Leaky bucket Algorithm, Computer networks, Simulator

1. GİRİŞ

Günümüzde, bilgisayar ağları üzerinden iletim sayesinde bilginin çok hızlı şekilde iletimi ve paylaşımı mümkün olmuştur. Bilginin iletimi sırasında, ağ üzerindeki iletişim bağlantıları, yönlendiricilerin bellek kapasiteleri, hız, işlem süresi gibi nedenlerden dolayı tıkanmalar oluştuğundan veri kaybı ya da geç iletimi, sistem kaynaklarının boşa kullanımı, ağ çöküntüsü gibi iletişim sorunları yaşanmaktadır. Bu noktada, bilgisayar ağlarında tıkanıklık kontrolünün önemi artmaktadır. Geliştirilen tıkanıklık kontrol stratejilerinden biri 1986 yılında Turner tarafından öne sürülen Delik Kova tıkanıklık algoritmasıdır.

Bilgisayar ağları tıkanıklığı, kaynak paylaşım problemi olarak bilinmektedir. Paket anahtarlamalı ağlarda kaynaklar, birbiriyle bağlantılı tüm bilgisayarlar arasında paylaşılır. Eğer ağdaki kaynaklar tüm kullanıcıların o anki isteklerine cevap veremiyorsa ağ tıkanıklığı oluşur (Yang, 1995). Bir başka ifadeyle, tıkanıklık, alt ağda (subnet) fazla paket bulunmasından dolayı oluşan performans düşmesidir. Alt ağda bir yönlendiriciye

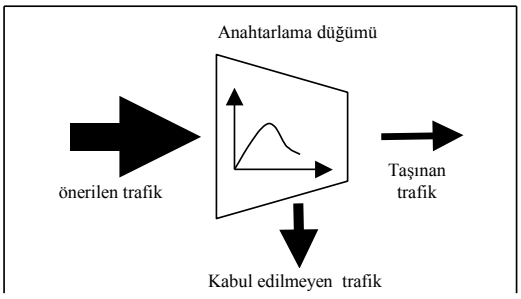
gelen ve yönlendiriciden gönderilen paket sayısı orantılı olmalıdır (Tanenbaum, 1996).

1. 1. Tıkanıklığın Nedenleri

Tıkanıklığın nedenleri, gelen paket oranının çıkan paket oranından fazla olması (Fortier, 1992), yavaş işlemler, paketlerin yeniden gönderimi (Filipiak, 1991) olarak özetlenebilir.

- Eğer 3-4 farklı girişten gelen paketlerin hepsi aynı çıkış hattını kullanmak isterse tıkanıklık oluşur ve paketlerin hepsini tutabilecek yeterli bellek yoksa paketler kaybolur.
- Fazla hat kapasitesi olsa bile yönlendiricinin işlemcisinin yavaş olması durumunda kuyruk oluşur. Hatların yenilenmesi, fakat işlemlerin değiştirilmemesi ya da tam tersi darboğaz yaratır. Ayrıca, tüm sistemin değil sadece bir bölümünün yenilenmesi de darboğazı sadece başka bir bölgeye kaydırır, çözümlemez.
- Eğer yönlendirici hiç boş tampona sahip değilse, yeni gelecek paketleri ihmal etmesi gerekecektir. Paket ihmal edilince kaynak yönlendiricinin süresi dolar ve pek çok kere yeniden gönderir. Paket atılamayacağından, geribildirim alana kadar, alıcı tarafındaki tıkanıklık, göndericiyi normal olarak boşaltılacak bir tamponun serbest bırakılmasından kaçınmak için zorlar (Tanenbaum, 1996, Fortier, 1992).

Şekil 1'de görüldüğü gibi, bir değere kadar trafik, önerilen trafiğe (offered traffic) eşit ya da hemen hemen eşit olmalıdır. O değerden sonra ihmal edilen paketler, yani reddedilen trafik hızla büyümeye başlar (Filipiak, 1991).



Şekil 1. Anahtarlama düğümü (Filipiak, 1991)

1. 2. Tıkanıklığın Etkileri ve Giderme Yolları

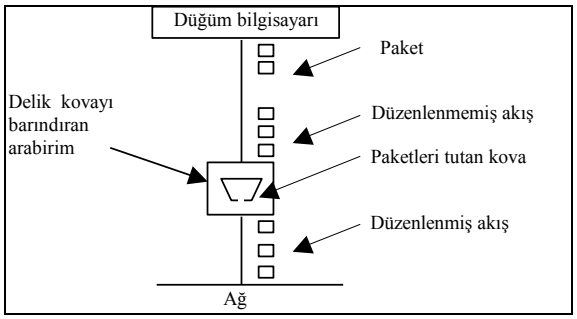
Ağ katmanında yeterli trafik kontrolü olmazsa, verimde düşme, doğru sonuç alamama ve kilitlenme (deadlock-lockup) gibi durumlar gözlenebilir. Bir düğüm, gelen paketleri gönderim hızından daha yüksek hızda alırsa ya alt ağ mevcut paketlerin işlemi bitene kadar gelen paketlerin tıkalı alana girişini engeller, ya da tıkalı yönlendiriciler sırada bekleyen paketleri yeni gelenleri alabilmek için ihmal edebilir (Meng).

Paket anahtarlamalı ağlarda gözlenen tıkanıklık problemleri, düğüm noktalarındaki paketlerin erişebileceği tampon alanını artırmakla çözümlenebilir. Ancak, yönlendiriciler sonsuz miktarda belleğe sahipse tıkanıklık daha da kötüleşecektir. Çünkü paketler kuyruğun önüne geldiklerinde zaten süreleri bitmiş olacaktır ve tekrar gönderileceklerinden yük artacaktır (Tanenbaum, 1996). Bu, çok fazla bellek alanı olması durumunun, çok az bellek alanı olmasından daha zararlı olduğunu göstermektedir (Jain, 1990). Uzun kuyruklar zaman kaybıdır ve daha karmaşık denetim gerektirir. Bununla birlikte, tıkanıklık sırasında oluşan problemler çözümlenmiş değil, sadece ertelenmiş olur (Filipiak, 1991).

Çok hızlı bağlantılara sahip olmak tıkanıklığı gidermede bir çözüm gibi görünse de yerel ağlar yavaş bağlantılarla birbirine bağlandığında, bağlantı noktalarında gene tıkanıklık problemi oluşacaktır (Jain, 1990). Tüm bağlantılar ve işlemciler aynı hıza sahip olsa bile tıkanıklık oluşabilir. Örneğin, tüm işlemcilerin ve bağlantıların hızının 1 Gb/s olduğu, A ve B düğümlerinin C yönlendiricisi üzerinden D hedef düğümüne paket yolladığı bir ağ modeli düşünülürse, C yönlendiricisinde tıkanıklık gözlenecektir. Çünkü yönlendiriciye gelen yük toplam 2Gb/s iken gönderebileceği yük 1Gb/s ile sınırlı olacağından, gelen paketlerde yığılma olacaktır.

2. DELİK KOVA ALGORİTMASI

Delik kova algoritması, ağdaki tıkalı noktalardan geri bildirim yollanmayan ve yeni trafiğin ne zaman kontrol edileceğine, hangi paketleri ne zaman ihmal edeceğine hedef düğüm ve hedefe giden yol üzerindeki orta noktaların karar verdiği tipte (açık devre hedef kontrollü) bir tıkanıklık kontrol algoritmasıdır (Yang, 1995). Çalışma mantığı, „altı delik bir kovaya su hangi hızda gelirse gelsin, suyun çıkış hızı sabit olacak ve kova dolu olduğunda gelen su taşacaktır" şeklinde açıklanabilir. Şekil 2'de de görüldüğü üzere, delik kova içeren bir arabirim ile ağa bağlı her düğüm, bir kuyruğa sahiptir. Düğüme kuyruk dolu iken gelen paketler ihmal edilir, atılır (Tanenbaum, 1996).

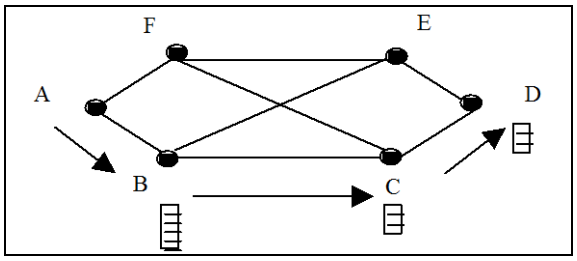


Şekil 2. Paket içeren delik kova (Tanenbaum, 1996)

Düğüm, her saat tıklamasında eşit büyüklükteki bir paketi ağa bırakır. Ancak farklı büyüklükteki paketler söz konusu olduğunda, bir saat tıklamasında sabit sayıda byte gönderilmesi daha uygundur. Eğer bir tıklamada 1024 byte kuralı varsa, tıklamada tek bir 1024-byte paketi ya da 2 tane 512-byte paketi gönderilebilir (Tanenbaum, 1996).

3. SİMÜLATÖRÜN YAPISI

Bu çalışmada tasarlanan Delik Kova tıkanıklık Kontrol Algoritmasının simulatörüne (Balcı, 2001) ait ağ modeli, çalışma kuralları aşağıda açıklanmaktadır. Simulatör Nesneye Dayalı Programlama Dillerinden DELPHI 4.0 ile tasarlanmıştır. Tasarım için Şekil3'de görülen altı tane yönlendiriciden oluşan sabit bir ağ modeli kullanılmıştır. Yönlendiricilerin altındaki kutular o yönlendiriciye ait bellek alanlarını göstermektedir.



Şekil 3. Simulasyonda kullanılan ağ modeli

3. 1. ÇalıŞma Kuralları

Algoritmanın işleyişinin kolay anlaşılabilmesi için, çalışma kuralları aşağıda sıralanmıştır.

1. Paket iletimi, A kaynak düğümünden, D hedef düğümüne doğru B ve C yönlendiricileri üzerinden yapılır. E ve F yönlendiricileri üzerinden paket gönderimi yapılmamıştır.
2. Ram bellekler B yönlendiricisinde 5, C ve D yönlendiricilerinde ise 3 tampon alanı ile sınırlandırılmıştır.
3. Paket üretimi, A kaynağında ve seriler halinde olur. Bir seride, 1-10 arasında paket üretebilir. Bu sayı rastgele seçilir.
4. Simulasyon 10 seri paket üretimi süresinde çalışır.
5. B ve C düğümleri belleklerindeki sayıda paketi bir sonraki düğüme iletmekle görevlidir.
6. D düğümü ise, tampon alanını, paketlerini rastgele belirlenen sayıda, sanal olarak var olduğu kabul edilen başka bir düğüme göndererek boşaltır.
7. D düğümünün göndereceği paket sayısı, tampon alanının doluluğuna bağlıdır. Rastgele seçilen sayı, belleğindeki paket sayısından büyükse, belleğindeki paket sayısı kadar paket boşaltabilir. Eğer rastgele seçilen sayı, belleğindeki paket sayısından küçükse, o zaman rastgele seçilen sayı kadar paket boşaltacaktır.
8. Düğümlerden, paket gönderimi 10 sn"de bir yapılır ve üretilen serideki sonuncu paket, 10. sn"de bir sonraki düğüme ulaşmak durumundadır.
9. Düğüm noktalarına gelen paket sayısı, o düğümün paket kapasitesinden büyükse, fazlası ihmal edilir. Bu durum, simulasyonda ihmal edilecek paketlerin koyu yeşil renk alarak aşağıya düşmesi ile ifade edilmiştir.
10. Düğümler arasındaki paket iletimi 6 adımda sağlanır. Altıncı adımda paket ya tampon alanına yerleşir ya da ihmal edilir.
11. Bir seride üretilen paketlerin bir sonraki düğüme ulaşmasının 10 sn ile sınırlandırılması sonucunda, düğümden kaç sn"de bir paket gönderileceği (timer.interval), serideki paket sayısına göre her 10 sn"de bir değişecektir. Bu değer, formül 1"den hesaplanmaktadır.

$$t = (p-1)*x + a*x \quad (1)$$

t : toplam süre (10000 ms)
p : gönderilecek paket sayısı
a : düğümler arası adım sayısı (6)
x : kaç sn"de paket gönderileceği (interval)

Bu formül maksimum 10 paket sayısı için uygulandığında, her p sayılık seri içindeki bir paketin adım atma hızı, Tablo 1"de gösterilmiştir.

Bir seride 5 paket gönderileceği varsayılırsa, A"dan B"ye bu paketlerin toplam 10 sn"de ve her birinin 6 adımda gitmesi için aralardaki bekleme süresinin 1000 ms olması gerekmektedir. Ayrıca paketlerin geliş zamanı, gerçek geliş zamanı, serviste kaldığı süreleri de göz önüne alınırsa, Tablo 2"de görülen daha detaylı bir çizelge oluşturulabilir.

Tablo 1. Delik Kova Algoritması Paket Sayısına Göre Hız Değerleri [7]

Paket sayısı- p	1	2	3	4	5	6	7	8	9	10
Hız- ms	1666	1428	1250	1111	1000	909	833	769	714	666

Tablo 2. Paketlerin Geliş Zamanı ve Servis Süresi [7]

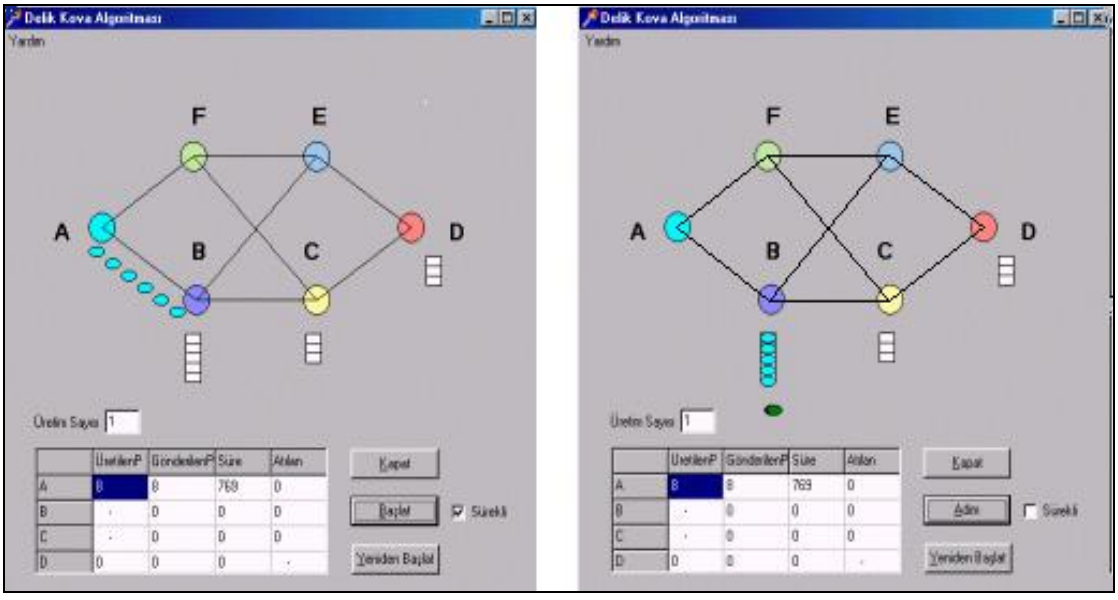
Zaman-sn	P1	P2	P3	P4	P5
A_i	0	1	1	1	1
G_i	0	1	2	3	4
S_i	6	6	6	6	6
D_i	6	7	8	9	10

Tablo 2'de A_i paketlerin geliş zamanını, G_i paketlerin gerçek geliş zamanını, S_i servis süresini yani hareket halinde geçirdiği süreyi ve D_i ise servisin bittiği zamanı göstermektedir. Toplam 5 paket olacağından ve 1 sn'lik aralıklarla adım atacaklarından, her paketin serviste kalacağı süre 6 sn olacaktır. Birinci paketin geliş zamanı ve gerçek geliş zamanı 0. sn, bitiş zamanı da 6. sn olmaktadır. İkinci paket, birincinin yollanmasından 1 sn sonra yolla çıkarıldığından geliş zamanı 1 sn'dir. Ancak, servis bitimi 7.sn'dir. Aynı şekilde, 5. paketin gerçek geliş zamanı, kendinden önce 4 paket olduğundan ve ilk paket 0. sn'de geldiğinden 4.sn olarak görülmekte, servis süresi 6 sn olduğundan servis bitimi 10. sn'ye rastlamaktadır.

3. 2. Simulatörün ÇalıGması

Arayüz tasarımında farklı renkteki dairelerle gösterilen yönlendiricilere ve *A...F* adları verilmiştir. Tampon alanları kutucuklar halinde bu düğümlerin altına sıralanmıştır. Şekil 4'de görülen arayüz üzerindeki tabloya çalışma sırasında düğümlere ait hız ve ihmal edilen paket sayısı bilgileri yazılmaktadır. "Sürekli" isimli checkbox işaretlenirse, simulasyon 10 seri üretimi süresince sürekli çalışacaktır. İşaretlenmezse, "Başlat" butonunun başlığı "Adım" olarak değişecek ve her seri üretimi ve paketlerin hareketini sağlamak için bu butona basılması gerekecektir. Program sürekli çalışma modunda iken, sürekli kutucuğundaki işaret kaldırılırsa, mevcut 10 sn'lik periyodun sonunda programın adım adım işlemesi sağlanabilmektedir. Kaçıncı paket serisinin üretildiği "Üretim Sayısı" alanında görülebilmektedir. "Yeniden Başlat" butonu, programın çalışması esnasında ve Adım modunda iken tüm değerlerin sıfırlanarak, programın çalışmasının durdurulmasını sağlar. "Yardım" alanından ise algoritmanın çalışma prensipleri, değer analizleri ve program hakkında bilgi alınabilmektedir. Kapat butonu ise, programın herhangi bir aşamasında kırılarak sonlandırılması amacıyla kullanılmaktadır.

Simulatörün çalışması aşağıdaki adımlarla açıklanabilir.



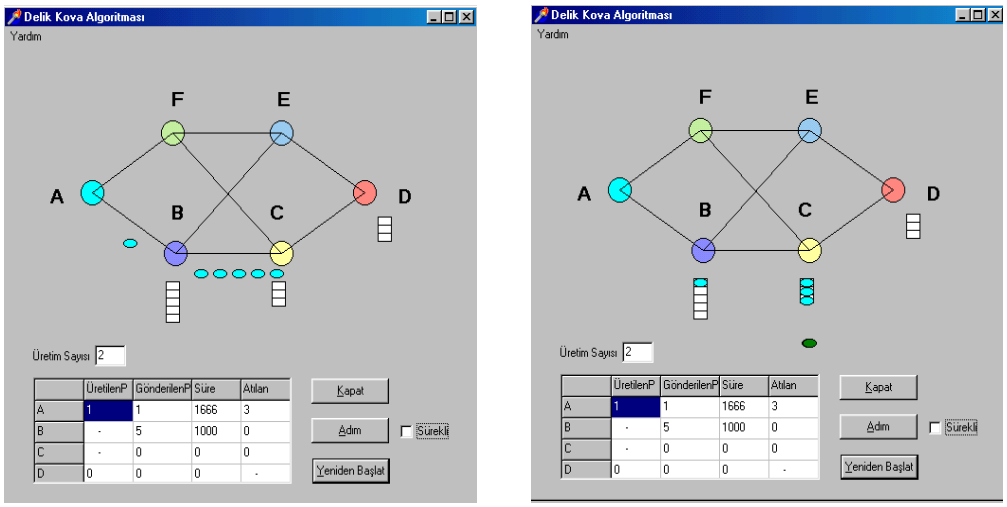
Şekil 4. Simulatörü Çalışması Birinci Adım (a); (b)

Adım 1 : Şekil 4 (a)'da simulatör *sürekli* modunda çalıştırılmış ve A düğümünde rastgele olarak toplam 8 paket üretilmiştir. Adım atma süreleri hesaplanıp 769 ms olarak tabloya yazdırılmıştır. (b)'de ise ilk adım sonunda, A düğümünden gönderilen sekiz paketten 5 tanesinin B tampon alanına yerleştirildiği kalan üç tanesinin ise atıldığı görülmektedir.

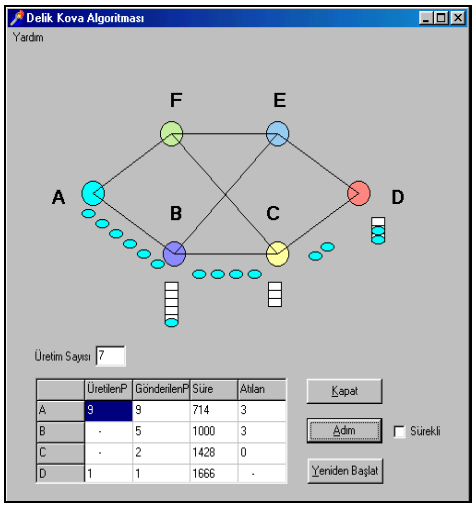
Adım 2 : Adım butonuna basıldığında, tabloya ilk adım sonucunda B düğümünde, A'ya ait atılan paket sayısı yazdırılır (Şekil 5a). A düğümünde, 1 paket içeren ikinci paket serisi üretilmiştir. Aynı anda hem A, hem de B düğümleri paketlerini yola çıkartırlar. A ve B'nin gönderdikleri paket sayıları ve hızları tabloya yazılır. İkinci adım sonunda B'den gönderilen 5 paketin 3'ü C tampon alanına yerleşirken 2'si ise dışarı atılacaktır (Şekil 5b).

Adım 3 : A düğümünde rastgele 4 paket üretilmiştir. A düğümü ürettiği 4 paketi, B tampon alanındaki 1 paketi ve C'de tampon alanına yerleşen 3 paketi göndereceğinden ve gönderilen paketler hedef düğümlerin tampon alanları kapasitelerini aşmadığından, hiçbir düğümde tıkanma oluşmayacak ve paket ihmali olmayacaktır.

Altıncı adım sonunda B tampon alanına 5 paket, C tampon alanına 2 paket ve D tampon alanına 3 paket yerleşmiştir. Bu değerleri dikkate aldığımızda, Şekil 6'da simulasyonun yedinci adımında A düğümünün ürettiği dokuz paketin en son altıncısının yola çıkarılmış olduğu görülmektedir. Burada dikkat edilecek nokta, bir önceki adımda D belleğinde 3 paket olduğu ve bu adımda D'nin rastgele 1 değerini seçerek, tampon alanından 1 paket boşaltmasıdır. Simulatör yeniden çalıştırıldığında, rastgele değerler üretilmesi nedeniyle, değişik işlemler yapmakta olduğu gözlenmiştir.



Şekil 5. Simulatörün çalışması ikinci adım (a); (b)



Şekil 6. Simulatörün çalışması yedinci adım

4. SONUÇ

Bu çalışmada elde edilen sonuçlar, delik kova tıkanıklık algoritması gerçek uygulama sonuçlarıyla aynı paralelliktedir. Ağ modelindeki paket iletiminde izlenen yol, yönlendiricilerin tampon alanı kapasiteleri, paket iletim hızı, düğümlerin hız ayarlamasında belirlenen kurallar tamamen birbirleriyle ilişkili olduğundan, bunlardan herhangi birinin farklı şekilde değerlendirmesi sonucunda simulasyon beklenen sonuçları vermeyebilir.

Gerçek bilgisayar ağlarında paket iletimi sadece bir yol üzerinden değil, birçok yol üzerinden sağlanabilmektedir. Ayrıca, bir yönlendirici üzerinden, farklı kaynaklardan gelen paketler

geçebilmektedir. Açıklanan çalışma kuralları, bu çalışmada tasarlanan ağ modeli için kabul edilmiş kurallardır. Bu çalışmada paket iletiminin tek hat üzerinden sağlanması ve bir yönlendirici üzerinden farklı kaynak paketlerinin geçmemesi kuralının kabul edilmesinin nedeni, bilgisayar ağları dersinde öğrencilerin simulatörlerde ele alınan tıkanıklık kontrol algoritmalarının işleyişini daha kolay analiz edebilmelerini sağlamaktır.

5. KAYNAKLAR

Balcı, B. 2001. "Bilgisayar Ağlarında Tıkanıklık Kontrol Algoritmaları İçin Simulatör", Yüksek Lisans Tezi, 88s. Marmara Üniv., Fen Bilimleri Enstitüsü, İstanbul, Türkiye.

Filipiak, J. 1991. "Real Time Network Management",Amsterdam, North-Holland, 27-37, 61-71.

Fortier, P. J. 1992. "Handbook of LAN Technology", *McGrawHill*, 2nd Ed.; New York, USA, 249-258.

http://www.comsoc.org/pubs/surveys/yang/yang-orig.html, Erişim Tarihi : 23.09.2000.

Jain, R. M. 1990. "Congestion Control in Computer Networks : Issues and Trends", *IEEE Network Magazine*, 24-30.

Meng, X. 2001. "Network Layer", *University of Texas-Pan American*, *http://www.cs.panam.edu/~meng/Course/CS6345/Notes/chpt-5/node1.html*, Erişim:10.05.2001.

Tanenbaum, A.S. 1996. "Computer Networks", *Prentice-Hall*, 3th Ed.; USA, 374-391.

Yang, C., Reddy, A.V.S.. July 1995. "A Taxonomy for Congestion Control Algorithms in Packet Switching Networks", *IEEE Network Magazine*,Vol. 9, No. 5.