

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

**TCP/IP'DE AKTİF KUYRUK YÖNETİM MEKANİZMALARI
VE İNTERNET İÇİN KALİTELİ HİZMET**

YÜKSEK LİSANS TEZİ

Müh. Bahri OKUROĞLU

504960603011

Anabilim Dalı: Kontrol ve Bilgisayar Mühendisliği

Programı: Kontrol ve Bilgisayar Mühendisliği

Tez Danışmanı: Y.Doç.Dr. Sema OKTUĞ

HAZİRAN 2000

ÖNSÖZ

Çalışmalarım süresince gösterdiği sabır, anlayış ve desteği için eşime; yakın ilgisi, yol göstericiliği ve motive edici tavrı için tez danışmanım Y.Doç.Dr Sema Oktuğ'a ve geliştirdikleri kodlar ve yorumları ile çalışmalarına katkıda bulunan Sean Murphy ve diğer NS geliştiricilerine teşekkür ederim.

Haziran 2000

Bahri Okuroğlu

İÇİNDEKİLER

| | |
|--|-------------|
| KISALTMALAR | v |
| TABLO LİSTESİ | vii |
| ŞEKİL LİSTESİ | viii |
| SEMBOL LİSTESİ | xii |
| ÖZET | xiii |
| SUMMARY | xiv |
| 1 GİRİŞ | 1 |
| 2 IP AĞLARI İÇİN KALİTELİ HİZMET | 5 |
| 2.1 Kaliteli Hizmet | 6 |
| 2.1.1 Hizmet kalitesi | 6 |
| 2.2 Temel QoS Mimarisi | 7 |
| 2.3 Uygulamaların QoS ihtiyaçları | 8 |
| 2.4 Farklı Tipte Hizmetler Sunmanın Yolları | 8 |
| 2.4.1 Fiziksel katman çözümleri | 8 |
| 2.4.2 Veri bağı katmanı çözümleri | 9 |
| 2.4.2.1 ATM | 9 |
| 2.4.2.2 Frame relay | 10 |
| 2.4.3 Ağ ve taşıma katmanı çözümleri | 11 |
| 2.4.3.1 FIFO kuyruklar | 12 |
| 2.4.3.2 Öncelikli kuyruklar | 13 |
| 2.4.3.3 Sınıflandırılmış kuyruklar | 13 |
| 2.4.3.4 Ağırlıklı adil kuyruk | 13 |
| 2.4.3.5 Ağırlıklı round-robin ve eksik ağırlıklı round-robin algoritmaları | 13 |
| 2.5 QoS Protokolleri | 14 |
| 2.5.1 Tümüleşik hizmetler | 15 |
| 2.5.1.1 Referans uygulama | 16 |
| 2.5.1.2 Garantili hizmet | 18 |
| 2.5.1.3 Kontrollü hizmet | 19 |
| 2.5.2 Farklılaştırılmış Hizmetler (Differentiated Services, DS) | 21 |
| 2.5.2.1 DS mimarisi | 22 |
| 2.5.2.2 Garanti edilen düğüm davranışı | 23 |
| 2.5.2.3 Çabuklaştırılmış PHB | 24 |
| 2.5.2.4 Müşteri tarafında hizmet paylaşımı | 25 |
| 2.5.3 IS ve DS'in karşılaştırılması | 28 |
| 2.5.4 Kaynak Rezervasyon Protokolü | 29 |
| 2.5.4.1 Çalışma şekli | 30 |
| 3 TIKANIKLIK DENETİMİ | 33 |
| 3.1 Tıkanıklık | 34 |
| 3.2 Tıkanıklık Çöküşü | 35 |
| 3.3 TCP ve TCP Tıkanıklık Denetimi Yapıları | 36 |
| 3.3.1 Tıkanıklık denetimi | 38 |
| 3.3.2 Kendinden-zamanlama | 38 |
| 3.3.3 Yavaş başlangıç | 39 |

| | |
|---|------------|
| 3.3.4 Tıkanıklık önleme | 40 |
| 3.3.5 Hızlı tekrar iletim ve toparlanma | 41 |
| 3.3.6 Tıkanıklığa tepki verme | 42 |
| 3.4 Kuyruk Yönetim Yapıları | 42 |
| 3.4.1 Aktif kuyruk yönetimi ihtiyacı | 43 |
| 3.4.2 Rastlantısal erken algılama | 45 |
| 3.4.2.1 Önceki çalışmalar | 45 |
| 3.4.2.2 Tasarım hedefleri | 46 |
| 3.4.2.3 RED algoritması | 47 |
| 3.4.2.4 Parametre seçimi | 49 |
| 3.4.2.5 RED'in başarımı | 50 |
| 3.4.3 Uyarlanabilir RED | 58 |
| 3.4.3.1 ARED algoritması | 60 |
| 3.4.4 RED üzerine diğer çalışmalar | 62 |
| 3.4.5 BLUE | 62 |
| 3.4.5.1 BLUE algoritması | 63 |
| 3.4.5.2 BLUE'nun başarımı | 63 |
| 3.4.5.3 Parametre seçimi | 68 |
| 3.5 Açık Tıkanıklık Bildirimi | 69 |
| 3.5.1 IP'de yapılan değişiklikler | 69 |
| 3.5.2 TCP'de yapılan değişiklikler | 70 |
| 3.5.3 ECN başarımı | 71 |
| 4 DS YAPISININ GERÇEKLENMESİ | 72 |
| 4.1 Örnek bir DS yönlendiricisi | 72 |
| 4.2 RIO | 74 |
| 4.2.1 RIO algoritması | 74 |
| 4.2.2 RIO'nun başarımı | 77 |
| 4.3 DS yönlendiricisi ayarları | 79 |
| 4.4 Benzetimler ile AF başarımı | 81 |
| 5 BIO | 87 |
| 5.1 Algoritma | 88 |
| 5.2 BIO Başarımı | 89 |
| 5.2.1 10 düğüm üzerinde 1'er aktif akış | 90 |
| 5.2.2 10 düğüm üzerinde 10'ar aktif akış | 92 |
| 5.2.3 10 düğüm üzerinde 10'ar aktif akış | 94 |
| 5.2.4 Analiz | 95 |
| 5.3 BIO için kullanılan NS konfigürasyonu | 96 |
| 6 SONUÇ | 97 |
| KAYNAKLAR | 100 |
| EK A NS AĞ BENZETİM ARACI | 106 |
| 6.1 Yapısı | 106 |
| EK B GECİKME DEĞİŞİMİ HESABI | 110 |
| EK C BLUE PARAMETRE ANALİZİ İÇİN GERÇEKLEŞTİRİLEN BENZETİM SONUÇLARI | 111 |
| ÖZGEÇMİŞ | 113 |

KISALTMALAR

| | |
|-------------|---|
| ABR | : Available Bit Rate |
| ACK | : ACKnowledgement |
| AF | : Assured Forwarding |
| ARED | : Adaptive RED |
| ATM | : Asynchronous Transfer Mode |
| BB | : Bandwidth Broker |
| BECN | : Backward Explicit Congestion Notification |
| BECN | : Backward Explicit Congestion Notification |
| BIO | : Blue with In and Out |
| CBQ | : Class Based Queuing |
| CBR | : Constant Bit Rate |
| CIR | : Committed Information Rate |
| CS | : Controlled-load Service |
| Cwnd | : Congestion Window |
| DE | : Discard Eligible |
| DFOF | : Drop Front On Full |
| DS | : Differentiated Services |
| DT | : Drop Tail |
| ECN | : Explicit Congestion Notification |
| EF | : Expedited Forwarding |
| ERD | : Early Random Drop |
| FECN | : Forward Explicit Congestion Notification |
| FIFO | : First In First Out |
| GS | : Guaranteed Service |

| | |
|----------------|---------------------------------------|
| IETF | : Internet Engineering Task Force |
| IP | : Internet Protocol |
| IS | : Integrated Services |
| ISDN | : Integrated Services Digital Network |
| İSS | : İnternet Servis Sağlayıcı |
| MSS | : Maximum Segment Size |
| MTU | : Maximum Transmission Unit |
| Nrt-VBR | : Non Real Time Variable Bit Rate |
| PHB | : Per-Hop Behaviour |
| QoS | : Quality of Service |
| RDOF | : Random Drop On Full |
| RED | : Random Early Detection |
| RIO | : RED with In and Out |
| RSVP | : Resource reSerVation Protocol |
| RTO | : Retransmission Timeout |
| RTT | : Round Trip Time |
| Rt-VBR | : Real Time Variable Bit Rate |
| Rwnd | : Receiver's Window |
| SLA | : Service Level Aggrement |
| SLS | : Service Level Specification |
| SQ | : Source Quench |
| TCP | : Transmission Control Protocol |
| TOS | : Type of Service |
| UBR | : Unspecified Bit Rate |
| UDP | : User Datagram Protocol |
| VC | : Virtual Circuit |
| WFQ | : Weighted Fair Queuing |
| WRR | : Weighted Round Robin |

TABLO LİSTESİ

| | <u>Sayfa No</u> |
|--|------------------------|
| Tablo 2.1 ATM QoS yapıları | 9 |
| Tablo 4.1 RED ve RIO iletim hızları | 77 |
| Tablo 4.2 ECN kullanıldığında RED ve RIO iletim hızları | 78 |
| Tablo 4.3 Örnek bir SLS | 79 |
| Tablo 5.1 Her düğümde tek FTP aktif olduğu durum için BIO ve RIO paket kayıp oranları | 91 |
| Tablo 5.2 Her düğümde 10 FTP aktif olduğu durum için BIO ve RIO paket kayıp oranları | 93 |
| Tablo A.1 NS benzetim aracı temel objeleri | 107 |

ŞEKİL LİSTESİ

Sayfa No

| | | |
|------------|---|----|
| Şekil 1.1 | Internet üzerindeki bilgisayar sayısı [1] | 1 |
| Şekil 2.1 | TCP/IP yapısındaki protokoller [6] | 6 |
| Şekil 2.2 | QoS mimarisi [8] | 7 |
| Şekil 2.3 | FIFO, öncelikli ve sınıflandırılmış kuyruklar | 12 |
| Şekil 2.4 | Tümleştirilmiş hizmetler için referans model..... | 16 |
| Şekil 2.5 | Jeton kovanı ile trafik şekillendirme birimi..... | 18 |
| Şekil 2.6 | Farklılaştırılmış hizmetler mimarisi [30] | 22 |
| Şekil 2.7 | AF için kullanılacak ölçücü ve paket işaretleyicisi | 24 |
| Şekil 2.8 | EF sınıflandırıcısı ve şekillendiricisi..... | 24 |
| Şekil 2.9 | Statik hizmet anlaşması ile AF hizmetinin işleyişi | 26 |
| Şekil 2.10 | Dinamik hizmet anlaşması ile EF hizmetinin işleyişi | 27 |
| Şekil 2.11 | Uç sistemler ve yönlendiricilerde RSVP | 30 |
| Şekil 2.12 | RSVP'nin temel işleyiş şekili | 31 |
| Şekil 2.13 | RSVP iletim hattında RSVP desteklemeyen ağlar ve RSVP isteklerinin birleştirilmesi | 32 |
| Şekil 3.1 | TCP ağlarının tıkanıklık durumunda yük, iş verimi, gecikme ve güç yapısı.. | 35 |
| Şekil 3.2 | Tepkisiz UDP akışlarının tıkanıklık çöküşüne neden oluşu benzetimi [45] | 36 |
| Şekil 3.3 | Pencere tabanlı iletim yapan protokollerde kendinden-zamanlama [37] .. | 39 |
| Şekil 3.4 | TCP'nin tıkanıklık penceresinin zaman içinde değişimine bir örnek..... | 40 |
| Şekil 3.5 | RED algoritması..... | 47 |
| Şekil 3.6 | RED parametreleri | 48 |
| Şekil 3.7 | DT kuyrukların patlamalı trafiğe davranışını inceleyen benzetim sonucu | 50 |

| | |
|---|----|
| Şekil 3.8 Rastantısal düşürmeli kuyrukların patlamalı trafiğe davranışını inceleyen benzetim sonucu | 51 |
| Şekil 3.9 RED kuyrukların patlamalı trafiğe davranışını inceleyen benzetim sonucu | 52 |
| Şekil 3.10 RED, ERD ve DT patlamalı trafiği düşürme davranışları | 52 |
| Şekil 3.11 EBONE ağı – müşteri hattı için RED ve DT kullanılan günlerin hat kullanımı [63]..... | 53 |
| Şekil 3.12 EBONE ağı – müşteri hattı için RED ve DT kullanılan günlerin paket düşürme oranları [63] | 53 |
| Şekil 3.13 EBONE ağı – müşteri hattının bir haftalık hat kullanımı [63]..... | 53 |
| Şekil 3.14 EBONE ağı – müşteri hattının bir haftalık paket kaybı grafiği [63] | 54 |
| Şekil 3.15 EBONE – müşteri hattının sürekli olarak RED koşturulan Cumartesi günü hat kullanım grafiği [63] | 54 |
| Şekil 3.16 EBONE – müşteri hattının sürekli olarak RED koşturulan Cumartesi günü paket kayıp grafiği [63] | 54 |
| Şekil 3.17 RED'in patlamalı trafiğe davranışını incelemek için yapılan benzetim ağı konfigürasyonu | 55 |
| Şekil 3.18 RED ve DT kuyrukların patlamalı ve normal trafik kaynakları için paket düşürme oranları | 56 |
| Şekil 3.19 RED ve DT kuyrukların patlamalı ve normal trafik kaynakları için başarılı iletim hızları | 56 |
| Şekil 3.20 RED ve DT gecikme değişimi..... | 57 |
| Şekil 3.21 RED'in akış sayısına tepki veremediğini gösteren benzetim için ağ yapısı | 59 |
| Şekil 3.22 RED'in $max_p=0,01$ için 8 ve 64 aktif akış için kuyruk davranışı..... | 59 |
| Şekil 3.23 RED'in $max_p=0,1$ için 8 ve 64 aktif akış için kuyruk davranışı..... | 60 |
| Şekil 3.24 ARED algoritması | 61 |
| Şekil 3.25 ARED kullanıldığında değişik sayıda aktif akış için RED'in duruma uyarlanabilmesi | 61 |
| Şekil 3.26 BLUE algoritması | 63 |
| Şekil 3.27 BLUE benzetim ağı | 64 |
| Şekil 3.28 Büyük RTT'ler için RED ve BLUE verimli iletim hızlarının kayıp oranlarının kıyaslanması..... | 64 |
| Şekil 3.29 Küçük RTT'ler için RED ve BLUE verimli iletim hızlarının kayıp oranlarının kıyaslanması | 65 |

| | |
|---|----|
| Şekil 3.30 Büyük ve küçük RTT için kuyruk boyu 160 paket iken RED kuyruğu..... | 66 |
| Şekil 3.31 Büyük ve küçük RTT için kuyruk boyu 160 iken BLUE kuyruk boyu | 66 |
| Şekil 3.32 ECN kullanılmadığında yüksek RTT'li ağ yapısında RED ve BLUE iletim hızları ve kayıp oranları | 67 |
| Şekil 3.33 ECN kullanılmadığında düşük RTT'li ağ yapısında RED ve BLUE iletim hızları ve kayıp oranları | 67 |
| Şekil 3.34 ECN kullanılmadığında büyük ve küçük RTT için kuyruk boyu 160 paket iken RED kuyruğu | 67 |
| Şekil 3.35 ECN kullanılmadığında büyük ve küçük RTT için kuyruk boyu 160 paket iken BLUE kuyruğu..... | 68 |
| Şekil 3.36 BLUE'nın 1000 aktif akış için değişik kuyruk boylarında değişik parametrelere tepkisi..... | 68 |
| Şekil 4.1 DS yönlendiricisinin temel blokları [70]..... | 73 |
| Şekil 4.2 RIO parametreleri | 75 |
| Şekil 4.3 RIO algoritması | 76 |
| Şekil 4.4 RIO test ağı yapısı | 78 |
| Şekil 4.5 Örnek hizmeti sunmak için gerekli yönlendirici yapısı..... | 80 |
| Şekil 4.6 AF benzetim ağı..... | 81 |
| Şekil 4.7 20Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları.. | 82 |
| Şekil 4.8 20Mbps darboğaz üzerinden 20 kısa süreli AF bağlantısı benzetim sonuçları..... | 83 |
| Şekil 4.9 40Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları.. | 84 |
| Şekil 4.10 20Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları (WRR, AF %80, BE %20)..... | 84 |
| Şekil 4.11 40Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları (WRR, AF %60, BE %40)..... | 85 |
| Şekil 5.1 BIO algoritması | 88 |
| Şekil 5.2 Her düğümde 1 akış aktif olduğu durum için BIO ve RIO başarılı iletim hızları | 90 |
| Şekil 5.3 Her düğümde tek FTP aktif olduğu durum için BIO ve RIO kuyrukları | 91 |
| Şekil 5.4 Her düğümde tek FTP aktif olduğu durum için BIO paket işaretleme olasılığı değişimi..... | 91 |
| Şekil 5.5 Her düğümde 1 akış aktif olduğu durum için BIO ve RIO başarılı iletim hızları | 92 |

| | |
|---|-----|
| Şekil 5.6 Her düğümde 10 FTP oturumu aktif olduğu durum için BIO ve RIO kuyrukları..... | 93 |
| Şekil 5.7 Her düğümde 10 FTP aktif olduğu durum için BIO paket işaretleme olasılığı değişimi..... | 93 |
| Şekil 5.8 Her düğümde 100 akış aktif olduğu durum için BIO ve RIO başarılı iletim hızları | 94 |
| Şekil 5.9 Her düğümde 100 FTP oturumu aktif olduğu durum için BIO ve RIO kuyrukları..... | 94 |
| Şekil 5.10 Her düğümde 10 FTP aktif olduğu durum için BIO paket işaretleme olasılığı değişimi..... | 95 |
| Şekil A.1 nam örnek ekran görünüşü..... | 108 |
| Şekil A.2 Örnek bir xgraph ekranı..... | 108 |
| Şekil C.1 d_i/d_d oranı 10 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı | 111 |
| Şekil C.2 d_i/d_d oranı 5 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı | 111 |
| Şekil C.3 d_i/d_d oranı 2 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı | 111 |
| Şekil C.4 d_i/d_d oranı 1 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı | 112 |
| Şekil C.5 d_i/d_d oranı 0,5 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı | 112 |
| Şekil C.6 d_i/d_d oranı 0,2 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı | 112 |

SEMBOL LİSTESİ

| | |
|-------------|--|
| w_q | : RED ortalama kuyruk boyu hesaplaması için kuyruk ağırlığı |
| \max_p | : RED için en yüksek işaretleme olasılığı |
| \min_{th} | : RED için alt eşik değeri |
| \max_{th} | : RED için üst eşik değeri |
| Q_{ave} | : ARED için ortalama kuyruk boyu |
| α | : ARED için işaretleme olasılığını düşürme katsayısı |
| β | : ARED için işaretleme olasılığını arttırma katsayısı |
| p_m | : BLUE paket işaretleme olasılığı |
| d_i | : BLUE işaretleme olasılığını arttırma katsayısı |
| d_d | : BLUE işaretleme olasılığı düşürme katsayısı |
| $p_{in,m}$ | : BIO için profil içi paketleri işaretleme olasılığı |
| $p_{out,m}$ | : BIO için profil dışı paketleri işaretleme olasılığı |
| $d_{in,d}$ | : BIO için profil içi paket işaretleme olasılığı azaltma katsayısı |
| $d_{out,d}$ | : BIO için profil dışı paket işaretleme olasılığı azaltma katsayısı |
| $d_{in,i}$ | : BIO için profil içi paket işaretleme olasılığı arttırma katsayısı |
| $d_{out,i}$ | : BIO için profil dışı paket işaretleme olasılığı arttırma katsayısı |

TCP/IP'DE AKTİF KUYRUK YÖNETİM MEKANİZMALARI VE İNTERNET İÇİN KALİTELİ HİZMET

ÖZET

Son on yıl içinde İnternet hızlı bir şekilde büyürken, üzerindeki uygulamaların da hem sayısı arttı hem de beklentileri değişti. Bu hızlı büyüme sonucunda TCP/IP protokolündeki sorunlar açığa çıktı. Ağ verimliliğini sağlamak ve paket kayıp oranını azaltmak temel sorunlardan biri olurken, İnternet'in mimarisi bugünün ihtiyaçlarına göre yapılmadığından uygulamaların yeni beklentilerini karşılayabilmek için yeni mekanizmalara da gereksinim duyulmaktadır.

Ağdaki tıkanıklığı önlemek için uç düğümlerde uzun süredir tıkanıklık denetimi algoritmaları koşturulurken, son yıllarda bu yapılara ek olarak yönlendiricilerin de aktif kuyruk yönetimi mekanizmaları koşturmaları önerilmektedir. Tıkanıklığın erken farkedilerek uç düğümlerin bu durumdan erken haberdar edilmesini sağlayan bu kuyruk yönetim mekanizmaları geleneksel sondan düşürmeli kuyruk yapısına göre bir çok üstünlüğe sahiptir.

İnternet'te sunulan elden geldiğince hizmet yapısı yanında değişik uygulamaların değişik seviyelerde hizmetler alabilmesi için IETF tarafından Farklılaştırılmış Hizmetler yapısı geliştirilmiştir. Bu alandaki daha önceki çalışmaların aksine daha ölçeklenebilir olan bu yapıyı desteklemek için yönlendiricilerde çizelgeleme algoritmaları ile hizmet verilen kuyruklar bulunmaktadır. Kuyrukların bazılarında sondan düşürme yapısı ve aktif kuyruk yönetimi mekanizmaları uygulanırken diğerlerinde hizmet farklılaştırılması için geliştirilmiş yeni algoritmalar koşturulmaktadır. Varolan aktif kuyruk yönetimi mekanizmalarının geliştirilmesi ile oluşturulan bu yapılar tek bir kuyruk üzerinde farklı hizmetler verilebilmesini sağlamaktadır.

Bu tez çalışmasında öncelikle tıkanıklık denetimi yapıları ve aktif kuyruk yönetimi yapıları incelenerek; yapılan benzetimlerle bu yapıların temel sorunları ve bu sorunlara çözümler irdelenilmiştir. Benzetimlerde özel olarak aktif kuyruk yönetimi yapılarının, ağın ve trafiğin niteliğine göre parametre seçimi sorunları ve çok sayıda akışın aktif olduğu yönlendiricilerde paket kaybını gerekli şekilde önleyememeleri durumu üzerinde durulmuştur.

Farklılaştırılmış Hizmetler yapısında hizmet farklılaştırmasını sağlama amacına yönelik olarak kullanılan kuyruk yönetimi mekanizmaları incelenerek benzetimler ile başarımları ölçülmüştür.

Bu tez çalışmasında bunların yanısıra, yönlendiricilerde tek kuyruk üzerinden hizmet farklılaştırılması yapan ve aktif bağlantı sayısının fazla olduğu yönlendiricilerde daha az paket kaybı ile kuyruğu daha iyi bir şekilde kontrol edebilen bir kuyruk yönetimi yapısı geliştirilmiştir.

ACTIVE QUEUE MANAGEMENT MECHANISMS FOR TCP/IP AND QUALITY OF SERVICE FOR THE INTERNET

SUMMARY

In the past ten years as the Internet grew rapidly, the number and the expectations of the applications on the Internet have changed. As a result of this rapid growth, the weaknesses in TCP/IP have become increasingly apparent. As the achievement of network efficiency and the reduction of loss rate became major problems, new mechanisms are also required to meet the expectations of today's applications since the architecture of the Internet is not designed to support these kind of applications.

In order to prevent network congestion, deployment of active queue management mechanisms on gateways is recently recommended in addition to the congestion control algorithms in use. The active queue management mechanisms, which detect congestion earlier and convey notification to sources before queue overflow and packet loss, have many advantages over the traditional drop-tail queues.

IETF has developed Differentiated Services architecture to offer different levels of service to the applications, besides the best-effort service. Scheduling algorithms at gateways, which serve to different queues, are used to support this new architecture, which is more scalable than the previous developments in this area. Besides the drop-tail and active queue mechanisms, new queue management algorithms, which are developed especially for service differentiation, are used on these queues. These new algorithms are based on the existing queue management algorithms and offer service differentiation on a single queue.

In this thesis, congestion control mechanisms and active queue management algorithms are studied, and the basic problems and possible solutions of the algorithms are investigated using simulations. The parameter selection problem according to the network and traffic properties, and the problem associated with the inability to prevent packet loss for high number of active flows are specifically concerned in simulations.

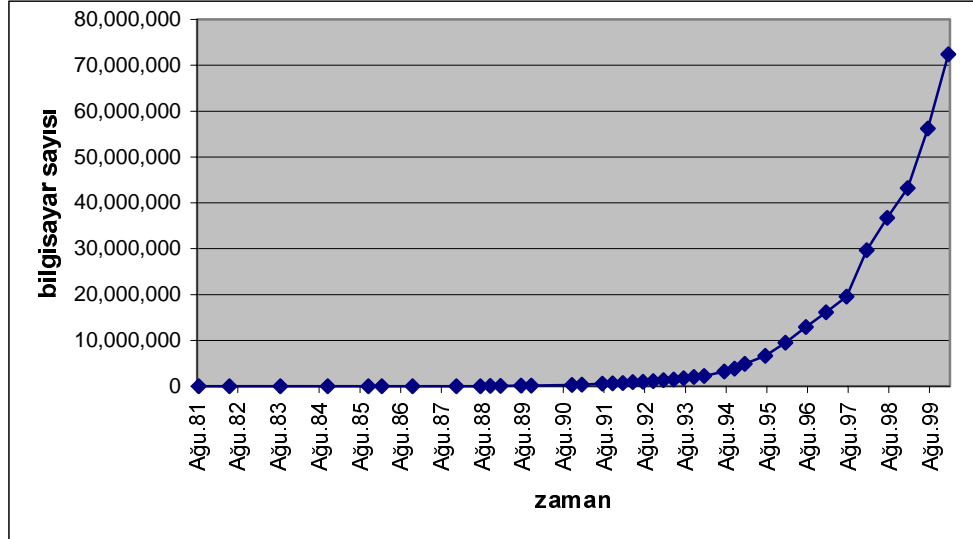
Queue management algorithms to offer service differentiation, are studied and performance of these algorithms are evaluated using simulations.

In this thesis, a new queue management algorithm, which offers service differentiation over a single queue, on gateways is developed. This new algorithm reduces the packet loss ratio, and successfully controls the queue size on the gateways that high number of flows is active on.

1 GİRİŞ

1973'te DARPA (Defense Advanced Research Projects Agency) değişik yapıdaki paket anahtarlamalı ağların birleştirilmesi için gerekli tekniklerin incelenmesi için bir araştırma projesi başlattı. Projenin amacı ağ üzerindeki bilgisayarların birbirine bağlı değişik tipteki ağlar üzerinden saydam bir şekilde haberleşebilmelerini sağlamaktı ve bu araştırma sonucunda ortaya çıkan ağlar topluluğu da Internet olarak isimlendirildi. Projenin hedeflerini gerçeklemek için TCP (Transmission Control Protocol) ve IP (Internet Protocol) protokolleri geliştirildi. 1986'da ise NSF (National Science Foundation) Internet'in temel altyapısını oluşturan NSFNET ağı oluşturuldu.

Amerikan hükümeti tarafından desteklenen Internet kullanıcıları ve altyapısı tüm dünya çapına yayılırken üssel bir büyüme kaydetti [1]. Bunun yanında başlangıçta sadece devlet kurumlarının bulunduğu Internet'te ağırlık ticari kurumlara kaydı.



Şekil 1.1 Internet üzerindeki bilgisayar sayısı [1]

Şekil 1.1'de görüldüğü gibi zaman içinde kaydettiği aşırı büyüme yanında başlangıçta sunulanların yanında çok değişik ve farklı beklentilere sahip uygulamalar da Internet üzerinden sunulmaya başlandı.

Internet'in, özellikle Web'in, yaygın kullanımı ile zaman içinde hem kullanıcı hem de üzerinde gerçekleştirilen uygulama bazında büyümesi ile TCP/IP protokolü yapısındaki sorunlar açığa çıktı. Ağ verimliliğini sağlamak ve paket kayıp oranını azaltmak temel sorunlardan biri olurken, Internet'in tasarımı yeni açığa çıkan beklentilere göre yapılmadığından uygulamaların ihtiyaçlarını karşılayabilmek için yeni mekanizmalara ihtiyaç duyulmaktadır.

Bağılantısız bir yapıya sahip olan IP protokolü üzerine inşa edilen Internet, bu bağlantısız yapının sağladığı sağlamlık ve esneklik sayesinde büyük bir hızla büyüyebilirken yine bu yapı nedeni ile sorunlarla da karşılaşmaktadır.

IP protokolü üzerinde güvenli bir iletim olanağı sağlayan TCP protokolü ağa alabileceği miktarda veri bırakmaya çalışır. Internet'in bağlantısız yapısı nedeni ile ağdan kullanılabilir kaynak hakkında bir bilgi alamayan TCP, paket kaybı ile karşılaşana dek iletim hızını arttırır. Bu sayede çok değişik hızlardaki bağlantılar üzerinde sağlıklı bir şekilde çalışabilen TCP tıkanıklık denetimi mekanizmaları üzerinde yeterli çalışma yapılmasaydı bugün bu kadar başarılı olamazdı.

Uç düğümlerin ağdaki tıkanıklığı hissetmesi ve buna tepki verebilmesini sağlayan yapılar 1986 yıllarda geliştirildi ve günümüze değin çok büyük değişiklikler geçirmedi. Ancak ağdaki tıkanıklık sorunu için bu yapılanlar yetersiz kalmaktadır ve yönlendiricilerin de tıkanıklık denetimi için aktif kuyruk yönetimi yapıları koşturmaları IETF (Internet Engineering Task Force) tarafından önerilmektedir.

TCP tarafından uygulamalara sunulan hizmet elden gelenin en fazlasını (best-effort) sunacak şekildedir. Akışlara sunulan hizmetler arasında bilinçli bir farklılaştırılma yapılmaz. Uygulamaların yeni beklentilerini karşılayabilmek için bu hizmet biçimini değiştirerek akışlara farklı tipte hizmetler sunulması gerekmektedir. Bazı ağlarda TCP/IP altındaki ATM gibi protokoller farklı tipte hizmetler sunulmasını sağlasa da bu TCP üzerindeki uygulamalara uygun biçimde sunulamamaktadır.

IETF'de bu ihtiyaçları karşılamak için çeşitli çalışmalar yapılmaktadır. Geliştirilen ilk kaliteli hizmet protokolü olan Tümlleşik Hizmetler yapısındaki sorunlar nedeni ile tam olarak yaygınlaşamamıştır. Buna alternatif sunacak daha basit protokol geliştirme çabaları sonunda ise üzerinde geliştirmeler halen devam eden Farklılaştırılmış Hizmetler yapısı ortaya çıkmıştır.

Yönlendiricilerde akışlara farklı hizmetler sunulmasını sağlamak için tıkanıklık denetimi için kullanılan aktif kuyruk yönetimi mekanizmaları ve bunların ihtiyaca göre değiştirilmiş halleri yanında çizelgeleme yapıları da kullanılmaktadır.

Bu tez çalışmasında, uç düğümlerde ve ağ üzerindeki düğümlerdeki TCP tıkanıklık denetimi yapıları ve uygulamalara farklı hizmetler sunmak için kullanılan protokoller ve yapılar incelenilmekte ve yönlendiricilerde değişik seviyelerde hizmetler sunmak için yeni bir mekanizma önerilmektedir.

Çalışmalarımızda belirli algoritmaların başarımlarını ölçümünü yapmak veya başka algoritmalar ile kıyaslamak için yapılan benzetimlerden faydalanıldı. Benzetim çalışmalarımız için kendimize özgü bir araç geliştirmek yerine bu tip çalışmalarda yaygın olarak kullanılan **NS** [2] isimli ağ benzetim aracı kullanıldı. NS değişik ağ araştırmalarında kullanılmış ve halen kullanılmakta olan bir benzetim ortamıdır. NS TCP, yönlendirme ve çoklu iletim (multicast) konularının benzetimi için C++ ve TCL ile yazılmış modüllerden oluşmaktadır.

1989'da REAL [3] benzetim aracının üzerine geliştirilmeye başlanan NS zaman içinde büyük değişikliklere uğramıştır. NS şu an VINT [4] (Virtual InterNetwork Testbed) projesinin bir parçası olarak USC/ISI (University of Southern California, Information Sciences Institute), XEROX PARC (XEROX Palo Alto Research Center), LBNL (Lawrence Berkeley National Laboratory), UCB (University of California, Berkeley) kurumlarından oluşan bir grup tarafından ortaklaşa geliştirilmektedir. VINT, mevcut ve gelecekteki protokollerin benzetimine olanak sağlayacak bir benzetim ortamı geliştirme amacına yönelik DARPA [5] tarafından desteklenen bir projedir.

Tezin genel organizasyonu şu şekildedir: Bölüm 2'de IP ağları üzerinde uygulamalara farklı hizmetler sunulma yöntemleri ve bunların yönetimi için geliştirilen protokoller incelenilmekte ve Farklılaştırılmış Hizmetler yapısı açıklanılmaktadır. Bölüm 3'te ise Internet'teki tıkanıklık sorunu açıklandıktan sonra bu sorunun çözümü için uç noktalar ve yönlendiriciler için önerilen yöntemler incelenilmektedir. Özellikle son yıllarda IETF tarafından yönlendiricilerde kullanılması önerilen ve kaliteli hizmet yapısı sunmada da kullanılan RED gibi oturmaş aktif kuyruk yönetim yapıları yanında bu alandaki yeni çalışmalar da detaylı olarak irdelenmektedir. Bölüm 4'te Farklılaştırılmış Hizmetler yapısının yönlendiricilerde desteklenilmesi için gereken yapılar incelenerek bu hizmetin başarımlarını yapılan detaylı benzetimler ile incelenilmektedir. Bölüm 5'te bu tez

alışmasında önerilen aktif kuyruk yönetimi yapısı BLUE temel alınarak geliştirilen BIO (BLUE with In and Out) algoritması sunulmaktadır. Bölüm 6'da yapılan alışmalar özetlenip bu konuda daha sonra yapılabilecek alışmalar hakkında bilgi verilmiştir.

2 IP AĞLARI İÇİN KALİTELİ HİZMET

İnternet üzerinden sunulan uygulamaların sayısı ve beklentilerinin değişimine uygun bir şekilde değişemedi. Ağa bırakılan trafik, ağ kullanıcılarının ve ağ üzerindeki uygulamaların sayısı ile birlikte arttı. Bunun yanında Moore yasasına¹ uygun şekilde işlem gücü artan kullanıcı sistemleri de artık ağa daha fazla veri bırakmaya başladılar.

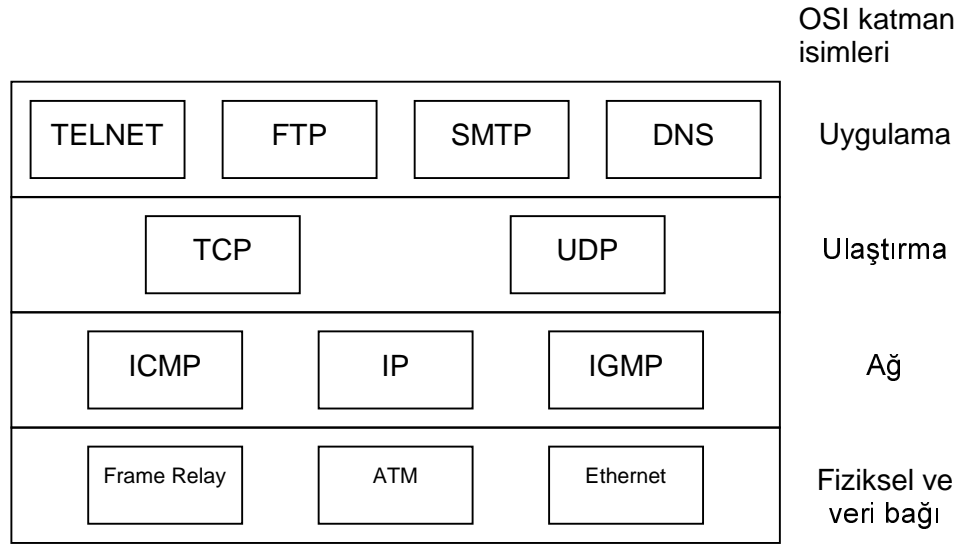
Yeni uygulamaların ihtiyaçlarını karşılamak için sadece bandgenişliğini arttırmak bir çözüm değil; çünkü İnternet üzerindeki trafik artarken özellikleri de değişti. Yeni uygulamalar yeni hizmet tiplerine ihtiyaç duymaya başladılar. İnternet mimarisi ise bu yeni uygulamaların ihtiyaçlarına paralel bir değişim göstermedi.

Bunların yanısıra uygulamaların özellikleri de değişmeye başladı. Yeni İnternet uygulamalarından bir bölümü çoğul-ortam (multimedia) uygulamalarıdır. Bunlar yüksek bandgenişliği isterlerken, bir diğer grup uygulamaların ise kesin zamanlama ihtiyaçları vardır. Bu nedenle artık *akılsız ağın* bir miktar *zeka* kazanması gerekmektedir.

İnternet, geliştirilmesi DARPA tarafından desteklenen TCP/IP protokolleri üzerine inşa edilmiştir. **Şekil 2.1**'de görüldüğü gibi telnet, ftp gibi kullanıcı uygulamaları ulaştırma katmanında yeralan TCP ve UDP (User Datagram Protocol) protokolleri üzerinde yeralmaktadır [6]. ATM, Frame-Relay ve Ethernet gibi değişik fiziksel seviye protokolleri üzerinde bulunan IP protokolü ulaştırma seviyesi protokollerine uçtan-uca bağlantısız bir iletim olanağı sunar.

Bu bölümde öncelikle kaliteli hizmet kavramı açılarak IP ağları üzerindeki düğümlerin farklı tiplerde hizmetler sunabilmesi için alternatifler incelenecektir. Daha sonra ise ağ düğümlerinin sunduğu farklı seviyelerdeki hizmetleri organize ederek uçtan-uca kaliteli hizmet sunulmasını sağlayan protokoller açıklanacaktır.

¹ Intel kurucularından Gordon Moore 1965'de her 18 ayda bir işlemcilerin transistör sayısının iki katına çıkacağını öngörmüştü. Moore'un bu öngörüsü bu güne dek gerçekleşti.



Şekil 2.1 TCP/IP yapısındaki protokoller [6]

2.1 Kaliteli Hizmet

Kalite, verinin iletiminin güvenli veya normalden daha iyi bir şekilde yapılmasıdır. Hizmet ise, ağ kullanıcılarına sunulan uçtan uca iletişim veya istemci-sunucu uygulamaları gibi yapılardır. Kaliteli hizmet (Quality of Service, QoS) ise, kullanıcıların bazı trafikler için daha iyi hizmet alabilmesini sağlayan, trafik ve hizmet tiplerinin ayırdedilebilmesi yeteneğidir [7]. Bir ağda QoS sağlanması için, ağdaki tüm düğümlerin ve tüm iletişim katmanlarının desteği gerekir. Verilen QoS garantisi ancak iki uç arasındaki en zayıf birim kadar iyidir.

Ek bandgenişliği sağlamayan QoS, ancak uygulama ihtiyaçları ve ağ yönetim politikalarına göre bandgenişliğinin kullanımını yönetir. Ayrıca aynı kaynağın farklı düğümlerce paylaşımı yapıldığında sunulabilecek hizmet kalitesinde tutarlılık sağlanamayacağından, belirli bir seviyede hizmeti garanti edebilmek için kaynakların rezervasyonu gereklidir.

2.1.1 Hizmet kalitesi

Sunulan hizmetin kalitesi, gecikme (delay), gecikme değişimi (jitter), bandgenişliği ve güvenilirlik (reliability) parametreleri ile belirlenir.

Gecikme, paketin yollayandan çıktıktan sonra alıcıya ulaşana dek geçen zamandır. TCP protokolünde yollayıcı iletim oranını ağdan geri dönen bilgiye göre ayarlar [37].

Gecikme arttıkça ağdan gelen paket alındıkları ile gelen bilgi de geciktiğinden, protokol ağdaki kısa süreli dinamik değişimlere duyarsız hale gelir.

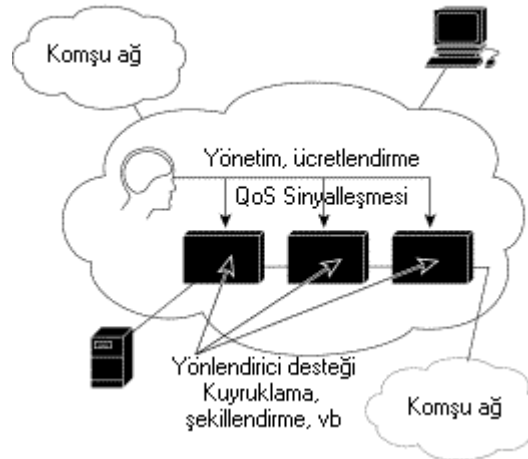
Gecikme değişimi (jitter), uçtan uca iletim gecikmesindeki değişim anlamına gelir. Bu değerin artması durumunda gerçek zamanlı çoğul-ortam uygulamalarında kullanılan kuyruğun (playback queue) büyütülmesini gerektirir.

Bandgenişliği, iki uç nokta arasında erişilebilen maksimum veri iletim hızıdır. Bu parametre ile fiziksel altyapı sınırlamasının yanında, beraber çalıştığı diğer akış²lardan kaynaklanan kaynak paylaşımı nedeni ile de değişir.

Güvenilirlik, iletim ortamının ortalama hata oranı olarak düşünülebilir. Hatalı iletimler TCP tarafından tekrarlanırken, UDP kullanan çoğul-ortam uygulamalarında hata orijinal işaretin bozulmasına neden olur.

2.2 Temel QoS Mimarisi

Temel QoS mimarisi, **Şekil 2.2**'de görülen üç ana bölümden oluşmaktadır [8]. Birinci bölüm ağ üzerindeki yönlendiricilerin farklı tiplerde hizmetler vererek sunduğu QoS desteği bölümüdür. İkinci bölüm uçtan-uca QoS desteği için kullanılacak QoS sinyalleşme teknikleri belirler. Son bölüm ise ağdaki uçtan uca trafiği kontrol etmek ve yönetmek için kullanılan QoS yönetim ve raporlama fonksiyonlarını yerine getirir.



Şekil 2.2 QoS mimarisi [8]

² Aynı kullanıcı aktivitesinden kaynaklanan ve aynı QoS'a ihtiyacı olan paketler topluluğu (flow)

2.3 Uygulamaların QoS ihtiyaçları

Gerçek-zamanlı uygulamalar paketlerin belirli zamanlarda düzenli olarak gelmesini beklerler. Asıl işaretin kaynaktan örneklenerek alıcıda gösterildiği uygulamalar bu tip uygulamalardır. Kullanıcıya eşzamanlı olarak ses veya görüntü gösterimi yapan bu tip uygulamalar için paketler geciktiklerinde artık kullanılamaz durumda olabilir. Ağda oluşan gecikme nedeni ile işaretin alıcı ile gönderen arasındaki gösterimlerinde bir zaman farkı vardır. Alıcı aldığı paketleri bir tamponda biriktirerek belirli aralıklarla gösterir. Bu nedenle, paketlerin gecikmelerindeki değişimlerin sabit olması istenmektedir. Kullanıcı tarafındaki uygulama belirlenen ortalama bir gecikmeye göre işlem yaptığından gösterim zamanından sonra gelen paketler gösterilemez.

Bu tip uygulamaların kalitesini iletim gecikmesi ve doğruluk belirler. Telefon görüşmesi gibi uygulamalar için gecikme önemli iken, play-back uygulamalar için önemsizdir. Gerçek-zamanlı uygulamalar doğruluğa olan toleranslarına göre de iki gruba ayrılmaktadır. Toleranssız uygulamalar için doğruluk son derece önemli iken tolere uygulamalar bir miktar hatayı ve gecikmeyi göze edebilir.

Uygulamaların bu tip ihtiyaçları yanında bir iletim hattının toplam kapasitesinin değişik akış gruplarına paylaşımı da önemli olabilir. Bir iletim hattı değişik kurumlarca birlikte kiralanmış olabilir ve hattaki tıkanıklık durumlarında bandgenişliğinin yapılan maddi harcamaya oransal olarak paylaştırılması isteniliyor olabilir. Bunun dışında aynı bağlantı üzerinden değişik protokoller (IP, IPX, SNA, vb) koşturan bir kurum bağlantısının protokoller arasında belirli bir oran ile paylaştırılmasını bekleyebilir. Buna benzer şekilde aynı protokol içinde farklı uygulamalara (FTP, Telnet gibi) farklı kaynak ayrılması istenebilir.

2.4 Farklı Tipte Hizmetler Sunmanın Yolları

Ağdaki trafiğe farklı hizmetler sunmanın çeşitli yolları vardır. Bu bölümde bunları, sırası ile TCP/IP katmanlarına ayırarak inceleyeceğiz.

2.4.1 Fiziksel katman çözümleri

Fiziksel seviyenin kablolamadan oluştuğu düşünüldüğünde, bu seviyede hizmet farklılaştırılmasının mümkün olamayacağı düşünülebilir. Ancak aynı hedefe doğru farklı fiziksel yollar kullanmak, farklı hizmetler sunmanın temel metotlarından birisidir.

Genel olarak yedekleme amacı ile aynı hedefe farklı fiziksel bağlantılar kurulmaktadır. Bu altyapı normal trafiğin yavaş olan bağlantıdan, öncelikli trafiğin hızlı bağlantıdan iletilmesi ile hizmet farklılaştırılmasında da kullanılabilir.

İnternet'te paketleri hedeflerine yönlendirme işlemi (routing) paketlerdeki hedef adresine bakılarak yapılır. Kaynak düğümün adresinin yol seçiminde bir etkisi bulunmadığından, ancak hedef düğüme göre hizmet kalitesi farklılaştırılabilir. Bu ise gelen ve giden trafiğin farklı seviyelerde hizmet almasına neden olur. TCP'de ağa veri bırakma hızını gelen ACK paketleri ile ayarlandığından, sonuçta gözlenen hizmet kalitesi iki yoldan daha düşük olanının kalitesine eşit olur.

2.4.2 Veri bağı katmanı çözümleri

ATM ve frame relay gibi 2. katmandaki protokoller sayesinde de farklı seviyelerde hizmet sunulması sağlanabilir.

2.4.2.1 ATM

ATM, yüksek anahtarlama yeteneği, trafik kontrol mekanizmaları, sanal devre (Virtual Circuit, VC) oluşturma ve bu VC'ler için çeşitli QoS seçeneklerine sahiptir. Ancak ATM genellikle sadece hızlı anahtarlama yeteneği nedeni ile kullanılmaktadır. Şu an ATM Forum tarafından tanımlanılmış bulunan 5 hizmet sınıfı bulunmaktadır.

Tablo 2.1 ATM QoS yapıları

| ATM Hizmet Sınıfı |
|---|
| Sabit bit miktarı (Constant bit rate, CBR) |
| Gerçek zamanlı değişken bit miktarı (Real-time Variable bit rate, rt-VBR) |
| Gerçek zamanlı olmayan değişken bit miktarı (Non-real-time VBR, nrt-VBR) |
| Uygun bit miktarı (Available bit rate, ABR) |
| Belirlenmemiş bit miktarı (Unspecified bit rate, UBR) |

ATM'in erişim protokolü olarak kullanılmasında ortaya çıkabilecek olumsuzluklar sinyalleşme ek yükü, verimsiz paketleme ve uygulama zorluklarıdır.

ATM 53 sekizliden oluşan her hücre için 5 sekizli başlık kullanmaktadır. Bunun yanında IP paketleri ATM gibi sabit uzunlukta olmadığından, bir IP paketine ilişkin ATM hücrelerinden sonuncusunda kullanılmayan bir alan kalır. Bir IP paketi 200 sekizli olarak düşünüldüğünde %10 başlık (header) yükü, %10 da şişirme (padding) yükünden toplamda %20'ye varan kayıplar oluşur. Bu da ATM'in toplam anahtarlama yeteneğinin büyük bir bölümünün bu ek yükler için harcanacağı anlamına gelmektedir.

ATM'in QoS yapısı çok çeşitli durumları kapsadığından ve mükemmeli sağlamaya yönelik olduğundan uygulamada zorluklara neden olabilecek derecede karmaşıktır. Bu karmaşıklığa karşın doğal ATM uygulamalarının sayısı çok fazla olmadığından, ATM'in tüm QoS yetenekleri tam anlamı ile kullanılamamaktadır. Alt seviyelerde kullanılan ATM protokolünün yeteneklerini kullanamayan uygulamalar nedeni ile de sadece belirli yeteneklerde (ABR, UBR gibi) yoğunlaşmaktadır.

CBR ve VBR'de önceden yapılan pazarlıklar sonunda ağın kabul edebileceği trafik parametreleri belirlenir. ABR'de ise alt ve üst limitler belirlendikten sonra zaman içinde ortamdaki tıkanıklık kontrol paketleri ile gözlenir ve iletilen veri miktarı bu iki limit arasında dalgalanabilir. UBR'da ise tıkanıklık oluştuğunda bu gruba ait paketler ilk atılacak paketler arasındadır. ABR kullanımı karmaşık iken, UBR'ın başarımı diğer bağlantı tiplerindeki trafik yoğunluğundan etkilenmektedir.

Alt seviyelerde ATM kullanılsa da, uçtan uca iletim TCP ile sağlanmaktadır. Bu ATM altyapısı ile QoS sağlamak olanaklı olsa da, uçtan uca tüm altyapının ATM olmaması nedeni ile ATM tarafından sağlanacak olan QoS'un etkileri tam anlamı ile gözlenemeyecektir.

2.4.2.2 Frame relay

Frame relay, aslen ISDN ağları için bir paket servisi olarak tasarlandı. İkinci seviyedeki hata sezme, paket tekrarlama ve akış kontrol özelliklerinin kaldırılarak bu işlemlerin 3. seviye protokolüne bırakılması ile ağ üzerindeki anahtarların ACK beklemeden, daha kısa tampon alanları ve daha hızlı bir şekilde çalışabilmelerine olanak tanır.

Frame relay sanal devrelerinin iletebileceği veri miktarını gösteren CIR (Committed Information Rate) isimli bir parametreye sahiptir. İletilecek verinin üst sınırı verilse de bu patlamalı (bursty) trafiğin oluşmasını engellemez. Gelen patlamalı trafik hattın izin verdiği ölçüde geçirilirken CIR ile belirlenen değerin üzerindeki paketlerin

gerektiğinde daha öncelikli olarak düşürülmesini sağlayacak şekilde DE (Discard Eligible) biti çekilir.

Frame relay, ağda tıkanıklık durumu oluştuğunda CIR'a uymaları için FECN (Forward Explicit Congestion Notification) ve BECN (Backward Explicit Congestion Notification) mekanizmaları ile alıcı ve yollayıcı düğümleri uyarır. Ağdaki tıkanıklık durumu giderilemezse önce CIR ile taahhüt edilen hizmet kalitesini bozmadan DE bayrağı çekilmiş bulunan paketleri; bu da yeterli olmazsa taahhüt edilen hizmet kalitesine bakmaksızın tüm paketleri atar.

Frame relay, ağda bir tıkanıklık oluştuğunda seçtiği bazı paketleri ağdan atmaktadır. Bu seçim işleminde TCP ve daha üst katmanlar etkin olmadığından uygulama açısından veri bağı katmanında frame relay kullanımının herhangi bir protokol kullanmaya göre üstünlüğü yoktur.

Uygulamanın, frame relay'in seçici atma kararına etkide bulunabilmesi ancak frame relay ağlarına geçiş sağlayan yönlendiricilerde frame relay başlığındaki DE bitinin, IP başlığındaki ilgili alana göre düzenlemesi ile mümkün olabilir.

ATM ve frame relay'in IP QoS'a katkılarının kısaca incelenmelerinden çıkan sonuç, TCP ve üst katman uygulamaların etkileşimi olmadan kullanılan ikinci katman protokolleri, yeteneklerine rağmen uçtan uca QoS desteğinde başarılı olamamaktadır.

2.4.3 Ağ ve taşıma katmanı çözümleri

Bugünün ağlarında üstün durumda bulunan protokol tartışmasız olarak IP'dir. QoS desteği verilmesi için en yaygın olan protokolün seçimi, uçtan uca sağlıklı olarak çalışan QoS sunulabilmesi ve bunun uygulama ve denetiminin daha kolay olmasını sağlar.

İnternet'te üç farklı sınıfta trafik taşınmaktadır [9]. Uzun süreli ve uyarlanabilir trafik ilk sınıftır. Uzun süren TCP akışlarının oluşturduğu bu sınıf toplam paket trafiğinin %1'ini oluştururken, toplam iletilen verinin %20'sini oluşturmaktadır. İkinci sınıf trafik ise ilk sınıfın kısıtlanılmış halidir. Web bağlantılarının oluşturduğu bu grupta bağlantı süresi oldukça kısadır ve bu süre içinde ağda oluşan değişimlere cevap verilemez. Bu trafik tipi de toplam paketlerin %60'ını oluşturmaktadır. Son sınıf ise uyarlanırlı olmayan, yani akış hızı uygulama tarafından kontrol edilen tek yönlü akışlardır. UDP akışlarının oluşturduğu bu trafik tipi de toplam veri trafiğinin %5'ini oluşturmaktadır.

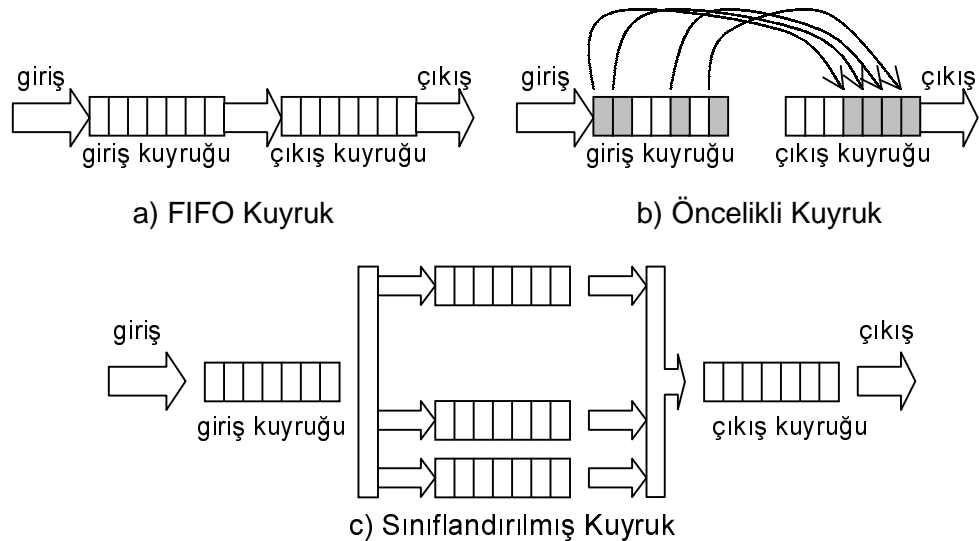
Bu üç farklı sınıftaki trafiği kontrol altında tutarak, ağda farklılaştırılmış hizmetler sunabilmek için farklı yaklaşımlar kullanılmalıdır. Ağın ilk sınıftaki akışlara yoldaki tıkanıklık hakkında bilgiler sunması gereklidir. İkinci tipteki akışlara ise iletimini kısıtlayıcı bir müdahalede bulunmak anlamlı olmaz. Çünkü bu müdahale ile akışın aktif kalma süresi uzayacak ve yapılacak sinyalleşme paketleri nedeni ile ağ kapasitesinin kullanılmasında veya tıkanıklık durumlarında pek bir fayda sağlanamamasına karşın akışın kalitesi bozulacaktır.

Üçüncü sınıftaki akışların kullanacağı kaynakların diğer tipteki trafiklerin kaynaklarından ayrılması ve başlangıçta yapılacak müzakereler ile akışın ağa kabul edilip edilmeyeceğinin belirlenilmesi gerekir.

Yönlendiricilerin akışlara farklı hizmet vermek için kullanabilecekleri en temel yol kuyruk yönetimi ve paket çizelgelemesi yapılarıdır. Farklı kuyruk tipleri **Şekil 2.3**'de gösterilmiştir.

2.4.3.1 FIFO kuyruklar

FIFO tipi kuyruklar tüm paketlere eş hizmet verilen tek öncelikli yapılarda kullanılmaktadır. Paketlerin ağda tıkanıklık yaşamadıkları durumda FIFO kuyruk yapıları yönlendiricilerde çok az işlemci gücü gerektirdiğinden oldukça işlevseldir. Ancak bu haliyle akışlara farklılaştırılmış hizmetler sunmaları mümkün değildir.



Şekil 2.3 FIFO, öncelikli ve sınıflandırılmış kuyruklar

Bunların yanısıra FIFO kuyruk yapısı ile uç düğümlerin iletim protokollerine tıkanıklık konusunda destek verilememektedir. Basitliği nedeni ile tercih edilen bu kuyruklar

yerine tıkanıklık konusundaki desteklerinden dolayı ileride açıklanacak olan *aktif kuyruk yönetim mekanizmaları* kullanımı önerilmektedir.

2.4.3.2 Öncelikli kuyruklar

Öncelikli hizmet alması gereken paketlerin belirlenerek kuyrukta daha önlere yerleştirilmesi esasına dayanan öncelikli kuyruklar, yoğun işlemci gücü gereksinimleri nedeni ile pratikte kullanımları oldukça verimsiz bir hal alır. Ayrıca mutlak bir öncelik tahsisi yapıldığından daha düşük öncelikli paketlerin hiç hizmet alamaması da olasıdır. Bu nedenle öncelikli kuyrukların kullanımı ancak ağa giren yüksek öncelikli paket miktarının bir kabul kontrol algoritması ile sınırlandırıldığı durumlarda anlamlı olabilir.

2.4.3.3 Sınıflandırılmış kuyruklar

Sınıflandırılmış kuyruklar [10] (Class Based Queuing, CBQ), öncelikli kuyruklara benzemektedir. Değişik önceliklerde farklı çıkış kuyrukları bulunan yapıda, her seferde kuyruklarından alınacak veri miktarı parametreler ile belirlenebildiğinden ve kuyruklara belirli ağırlıklara göre hizmet verildiğinden öncelikli kuyrukların aksine düşük öncelikli akışların hizmet alamaması engellenmiş olur. Alınan her paket hangi sınıfa ait olduğunun anlaşılması için bir dizi kontrolden geçtiğinden yönlendiricide işlemciye bir ek yük oluşturulmaktadır ve bu da CBQ'nun yüksek hızlarda paket kabul eden ağ merkezlerindeki yönlendiricilerde kullanımını engellemektedir [7].

2.4.3.4 Ağırlıklı adil kuyruk

Ağırlıklı adil kuyruk (Weighted Fair Queuing, WFQ) yapısında veri miktarı daha az olan akışların daha öncelikli olarak hizmet alması sağlanmaktadır. Böylece yüksek miktarda veri ileten akışların ağ kaynaklarını tüketerek diğer akışların hizmet almalarını engellemesi durumu engellenilmiş olur. Fakat bu yapıda da paketler üzerinde detaylı kontroller yapıldığından ölçeklendirme sorunları bulunmaktadır.

2.4.3.5 Ağırlıklı round-robin ve eksik ağırlıklı round-robin algoritmaları

İdeal yaklaşım her akışa bağıl bir ağırlık vermek ve yönlendiricilerde her akış için ayrı bir kuyruk kullanarak çizelgeleyicinin tüm kuyruklara bit bazında round-robin olarak akışların ağırlıklarına göre hizmet vermesidir.

Çeşitli çizelgeleme teknikleri GİP (Genelleştirilmiş İşlemci Paylaşımı) [11] ideal durumuna yakınsamaya çalışır. Temel yaklaşım paketleri IP başlığındaki TOS (Type of Service) alanına göre kuyruklara ayırmak ve bu kuyruklara ağırlıklı round-robin yapısıyla hizmet vermektir. Tüm paketlerin eş uzunlukta olmaması nedeniyle bu yapı GİP'a çok yakınsayamaz.

Round-robin algoritmasının değiştirilmiş bir hali ise (Deficit Round Robin, DRR) bir akışın paket uzunlukları nedeni ile bir adımda kullanamadığı hizmeti bir dahaki adımda ona sunarak uzun vadede GİP'e oldukça yakınsar.

2.5 QoS Protokolleri

Ağ üzerindeki düğümlerde hizmet farklılaştırılması, uygulamaların değişik seviyelerde hizmet alması için yeterli değildir. Uygulamaların ağa ne tipte hizmet istediklerini bildirmeleri için bir ortam da kurulmalıdır. Bu tip bir ortamı oluşturmak için izlenebilecek üç farklı yol vardır:

1. İlk seçenekte akışın yolu üzerindeki yönlendiriciler yerel bazı kurallara göre paket başlıklarında yaptıkları inceleme sonucunda pakete nasıl bir hizmet sunacaklarına kendileri karar verirler. Yönlendiriciler bunu sağlamak için her servis tipi için ayrı kuyruklar bulundururlar. Bu yöntem ancak düğüme gelen toplam trafik miktarı çıkış arabiriminin bandgenişliğinden fazla olduğunda hizmet farkı sunabilir. Ağdaki trafik miktarı az ise trafik sınıfları arasında belirgin bir kalite farkı gözlenemeyecektir. Bu yöntem ağdaki her yönlendiricide aynı kuralların kurulması gerektiğinden ve paket başlıklarında yapılan, karmaşık olabilen, kontrollerin neden olduğu işlem yükü nedeni ile ölçeklenebilir değildir.
2. İkinci yöntemde ise bir dizi yönlendiricide akışlara ilişkin durum bilgileri tutulur ve paketler hangi akışlara ait olduklarını gösterecek şekilde işaretlenirler. Bu yaklaşımda veri paketleri, akışın özelliklerini belirten bir paketin ağa bırakılarak yönlendiricilerde akışlara ilişkin durum oluşturulmasından sonra ağa bırakılır.
3. İlk iki yöntemin olumsuz yönlerini gideren üçüncü yaklaşımda, ağın girişinde bulunan yönlendiriciler paket başlıklarında yaptıkları kontroller sonucunda paketin alması gereken servisi belirlerler ve buna uygun bir şekilde paket

başlıındaki bir alanı doldururlar. Ağdaki diğer düğümler ise sadece bu alana bakarak pakete ne şekilde hizmet verileceğine karar verirler.

Bu yaklaşımların ikincisi farklılaştırılmış hizmetler (Differentiated Services, DS), üçüncüsü ise tümleşik hizmetler (Integrated Services, IS) protokolleri tarafından esas alınmıştır.

2.5.1 Tümleşik hizmetler

Internet'te değişik hizmetler sunma arayışları sonucu 1994'te IETF Tümleşik Hizmetler Çalışma Grubu [12] kuruldu. Bu grup tarafından geliştirilen tümleşik hizmetler yapısı, elden geldiğince (best-effort) hizmet yanında garantili ve kontrollü hizmetlerin sunulması için gerekli altyapıyı oluşturmaktadır.

Bu yapıda yönlendiricilerin değişik trafik akışlarına değişik hizmetler sunabilmesi için kaynakların rezerve edilmesi ve bunun için de yönlendiricilerde akışlara ilişkin durum bilgilerinin tutulmasının bir gereklilik olduğu belirtilir [13]. Yönlendiricilerde akışlara ilişkin durum bilgisinin tutulması, Internet'in tüm durum bilgilerinin uç düğümlerde tutulduğu mevcut yapısına göre büyük bir değişiklik anlamına gelmektedir. IS yapısının temel taşlarından bir diğeri ise trafiğin ağa kabul kontrolüdür.

IS, toleranssız uygulamalar için oluşabilecek gecikme için bir üst sınır belirleyebilen garantili hizmeti (Guaranteed Service, GS) [14] sunmaktadır. Tolere edilebilir uygulamalar için ise güvenli, ancak GS kadar güvenli olmayan kontrollü hizmeti (Controlled-Load Service, CS) [15] sunulmaktadır.

Elastik uygulamalarda ise paketlerin geliş süresinin paketin kullanılabilirliğinde bir etkisi yoktur. Bunlar içinse değişik önceliklerdeki "best-effort" hizmet kullanılır.

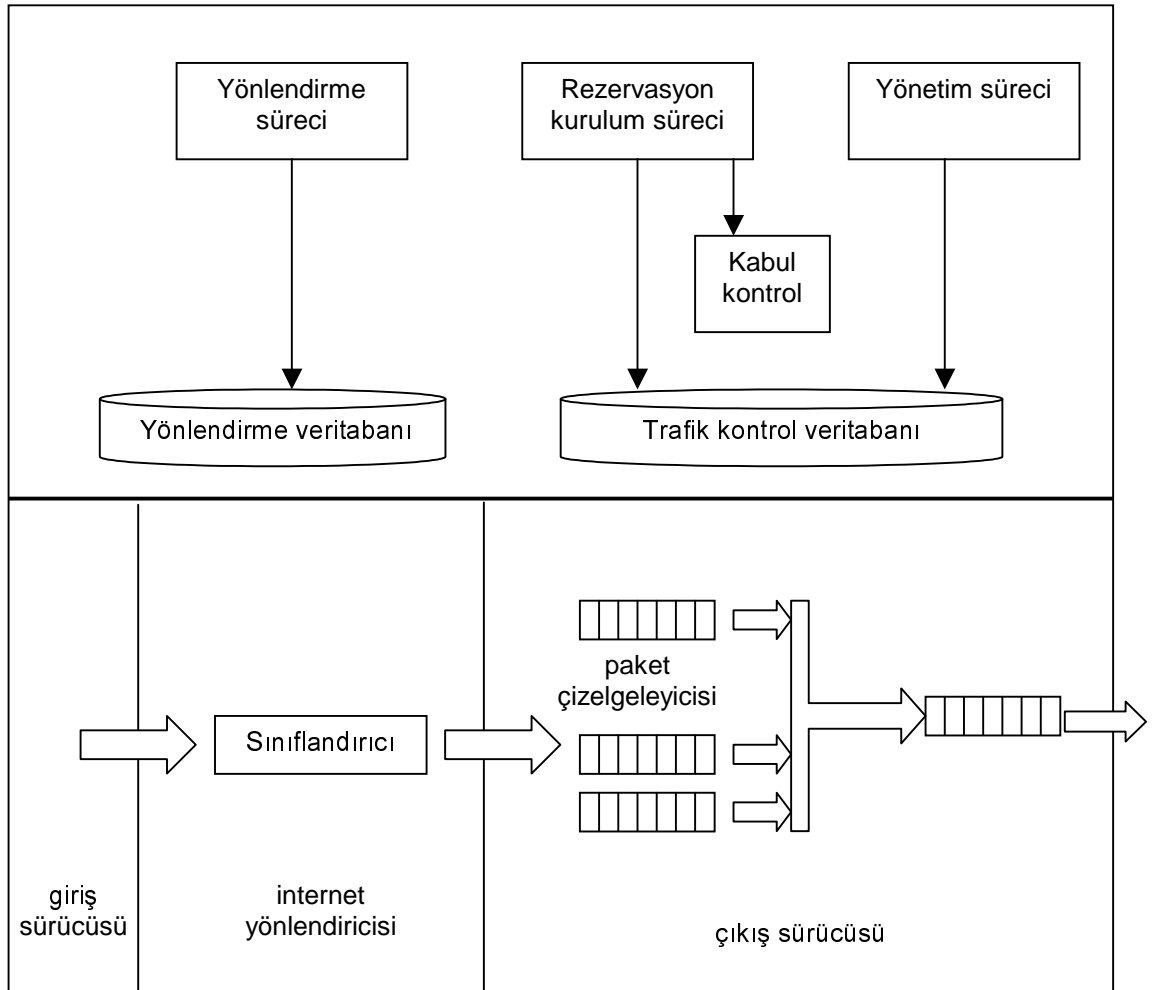
IS, her akışa kaynak paylaşımı yapabildiği gibi bir iletim hattının toplam bandgenişliğinin değişik akış gruplarına nasıl paylaştırılacağı da belirleyebilir. Gerçek-zaman taahhütlerinin sağlanabilmesi için gereken kabul kontrol algoritması belirli grupların kendi paylarını aşmaması için burada da kullanılmalıdır.

Bu bölümde ilk olarak bir IS yönlendiricisinde bulunması gereken temel birimleri açıklayan bir referans uygulama verilmektedir. Daha sonraki iki bölümde ise IS tarafından sunulabilen iki farklı hizmet olan GS ve CS açıklanılmaktadır.

2.5.1.1 Referans uygulama

IS'de kullanılacak paket çizelgeleyici, kabul kontrol, sınıflandırıcı, rezervasyon kurulum protokolü bileşenlerinden oluşan bir IS yönlendiricisinin genel yapısı [13]'te verilmiştir.

Şekil 2.4'te yönlendiricinin hem veri iletim hem de yönetim blokları verilmiştir. Paketler girişte sınıflandırıldıktan sonra gerekli kuyruklara yerleştirilir ve kuyruklara da belirli bir yapıda hizmet verilir. Sınıflandırma, kuyruğa yerleştirme gibi işlemlerde ise yönetim bloğundaki yapılar etkin olmaktadır.



Şekil 2.4 Tümlleştirilmiş hizmetler için referans model

Paket çizelgeleyici, çeşitli kuyruklar ve çizelgeleme yapıları ile değişik paket akışlarının yönlendirilmesini yönetir. Paketlerin değişik seviyelerde hizmet almasının sağlandığı bu bileşen paketlerin kuyruklandığı çıkış arayüzlerinde koşturmaktadır ve ikinci seviyede işlem gerçekleştirmektedir.

ATM veya frame-relay gibi bandgeniřlięi rezerve edilmesi yeteneęine sahip olan veri baęı protokollerinde protokole özel iřlemler yapılması gerekebilir [16, 17, 18]. rnek olarak ATM iin CBR iletimi yapacak bir VC kurulabilir. Bandgeniřlięi ayırma yeteneęi bulunmayan veri baęı protokollerinin kullanılması durumunda ncelikli kuyruklar veya aęırlıklı round-robin tipi kuyruklar kullanılan bir paket izelgeleyicisi kullanılabilir. Paket izelgelenmesini veya kabul kontrol algoritmasını kontrol etmek iin istatistik elde eden tahmin birimi de izelgeleyicinin bir parası olarak grlebilir.

Sınıflandırıcı, akıřları alacakları hizmet sınıflarına gre ayırır. Bir telefon grřmesi bir sınıf olabilirken, bir řirkete ait tm paketler de bařka bir sınıf olabilir. Paketlerin dahil oldukları sınıflar her ynlendiricide farklı olabilir. Aęın kenarlarına yakın ynlendiricilerde farklı sınıflarda bulunan birok paket, aę merkezindeki ynlendiricilerde aynı sınıfa dahil olacak řekilde birleřtirilebilir.

Akıřların sınıflandırılmasında paket bařlıklarındaki kaynak ve hedef adresleri ve port numaraları gibi alanların kontrol edilmesi veya IPv6'daki akıř tanımlayıcısı alanı kullanılabilir.

Kabul kontrol birimi ise yeni bir akıřın nceki garantileri etkilemeden aęa kabul edilip edilemeyeceęini belirleyen algoritmayı kořturur. Bu algoritma akıřın geeceęi her ynlendiricide kořmalıdır. Kabul kontrol algoritmalarında geleneksel yaklařım, hizmet verilen tm akıřlara iliřkin bilgilerin saklanarak en kt durum gznne alınarak yapılan bir hesap sonucu karar verilmesidir. Ancak son zamanlarda en kt durumun gznne alınması yerine, akıřların kullanım istatistikleri temel alınarak yapılan hesapların kullanılması da ne srlmektedir [15].

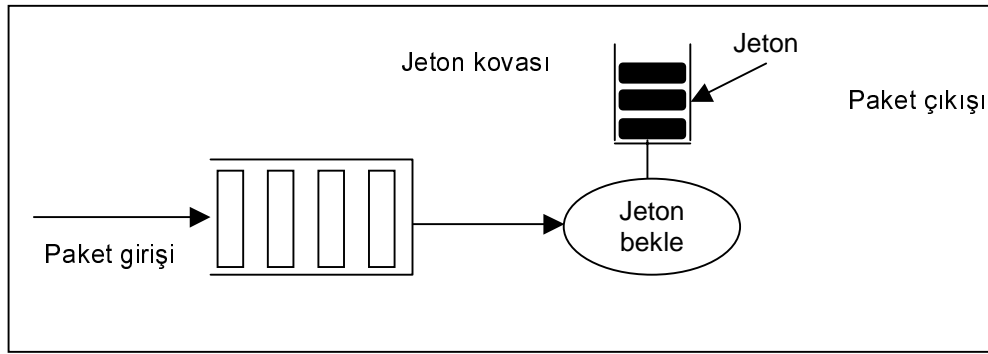
Son bileřen ise u noktalarda ve ynlendiricilerde akıřlara iliřkin durum bilgilerinin oluřturulmasında kullanılan rezervasyon kurulum protokoldr (RSVP , Kaynak Rezervasyon Protokol). Kaynak uygulamanın, bekledięi hizmet kalitesini belirtmek iin kullandığı bilgiler rezervasyon protokol ile tařınır ve kabul kontrol birimine sunulur. Akıř aęa kabul edildięinde saęladıęı bu bilgiler paket izelgeleyicisini dzenlemek iin kullanılır. İhtiyacı karřılamak iin yapılacak kaynak ayırımlarında alttaki katmanın yetenekleri de nemlidir. Eęer alttaki katman ATM gibi QoS desteęi sunuyorsa baęlantı katmanı, baę katmanında gerekli ayarları yapmalıdır. Altteki katmanın bu tip yetenekleri yok ise QoS desteęinde ynlendiricinin paket izelgeleyicisi kullanılır.

řekil 2.4'te de grldę zere ynlendiricilerde IS desteęi sunmak iin byk ekler yapılması gerekmektedir.

2.5.1.2 Garantili hizmet

Garantili hizmet (GS) [14], paketlerin başlangıçta verdikleri trafik profili içinde kaldıkları sürece, *garanti edilen süre içinde* ve *herhangi bir kayba uğramadan* hedeflerine ulaştırılacağını garantiler. GS gecikme farkını minimize etmeye çalışmaz, sadece maksimum gecikmeyi kontrol eder. Bu seviyede bir garanti verilebilmesi için paketin yolu üzerindeki tüm düğümlerin GS desteği sunması gerekmektedir.

GS desteği sunan bir düğüm bir akışa diğer akışlardan hiçbir şekilde etkilenmeyecek şekilde sabit R iletim hızını sunar. Sıvı modeli olarak da adlandırılan bu model bandgeniřlięi R olan bir iletim hattını benzetir. GS, kova hızı (r) ve kova derinlięi (b) parametreleri ile verilen bir jetonlu kova modeline uyan bir akışın alacaęı gecikmenin en fazla b/R olabileceęi kuralını temel almaktadır.



Şekil 2.5 Jeton kovası ile trafik şekillendirme birimi

Sıvı modelinde b/R olarak verilen gecikmeyi gerçek dünyaya aktarabilmek için bu gecikmeyi $b/R + C/R + D$ olarak deęiřtirmek gerekir. Burada C akış hızına baęlı olan hata terimidir. Bu tip bir hata terimine örnek olarak gelen IP paketlerini ATM aęında iletmek için paketlerin ATM hücrelerine bölmekten kaynaklanan gecikme verilebilir. D ise akış hızından baęımsız olan hata terimidir. C terimi byte, D ise mikrosaniye birimindedir.

Tüm iletim yolu üzerindeki düğümlerden kaynaklanan C ve D terimleri toplanarak C_{tot} , D_{tot} ve C_{sum} , D_{sum} deęerleri bulunur. Bu parametreler paketlerin alacakları gecikmenin üst limitini ve akışa aę düğümlerinde ayrılacak tampon alan uzunluęunu hesaplamada kullanılır.

Paketlerin GS alabilmesi kararı kabul kontrol algoritmaları sonucunda verilir. Aę sınırında akışın trafik profiline uygunluęu sınanır (policing). Bu sınamada akışın verilen jetonlu kova parametrelerine uygunluęu test edilir. Profil dıřı paketler normal

hizmet alırlar. Verilen jetonlu kova parametreleri gereğince gönderilen veri miktarı $M + \min[pT, rT + b - M]$ değerini aşamaz. Bu limiti aşan paketler profil dışı olarak işaretlenir.

Ağ üzerinde gerekli yerlerde ise akışa ilişkin paketlerin şekillendirmesi (reshaping) yapılır. Ağ içinde akışın verilen profile uygunluğunu sınamak anlamlı değildir. Çünkü ağdaki kuyruklar nedeni ile ağa girdiğinde profile uygun olan bir akış zamanla profile uyumsuz hale gelebilir. Bu nedenle ağ içinde akış yapılan şekillendirmeler ile T_{Spec} 'e uygun hale getirilir. Trafik yeniden şekillendirmesi için bir tampon alan ile bir jeton kovası kullanmak yeterlidir.

2.5.1.3 Kontrollü hizmet

Kontrollü hizmet (Controlled-load Service, CS) [15], bir trafik akışına ağın yüklü olmadığı durumlarda alacağı hizmet kalitesini sunar ve bunun için kabul kontrol algoritmalarını da kullanır. Bu hizmet sayesinde trafik kaynakları gönderilen paketlerinin *büyük bir yüzdesinin* başarılı olarak karşıya ulaştırılacağını ve paketlerin karşılaştıkları gecikmenin belirtilen üst gecikme sınırını geçmeyeceğini varsayabilirler.

Bu nitelikte bir hizmet sağlanabilmesi için trafik kaynakları hedefe kadar olan ağ düğümlerine iletecekleri trafik miktarının bir tahminini (T_{Spec}) verirler. Bunun karşılığında CS, ağ düğümlerinin bu nitelikte trafiğe gerekli hizmeti verebilmek için yeterli kaynağın her zaman kullanılabilir olacağını garantiler.

Bu tipte bir garanti verilebilmesi için ağa giren paketlerin CS istekleri için kabul kontrol algoritmaları koşturur. Ağa ancak vaad edilen garantiler sağlanabildiği sürece yeni akış kabul edilir. Normal trafik koşullarında paketlerin karşı tarafa iletilmesi garantili olmadığından, bu CS'de de garantilenmemektedir.

Kabul kontrol algoritmalarının yaptıkları hesaplamalarda akışın iletme başlarken T_{Spec} parametresi ile verilen üst iletim hızı kullanabileceği gibi akışların anlık olarak yaptıkları iletim miktarı da kullanılabilir. Bir akış için 100Kbps'lik bir iletim kapasitesi talebinde bulunduğu halde genellikle bunun sadece 70Kbps'i kullanılıyor olabilir. Hesaplamalarda anlık iletim hızının kullanıldığı durumlarda T_{Spec} parametrelerine göre mümkün olandan daha fazla akışın ağa kabul edilmesi de mümkün olabilir. Kabul kontrol algoritmalarında T_{Spec} ile verilen parametrelerin kullanılması, düğüm üzerindeki CS akışı sayısının az olduğu durumlarda daha uygundur. Çünkü bu

durumda akışların toplamı büyük ölçüde değişken olmaktadır ve bu da istatistiksel bir çoğullama yapılması mümkün olamamaktadır.

CS, daha önce de bahsedildiği gibi normal ağ koşullarında sağlıklı çalışabilen ancak ağın yüklü olduğu durumlarda başarımlarının yetersiz olduğu uyarlanabilir gerçek-zamanlı uygulamalar için elverişlidir.

CS'e oluşturulacak trafik ile ilgili bilgilerin verilmesi için T_{Spec} kullanılır. Bu T_{Spec} ile jeton-kovası parametreleri ile trafiğin tepe değeri (p), ölçülen minimum birim (m) ve maksimum paket büyüklüğü (M) parametreleri verilir. Jeton kovası için ise kova hızı (r) ve kova derinliği (b) bilgileri verilir.

Diğer hizmetler ile uyumluluk için verilen p parametresi şu an için CS yapısında kullanılmamaktadır. Büyüklüğü m değerinden küçük tüm paketler jeton kovasında ölçülürken m uzunluğunda sayılır.

Ağ düğümleri, trafik akışlarına ilk başta T_{Spec} ile verdikleri trafik profiline uydukları sürece belirli bir hizmet sunarlar.

Verilen parametrelere göre bir trafik akışı T süresi içinde $(rT + b)$ miktarından daha fazla trafik iletemez. Buna göre $(rT + b)$ sınırını aşan paketler profil dışı olarak saptanır. Buna ek olarak M değerinden ve bağlantının MTU'sundan daha büyük paketler de profil dışıdır. Ağ düğümleri kendilerine gelen profil dışı trafiği, trafik profillerine uyan diğer CS alan akışları ve diğer normal trafik akışlarını etkilemeyecek şekilde normal trafikmişçesine iletmeye devam etmelidir.

Profil dışı trafiğin oluşması normal olmasa da bazı durumlarda bu tip paketler ile karşılaşılabilir. Bu nedenle ağdaki her düğüm, kendinden önceki düğümlerin trafiğin profile uygunluğunu kontrol etmiş olmasına güvenmeden paketlerin profil içinde olup olmadığını kontrol etmelidir.

Ağ düğümlerinde CS desteği sunmak için iki farklı öncelik seviyesinde kuyruk kullanılması yeterlidir. Düşük öncelikli kuyruk normal trafik için kullanılabilirken, üzerindeki veri miktarı kabul kontrol algoritmaları ile sınırlı tutulan yüksek öncelikli kuyruk da CS paketleri için kullanılabilir.

Yönlendiricilerde WFQ veya CBQ gibi yapılar var ise, CS akışlarını uygun sınıflara eşleyerek mevcut yapılar da kullanılabilir. Bu durumda her CS akışını farklı kuyruklara yerleştirip kuyruk ağırlıkları uygun olarak belirlendiğinde, trafik çizelgeleyicisi aynı zamanda trafik denetimi işlemini de gerçekleştirmiş olur.

2.5.2 Farklılaştırılmış Hizmetler (Differentiated Services, DS)

Ölçeklenebilirlik sorunları nedeni ile *ağların merkezi düğümlerinde kullanılamayan* IS'e alternatif olarak açığa çıkan DS, daha basit ve daha az yetenekli ancak uygulaması daha kolay ve ölçeklenebilirliği daha fazladır. Bu ölçeklenebilirlik, DS yapısındaki karmaşıklığın çok büyük miktarda verinin işlendiği ağın merkezinin dışına, sınırlardaki yönlendiricilere bırakılması ile gerçekleştirilir [19]. DS, paketlerin göreceği gecikmeye ilişkin bir garanti vermez, ancak belirli paketlerin diğerlerinden daha iyi veya kötü hizmet almasını sağlamaktadır.

IPv4 (IP sürüm 4) başlığında nispi hizmet tercihini belirten TOS (Type of Service) alanı bulunmaktadır. IS ve DS gibi çalışmalardan önce de, uç düğümlerdeki uygulamalar bu alan yardımı ile paketin ağda almasını istediği hizmeti belirtebiliyordu. DS bu modelin daha iyileştirilmiş halidir [20].

Bir DS düğümünün, DS akış gruplarına verdiği yönlendirme hizmetinin niteliğine düğüm davranışı (Per-Hop Behaviour, PHB) denilir. Bir PHB en basitinden, bandgenişliğinin belirli bir kısmının bir DS grubu tarafından kullanılmasını taahhüt ediyor olabilir. Bir PHB'nin bir düğümde uygulanması çeşitli kuyruk ve bandgenişliği yönetimi algoritmalarının kullanılması ile olmaktadır.

Farklı PHB'lerde alınan hizmetin görünür şekilde farklı olması ancak, değişik trafik gruplarının sınırlı kaynaklar için yarıştığı durumlarda gözlenebilir. Ağın yüklü olmadığı durumlarda ise bir fark gözlenemez.

PHB'ler diğer PHB'lerle kaynak paylaşımındaki önceliklerine veya gecikme, kayıp gibi diğer trafik karakteristiklerinin bağlı değerlerine göre tanımlanabilir.

Paketin alacağı PHB, paket başlığındaki DS alanının değerine göre kararlaştırılır. DS alanı olarak IPv4'te [21] ile belirlenen TOS (Type of service) alanı kullanılırken, IPv6'da [22] ise trafik sınıfı alanı kullanılmaktadır. DS alanı aynı olan paketlerin tümü aynı DS grubuna aittir ve ağ düğümlerinde aynı PHB'ye tabidirler.

Önerilmiş olan başka PHB'ler (LBE [23], Dropping EF [24], PHB-i [25], bDelay [26]) bulunsa da, şu an için standartlaşmış garanti-edilen (Assured Forwarding (AF) PHB) PHB [27] ve çabuklaştırılmış (Expedited Forwarding (EF) PHB) PHB [28] olmak üzere iki PHB bulunmaktadır.

Bu bölümde öncelikle DS mimarisinin yapısı açıklanmaktadır. Sonraki iki bölümde ise DS yapısının standartlaşmış tek PHB'leri olan AF ve EF PHB'leri detaylı olarak

irdelenilmektedir. Son olarak ise, DS alanlarının başka DS alanlarından aldığı hizmeti kendi içlerinde paylaşımı için kullanılan yapılar örneklenilmektedir.

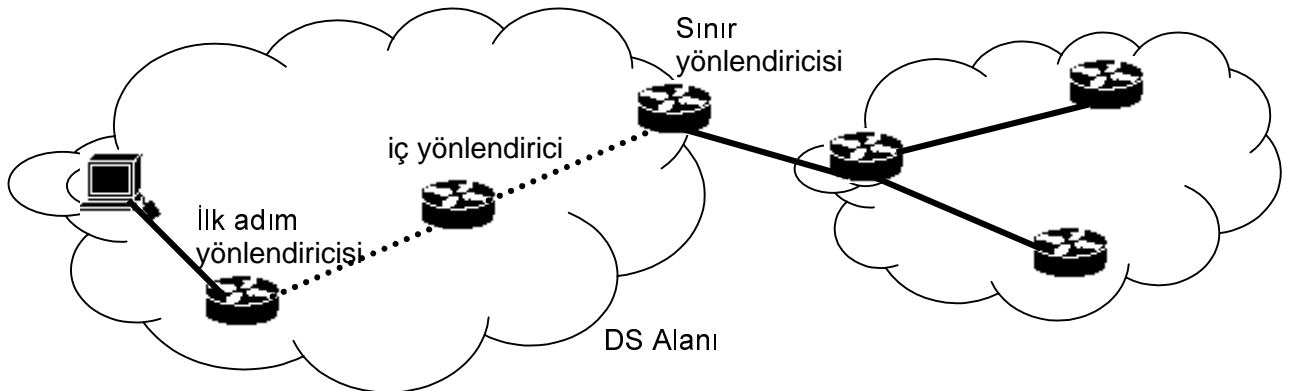
2.5.2.1 DS mimarisi

DS mimarisinin temel düğümleri ve aralarındaki ilişki **Şekil 2.6**'da görüldüğü gibi tanımlanmıştır [20]. Günümüzde ağ trafiği, kullanıcı sistemleri, şekilde görüldüğü gibi ev ve ofis ağları, kampüs ağları ve daha büyük ağlarından oluşan bir yol izlemektedir. Küçük ağlar daha büyük ağların müşterisi, hizmet alanı durumundadır.

DS mimarisi, ağı giren trafiğin ağ sınırlarında sınıflandırılarak farklı hizmetler alacak şekilde atandığı basit bir modelden oluşmaktadır. Bu basit yapıda, ağın merkezindeki yönlendiricilerde kompleks işlemler yapılmadan sadece IP paket başlığındaki DS alanına [29] göre hizmet verirler.

DS alanları, ağ içinde sınırları açıkça belirlenmiş, tüm düğümleri DS desteklemekte ve tek elden yönetilmekte olan parçalardan oluşmaktadır. DS alanlarını, DS destekleyen ve desteklemeyen alanlara bağlayan yönlendiricilere sınır yönlendiricileri ve DS alanı içinde kalan ve diğer iç ve sınır yönlendiricilere bakan yönlendiricilere iç yönlendiriciler denilmektedir [20].

Her iki tip yönlendirici de, paketlere farklılaştırılmış hizmetler verme yeteneğine sahiptir; ancak sınır yönlendiriciler ve ilk adım yönlendiriciler (bunlar da yaptıkları işler bakımından sınır yönlendiricisi olarak düşünülebilir) buna ek olarak paket sınıflandırılması, trafik denetleme ve şekillendirmesi de yapmaktadır.



Şekil 2.6 Farklılaştırılmış hizmetler mimarisi [30]

Çıkış sınır yönlendiricileri karşı alanla yapılan anlaşmaya uyumu sağlamak için trafik şekillendirmesi yaparken, giriş sınır düğümü karşı taraftan gelen paketlerin yapılan anlaşmaya uygunluğunu kontrol eder (traffic policing). İç ve sınır yönlendiricilerin her

ikisi de paket sınıflandırması yapmaktadır. Sınır yönlendiriciler paketlerin IP adresi port numarası gibi birçok alanına bakarak sınıflandırma yapar ve bu sınıflandırma sonucunda IP başlığı içindeki DS alanını uygun şekilde doldurarak paketin alması gereken hizmeti belirlerler. Bu tip sınıflandırmaya çok alanlı (Multi Field, MF) sınıflandırma denilmektedir. İç yönlendiriciler ise sadece DS alanına bakarak sınıflandırma yapmaktadırlar. Tek alan üzerinde yapılan bu sınıflandırma (Behaviour Aggregate, BA) sonucunda pakete gereken hizmet verilir. Tek alan üzerinde sınıflandırma çok daha az işlem gücü gerektirdiğinden trafiği yoğun olan merkezi ağ yönlendiricilerinde de sorunsuz bir şekilde kullanılabilir.

2.5.2.2 Garanti edilen düğüm davranışı

AF PHB grubu paketlere farklı seviyelerde yönlendirme kaynağı sunulmasını sağlar. Herbirine belirli kaynaklar (bandgenişliği ve kuyruk) ayrılmış toplam dört AF sınıfı bulunmaktadır. Her AF sınıfı içinde ise üç farklı öncelik bulunmaktadır. Bu öncelikler ağdaki bir tıkanıklık durumunda aynı AF grubu içinde bağlı olarak paketlerin düşürülme önceliğini belirler. Her AF sınıfına yönlendiricilerde belirli bir miktarda kaynak ayrılır. Her sınıfın alacağı maksimum yönlendirici kaynağı sabit kalmakla birlikte, yönlendiricinin fazlada kalan kaynaklarından kullanabilir.

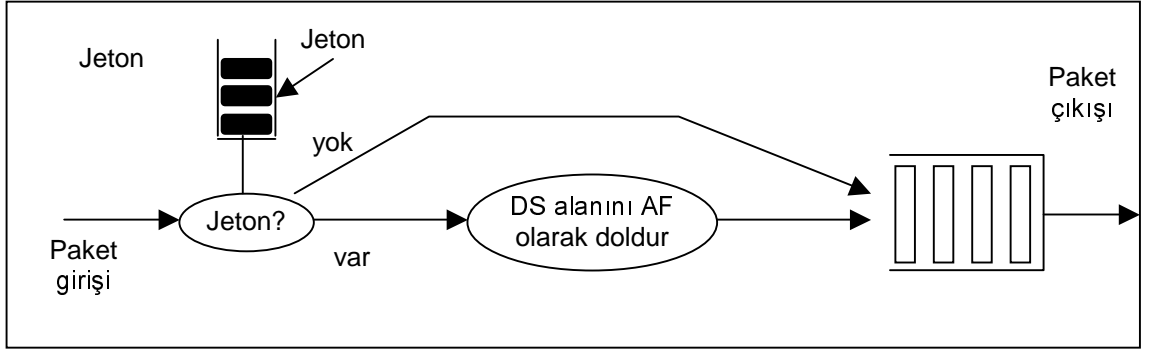
Bir DS düğümünde AF PHB hizmeti alan bir IP paketinin alacağı hizmeti 1) paketin ait olduğu AF sınıfına ne kadar kaynak ayrıldığı, 2) AF sınıfının ne kadar yüklendiği ve 3) paketin düşürülme seviyesi belirlemektedir.

AF PHB, patlamalı trafikten kaynaklanan kısa vadeli tıkanıklığı, paketleri kuyruklarda bekleterek gözardı ederken; uzun vadeli tıkanıklıkta paketleri düşürür. Bunu gerçeklemek için tıkanıklık seviyesinin ağırlık ortalaması hesaplanarak bu yeni seviyeye göre paketlerin atılması işlemi gerçekleştirilir. Bu tarz bir algoritma için çift seviyeli RED algoritması olan RIO kullanılabilir.

AF hizmeti alacak olan paketlerin profil dışı/içi olarak sınanarak uygun şekilde işaretlenmesini yönlendiricide **Şekil 2.7'**deki gibi bir jeton-kovası (token bucket) kullanarak gerçekleştirilebilir.

Bir ISP müşterilerine AF PHB kullanarak normal trafikten daha öncelikli bir hizmet sunabilir [19]. ISP'nin web tabanlı içerik sunan müşterisi, ziyaretçilerine diğer içerik sağlayıcılardan daha hızlı yanıt vermek için böyle bir hizmeti satın alabilir. Müşteri ile ISP arasındaki HA sonucu, müşterinin ISP'de baktığı yönlendiricide yapılan konfigürasyon şu şekilde gösterilebilir.

AF11 kodu : 1Mbps : herhangi bir çıkış : profil dışı trafik AF13 olarak kodlanacak

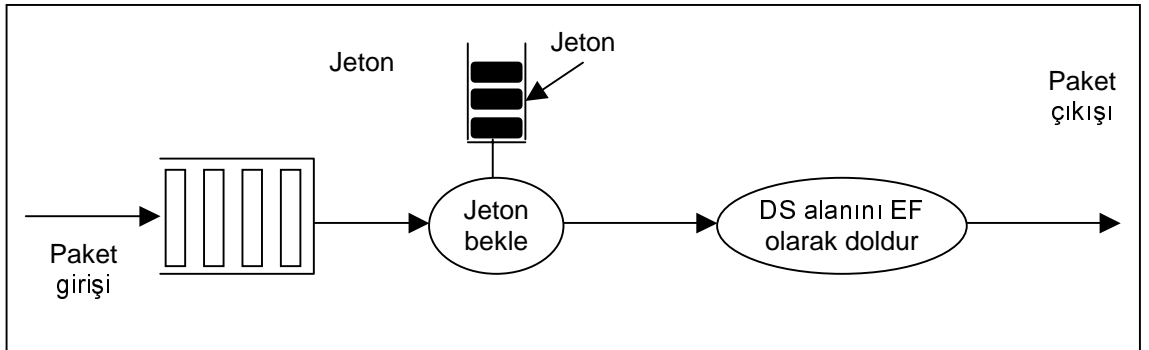


Şekil 2.7 AF için kullanılabilir ölçücü ve paket işaretleyicisi

Bu hizmeti alacak paketler ISP'nin giriş yönlendiricisine DS alanı kodlanmış olarak geleceklerdir. ISP, 1Mbps'e dek olan trafiği herhangi bir çıkışa dek normal trafikten daha öncelikli olarak iletmeyi taahhüt eder. 1Mbps'i geçen trafik AF13 olarak işaretlenir. Bunu sağlamak için servis sağlayıcı, trafiğin giriş noktasında paketleri kontrol eder. AF11 ve AF13'ün tüm ağda desteklenmesi için yönlendiricilerin konfigüre edilmesi gereklidir. AF11 ve AF13'ü gerçeklemek için tek bir RIO kuyruğu kullanılması yeterlidir.

2.5.2.3 Çabuklaştırılmış PHB

EF, özel bir hat kullanıyormuş gibi, uçtan uca düşük gecikmeli, düşük gecikme değişimli ve garantili bir hizmet sunar.



Şekil 2.8 EF sınıflandırıcısı ve şekillendiricisi

Paketlerin karşılaşılabileceği gecikme ve kayıplar yönlendiricilerdeki kuyruklardan kaynaklandığından, bunları minimize etmek için ya hiç kuyruk kullanılmamalı, ya da çok kısa kuyruk kullanılmalıdır. Kuyruk kullanımı gelen trafiğin, çıkan trafikten fazla olduğu durumlarda gerektiğinden, EF PHB hizmeti sunmak için çıkış bandgenişliğinin her zaman için (ağın ve yönlendiricinin yükünden bağımsız olarak),

trafiğin geliş hızından fazla olması gerekir. EF PHB'nin bunu sağlamak için yaptığı çabalar tek başına yeterli değildir. Ağa giren EF trafiğinin de sınırlandırılması gerekir, ancak bu EF'in konusuna girmez.

EF hizmetine ilişkin işaretleyici **Şekil 2.8**'deki gibidir. Yönlendiricilerin EF hizmeti sunmaları için, temel öncelikli kuyruk kullanmaları yeterlidir. Ancak EF'e ihtiyacı olan trafiği sunacak şekilde konfigüre edilmiş CBQ ve WRR kuyruklar da kullanılabilir.

2.5.2.4 Müşteri tarafında hizmet paylaşımı

Müşteri tarafı, ISP tarafından kendisine sunulan hizmeti kendi içinde paylaştırmada iki yaklaşımı benimseyebilir: Her düğüm alacağı hizmeti kendisi belirleyerek DS alanını kendisi doldurabilir veya kaynak paylaşımı merkezi bir otorite, bandgenişliği simsarı (BB - bandwidth broker) [30], tarafından yapılır.

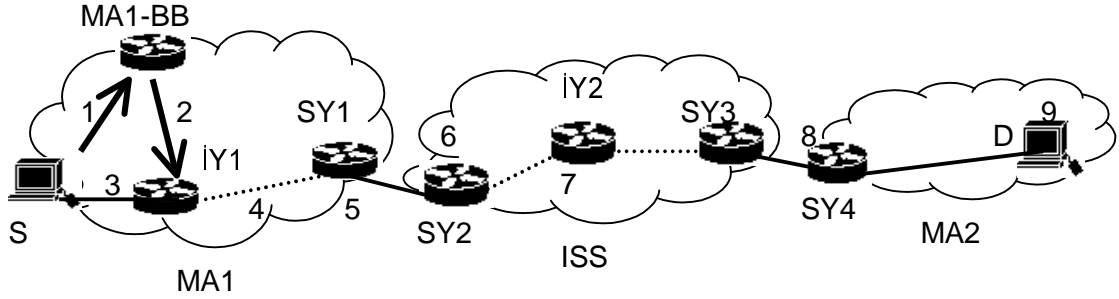
Tüm düğümler, ağın kaynak paylaşımı politikasını ve ağın mevcut kaynak kullanımını bilemeyeceğinden, BB kullanımı teknik olarak daha uygundur. BB kullanımı yönetimi de kolaylaştırmakta ve akışlara ilişkin tutulan durum bilgilerinin her yönlendirici yerine, sadece merkezi olarak tutulmasını sağlar.

BB'nin iki görevi vardır: ISP'den alınan hizmeti paylaştırarak, yönlendiricileri uygun şekilde konfigüre etmek ve dinamik HA kullanıldığında ISP'deki BB ile konuşmak.

DS'nin ilk kurulumlarında, uç düğümlerin DS alanını doldurmaları gerekmez. ISP'ye bakan yönlendiricinin belirli kurallar ışığında paketlerin alacağı hizmeti belirleyerek DS alanını doldurması yeterlidir. Daha sonraki kurulumlarda, uç sistemlerde bir sinyalleşme veya işaretleme mekanizması bulunabilir. Kullanıcı sistemi bir paket yollamadan önce alacağı hizmeti kendisi belirleyerek paketi uygun şekilde işaretleyebilir.

Bu yapılmadığından, kullanıcı sistemi almak istediği hizmeti ve süreyi BB'ye bildirir. BB önce paketi yollayanın yetkilerini sınar ve yeterli kaynak bulunup bulunmadığını sınar. Paket ISP üzerinden geçecek ve HA dinamik ise, ISP BB'si ile gerekli sinyalleşme yapılır. Kullanıcı sistemi ve BB arasında ve BB ile ISP BB'si arasındaki sinyalleşme RSVP kullanılarak yapılabilir. Daha sonra kullanıcı sistemin baktığı yönlendirici paketleri uygun şekilde işaretlemek için konfigüre edilir. Bu konfigürasyon periyodik olarak tazelenir.

BB kullanımı daha kolay açıklamak için aralarında statik HA bulunan ağlar üzerinde AF hizmeti kullanımını **Şekil 2.9**'daki ağ yapısı ile örnekleyeceğiz. Şekilde CN1 ağındaki S makinası, CN2 ağındaki D makinası ile AF kullanarak iletişim kurmak istemektedir. CN1 ile ISP arasındaki HA ise statiktir.

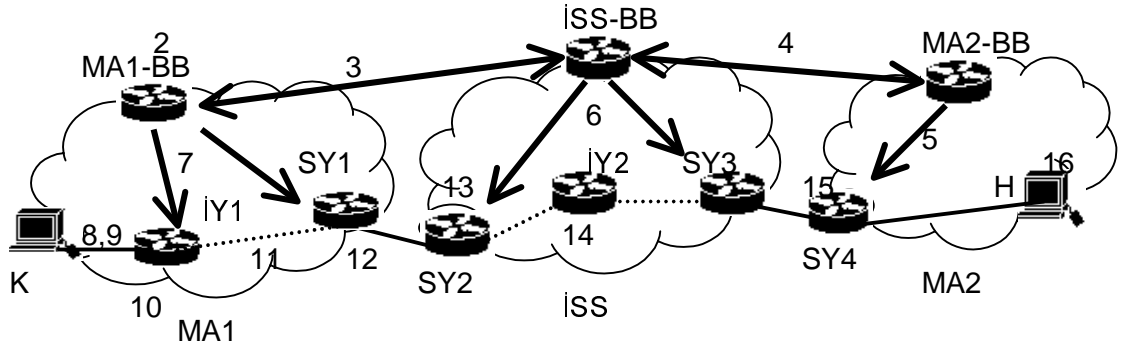


Şekil 2.9 Statik hizmet anlaşması ile AF hizmetinin işleyişi

1. K, MA1-BB'ye, H'ye AF hizmeti ile bağlantı kurmak istediğini ve istediği AF hizmetinin özelliklerini (AF sınıfı, önceliği, hizmeti alacak trafik miktarı, iletişim süresi ve hizmeti alacak olan trafiği tanımlayıcı bilgiler – AF11, 1Mbps, 2 saat, S:2000-D:80) bildirir.
2. MA1-BB istenilen hizmetin verilebileceğine karar verirse (koşturulan kabul kontrol algoritması sonucu), İY1 yönlendiricisini uygun şekilde konfigüre eder ve K'ya istediği hizmetin sağlanabildiğini bildirir. Hizmet sunulamıyor ise, bu durum da bir hata mesajı olarak K'ya bildirilir.
3. K veri yollamaya başlar.
4. İY1, K'dan gelen paketler üzerinde MF sınıflandırma ve trafik şekillendirme işlemlerini yürütür. Gelen paket AF hizmeti alacak akışa dahil ise, trafik 1Mbps altında oldukça AF11 olarak işaretlenir. 1Mbps üzerindeki trafik ise aynı sınıf içinde daha düşük öncelikli (AF12 veya AF13) olarak veya normal trafik olarak işaretlenebilir. [see marker at AF PHB].
5. SY1 de dahil olmak üzere, İY1'den SY1'e dek olan tüm yönlendiriciler BA sınıflandırması yaparlar ve gelen tüm AF1 paketlerini üzerinde hizmet farklılaştırılması amacına yönelik algoritmaların koştugu bir kuyruğa sokarlar.
6. SY2 trafiğin HA'a olan uygunluğunu kontrol eder. AF1 trafiği HA'da belirtilen AF1 limit altında değilse, limiti geçen trafikler daha düşük öncelikli veya normal trafik olarak işaretlenir. (HA'na bağlı olarak)

7. SY2'den SY3'ye dek olan tüm yönlendiriciler (SY3 dahil) BA sınıflandırması yaparlar ve kuyruklarında hizmet farklılaştırması yaparlar.
8. SY4, İY2 ile aynı işlemleri gerçekleştirir.
9. Paketler H makinasına ulaştırılır.

Dinamik HA kullanımı **Şekil 2.10**'daki örnek ağ yapısı üzerinde incelenecektir. Bu örnekte S, D'ye EF hizmeti ile bağlantı kurmak istemektedir.



Şekil 2.10 Dinamik hizmet anlaşması ile EF hizmetinin işleyişi

1. K düğümü, MA1-BB düğümüne, H'ye EF hizmeti ile bağlantı kurmak istediğini ve istediği EF hizmetinin özelliklerini (trafiğin tepe değeri, iletişim süresi ve hizmeti alacak olan trafiği tanımlayıcı bilgiler –1Mbps, 2 saat, S:2000-D:80) bildirir.
2. MA1-BB kabul kontrol testi yapar. Sonuç olumsuz ise K düğümünü bilgilendirir.
3. İstenilen hizmetin verilebileceğine karar verirse (koşturulan kabul kontrol algoritması sonucu), İSS-BB' aynı isteği gönderir.
4. İSS-BB'in yaptığı kabul kontrol testi sonucu olumsuz ise MA1-BB'ye hata mesajı gönderilir. MA1-BB de K'yı bu konuda bilgilendirir. Kabul kontrol testi sonucu olumlu ise İSS-BB aynı isteği MA2-BB'ye gönderir.
5. MA2-BB kabul kontrol testi yapar. Test sonucu olumlu ise SY4 konfigüre edilir ve İSS-BB'e hizmetin sağlanacağı bildirilir.
6. İSS-BB onayı alınca SY2 ve SY3'i konfigüre ederek MA1-BB'i bilgilendirir.
7. MA1-BB onay aldığı anda İY1 ve SY1'i konfigüre eder ve K'yı bilgilendirir.
8. K onay aldığı anda veri yollamaya başlayabilir.

9. K, İY1'e veri yollar.
10. İY1, K'dan gelen paketler üzerinde MF sınıflandırma ve trafik şekillendirme işlemlerini yürütür. Gelen paket EF hizmeti alacak akışa dahil ise, trafik 1Mbps altında oldukça EF olarak işaretlenir ve EF kuyruğuna koyulur. 1Mbps üzerindeki trafik ise düşürülür.
11. SY1 de dahil olmak üzere, İY1'den SY1'e dek olan tüm yönlendiriciler BA sınıflandırması yaparlar ve gelen tüm EF paketlerini EF kuyruğu üzerinden iletirler.
12. SY1 gelen trafiğin HA'a uygunluğu için trafik şekillendirmesi yapabilir. Bu farklı kaynaklardan gelen EF trafiğin İSS ile olan hizmet anlaşmasında belirtilen EF limitini aşmaması içindir.
13. SY2 trafiğin HA'a olan uygunluğunu kontrol eder. Limit üzerindeki EF paketleri atılır.
14. SY2'den SY3'ye dek olan tüm yönlendiriciler (SY3 dahil) BA sınıflandırması yaparlar ve SY3 trafik şekillendirmesi yapar (SY1'e benzer şekilde).
15. SY4, SY2 ile aynı işlemleri gerçekleştirir.
16. Paketler H'e ulaştırılır.

2.5.3 IS ve DS'in karşılaştırılması

IS mevcut Internet altyapısını ATM ağlarına benzer şekilde ağ üzerindeki düğümlerde akışlara ilişkin durum bilgilerini tutulacağı biçimde değiştirmektedir. Tolere edilebilen ve tolere edilemeyen gerçek zamanlı trafik akışlarını normal trafik ile birlikte aynı ağda iletmeyi hedefleyen IS, akışlara kesin garantiler verilebilmesi için bu tip değişikliğin kaçınılmaz olduğu görüşünü savunmaktadır. Sinyalleşme protokolleri yardımı ile her trafik akışı ağdan farklı seviyede hizmet talebinde bulunabilmektedir.

Ancak tutulan durum bilgilerinin yüksek miktarda verinin geçtiği merkezi ağ düğümlerinde çok yüksek boyutlara ulaşması IS'nin ölçeklenebilirlik sorunları nedeni ile yaygınlaşamamasına neden olmuştur. Bunun yanında yönlendiriciler kendilerine gelen paketin sınıflandırılması ve taahhüt edilen trafik profillerine uygunluğunu

sınamakla da yükümlüdürler. Bu ihtiyaç yönlendiricilerde ciddi işlem gücü kaybına neden olmaktadır.

Bu olumsuzluklara ek olarak uçlardaki uygulamaların veri akışına başlamadan önce sinyalleşme yapacak şekilde değiştirilmesi gerekliliği de bulunmaktadır. Ayrıca gerek farklı hizmetler sunulması ve gerekse de yeni sunulan sinyalleşmeler nedeni ile yönlendiricilerde de büyük değişikliklerin yapılması gerekmektedir.

DS ise ağ üzerindeki tüm karmaşıklığı daha az miktarda trafiğin işlendiği ağ sınırlarına iterek ağ içindeki düğümlerde karmaşık işlemlerin yapılması gerekliliğini gidermiştir. Tutulması kaçınılmaz durum bilgileri de BB gibi merkezi yerlerde tutulmaktadır. Buna karşın akışlara sunulan QoS imkanları da birleştirilmiştir. Akışlar bireysel olarak değil de bir grup içinde hizmet almaktadır. Ağ düğümlerinde durum bilgisi tutulmadığından gecikme gibi parametreler için uçtan uca net bir garanti verilememektedir.

DS'in yönlendiricilerde durum bilgisi tutmadan, IS hizmetlerine yakınsayan hizmetler verebilmesi için çalışmalar [31, 32] bulunsa da bunlar standartlaşmamıştır. Bu yapıda tutulması gereken durum bilgisi yönlendiriciler yerine paket başlığındaki bir alanda tutulmaktadır.

Bunun yanında DS'in akışlara sunduğu hizmet tek yönlüdür. İletilen veri normalden kaliteli bir şekilde hizmet alırken, veriye ilişkin paket alındıları normal şekilde hizmet aldığı anda genel iletim kalitesi bundan etkilenmektedir [33]. Özellikle akışların RTT'si düşük olduğunda ağın ağır yüklü olduğu durumlarda akışın almış olduğu hizmetin niteliğini, paket alındılarının almış olduğu hizmet etkilemektedir. Geliş yönünde trafiğin daha fazla olduğu Web gibi uygulamaları kullanan uç noktalar kaliteli hizmet için ücret ödeseler de bu ancak gidiş yönünde kaliteli hizmet almalarını sağlayacaktır. Bu sorunu çözmek için dönüş paketlerinin de uygun şekilde işaretlenilmesini ve buna uygun hizmet almasını sağlayacak yeni yapılara ihtiyaç vardır [34], ancak bu konu DS çalışma grubunun çalışma alanı dışındadır.

2.5.4 Kaynak Rezervasyon Protokolü

Daha önceden bahsedildiği gibi IS, uygulamaların çeşitli QoS seviyelerinde hizmetler arasından seçim yapmasına olanak tanıyan bir altyapı oluşturmaktadır. Bunun için öncelikle ağ üzerindeki düğümlerin farklı hizmetler (GS ve CS) sunması gereklidir. İkinci ihtiyaç ise uygulamaların ağ düğümlerine istedikleri hizmeti belirtebilmelerini sağlayan bir mekanizmadır. Bunun sağlanması için Kaynak

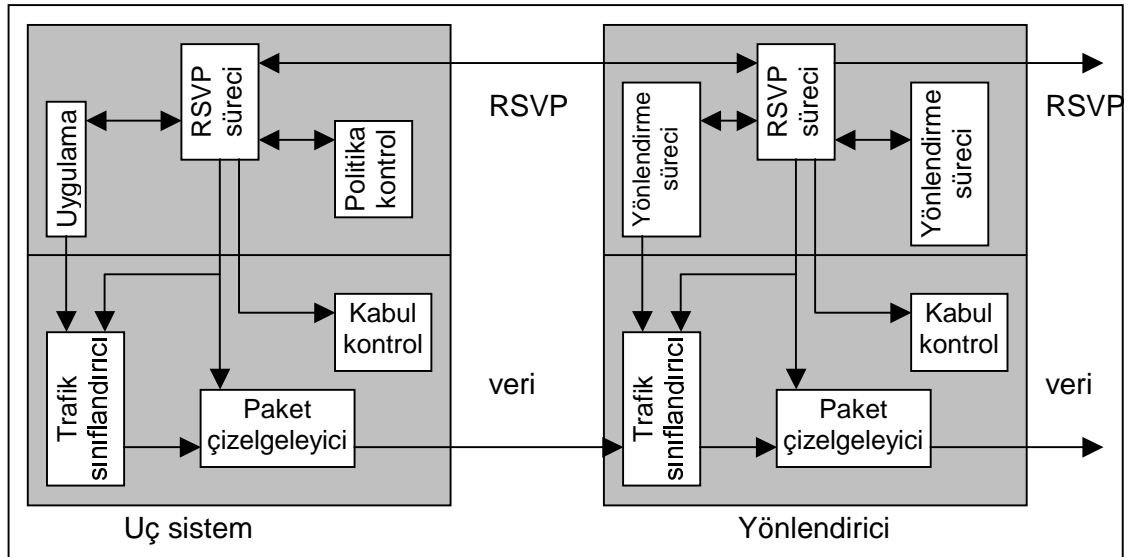
Rezervasyon Protokolü (Resource reSerVation Protocol, RSVP) [35] kullanılmaktadır.

RSVP sadece IS ile birlikte kullanılmak üzere değil çeşitli QoS prokolleri ile birlikte kullanılmak üzere tasarlandı. Bu nedenle RSVP aktarılan mesajın iç yapısını tanımlamaz. Yani RSVP sadece bir sinyalleşme protokolüdür ve QoS kontrol bilgisi de sadece sinyal içeriğidir. RSVP bir yönlendirme protokolü değildir ama yönlendirme protokolleri ile etkileşim içinde çalışır.

Genel olarak RSVP veri gönderen ve alıcısı arasındaki tüm düğümlere QoS isteklerin iletilmesini ve bu hizmetin sunulabilmesi için gerekli olan bilgilerin yönlendiricilerde tutulmasını sağlamaktadır. Bu durum bilgileri “soft state”dir, yani periyodik mesajlarla tazelenmediğinde otomatik olarak yokedilmektedir. RSVP ayrıca dinamik QoS istekleri de sağlamaktadır. Bu sayede RSVP alıcısı ve göndereni QoS parametrelerini değiştirebilir, bir çoğa gönderim grubunda yeni bir yollayıcı ve alıcı grubun QoS’undan daha yüksek seviyede bir QoS ile veri alabilir ve yollayabilir.

2.5.4.1 Çalışma şekli

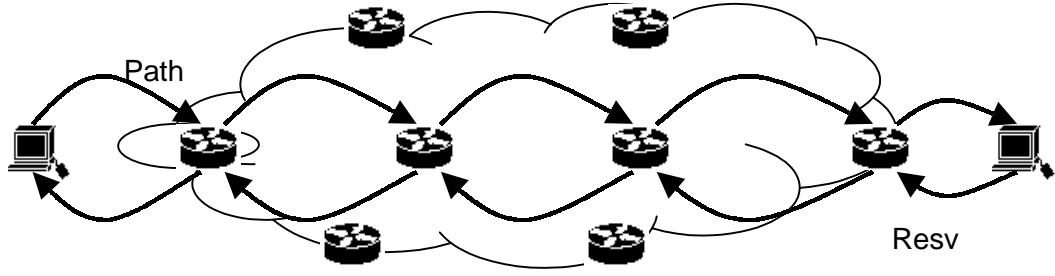
Şekil 2.11'de RSVP'nin uç sistemlerdeki ve yönlendiricilerdeki yeri ve işleyişi görülmektedir.



Şekil 2.11 Uç sistemler ve yönlendiricilerde RSVP

Bir RSVP göndericisi **Şekil 2.12**'de olduğu gibi, alıcıya bir Path mesajı gönderir. Path mesajının izleyeceği yol yönlendirme protokolleri tarafından belirlenir. Path

mesajları yol üzerindeki yönlendiricilerde yola ilişkin bilgilerin (paketin geldiği düğümün adresi) tutulması için kullanılır. Path mesajı taşıyan paketlerde bir önceki düğümün adresine ek olarak trafik akışını tanımlayan ve trafiğin özelliklerini belirten parametreler de taşınır. Bu bilgilerin yanında seçimlik olarak yol üzerindeki yönlendiricilerin özelliklerinin belirlenilmesi için ek alanlarda bulunabilir. Paketi iletmek için alan yönlendirici bu alanları doldurmaktadır.



Şekil 2.12 RSVP'nin temel işleyiş şekili

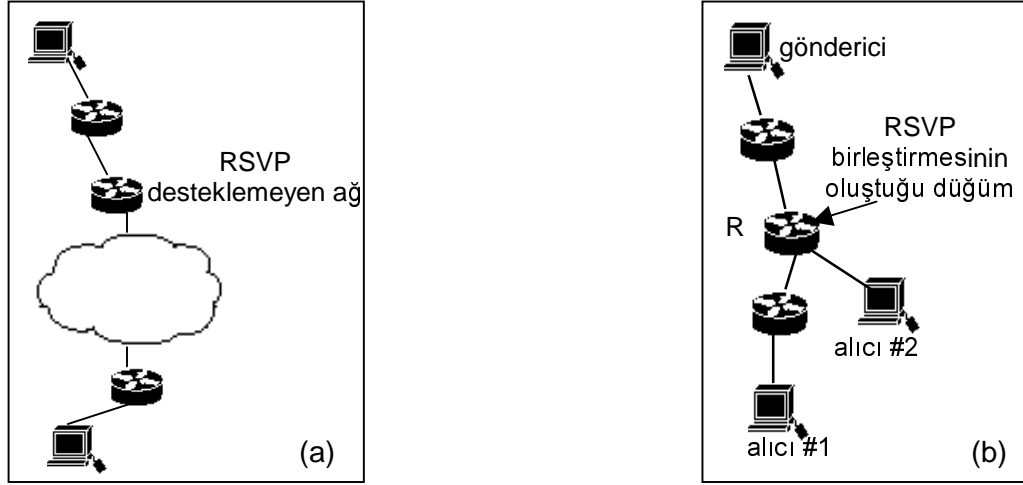
Bu mesajı alan alıcı düğüm göndericiye bir Resv mesajı yollar. Path mesajı ile tutulan yol bilgisi sayesinde Resv mesajı Path mesajının geldiği yol kullanılarak geriye gönderilir. Bu mesajı alan yönlendiriciler istenilen hizmet için kaynak ayırımını da gerçekleştirirler. İstenilen QoS seviyesini belirleyen yolun kaynak durumunu da gözönüne alarak alıcı olduğundan, Resv mesajında istenilen QoS seviyesi de belirtilir. Bunun yanında alıcının talep ettiği QoS'ı alacak olan akışı ve istediği trafik özelliklerini belirleyen parametreler de taşınır.

Path ve Resv mesajlarına ek olarak PathErr, ResvErr (hata durumları için); PathTear, ResvTear (kaynakların çözülmesi için) ve ResvConf (rezervasyon onayı) mesajları bulunmaktadır. RSVP mesajları 46 protokol numarası ile IP üzerinden yollanılır. IP üzerinden iletim gerçekleştiremeyen sistemler ise RSVP paketlerini UDP üzerinde yollayabilir.

Çoğa gönderim paketlerinin iletiminde paket çoğullaması yapıldığından aynı çoğa gönderim grubuna dahil alıcıların göndericiye olan yollarının birleştiği düğümlerde RSVP istekleri ve dolayısıyla ayrılan kaynaklar birleştirilir. Şekil 2.13'te R yönlendiricisinde alıcı 1 ve alıcı 2 düğümleri tarafından yapılan RSVP Resv istekleri birleştirilir. Yapılan birleştirmede QoS isteği daha yüksek olanın QoS parametreleri temel alınır.

Alıcılar Resv paketi ile rezervasyon stilini de belirleyebilmektedir. Burada bir seçenek farklı alıcılardan gelen trafik akışları için farklı rezervasyonlar yapılması

iken, diğer seçenek ise değişik yollayıcılar tarafından paylaşılan bir rezervasyon yapmaktır. Farklı rezervasyon stilleri ile yapılan RSVP istekleri birleştirilmez.



Şekil 2.13 RSVP iletim hattında RSVP desteklemeyen ağlar ve RSVP isteklerinin birleştirilmesi

iletim yolu üzerinde RSVP desteklemeyen ağların bulunması durumunda da RSVP işlevselliğini yitirmez. Kaynak ayırımı tam olarak yapılamasa da RSVP desteklemeyen düğümlerin gerçek zamanlı trafik akışlarına sunacakları kaynakları bulunabilir. Bu durumda RSVP desteklemeyen ağın iki ucunda bulunan RSVP yönlendiricileri bir önceki düğüm olarak birbirlerini göreceklerinden $Resv$ mesajlarını geriye gönderilebilmesi mümkün olabilmektedir.

3 TIKANIKLIK DENETİMİ

Internet, IP protokolü üzerinde uçtan uca bağlantısız bir mimari üzerine inşa edilmiştir. Bu bağlantısız yapının faydaları olan esneklik ve sağlamlık sayesinde bugünlere gelen Internet, kusursuz değildir. Uygun tasarlanmadığında IP ağları yapısındaki basitlikten kaynaklanan sorunlar nedeni ile ağır yük altında iyi hizmet verememektedir.

IP ağlarının tümüyle çökmesine de neden olabilen bu sorunlar ilk olarak 80'li yılların başlarında fark edildi [36]. Bu yıllardaki TCP yapısı ağ üzerindeki trafikteki ani değişimlere uygun cevabı verememekteydi. Ağın tamamen kullanılamaz hale gelmesine neden olan bu durum tıkanıklık çöküşü (congestion collapse) olarak isimlendirilmiştir. 80'li yılların başlarında Internet hatlarındaki fazla kapasite nedeni ile TCP/IP yapısındaki bu sorun ağlarda herhangi bir etki yaratmadı.

İlk olarak John Nagle [36] tarafından bahsedilen bu olay 1986 yıllarında gerçek oldu [37] ve bunun üzerine soruna çözüm çalışmaları yapıldı. Halen kullanılan TCP tıkanıklık denetimi (congestion control) yapıları bu yıllarda Van Jacobson tarafından geliştirilmiştir [37]. Jacobson'un geliştirdiği bu yapılar bugün IP yönlendiricileri tarafından uyulması gereken yapılar olarak standartlaştırılmıştır [38, 39].

Internet, basitliği sağlayabilmek için karmaşıklığı ve gereken zekayı ağ sınırlarına iter. Bunun sayesinde ağ üzerindeki düğümler basit halde kalabilmiştir. Tıkanıklık denetimi için ağ sınırlarında yapılanlar, sorunların çözümü için yeterli olamamaktadır ve bu nedenle TCP protokolünde yapılan değişikliklere ek olarak ağ üzerindeki düğümlerde de kuyruk yönetimi yapıları gibi değişiklikler yapılmıştır.

Son yıllarda hızlı bir şekilde büyüyen Internet'in yapısındaki zayıflıklar üzerine 80'li yıllardan itibaren bu çalışmalar yapılmasaydı, Internet'in bugünkü başarısına ulaşması mümkün olamazdı. Bunun sayesinde IP ağları her türlü hızdaki bağlantılar üzerinde başarılı bir şekilde çalışabilmektedir.

Bu bölümde öncelikle tıkanıklık ve tıkanıklık denetimi kavramları açıklanacaktır. Bölüm 3.3'te TCP protokolü tıkanıklık denetimi mekanizmalarını, bölüm 3.4 ise TCP

tıkanıklık denetimi yapılarını destekleyen yönlendiricilerdeki tıkanıklık denetimi yapıları incelenecektir.

3.1 Tıkanıklık

Bir kaynak üzerindeki talep, kaynak kapasitesini aştığında kaynak üzerinde tıkanıklık gerçekleşir [40, 41]. Bu matematiksel olarak şu şekilde ifade edilebilir:

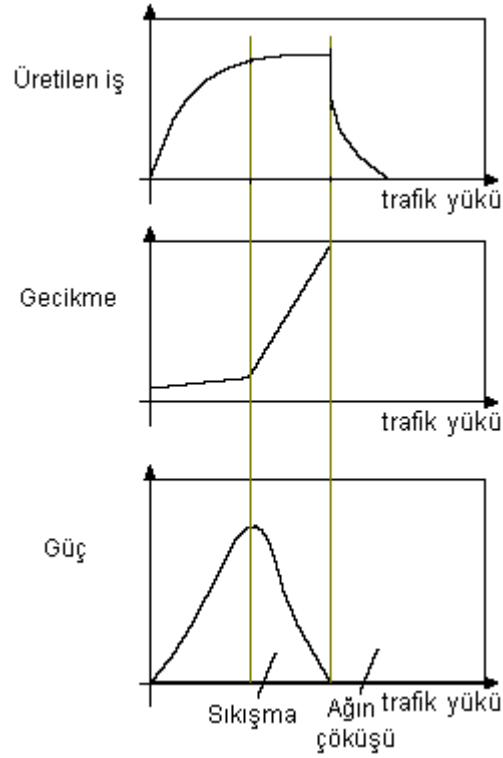
$$\sum \text{Talep} > \text{Kullanılabilir kaynak kapasitesi} \quad \text{Denklem 3.1}$$

Bilgisayar ağlarında tampon bellekler (buffers), hat kapasitesi (link bandwidth) gibi birçok kaynak bulunmaktadır. Kısa bir süre için hedefteki tampon bellek alanı gelen trafik için yetersiz hale gelirse paketler düşürülür (packet drop). Benzer şekilde, hatta girmek isteyen toplam trafik miktarı hattın kapasitesini aştığında da tıkanıklık gerçekleşir.

Toplam iletim gecikmesini arttıran, paketlerin düşürülmesi nedeni ile boş yere kaynak harcanmasına neden olan tıkanıklık, gerekli denetim mekanizmalarının bulunmaması durumunda ağın çalışmasını tamamen engelleyebilir [37, 36].

Tıkanıklığın daha matematiksel bir tanımı ağın başarımına bakılarak verilebilir [42]. **Şekil 3.1**'de ağın yük (load), üretilen iş (throughput) grafiği görülmektedir. Bir numaralı bölgede yük arttırıldıkça iletim kapasitesi giderek artmakta ancak bölgenin son kısımlarında bu artış giderek azalmaktadır. Ağ üzerindeki yük daha fazla arttırıldığında yönlendiricilerdeki kuyruklar dolmaya başlar ve daha ileri safhalarda paketler düşürülmeye başlar. **Şekil 3.1**'de ikinci bölge olarak gösterilen bu durum tıkanıklık durumudur. Yükün daha fazla arttırılması durumunda iletim kapasitesi tepe değerinin hemen arkasından keskin bir iniş ile sıfıra yakın değerlere inecektir. Bu durum da tıkanıklık çöküşü olarak isimlendirilir.

Aynı şekilde, her üç durum için ortalama dönüş zamanı (Round Trip Time, RTT) ve iletim kapasitesinin ortalama dönüş zamanına oranını gösteren güç (power) terimlerinin de değişimini görülmektedir. Eşik değerinin altında yavaşça artan gecikme tıkanıklık durumunda hızla artmaktadır. Çöküş aşamasında ise gecikme sonsuza gitmektedir. Benzer şekilde güç, tıkanıklık öncesinde tepe değerine ulaşırken tıkanıklık da giderek düşmektedir.



Şekil 3.1 TCP ağlarının tıkanıklık durumunda yük, iş verimi, gecikme ve güç yapısı

3.2 Tıkanıklık Çöküşü

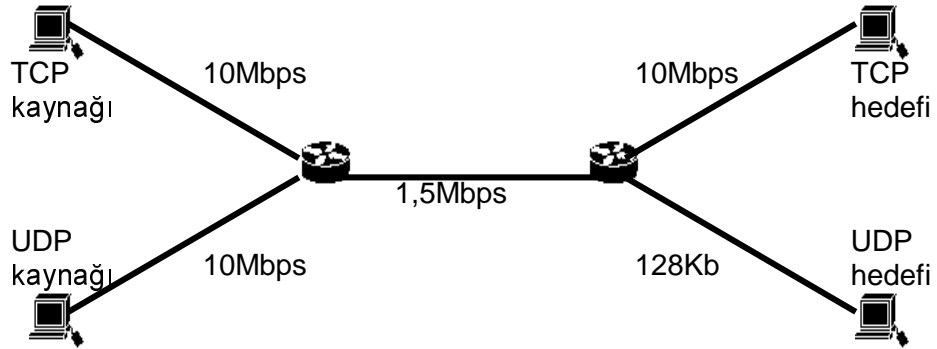
Uygun tıkanıklık denetimi yapıları olmadığı takdirde ağların karşılaşılabileceği sorunlar [36]'da açıklanmıştır. O zamanlar için gerçek olmayan bu tahminlerin doğruluğu sonraki yıllarda ortaya çıkmıştır.

TCP paket alındısı (Acknowledgement, ACK) aldıkça yeni paket göndermektedir. Belirli bir süre aşıldığında ise paketin yolda kaybolduğu düşünülerek tekrar iletim gerçekleştirilir. Bu süre ortalama dönüş zamanına göre ayarlanılır. RTT ise ağdan alınan örneklerle ölçülmektedir. Ağa aniden giren büyük miktardaki trafik RTT'nin aniden artmasına neden olur. Beklenen zamanında yerine ulaşamayan paketlerin düşürüldüğü varsayılarak tıkanıklık denetimi olmayan TCP tarafından tekrar iletilir. Ancak paketler halen ağıdadır ve yeni iletilen paketler gecikmenin daha da artmasına neden olur. Sonuçta bu tetikleme artarak devam eder ve ağı tıkanıklık çöküşü durumuna götürür. Çöküş durumu kararlı bir durumdur ve bir etkiye bulunulmadığında ağ bu durumda kalır.

Bugün TCP protokolünde tıkanıklık denetimi yapıları sayesinde Nagle'ın yukarıda detaylandırılan senaryosu gerçekleşmese de, giderek daha fazla kullanılmaya

başlanan tepkisiz (unresponsive) UDP akışları hala Internet için çöküşü olası kılmaktadır [43].

[44, 45] da yapılan benzetimlerde ağ kolaylıkla tıkanıklık çöküşü durumuna sokulabilmiştir. **Şekil 3.2**'deki yapıda bir ağ kullanarak yapılan benzetimde 1,5 Mbps'lık bir bağlantıyı paylaşan 2 kaynak karşıdaki 2 hedefe paket yollamaktadır. Kaynaklardan birisi UDP protokolü ile iletim yapmaktadır ve bunun hedef düğümü paylaşılan bağlantıya sadece 128KB ile bağlıdır. Diğer düğümlerin bağlantısı ise 10Mbps'dır. UDP hedefinin bağlantı hızı, paylaşılan bağlantının yaklaşık dokuzda biri olduğundan bunun üzerindeki hızlar ile iletilen tüm UDP paketleri yavaş bağlantının girişinde düşürülecektir.



Şekil 3.2 Tepkisiz UDP akışlarının tıkanıklık çöküşüne neden oluşu benzetimi [45]

Bu nedenle UDP kaynağının iletim hızı artırıldıkça, UDP'nin varış oranı artmadığı halde TCP'nin varış oranı düşmektedir. Böyle bir sorunu TCP'nin tıkanıklık denetimi yapıları da çözemez. Bunun çözümü ağa sadece uçtan uca iletimi gerçekleştirilebilecek trafiğin alınmasını sağlamaktır [45].

3.3 TCP ve TCP Tıkanıklık Denetimi Yapıları

Verilerin güvenli iletimi için kullanılan TCP, paketlerin geciktirilip, düşürülebileceği, sıralarının değiştirilebileceği veya bozulabileceği bir ortamda sıralı ve güvenli bir iletim sunma görevlerini üstlendiğinden oldukça karmaşıktır.

Paketlerin doğru sıra ile iletimi için TCP, paket başlığındaki sıra numaralarını (sequence numbers) kullanır. Güvenli iletim için ise, paket alındıları (ACK) kullanılır. Alınan her ACK paketi o ana dek gönderilen hangi paketlerin sağlam olarak hedefine ulaştırıldığını gösterir.

Belirli bir süre içinde alındısı gelmeyen paketin iletim sırasında bozulduğu veya düşürüldüğü varsayılarak paket tekrar iletilir. Bu tür bir iletim *zaman aşımli iletim* olarak isimlendirilir. Bu yapıda tutulan RTO (Retransmission TimeOut) süresi içinde gelmeyen paketler için tekrar iletim yapılır. RTO'nun uygun olarak belirlenemediği durumlarda oluşabilecek sorunlar Nagle tarafından gösterilmiştir [36].

TCP belirtilen hedeflerini gerçekleştirirken aynı zamanda yüksek başarımlı da sağlamalıdır. Başarımlı kaygısı bulunmadığında sıralı ve güvenli iletim için bir paketin alındısı gelmeden yeni paket gönderilmemesi yeterlidir. Ancak bu tip bir iletim son derece verimsizdir. Ortalama paket boyu b ve RTT de T saniye olarak isimlendirildiğinde iletim hattı çok hafif yüklü olduğu için paket kayıpları bulunmasa da, elde edilebilecek en yüksek iş

$$p = b / T \quad \text{Denklem 3.2}$$

olacaktır. $b = 512$ ve $T = 100\text{msn}$ olarak varsayıldığında $p = 5120$ bytes/sec olarak bulunacaktır. Bu durumda iletim hattının kapasitesi 10Mbps bile olsa bunun ancak 5Kbps'lık bir kısmı kullanılabilir.

Bu sorunu aşmak için TCP öncelikle paketleri mümkün olduğunca büyük olarak yollar. Her iletim bağının maksimum iletim birimi (Maximum Transmission Unit, MTU) isimli bir değeri bulunmaktadır ve bu değerin üzerindeki boylardaki paketler parçalanmaktadır. Yol üzerinde paketlerin parçalanıp tekrar birleştirilmesi gibi pahalı bir işlemi engellemek için, TCP bağlantısı kurulurken karşılıklı olarak iki tarafın almaya hazır olduğu en büyük paket boyu belirlenir. MSS (Maximum Segment Size) olarak isimlendirilen bu değer genellikle iletim yolu üzerindeki hatların en düşük MTU'sundan daha küçüktür.

Tüm paketler MSS boyunda da iletilse iletim hattı verimli olarak kullanılamaz. MSS değeri genellikle 2K – 4K gibi değerlerdedir ve bu da yetersizdir. Daha iyi başarımlı ancak aynı anda birkaç paket gönderilerek elde edilir. Aynı anda k adet paket gönderildiğinde elde edilen iş

$$p = k * b / T \quad \text{Denklem 3.3}$$

olacaktır. Bu sayede k değeri ile oynayarak herhangi bir hızda iletim gerçekleştirilebilir.

k değerinin belirlenmesinde hedefin ne miktarda veri kabul edebileceği ve ağın ne miktarda veri iletebileceği önemlidir. İlk sorun akış denetimi, diğeri ise tıkanıklık denetimi olarak isimlendirilir.

TCP akış denetimi için alıcının gönderdiği ACK paketlerinde bir alıcı penceresi (receiver window, rwnd) alanı tutar. Bu alıcının ne miktarda veri almaya hazır olduğunu gösterir ve kullanılabilir durumdaki tampon bellek alanı ve alıcı uygulamanın ne hızla veri tükettiğine bağlı olarak belirlenir.

Alıcı TCP, sırasız olarak gelen paketlerin sıralamasını gerçeklemek için paketleri bekleyen uygulamaya vermeden bekletebilmektedir ve bu nedenle kullanılabilir durumdaki tampon bellek miktarı sürekli olarak değişebilmektedir. Bu değişiklikler alındı paketleri ile gönderen tarafa bildirilir.

3.3.1 Tıkanıklık denetimi

Genellikle iletim hızını alıcının ne kadar veri kabul edebileceği değil, iletim hattının ne kadar veri kabul edebileceği sınırlamaktadır. TCP [46] tıkanıklık denetimini, RTO değerini RTT'nin birkaç katına ayarlayarak ve geciken paketlerin tekrar iletimini yaparak gerçekleştirir. Ayrıca RTO aşıldığında iki katına çıkartılarak ağır tıkanıklık durumlarında iletim hızının azaltılması da sağlanır.

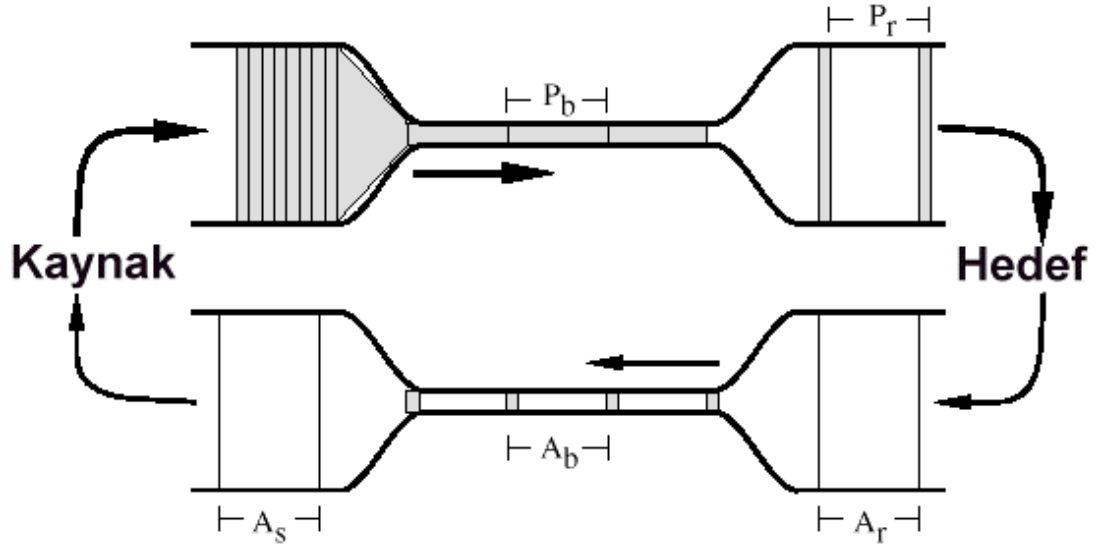
[37] ve [36]'da bu yapının olumsuzları anlatılmıştır. Jacobson TCP'nin bu haliyle ağ kaynaklarını tekrar iletilen paketlerle harcadığını ve tıkanıklığı düzgün olarak farkedemediğini göstermiş ve bunların çözümleri için tıkanıklık penceresi (congestion window, cwnd) isimli yeni bir pencere kullanılması gerektiğini belirtmiştir.

cwnd, alıcının penceresinden ayrıdır ve herhangi bir anda ağın ne kadar veri alabileceğini belirtmektedir. TCP iletim yaparken bu iki pencereden küçük olanına göre işlem yapmaktadır. Alıcı penceresinin aksine tıkanıklık penceresi ağ tarafından gönderene bildirilmemektedir. Jacobson bu pencerenin değerinin belirlenmesinde ağdaki paket kayıplarından faydalanmıştır.

3.3.2 Kendinden-zamanlama

Yüksek hızlı iki ağın düşük bant genişliğine sahip bir hat ile birbirine bağlanması **Şekil 3.3**'te gösterilmiştir. Bu yapıda yatay doğrultu zamanı gösterirken dikey boyut da bant genişliğini göstermektedir. Bant genişliği ve zaman çarpımı bit sayısını

gösterdiğinden ve bu da bağlantılar üzerinde değişmediğinden şekilde taralı olarak gösterilen paketlerin alanı sabittir.



Şekil 3.3 Pencere tabanlı iletim yapan protokollerde kendinden-zamanlama [37]

Kendinden-zamanlama (self-clocking) ile anlatılmak istenen hedefe gönderilen paketlerin darboğaz yaratan hattın bir kez P_b aralığı ile gönderilmeye başlanmasından sonra, bu zamanlamanın bozulmamasıdır. Birbirine eşit olan P_r ve P_b değerleri paketlerin gönderilme sıklığı için optimum değerdir. Paketler kaynak düğümünden daha sık aralıklarla gönderilse, kuyruklarda bekleyeceklerdir. Tersine daha az sıklıkla gönderilse bu kez de darboğaz oluşturan hattın kapasitesi tam olarak kullanılamayacaktır.

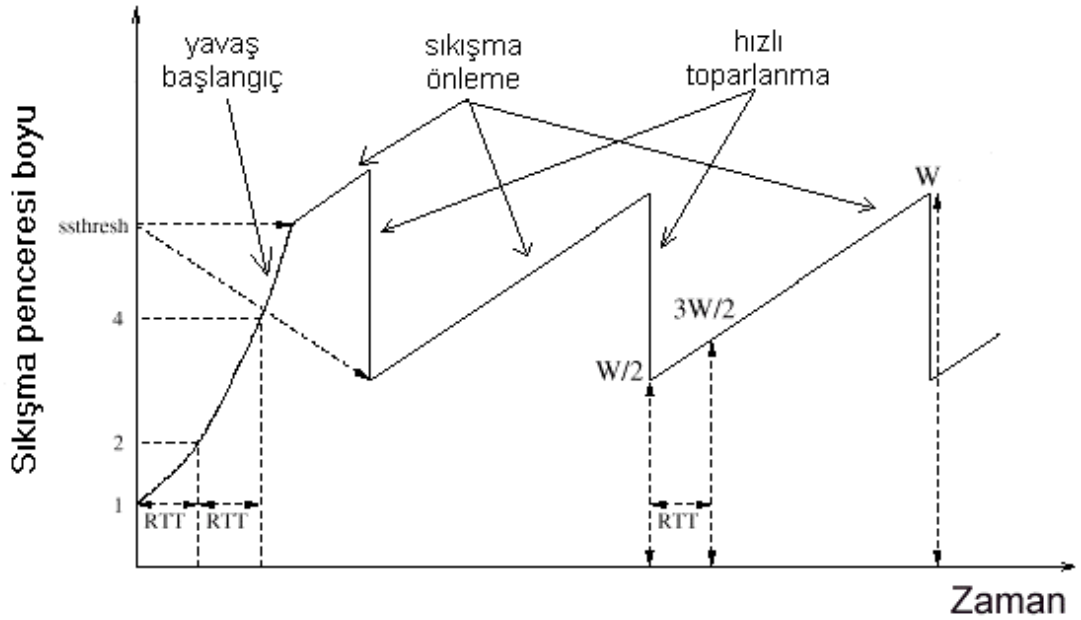
Pencere tabanlı iletim yapılarının iyi bir yanı olarak $P_r = P_b$ aralığı ile gönderilen paketler için, alıcı düğüm $A_r = P_r = P_b$ sıklığı ile aldığı paketleri üretecektir. Kaynak düğümüne $A_s = P_b$ aralığı ile ulaşan alındılar ise iletim penceresinin kaydırılarak yeni bir paketin iletilmesini sağlar. Bu sayede iletim kendinden-zamanlamalı duruma geçtiğinde, yeni paketler otomatik olarak hattın kapasitesine uygun hız ile gönderilir.

3.3.3 Yavaş başlangıç

Jacobson, kendi önerdiği tıkanıklık penceresinin yönetimi için iki algoritma sunmuştur. Bunlardan ilki olan yavaş başlangıç (slow start), cwnd'nin başlangıçta nasıl ayarlanması gerektiğini belirlerken; ikinci algoritma tıkanıklığın farkedilmesi durumunda hızın nasıl azaltılacağını belirlemektedir.

Kendinden-zamanlamayı işler hale getirmek için kullanılan yavaş başlangıç, TCP'nin iletme $cwnd = 1$ olarak başlamasını ve gelen her paket alındısı için $cwnd$ 'yi bir arttırmasını gerektirir. İsmi yavaş olsa da, yavaş başlangıç aslında $cwnd$ 'yi hızla arttırır. Yolda paket kaybının olmadığı bir durum için ilk RTT'de $cwnd$ 1 iken, ikinci RTT'de 2, daha sonra da 4, 8, 16 şeklinde exponansiyel olarak devam edecektir. Bu artış $cwnd$ alıcının penceresini aşana, yolda bir kayıp olana veya TCP uyarlaması tarafından belirlenen $ssthresh$ değerine ulaşana dek devam eder.

Yavaş başlangıç TCP akışını kendinden-zamanlamalı duruma geçirmek için kullanılır. Bu yüzden yavaş başlangıç aşamasında kendinden-zamanlamalı olarak çalışılmaz. Doğru P_b değeri bilinmediğinden iletim penceresi sürekli olarak artırılarak doğru değerin bulunmasına çalışılır.



Şekil 3.4 TCP'nin tıkanıklık penceresinin zaman içinde değişimine bir örnek

3.3.4 Tıkanıklık önleme

Jacobson'ın diğer mekanizması tıkanıklığa karşı nasıl tepki verileceğini belirler. Tıkanıklık durumunda iletim yolu üzerindeki düğümler paketleri geldikleri hızda işleyememekte ve kuyruklarda bekletmektedir. Tıkanıklık bir süre devam ettiğinde kuyruklar dolacak ve gelen paketler düşürülecektir.

TCP, yavaş başlangıç ile $ssthresh$ değerini aşmasıyla tıkanıklık önleme (congestion avoidance) durumuna geçer. $Ssthresh$ değeri kabaca ağın kabul edebileceği toplam veri miktarını göstermektedir. Bu nedenle $cwnd$ boyu bu değere ulaştığında iletimin

üstel olarak arttırılması uygun olmaz. Bunun yerine her paket alındısı ile $cwnd$, $1/cwnd$ kadar arttırılarak penceredeki tüm paketlerin başarılı olarak gelmesi ile pencere boyutunun sadece 1 arttırılması sağlanır. *ssthresh* ağın yaklaşık olarak kapasitesinin tamamının kullanılmış olduğu pencere değeridir. Bu nedenle kaynak TCP *ssthresh* sonrasında pencereyi çok daha yavaş arttırır.

3.3.5 Hızlı tekrar iletim ve toparlanma

TCP, zaman aşımaları dışında hızlı tekrar iletim (fast retransmit) [47, 38, 39, 48] isimli bir başka mekanizma daha kullanmaktadır. Bu TCP protokolünün bir parçası olmamasına karşın yaygın olarak kullanılmaktadır. Hızlı tekrar iletim TCP bağlantılarının kayıplar sonrasında yaşadıkları uzun beklemleri giderme amacı gütmektedir. Bağlantılar paket kayıplarını, alındının RTO süresi içinde gelmemesi sonucunda algılamaktadır. RTO, en yüksek RTT'den daha büyük olmalıdır ve RTT'nin en yüksek değerini belirlemek tam olarak mümkün değildir. Bu nedenle RTO'yu gerekenden düşük tutarak gereksiz iletimlere neden olmaksızın, RTO'nun gerekebilenden daha büyük olması tercih edilmektedir.

Bu sorunu çözmek için TCP'nin tıkanıklık durumunda açık olarak haberdar edilmesi gerekmektedir. İletilen paketlerden bir kısmı yolda düşürüldüğünde, alıcı taraf düşürülen paketi beklerken bu paketlerden sonraki paketleri alacaktır. Paket düşürülmesi dışında, paketlerin sıralarının izledikleri farklı yollar nedeni ile değişmesi sebebiyle oluşabilecek bu durum, kaynak düğüme en son gönderilen paket alındısının tekrar edilmesi ile bildirilir. Paket kaynağına gelecek çift paket alındıları (dup, duplicate ack) kaynağı yoldaki bir tıkanıklıktan haberdar eder. Çift paket alındılarının sayısı belirli bir değere (N_d) ulaştığında kaynak TCP bu noktadan sonraki paketlerin kaybolduğunu anlayarak *ssthresh* = $cwnd$ yaparak gerekli paketleri tekrar iletir. Paketlerin karışık sıra ile gelmesinin paket kaybı olarak algılanmaması için N_d çok küçük seçilmemelidir. N_d 'nin çok büyük olması ise kaynağın paket kaybını geç algılamasına neden olacağından mevcut TCP uyarlamaları bu iki durum arasında bir değer olarak $N_d = 3$ kullanmaktadır.

Jacobson, bu yapı üzerine bir de hızlı toparlama (fast recovery) algoritmasını eklemiştir. N_d adet çift paket alındısından sonraki her ACK alıcıya yeni bir paketin ulaştığını haber vermektedir. Bu nedenle paketlerin tekrar iletimi yapıldıktan sonra *ssthresh*, ($N_d * \text{paket boyu}$) kadar arttırılmaktadır.

3.3.6 Tıkanıklığa tepki verme

İletim yolunda tıkanıklık olduğu paketlerin düşürülmesi ile anlaşılmaktadır. Paketin düşürüldüğü ise iki şekilde anlaşılmaktadır:

1. RTO zaman aşımının dolması ile
2. Gelen çift paket alındıları ile

RTO, RTT'ye göre oldukça büyük olduğundan, birinci durumda iletimde paket olmadığı kesindir. Bu nedenle kendinden-zamanlamayı tekrar başlatmak için yavaş başlangıç aşamasına geçilir.

İkinci seçenek için ise halen iletimde paket bulunmaktadır. Bu nedenle yavaş başlangıç yapılması gerekmez. Bu durumda sadece *cwnd* ve *ssthresh* yarılanır.

Gelen çift alındılar bir yandan paket kaybı anlamına gelirken, bir yandan da başka paketlerin iletim hattını terkederek hedefteki tampon alanlara yerleştirildiğini ifade eder. Bu nedenle *cwnd* gelen her çift paket alındısı için bir artırılır.

3.4 Kuyruk Yönetim Yapıları

Tıkanıklık exponensiyel olarak büyüdüğünden, tıkanıklığın erken fark edilmesi uç noktalarda küçük pencere ayarları ile tıkanıklığın giderilmesini sağlayacağından önemlidir. Aksi halde pencerelerde büyük azaltmalar yapmak gerekecektir.

Tıkanıklık hakkında erken bilgi sahibi olabilecek tek birim ağ üzerinde tıkanıklığın yaşandığı düğümlerdir. Bu nedenle uç noktalarda tıkanıklık denetimi için TCP yapısında yapılan değişikliklerin yanında, ağ yönlendiricilerinin de tıkanıklık denetimi yapılarına sahip olması kaçınılmazdır.

Yönlendiriciler üzerindeki yapılar tıkanıklığı erken fark ederek paket kayıplarını azaltmalarının yanında, tıkanıklık denetimi algoritmaları koşturmayan iletim protokollerinin diğer protokolleri mağdur etmesini de engelleyebilecek tek çözümdür. Aksi halde TCP gibi tepki veren protokoller hızlarını tıkanıklık nedeni ile kesseler de, UDP gibi tepkisiz protokollerle iletim gerçekleyen akışlar hızlarını kesmeyeceklerinden UDP akışları paylarından daha da fazla iletim yapacaklardır.

3.4.1 Aktif kuyruk yönetimi ihtiyacı

Yönlendiricilerin geleneksel kuyruk yönetim yapısı kuyruktan düşürme (Drop-Tail, DT) yapısıdır. Bu yapıda kuyruk için bir üst limit belirlenerek kuyruk boyu bu sınırın altında iken gelen paketler kabul edilir. Kuyruk limit değere ulaştıktan sonra ise gelen paketler düşürülür. Uzun yıllar kullanılan bu yapının iki temel sorunu bulunmaktadır:

1. Kilitlenme: Senkronizasyon veya diğer zamanlama sorunları nedeni ile birkaç bağlantının tüm kuyruğu ele geçirerek, diğer bağlantılara ait paketlerin iletimini engelleyebilmesi mümkündür.
2. Kuyruk doluluğu: DT kuyrukları uzun süre dolu olarak kalmaktadır. Çünkü TCP gibi tepki veren protokollerin iletim hızlarını kesmesi ancak paket düşürülmesi ile olmaktadır. Kuyruk yönetimi algoritmalarının temel amacı kuyruk boyunu kontrol etmek olduğundan, kuyruğun kararlı durumdaki boyu tepe değerlerde bulunmamalıdır.

TCP pencere tabanlı iletim yapısı nedeni ile patlamalı bir iletim yapmaktadır. Yönlendirici kuyrukları sürekli dolu veya doluluğa yakın durumda ise patlamalı şekilde gelen bir paket grubu aynı anda birçok paketin düşürülmesine neden olabilmektedir. Bu durum patlamalı iletim yapan akışların DT tipi kuyruklar kullanıldığında genel trafik içindeki oranlarından daha fazla bir şekilde düşürülmesine neden olmaktadır. Kuyruk dolu olduğu için aynı anda birçok farklı akışa ait paketlerin düşürülmesi ise küresel eşzamanlılık (global synchronization) isimli olaya neden olmaktadır.

Küresel eşzamanlılık, dolu kuyruklar nedeni ile aynı anda birçok farklı akışa ait paketlerin aynı anda düşürülmesi sonucunda, bu TCP akışlarının aynı anda iletim hızlarını kesmesi ve daha sonra da aynı şekilde arttırıma gitmeleridir. Bu durum sonucunda akışlar kuyruğun periyodik olarak dolup boşalmasına neden olmaktadır.

Yönlendiricilerde kuyrukların varoluş amacı veri patlamalarını tolare etmek ve daha sonra iletim hattı boşaldığında kuyrukta bekletilen verilerin iletimini yapmaktır. Aksi halde patlamalı bir şekilde veri transferi yapmak mümkün olamayacaktır. Kuyrukların sürekli dolu halde olması da bu amaca ulaşılmasını engellemektedir. Ayrıca kuyrukların üst sınırları sürekli olarak bu durumlarda kalınması amaçlı değil anlık olarak bu durumlara ulaşılması içindir.

DT'de ufak deęişiklikler ile RDOF (Random Drop on Full) ve DFOF (Drop Front on Full) kuyruk yapıları elde edilmiştir. Kuyruk dolduęunda gelen paketin düşürölmesi yerine RDOF kuyruk içinden rasgele bir paketin düşürölken, DFOF kuyruęun en önündeki paketi düşürmektedir. Bu iki yapı da kilitlenme ve patlamalı iletim yapan akışlara karşı gösterilen adaletsizlik sorunlarını çözse de kuyruęun sürekli olarak doluluęa çok yakın bir durumda kalmasını engelleyemez.

Aslında kuyrukların doluluk sorununun çözölümü, TCP gibi aędan gelen işaretlere tepki verebilen iletim protokolleri için basittir. Bu protokoller paketlerin düşürölmesi ile aęda bir tıkanıklık olduęunu anlayarak iletim hızlarını kestiklerinden, kuyruk tam dolu hale gelmeden paket düşürölmesi kuyruęun boyunu kontrolde tutabilmek için yeterlidir. Bu tip bir yapı aktif kuyruk yönetimi olarak isimlendirilmektedir. Aktif kuyruk yönetimi yapıları yönlendiricilerin kuyruk boyunu kontrol edebilmek için paketleri ne zaman ve ne miktarda düşüreceklarını belirleyebilmelerini sağlar.

Aktif kuyruk yönetimi yapılarının tepki veren (responsive) protokoller ile sağladıkları faydalar temel olarak řu řekilde sıralanabilir [43]:

1. Yönlendiricilerde düşürölün paket sayısı azalır.

Paket kaybı paket tabanlı aęlar için önü kesilemeyecek bir durumdur. Aktif kuyruk yönetimi yapıları kuyrukları kısa tutarak doęal olarak oluřan paket patlamalarını paket düşürölmeden geçiřtirebilmektedirler. Ayrıca sürekli yeni paketler için yer bulunabildięinden DT kuyruklarda oluřan küresel eşzamanlılık da oluřmaz. Bu sayede uçlarda tıkanıklık denetimi yapıları bulunan bir durumda aktif kuyruk yönetimi mekanizmaları toplam paket düşürölme oranını da azaltarak, baęlantıların daha verimli kullanılmasını da sağlamaktadır.

2. Düşük gecikme sağlanır.

Ortalama kuyruk boyu daha kısa olacaęından, karşılaşılan gecikme de azalır.

3. Kilitlenme durumu oluřmaz.

Belirli akışların dięer akışların iletimini engellemesi durumu, aktif kuyruk yönetimi yapılarında sürekli olarak kuyruklarda boş yer olacaęından oluřmaz. Bu sayede yönlendiricilerin akışlara adil olmayan bir řekilde hizmet vermeleri engellenilmiş olur. Tam olarak adil olunabilmesi için yönlendiricilerde her akışa ait durum bilgisi tutulmalıdır. Aksi halde FIFO yapısı ile hizmet verilen

kuyruklarda farklı RTT'ye sahip iki TCP akışı çok farklı seviyelerde hizmet alırlar.

Adilliğin sağlanması için akışlara ilişkin durum bilgilerinin tutulduğu zamanlama mekanizmalarının bulunması durumunda da aktif kuyruk yönetimi yapılarına olan ihtiyaç yok olmaz. Bu zamanlama yapıları sadece adaleti sağlarlar, ancak kuyruk boyunu kontrol edemezler.

3.4.2 Rastlantısal erken algılama

Bugün üzerinde en çok çalışma yapılan ve en çok kullanılan aktif kuyruk yapısı olan Rastlantısal Erken Algılama (Random Early Detection, RED) [49], çıkış kuyruklarının ortalama boyunu denetleyerek, rastlantısal olarak tıkanıklıktan haberdar edilecek bağlantıları belirler. Uzun süreli tıkanıklık ortalama kuyruk boyunun artmasına neden olarak kaynak düğümlerin hızlarını kesmelerine neden olurken; kısa süreli tıkanıklık herhangi bir değişime neden olmaz. Herhangi bir akışın tıkanıklıktan haberdar edilme olasılığı, akışın toplam trafik içindeki oranı ile orantılıdır.

Akışların tıkanıklıktan haberdar edilmesi için kullanılan tek mekanizma paket düşürülmesi değildir. Açık tıkanıklık bildirimi (Explicit Congestion Notification, ECN) yapısı ile paket düşürmeden de akışların haberdar edilmesi mümkündür.

Kuyruk boyunun belirli bir limiti aşması durumunda ise tüm paketler işaretlenilir. İşaretlenme ya paketlerin düşürülmesi ya da ECN ile ilgili bitlerinin işaretlenilmesi anlamına gelmektedir. Paketlerin düşürülerek işaretlenilmesi durumunda üst kuyruk sınırı sayesinde, kaynak düğümlerin tıkanıklık denetimi yapılarına sahip olmamaları durumunda bile RED kuyruk boyunu kontrol edebilmektedir.

3.4.2.1 Önceki çalışmalar

RED'den daha önce de Erken Rastlantısal Düşürme (ERD, Early Random Drop) isimli yapılar bulunmakta idi. Bu yapıda yönlendiricinin kuyruğu yarı oranda dolu olduğunda ERD paketleri 0,02 olasılığı ile düşürür. [41]'de bu yapının işbirliği içinde bulunan ve bulunmayan uç nokta protokolleri arasında adil olmadığını gösterilmiştir.

Bunun yanında ICMP [50] protokolündeki SQ (Source Quench) mesajlarının da tıkanıklık denetimi amaçlı kullanılabileceği düşünülmüştü. Yönlendirici paket düşürdüğünde geriye doğru iletim hızının kısılması için SQ mesajı gönderir. [51]'de ise kuyruk boyunun bir değeri aşması durumunda geriye doğru SQ mesajı

gönderilmesi önerilmekteydi. Ancak [52]'de SQ mesajlarının fayda sağlamadığı ve gereksiz yere kaynak yemesi nedeni ile gönderilmemesi önerilmektedir. [53] de ise tıkanıklık fark edildiğinde ileriye doğru ECN işareti gönderilirken, geriye doğru da SQ mesajı gönderilen bir yapı önermiştir. [BECN] günümüzde işlem gücünün önemli bir sorun olmaması ve kullanılan yapı nedeni ile de çok fazla SQ mesajı oluşturulmadığı için SQ mesajlarının kullanılabileceğini belirtmiştir.

[54]'de açıklanan DECbit isimli yapıda yönlendiriciler son iki periyot gözönüne alınarak ortalama kuyruk boyunu hesaplar. Eğer hesaplanan değer 1'den fazla ise paketlerdeki tıkanıklık bildirim biti çekilir. Gönderen TCP tarafı, son pencere içinde gelen paketlerin yarısından fazlasında tıkanıklık bildirim biti çekili ise, pencere yarılanır. Aksi halde pencere 1 artırılır.

RED'in DECbit'ten temel iki farkı bulunmaktadır. DECbit ortalama kuyruk boyunu hesaplamada sadece son iki periyodu gözönüne alırken, RED tüm geçmişin ağırlıklı ortalamasını almaktadır. Ayrıca RED tıkanıklığın kaynak düğüme bildirilmesinde de paketlerin özel yerlerini işaretlemenin yanında paketleri düşürme yolunu da seçebilmektedir. Paketler düşürülerek tıkanıklık bildirildiğinde uç noktalardaki TCP uyarlamalarında herhangi bir değişiklik yapılması gerekmemektedir. Bu sayede RED'in Internet çapında parça parça kullanıma verilmesi de mümkündür.

3.4.2.2 Tasarım hedefleri

RED'in ana amacı kuyruk boyunu kontrol ederek tıkanıklığı denetlemektir. Bunun yanında kuyruklarda patlamalı trafiğe karşı gösterilen adaletsizliği ve evrensel eşzamanlılığı önlemektir.

Bir tıkanıklık denetim yapısı ağı düşük gecikme ve yüksek iş veriminde tutmaya çalışır. Ağ üzerindeki trafiği en iyi yönlendiriciler gözlemleyebildiğinden tıkanıklığı zamanında fark ederek gerekli işlemleri yapacak olan da yönlendiricilerdir.

Yönlendiricinin amacı uzun süreli tıkanıklığı fark etmektir. Bundan sonraki adım hangi akışların tıkanıklıktan haberdar edileceğinin belirlenilmesidir. Tıkanıklık kuyruklar tam dolmadan fark edilebiliyorsa, tıkanıklık kaynaklara paketlerinin düşürülmesi ile bildirilmek zorunda değildir. RED, bu yüzden hem paket düşürülmesini hem de paketlerin ECN alanlarının işaretlenilmesini desteklemektedir. RED paketleri toplu olarak düşürmek yerine zaman içerisinde rastlantısal ve dağınık olarak işaretlediğinden DT kuyruklarda gözüken küresel eşzamanlılık RED kuyruklarında gözükmez.

RED kuyruk boyunu, kuyruklarda sürekli boş yer bulunabilecek şekilde kontrol ettiğinden her zaman gelecek yeni bir paket için boş yer vardır. Bunun sayesinde patlamalı şekilde gelen paketler düşürülmeyecektir.

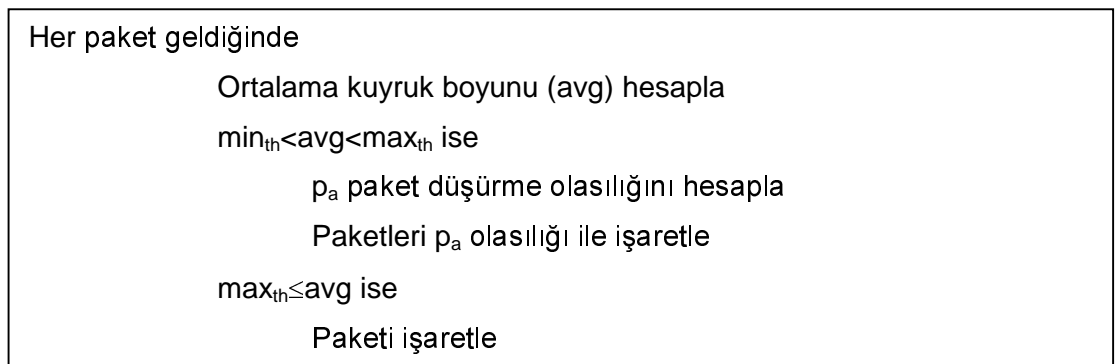
TCP pencere tabanlı iletim yapısı nedeni ile patlamalı bir trafik üretmektedir. DT kuyruklar ise sürekli olarak doluluğa yakın olduğundan patlamalı bir şekilde iletim yapan akışlar diğerlerine göre daha çok düşürülmektedir. RED, kuyruk boyunu kısa tutarak bu sorunu da çözmektedir.

3.4.2.3 RED algoritması

RED kullanan yönlendiriciler her çıkış kuyruğu için, zaman içinde sürekli olarak ortalama kuyruk boyunu hesaplamaktadır. Ortalama kuyruk boyu verilen iki parametre değeri arasında oldukça belirli olasılıkla paketlerin işaretlenilmesi yapılır. Kuyruk boyu alt eşik değerinin (min_{th}) altında ise paketler hiçbir şekilde işaretlenilmez. Kuyruk boyu üst eşik değerine (max_{th}) ulaştığında ise gelen tüm paketler işaretlenilir.

Kuyruk boyu iki eşik değeri arasında iken paketlerin düşürülme olasılığı kuyruk boyu ile doğru orantılıdır ve bu olasılığın üst sınırı da kullanıcı tarafından belirlenen bir RED parametresidir (max_p).

Floyd 2000 yılında, ilk olarak RED'in ATM ağları için olan bir uyarlamasında [55] kullanılan *gentle* parametresinin asıl RED algoritmasında da kullanılmasını önermiştir [56]. Bu yapıda, ortalama kuyruk boyu üst eşik değerini aştıktan sonra "1" olasılığı ile düşürülmek yerine, max_p ile 1 arasında ortalama kuyruk boyuna göre değişen bir olasılıkla düşürülmektedir. Floyd, bu yapının RED'i max_{th} ve max_p parametrelerinin yanlış seçimine karşı daha güvenli kıldığını iddia etmektedir.



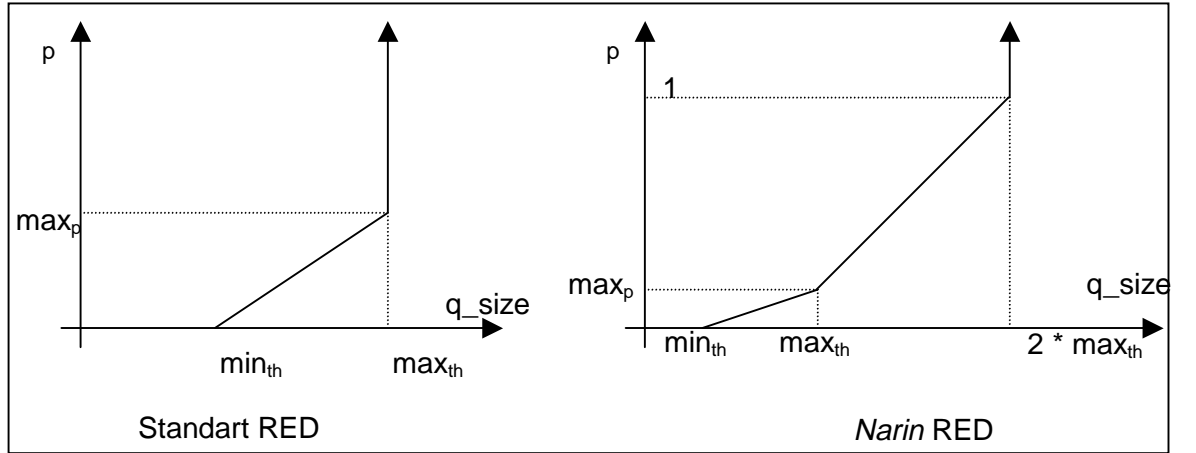
Şekil 3.5 RED algoritması

Buradan görüleceği şekilde RED ortalama kuyruk boyu hesaplama ve paket düşürme olacak şekilde iki temel algoritmadan oluşmaktadır. Paket işaretleme hem tıkanıklığı kontrol altında tutacak kadar çok olmalı hem de evrensel eşzamanlılığı önleyecek şekilde zaman içinde dağıtılmalıdır. Algoritmanın detayları **Şekil 3.5**'te görülmektedir.

Ortalama kuyruk boyu hesabı her yeni paket gelişinde yapıldığından, her hesaplama da hattın boş geçtiği zamanlar da hesaplamaya katılır. Ortalama kuyruk boyu ağırlıklı olarak hem ortalama kuyruk boyu hem de anlık kuyruk boyu hesaba katılarak hesaplanır.

$$avg \leftarrow (1 - w_q) * avg + w_q * q \quad \text{Denklem 3.4}$$

Bu sayede kuyruk boyundaki artışlar ortalama kuyruk boyuna olduğu gibi yansımaz. Böylece anlık patlamalardan oluşan tıkanıklık gözardı edilerek sadece uzun süreli tıkanıklığa tepki verilir. Anlık kuyruk boyunun ortalama kuyruk boyuna etkisini gösteren w_q da değiştirilebilen bir parametredir.



Şekil 3.6 RED parametreleri

Paket düşürme olasılığının hesaplanılmasında hem ortalama kuyruk boyu hem de en son ne zaman paket işaretilendiği bilgisi dikkate alınır.

$$p_b \leftarrow max_p (avg - min_{th}) / (max_{th} - min_{th}) \quad \text{Denklem 3.5}$$

$$p_a \leftarrow p_b / (1 - count * p_b) \quad \text{Denklem 3.6}$$

count değişkeni en son paket işaretlenilmesinden bu yana gelen paket sayısını göstermektedir ve bunun sayesinde son paket işaretlenilmesinden itibaren geçen zaman arttıkça, paket işaretleme olasılığı artmaktadır.

3.4.2.4 Parametre seçimi

RED'in temel parametreleri olan min_{th} , max_{th} , max_p ve w_q değiştirilerek tıkanıklığa çok değişik tepkiler verilmesi mümkündür. Ancak [57] ve [58]'de belirtildiği gibi istenilen etkilerin elde edilmesi için parametrelerin nasıl seçileceği kolay bir iş değildir. Parametreler yönlendirici üzerindeki trafiğin karakteristiğine ve aktif akış sayısı [59] gibi etkenlere göre değiştirilmelidir. Ancak RED'in hedefleri gözönüne alındığında parametre seçimi şu şekilde yapılmalıdır [49, 60]:

1. Ağın etkin kullanımını arttırmak

Kuyruğun boş kalması nedeni ile hattın boşta kalmasını engellemek için min_{th} parametresi büyük seçilmelidir. Bu şekilde kuyrukta sürekli iletilecek veri olacaktır.

2. Kuyruklama gecikmesini en aza indirmek

Kuyruk boyunun az olması yaşanan gecikmeyi azaltacağından min_{th} küçük seçilmelidir.

UDP gibi tepkisiz akışlar paketlerin düşürülmesine tepki vermediklerinden, bu tip akışların bulunduğu kuyruklarda kuyruk boyunu kısa tutmak için max_{th} küçük tutulmalıdır.

3. Evrensel eşzamanlılığı engellemek

p_q 'nın homojen bir şekilde dağıtılmasını sağlanılmalıdır.

Yönlendirici üzerinden geçen trafiğin bir RTT'de oluşturduğu patlamaları tolare edebilmek için $(max_{th} - min_{th})$ farkının büyük olması sağlanılmalıdır. Aksi halde kuyruk DT kuyruğuna benzer bir davranış gösterir. [49]'da max_{th} 'in min_{th} 'in en az iki katı olması önerilirken; yine Floyd tarafından [61]'de bu oranın 3 olması gerektiği belirtiliyor.

max_p , kuyruk min_{th} ile max_{th} arasında kaldığında yeterince işaretlenebilecek şekilde seçilmelidir. [49]'da yapılan simülasyonlarda $max_p = 0,02$ olarak

kullanılırken, [61]'de $max_p = 0,1$ olması gerektiği belirtiliyor. [59] max_p 'nin trafik yoğunluğuna göre dinamik olarak belirlenmesini önerir.

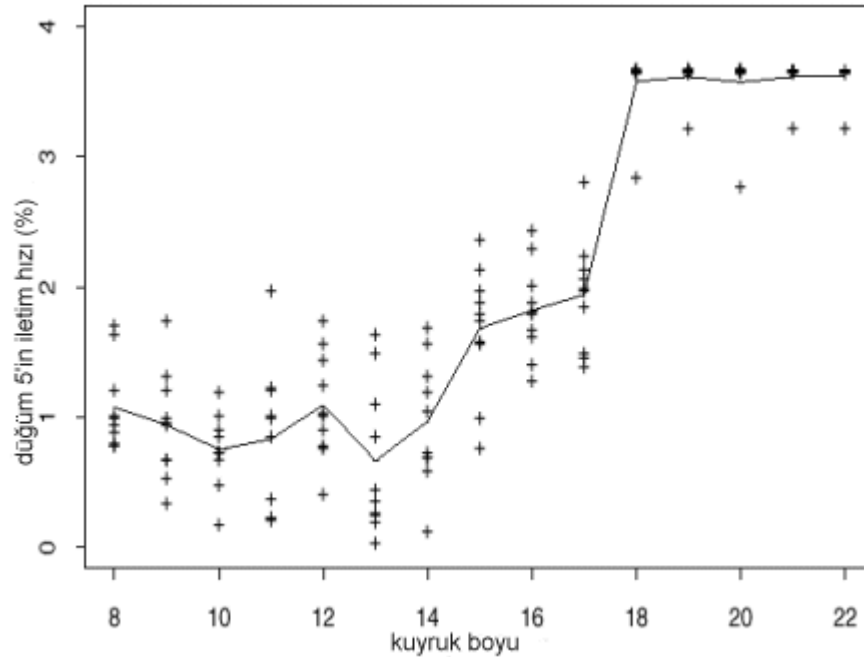
4. Patlamalı trafiğe de adil davranmak

w_q doğrudan izin verilecek patlamalı trafik miktarını belirlemektedir. w_q 'nın küçük seçilmesi anlık kuyruk boyunun, ortalama kuyruk boyuna daha az etkide bulunmasını ve bu sayede daha fazla patlamaya izin verilmesini sağlar.

Bu parametrelerin değiştirilmesi ile hedeflenenlerde gözlenen değişimler birbiri ile çelişmekte olduğundan ağda hangi isteğin önemli olduğuna karar vererek, buna uygun konfigürasyonu yapmak gerekmektedir.

3.4.2.5 RED'in başarımı

RED'in kuyruk boyunu kontrol etmede başarılı olduğu ve patlamalı iletim yapan trafiğe karşı daha adil davrandığı [49]'da yapılan benzetimler ile gösterilmiştir. Bu benzetimlerde RED'in DT ve ERD kuyruklara göre patlamalı trafiğe daha iyi olduğu görülmüştür.

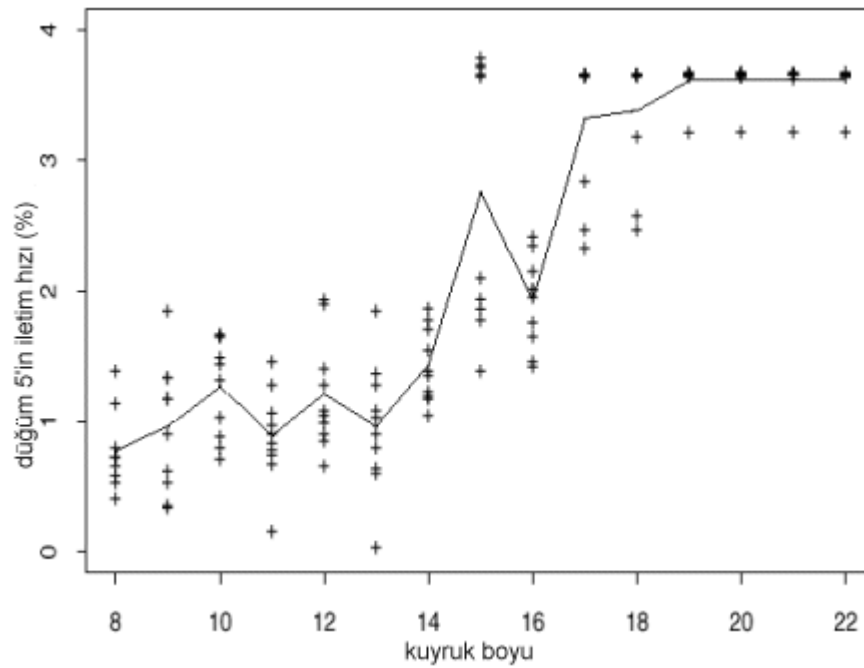


Şekil 3.7 DT kuyrukların patlamalı trafiğe davranışını inceleyen benzetim sonucu

Benzetimde 5 kaynak düğüm 45Mbps'lik bir darboğaz hat üzerinden tek bir hedef düğüm ile FTP koşturmaktadır. 5 numaralı kaynak düğümün patlamalı bir iletim gerçekleştirmesi için RTT'si diğer kaynakların yaklaşık 6 katı, ulaşabileceği en

yüksek TCP pencere değeri de 8 olarak ayarlanılmıştır. Diğer kaynakların en yüksek pencere değeri 12'dir.

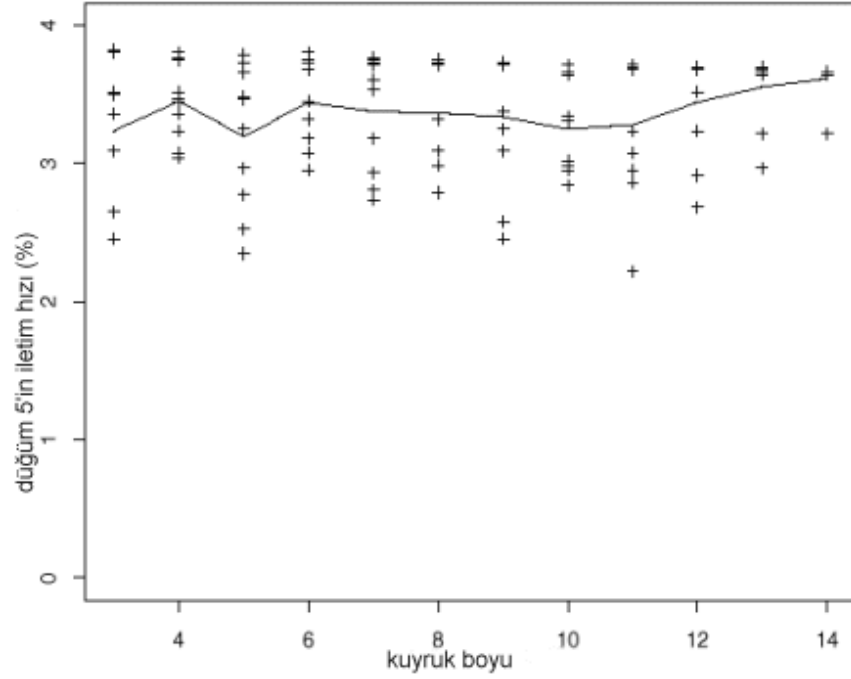
Şekil 3.7, Şekil 3.8 ve Şekil 3.9'da diğer kaynaklara göre daha patlamalı bir iletim gerçekleyen 5 numaralı kaynağın toplam iletim içinde yüzde olarak verimli iletim hızının (throughput) darboğaz hattında kullanılan kuyruk boyuna göre değişimi görülmektedir. RED ve diğer kuyrukları kuyruk boyları üzerinden kıyaslamak sağlıklı olmadığından, RED'in ortalama kuyruk boyunun diğer kuyruk tiplerinin kuyruk üst sınırına yakın olacağı şekilde kıyaslama yapılmıştır. Bu nedenle, RED grafiğinde x eksenini min_{th} parametresini göstermektedir.



Şekil 3.8 Rastantısal düşürmeli kuyrukların patlamalı trafiğe davranışını inceleyen benzetim sonucu

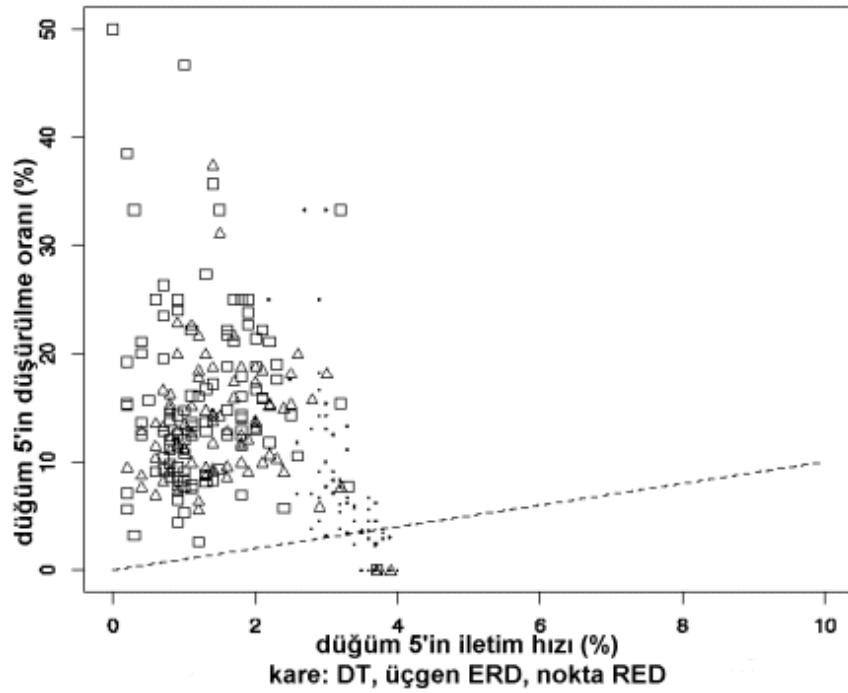
Sürekli çizgi 5 numaralı akışın ortalama iletim hızını göstermektedir. Benzetim 10 sn süresince koşturulmuştur ve grafiklerdeki her "+" işareti bir saniyelik süre için iletim hızını göstermektedir. Sürekli çizgi ise bu değerlerin ortalamalarından geçmektedir.

DT ve rastlantısal düşürmeli kuyruklarda 5 numaralı düğümden paket geldiğinde kuyruklar taşmaktadır, bu nedenle bu akış adil bir oran ile iletim yapamamaktadır. Ancak kuyruk boyu yeterince büyük olduğunda kuyrukların taşma durumu azaldığından patlamalı akış yeterince iletim yapabilmektedir. RED ise her kuyruk boyu için patlamalı trafiğe mümkün olduğunca eşit oranda iletim olanağı sağlamaktadır.



Şekil 3.9 RED kuyruklarının patlamalı trafiğe davranışını inceleyen benzetim sonucu

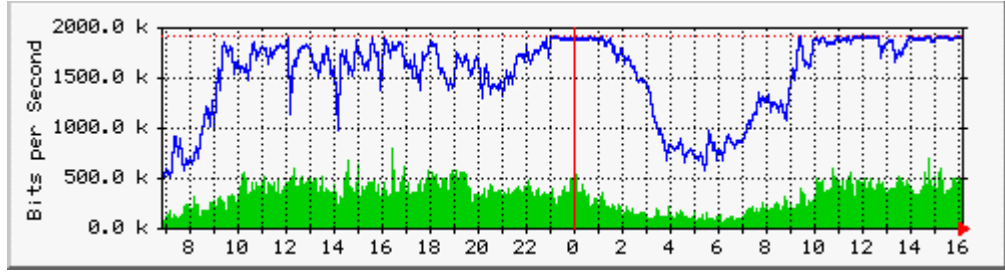
Değişik kuyruk yapılarının patlamalı akışı düşürme oranlarının gösterildiği **Şekil 3.10**'da RED'in patlamalı akışı, genel trafik içindeki payına uygun olarak düşürdüğü görülmektedir. Kesik çizilen sürekli çizgi akışın tam adil olarak düşürülmesi durumunda çıkacak olan çizgiyi göstermektedir.



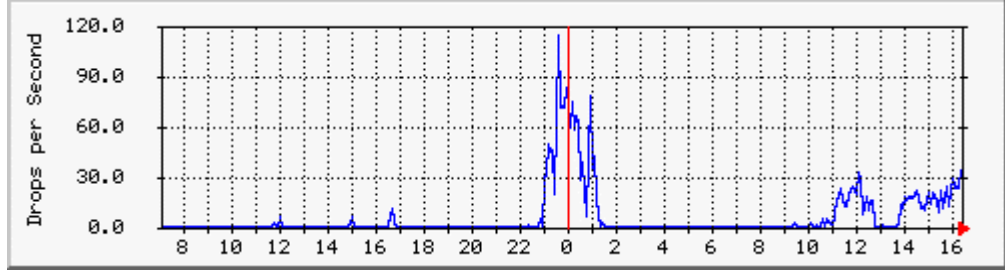
Şekil 3.10 RED, ERD ve DT patlamalı trafiği düşürme davranışları

RED'in gerçek bir ağ üzerinde çalıştırılması sonucunda elde edilen başarımların değerleri [62 ve 63]'te sunulmuştur. **Şekil 3.11'**den **Şekil 3.16'**e kadar olan grafikler EBONE şirketinin 1920Kbps'lik müşteri bağlantısının paket kayıp ve iletim hızlarını göstermektedir. Müşteriden gelen trafik içi dolu çizgi ile gösterilirken, müşteriye giden trafik ise sadece çizgi ile gösterilmiştir.

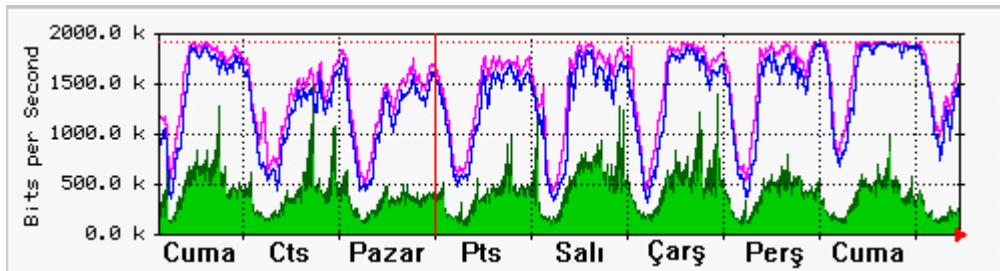
Şekil 3.11'de ilgili hattın Perşembe ve Cuma günleri için kapasite kullanım grafiği verilmiştir. Normalde DT kullanılan hatta, Cuma günü saat 10'da RED devreye sokulmuştur. Bağlantı üzerinde RED'in aktive edilmesi ile hat verimliliği %100'lere yaklaşmıştır. Aynı zaman aralığı için, saniyede düşürülen paket sayısını gösteren **Şekil 3.12'**de, RED'in DT'e karşı üstünlüğü görülmektedir. RED hat verimini %100'lere çıkartırken, aynı verim için DT'e göre çok daha az paket düşürmüştür.



Şekil 3.11 EBONE ağı – müşteri hattı için RED ve DT kullanılan günlerin hat kullanımı [63]

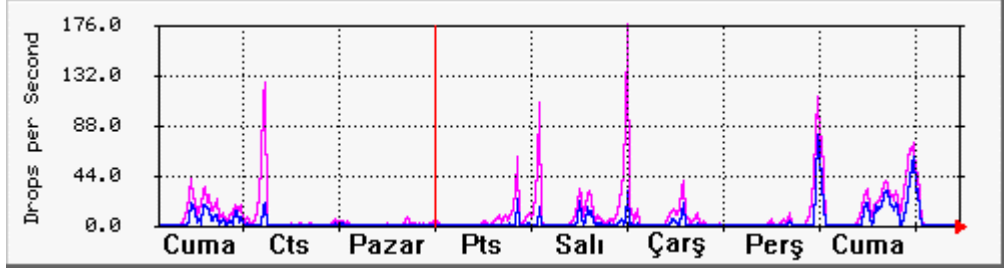


Şekil 3.12 EBONE ağı – müşteri hattı için RED ve DT kullanılan günlerin paket düşürme oranları [63]



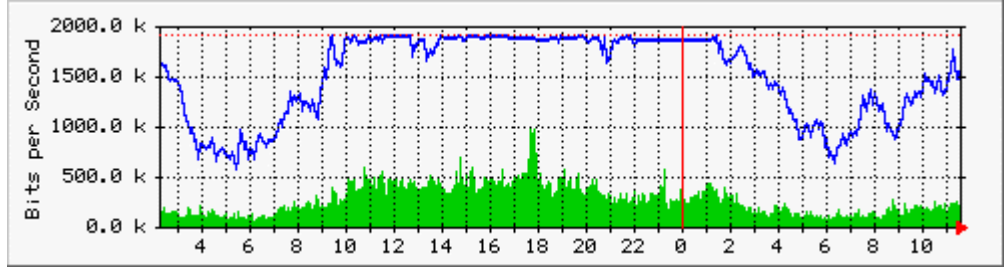
Şekil 3.13 EBONE ağı – müşteri hattının bir haftalık hat kullanımı [63]

Perşembe gününün özel olarak kötü bir gün olmadığı da aynı grafiklerin bir haftalık zaman aralığı için olan **Şekil 3.13** ve **Şekil 3.14**'te görülmektedir. Hat, DT kullanılan diğer günlerde de Perşembeye benzer şekilde davranmıştır.

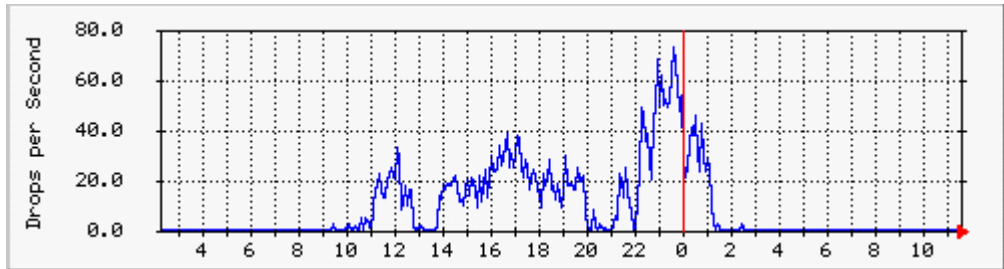


Şekil 3.14 Ebone ağ – müşteri hattının bir haftalık paket kaybı grafiği [63]

Cumartesi tüm gün RED koşturulduğunda elde edilen grafikler ise **Şekil 3.15** ve **Şekil 3.16**'da görülmektedir.



Şekil 3.15 Ebone – müşteri hattının sürekli olarak RED koşturulan Cumartesi günü hat kullanım grafiği [63]

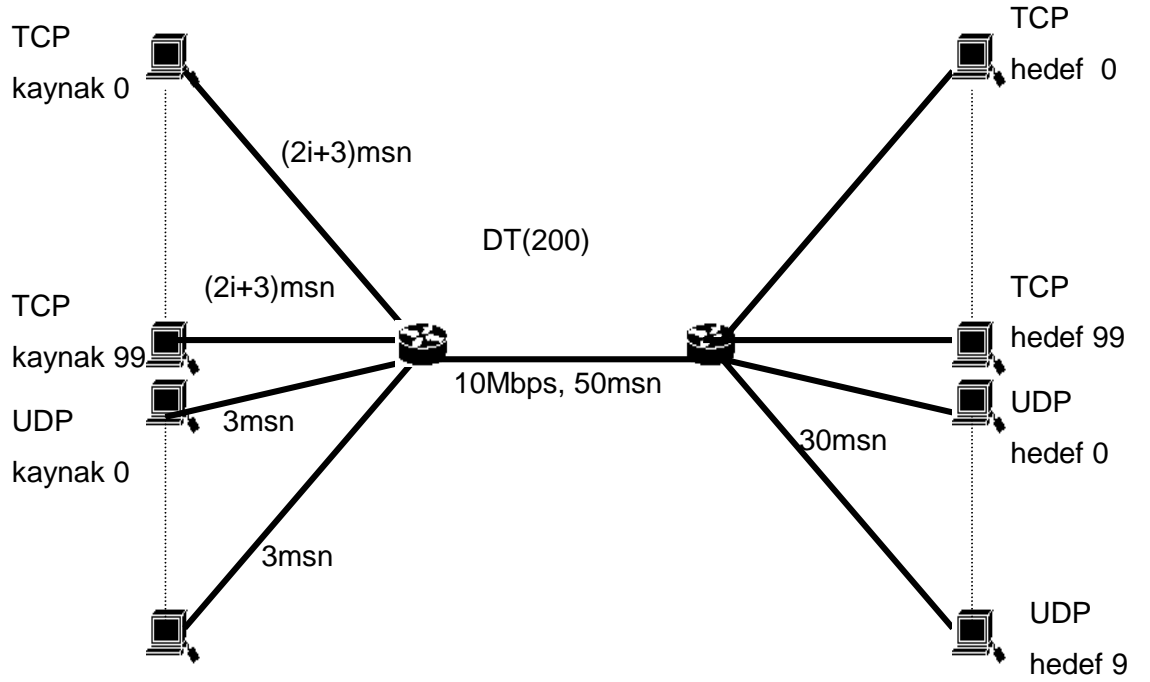


Şekil 3.16 Ebone – müşteri hattının sürekli olarak RED koşturulan Cumartesi günü paket kayıp grafiği [63]

[64]'te, [63]'te yapılan ölçümlerin sadece yönlendirici başarımını gösterdiği ve bunun uçtan uca iletim verimliliğini yansıtamayacağı iddia edilmektedir. Gerçek bir ağda yapılan ölçümlerde de iddialara paralel olarak RED kullanılan yapının DT kullanılan yapıya göre uçtan uca verimlilik adına çok büyük bir üstünlük sağlamadığı gösterilmektedir.

Analitik ve benzetim çalışmaları ile RED'in patlamalı trafiğe karşı gösterdiği davranışı [58]'de incelenilmiştir. May, RED'in patlamalı trafiğe diğer kuyruk tiplerine göre daha adil davrandığını ancak bunu patlamalı trafik için paket düşürme oranını azaltarak değil, normal trafiğin düşürülme oranını artırarak yaptığını göstermiştir.

Şekil 3.17'deki gibi bir ağ yapısı üzerinde darboğaz yaratan hatta RED ve DT kullanarak yapılan simülasyonda da May'ın bulduklarının doğruluğu görülmüştür.

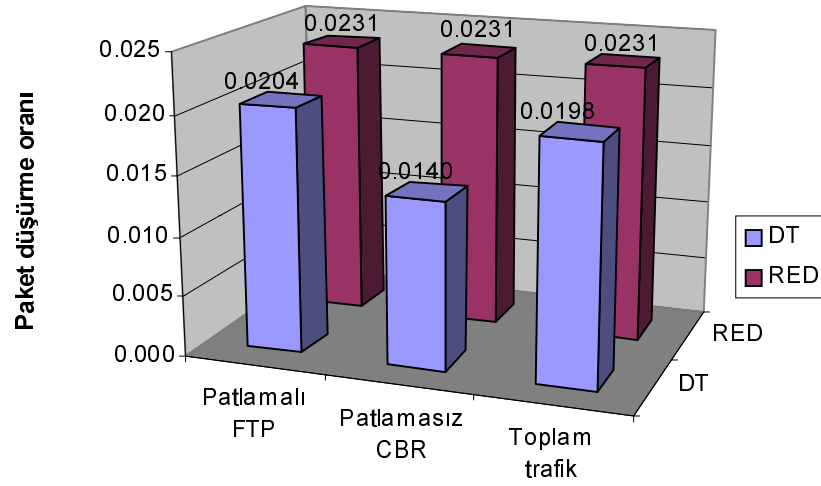


Şekil 3.17 RED'in patlamalı trafiğe davranışını incelemek için yapılan benzetim ağı konfigürasyonu

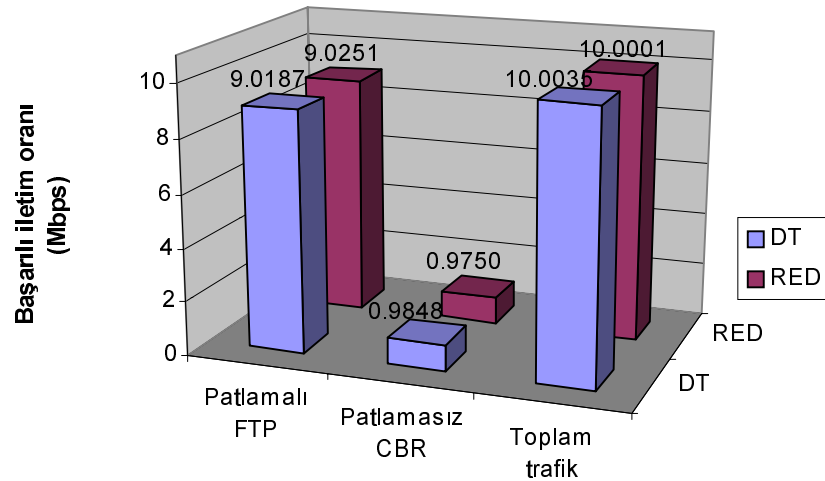
TCP üzerinden iletim yapan 100 adet ftp kaynağı ile UDP üzerinden iletim yapan 10 CBR kaynağı 10Mbps kapasitesinde ve 50msn gecikmesine sahip olan darboğaz hattı üzerinden karşıdaki 110 hedef düğüm ile konuşmaktadır. Hattın kapasitesinin %10'unu tüketecek şekilde ayarlanılan CBR kaynakları patlamasız bir trafik oluşturmaktadır ve sonsuz uzunluktaki bir dosyanın iletimini yapan ftp kaynakları da patlamalı iletim yapmaktadır. TCP kaynaklarını daha da patlamalı yapabilmek için TCP'nin maksimum pencere boyu 8 ile sınırlandırıldı. ftp kaynaklarını darboğaz hatta bağlayan hatların gecikmeleri $2 * i + 3$ 'tür, $i:0..99$. Hedef düğümlerin darboğaz hattına bağlantıları ise 30msn gecikmeye sahiptir. CBR kaynaklarının darboğaz hattına bağlantısı 3msn gecikmeye sahip iken, hedef düğümleri 30msn gecikmeye sahiptir.

Aynı ağda, darboğaz hattında RED ve DT kuyruk yönetimi yapılarının kullanılması ile tablolardaki veriler elde edilmiştir. 1Kbyte'lık paketlerin kullanıldığı benzetimde, DT için kuyruk boyu olarak 200 paket alınmıştır. RED parametreleri ise, $min_{th} = 50$, $max_{th} = 150$, kuyruk boyu = 200, $w_q = 0,02$, $max_p = 0,1$ olarak kullanılmıştır.

Şekil 3.18'de de görüldüğü gibi, RED gerçekten de patlamalı ve patlamasız akışları eşit oranlarda düşürmektedir. Bu daha önce de [49]'da gösterilmişti. Ancak burada dikkat çekici olan bu eşitliğin, patlamasız trafiğin düşürülme olasılığının artırılması ile sağlandığıdır.



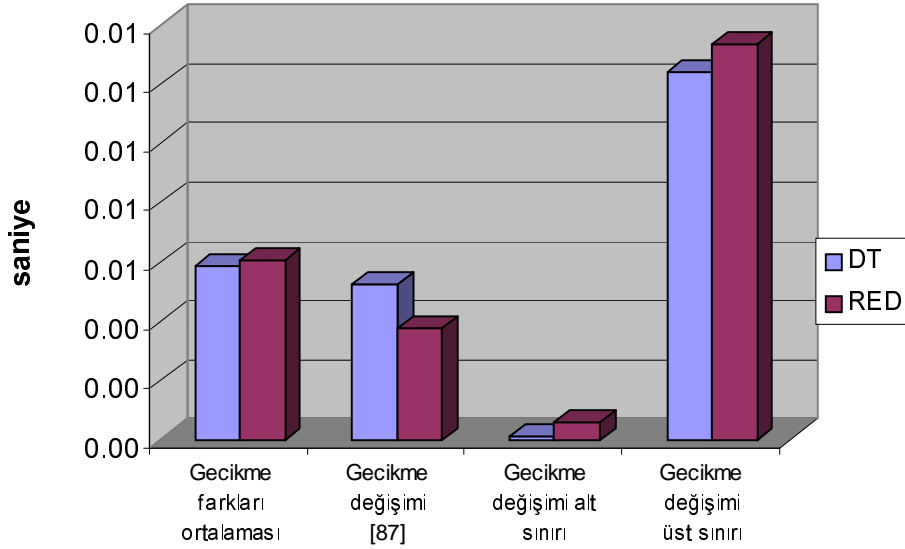
Şekil 3.18 RED ve DT kuyrukların patlamalı ve normal trafik kaynakları için paket düşürme oranları



Şekil 3.19 RED ve DT kuyrukların patlamalı ve normal trafik kaynakları için başarılı iletim hızları

RED'in kuyruk boyunu kontrol etmedeki başarısı May tarafından da gözlenilmiştir. Ancak RED bunu yaparken gecikme değişiminin (jitter) artmasına neden olmaktadır. Denklem B.4 ile yapılan hesaplama sonucunda RED gecikme farkı 0,003764 ve DT gecikme farkıda 0,005225 olarak bulunmaktadır. Ortalama RED paket iletim gecikmesi 0,178239 ile DT'in ortalama gecikmesinin (0,210449) %85'i iken, RED'in paket gecikme değişimi de DT kuyruğunun altındadır.

Ancak, RED gecikme farkı zaman içinde incelendiğinde çok dalgalı hareket ettiği görülmektedir. [82]'deki formülasyon ile zaman içinde hesaplanan gecikme değişimi 0,000572 ile 0,013350 arasında değişmektedir. DT gecikme değişimi ise 0,000143 ile 0,012413 değerleri arasındadır. Benzetim sonlarında gecikme değişiminin RED için düşük olduğu zamanlardır. Bu nedenle gecikme farkı ortalamalarına bakarak DT'in genel olarak daha düşük gecikme değişimine sahip olduğunu düşünebiliriz.



Şekil 3.20 RED ve DT gecikme değişimi

Ortalama gecikmenin düşürülmesi genel olarak faydalı olsa da, gecikme değişiminin artması birçok gerçek zamanlı uygulama için sorun yaratabilmektedir.

RED ortalama kuyruk boyunu sürekli olarak kuyruğun sınırlarının altında tutmaya ve böylece yeni paketler için her zaman yer kalmasına olanak sağladığından, kuyruk boyu hem artabilmekte, hem de azalabilmektedir. Ancak DT kuyruklarda kuyruk dolu veya doluluğa yakın bulunduğundan, kuyruk boyu ancak azalabilmektedir. RED'in DT'ye göre daha fazla gecikme değişimine neden olmasının ana nedeni budur.

3.4.3 Uyarlanabilir RED

RED gibi aktif kuyruk yönetimi mekanizmalarının zayıf yönü, tıkanıklığa verdikleri tepkinin yönlendirici üzerindeki aktif akış sayısına bağlı olmamasıdır. Ancak tıkanıklığın yönlendirici üzerindeki aktif akış sayısı ile orantılı miktardaki akışa bildirilmesi sağlanmalıdır.

10Mbps kapasitesindeki bir darboğaz hattının bir miktar akış tarafından eşit olarak paylaşıldığını düşünelim. 100 TCP akışı bulunduğunda, bir akışa ECN ile paket işaretlenilmesi veya paket düşürülmesi ile tıkanıklık olduğu bilgisinin gönderilmesi sonucunda toplam yük sadece 1/200 oranında düşerek 9,95Mbps olacaktır. Aynı hat üzerinde sadece iki akışın aktif olduğu durumda ise, bir akışın tıkanıklık nedeni ile hızını yarılaması yükün 7,5Mbps'e düşmesine neden olmaktadır.

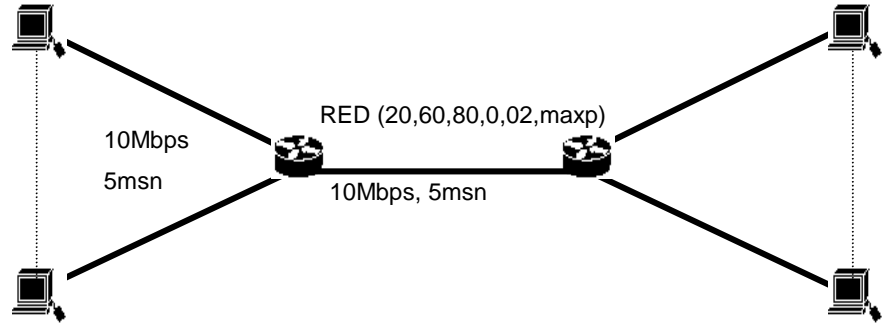
Bu durumu genellersek, N adet akışın bir darboğaz hat üzerinde aktif olduğu durumda, aktif kuyruk yönetimi mekanizması bir akış paket düşürme veya işaretleme ile tıkanıklıktan haberdar ettiğinde, hattın yükü $1-1/2N$ oranında düşürülecektir.

RED aktif akış sayısı hakkında bilgi sahibi olmadığından, agresif olmayan bir RED konfigürasyonu az sayıdaki akış için başarılı bir şekilde kuyruk boyunu kontrol edebilirken, akış sayısının artırılması ile bu konfigürasyon başarısız olacaktır. Benzer şekilde çok sayıda akış için agresif bir RED konfigürasyonu yapıldığında kuyruk boyu başarılı bir şekilde kontrol edilecektir. Ancak akış sayısının azalması ile gereğinden fazla akış düşürüleceğinden hattın verimsiz kullanımı ortaya çıkacaktır. Hat üzerindeki akış sayısı sabit ve önceden bilinen olmadığından konfigürasyon da uygun şekilde ayarlanamaz.

RED'in bu davranışını daha iyi anlayabilmek için farklı sayıda düğümün tek bir darboğaz hat üzerinden aynı sayıdaki hedef düğüme bağlandığı **Şekil 3.21**'deki gibi bir ağ üzerinde benzetim yapıldı. Hedef ve kaynak düğümler darboğaz hatta 10Mbps kapasitesinde ve 5msn gecikmeye sahip hatlar ile bağlanmıştır. Darboğaz hattın kapasitesi ise 10Mbps ve gecikmesi 5msn'dir. Darboğaz hattının giriş kuyruğu RED koşturmaktadır. RED parametresi olarak $min_{th} = 20$, $max_{th} = 60$, $kuyruk\ limiti = 80$, $w_q = 0,002$ seçilmiştir. Max_p olarak 0,01 ve 0,1 seçilerek benzetimler yapılmıştır.

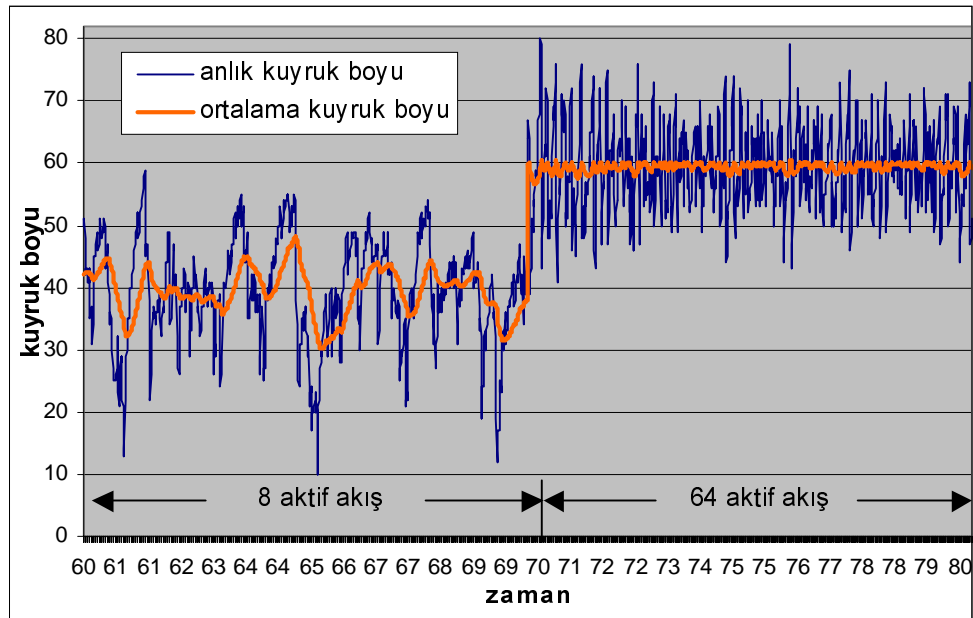
Şekil 3.22'de RED kuyruğunun ortalama ve anlık kuyruk boyunu göstermektedir. Benzetim 100 sn koşturulmuştur. **Şekil 3.22**'de 8 akışın aktif olduğu benzetimden

60-70 zaman aralığını, 64 akışın aktif olduğu benzetimden 70-80 zaman aralığını göstermektedir.



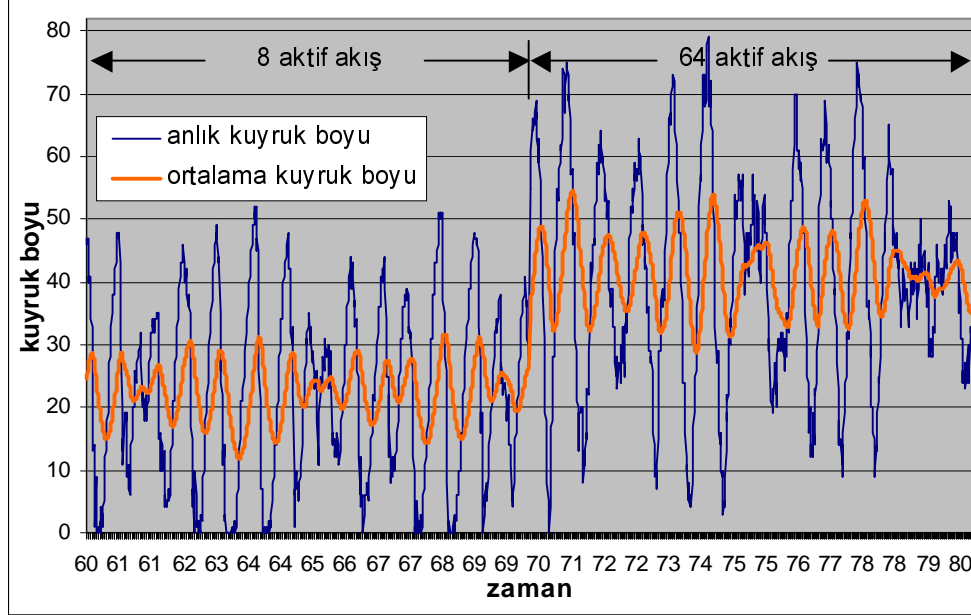
Şekil 3.21 RED'in akış sayısına tepki veremediğini gösteren benzetim için ağ yapısı

Bu benzetimlerde $max_p = 0,01$ olarak seçilmiştir. Şekilden de görüldüğü üzere agresif olmayan bir konfigürasyon 8 akış için başarılı bir şekilde kuyruk boyunu kontrol edebilmekte ve min_{th} ile max_{th} aralığında tutabilmektedir. Ancak 64 akış için bu konfigürasyon yetersiz kalmakta ve kuyruk boyu kontrol edilememektedir. Ortalama kuyruk boyu sürekli max_{th} değerine yakın kalmaktadır. Hem bu nedenden, hem de anlık kuyruk boyu kuyruk sınırlarına dayandığından paket kaybı çok olmaktadır.



Şekil 3.22 RED'in $max_p=0,01$ için 8 ve 64 aktif akış için kuyruk davranışı

Şekil 3.23 ise daha agresif bir RED konfigürasyon ile koşturulan benzetime ilişkin kuyruk boyunu göstermektedir. Konfigürasyonu agresif hale getirebilmek için $max_p = 0,1$ olarak seçilmiştir.



Şekil 3.23 RED'in $max_p=0,1$ için 8 ve 64 aktif akış için kuyruk davranışı

Bu konfigürasyon 64 akış için uygun sonuçlar verirken, 8 akış için gereğinden fazla paketin düşürülmesine ve bu nedenle hattın zaman zaman boş kalarak verimsiz kullanılmasına neden olmaktadır.

Bu sorunun çözümü için [59]'da RED parametrelerinden max_p 'nin dinamik olarak değiştirilmesi önerilmektedir. Uyarlanabilir RED (Adaptive RED, ARED) olarak isimlendirilen bu yapı sayesinde hem az sayıda akış için agresif bir konfigürasyon ile hattın verimsiz kullanılması önlenmekte, hem de çok sayıda akış için yeterince agresif olmayan konfigürasyonlarla kuyruk boyunun kontrolden çıkmasının önü kesilmiş olmaktadır.

3.4.3.1 ARED algoritması

Yukarıdaki benzetimde de gösterildiği gibi RED parametrelerinin yüke göre değiştirilmesi akışın başarımı üzerinde olumlu etki yapacaktır. ARED, RED konfigürasyonunu ortalama kuyruk boyunun değişimine göre ayarlayarak RED'in gerektiğinde agresif olması sağlamaktadır. Ortalama kuyruk boyunun min_{th} civarında dolaşması konfigürasyonun gereğinden fazla agresif olduğu anlamına gelirken, bunun max_{th} etrafında dolaşması da konfigürasyonun yeterince agresif olmadığı

anlamına gelmektedir. Detaylı algoritması **Şekil 3.24**'te verilen ARED gözlenen bu değişimlere karşı max_p parametresini sabit α ve β katsayıları ile ayarlayarak RED'in gerektiği kadar agresif olmasını sağlamaktadır.

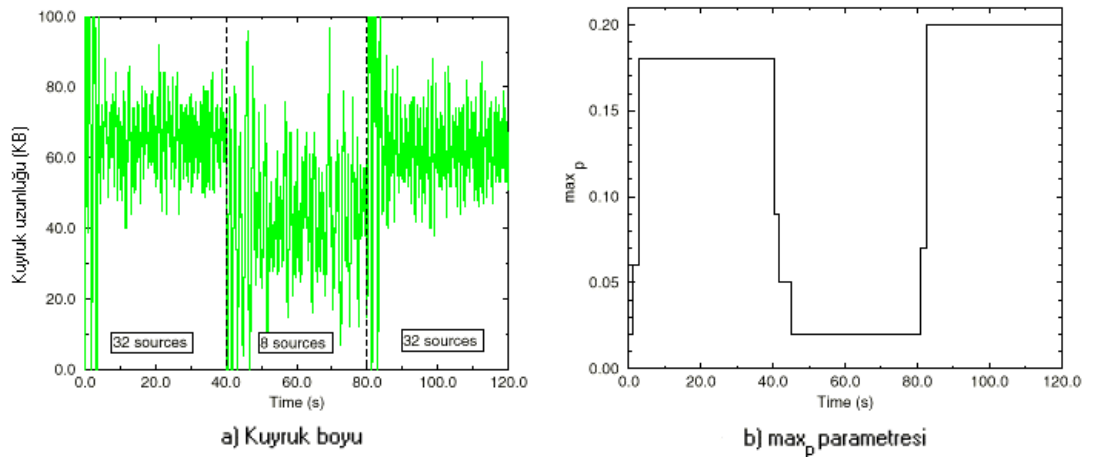
```

Her Qave güncellenmesinde:
    if (minth < Qave < maxth)
        status = between;
    if (Qave < minth && status != below)
        status = below;
        maxp = maxp /  $\alpha$ ;
    if (Qave > maxth && status != above)
        status = above;
        maxp = maxp *  $\beta$ ;

```

Şekil 3.24 ARED algoritması

Algoritmanın yeterliliğini göstermek için yapılan benzetimlerle ARED'in RED parametrelerini gerektirdiği şekilde değiştirebildiği ve değişik ağ yüklerinde bile kuyruk boyunu gerektiği gibi kontrol edebildiği **Şekil 3.25**'de gösterilmiştir. **Şekil 3.25a**'da 32 akışın aktif olduğu ARED koşturan bir yönlendiricide 40-60 saniye aralığında akış sayısının 8'e düşürülmesi ile yapılan benzetim süresindeki kuyruk boyu değişimi görülmektedir. ARED **Şekil 3.25b**'de görüldüğü gibi RED'in max_p parametresini uygun şekilde ayarlayarak kuyruğun boş kalmasını veya üst sınırlara dayanmasını engellemektedir.



Şekil 3.25 ARED kullanıldığında değişik sayıda aktif akış için RED'in duruma uyarlanabilmesi

3.4.4 RED üzerine diğer çalışmalar

[65]'te RED algoritmasını geliştiren Jacobson'ın da içinde bulunduğu grup tarafından kuyruk yönetiminin detaylı olarak analiz edilmesi ve bunun sonucunda RED algoritmasına önerdikleri değişiklikler yer almaktadır.

[66]'da ise hat kapasitesinin akışlar arasında adil bir şekilde paylaşılmasını sağlayan uygulaması basit ve ölçeklenebilir bir algoritma sunulmaktadır.

3.4.5 BLUE

RED tıkanıklığının boyutunu kuyruk boyuna göre algılamaktadır. [67] kuyrukların dolu olması tıkanıklık olduğu anlamına gelse de kuyruk boyu tıkanıklığının boyutu hakkında bilgi vermediğini iddia etmektedir. Yoğun veri ileten tek bir akış bile hattın tüm kapasitesini tüketerek kuyruğun dolmasına neden olabilir.

Yine [67]'ya göre RED tıkanıklığının boyutunu anlamak için kuyruk boyunu kullandığından sağlıklı çalışmaz. Bunun yanında çok sayıda TCP akışının aktif olduğu bir yönlendiricide kuyruk boyu **Şekil 3.30a**'da gösterildiği gibi son derece salınımlı bir şekilde değişmektedir. RED'in sağlıklı çalışabilmesi için birçok parametrenin ağırlık yapısı ve trafik yapısına göre ayarlanması gerekmektedir. Bu sorun Feng dışında [57] ve [58] gibi çalışmalarda da dile getirilmiştir.

Tıkanıklık nedeni ile paket düşürüldükten sonra, kaynağın buna verdiği tepki tıkanıklık oluşan düğüme ulaşana dek olan trafiği tutabilmek için RED yeterince büyük kuyruklara ihtiyaç duyar.

BLUE, RED'in kuyruk boyuna göre düşürme olasılığını değiştirmesine alternatif olarak tek bir düşürme olasılığı tutmaktadır ve bu olasılığı kuyruğun paket düşürme ve hat kullanım geçmişine bağlı olarak değiştirmektedir. Eğer kuyruk boyu sürekli limitlerde dolaştığı için paketler düşürülmekte ise, BLUE bu olasılığı arttırmaktadır. Ters durumda kuyruk sürekli boş kaldığı için hattın kapasitesi tam olarak kullanılamıyor ise paket işaretleme olasılığı düşürülür.

BLUE tıkanıklığı kaynak düğüme haber vermek için RED gibi paket düşürme veya ECN ile paketleri işaretlemeyi kullanabilmektedir.

3.4.5.1 BLUE algoritması

BLUE bilinen tüm kuyruk yönetimi mekanizmalarının tersine kuyruk yönetimini paket düşürme ve hattın kullanım geçmişine dayanarak yapmaktadır. BLUE paketleri işaretlemek için tek bir p_m olasılığı tutmaktadır ve bu olasılık kuyruğun taşması veya boş kalmasına göre değiştirilmektedir. Kuyruk sürekli dolu olduğundan paketler düşürülüyorsa mevcut işaretleme olasılığının tıkanıklığı önlemek için yeterli olmadığı anlaşılmaktadır. Bu nedenle p_m olasılığı artırılır. Hattın trafik profiline göre paketler yüksek olasılıkla işaretlenilmekte ise p_m olasılığı azaltılır. Bu yapı BLUE'nun mevcut trafik profiline göre nasıl işaretleme yapması gerektiğini öğrenmesini sağlar.

Şekil 3.26'da BLUE algoritması görülmektedir. Algoritmanın 3 parametresi bulunmaktadır. Bunlardan ilki olan *freeze_time* parametresi kuyruktaki anlık değişimlerin gözardı edilmesini ve kuyruktaki değişimlere gereğinden fazla tepki verilmesini önlemek için kuyruğun belirli aralıklarla güncellenmesini sağlar. Bu parametre şu an sabit değerdir, ancak Feng evrensel eşzamanlılığı önlemek için bu parametrenin rastlantısal olarak değiştirilmesi gerektiğini ifade etmiştir.

```
Paket kaybında (veya  $Q_{len} > L$ ):  
    if ((now – last_update) > freeze_time )  
         $p_m = p_m + d_i$ ;  
        last_update = now;  
Link boş kaldığında  
    if ((now – last_update) > freeze_time )  
         $p_m = p_m + d_d$ ;  
        last_update = now;
```

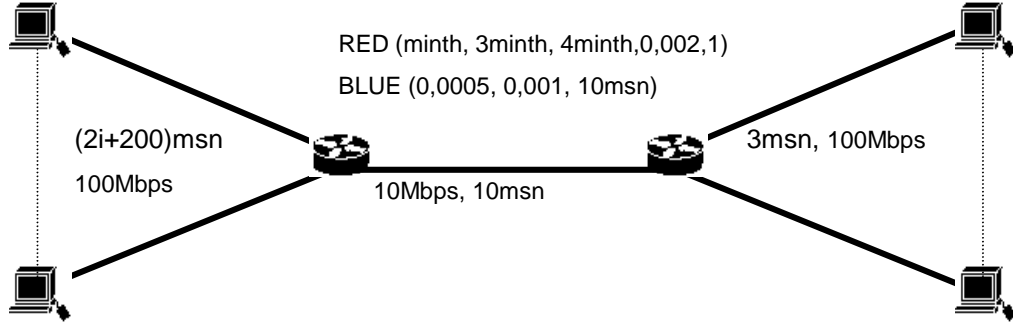
Şekil 3.26 BLUE algoritması

Diğer parametreler ise paket işaretleme olasılığının arttırma ve azaltma değerleri olan d_i ve d_d 'dir. d_i paket işaretleme olasılığı arttırılırken kullanılırken, d_d azaltılırken kullanılır. BLUE algoritması son derece basit görünmesine karşın benzetimlerde başarılı olduğu görülmüştür.

3.4.5.2 BLUE'nun başarımı

BLUE'nın başarımlarını değerlendirmesini yapmak için Feng'in [67]'de yaptığı benzetimlere benzer bir ağ yapısı kurulmuştur. Ağ yapısı **Şekil 3.27**'de verilen benzetimde 20 düğüm üzerinden 50şer TCP kaynağı darboğaz hat üzerinden

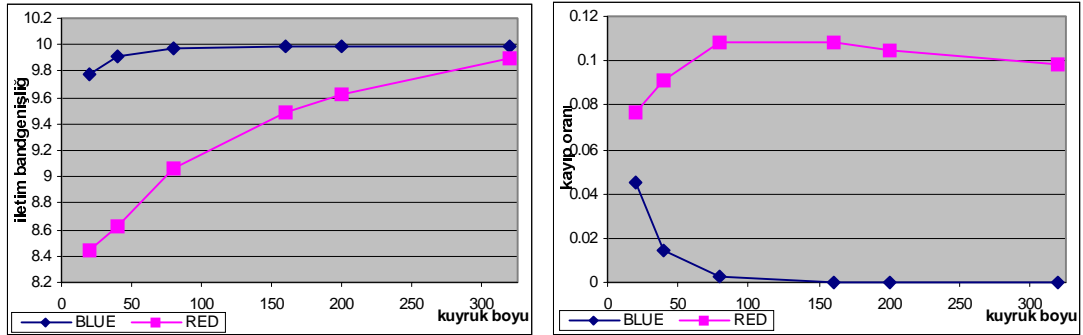
karşıdaki hedef düğümlere bağlanmıştır. Kaynak uygulaması olarak 2msn kapalı 3msn açık iletim yapan Pareto trafik kaynakları kullanılmıştır. ECN'in aktive edildiği yapıda darboğaz hattında RED ve BLUE kullanarak verimli iletim hızı ve düşürme oranları kıyaslanıldı. RED için $max_{th} = 3min_{th}$ ve $q_{lim} = 4min_{th}$ seçilirken. max_p ise bu sayıdaki kaynak için en uygun değer olan 1 seçilmiştir. BLUE için ise $d_i = 0,0005$ ve $d_d = 0,001$ ve $freeze_time = 10msn$ seçilmiştir.



Şekil 3.27 BLUE benzetim ağı

Kaynakları ilk yönlendiriciye bağlayan hatların gecikmesi değiştirilerek ($2i+200$ msn ve $2i+5$ msn) düşük ve büyük RTT için iki ölçüm yapılmıştır. Büyük RTT'ye sahip kaynakların değişik kuyruk boyları için yapılan ölçümlerinde BLUE belirgin şekilde daha iyi başarımlar göstermektedir. RED ancak kuyruk boyu çok büyütüldüğünde BLUE'nın başarımına yaklaşabilmektedir.

Paket düşürme oranlarının kıyaslanıldığı Şekil 3.28'de BLUE'nın paket düşürme oranının 0'a yakınsadığı görülmektedir. Buna karşın RED %10lar civarında paket düşürme oranına sahiptir.



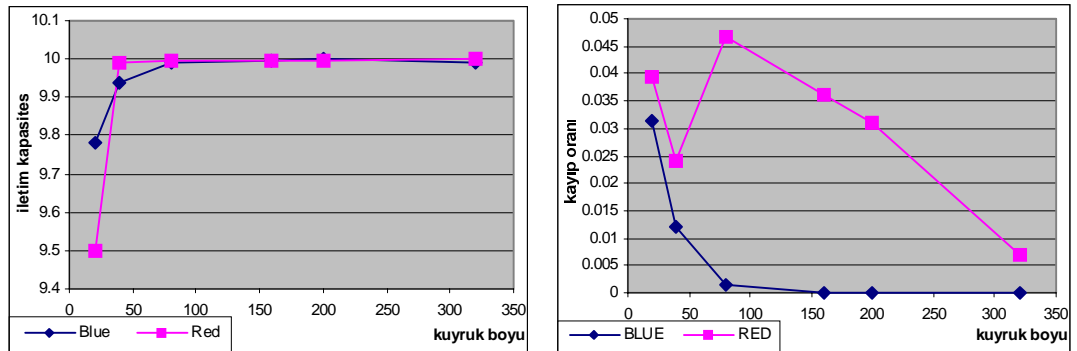
Şekil 3.28 Büyük RTT'ler için RED ve BLUE verimli iletim hızlarının kayıp oranlarının kıyaslanması

Bu kıyaslamaların ışığında yönlendiricilerde BLUE kullanılması hem daha az düşürme oranlarının hem de daha yüksek verimli iletim hızlarının daha küçük kuyruklarla elde edebilmesini sağlayacağı görülmektedir.

Ancak ağ konfigürasyonunda uçtan uca olan gecikme azaltıldığında BLUE, RED'e karşı bu üstünlüğünü koruyamamaktadır. **Şekil 3.27**'deki ağ yapısında kaynak düğümlerin darboğaz hatta olan bağlantıların gecikmesi $(2i+5)$ msn şeklinde değiştirilerek aynı ölçümler tekrar alınmıştır.

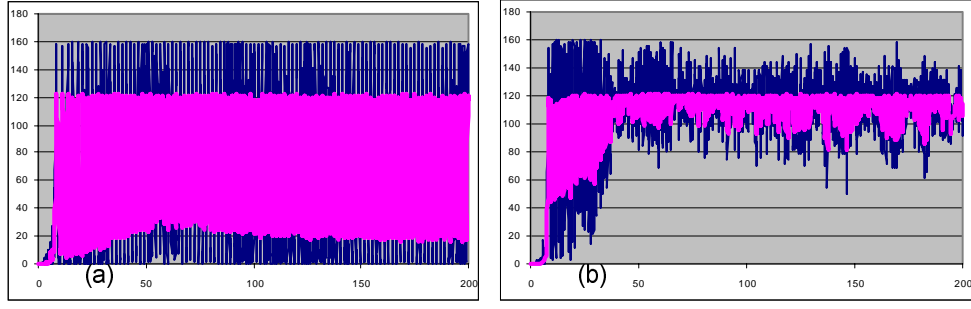
Gecikmenin düşmesi ile RED **Şekil 3.29**'da görüldüğü , BLUE'ya karşı daha yüksek iletim hızına ulaşmaktadır. Ancak BLUE kayıp oranı kıyaslamasında hala RED'e karşı üstünlüğünü korumaktadır. BLUE'nın iletim hızı olarak RED'e yaklaşmasının gerçekleştiği kuyruk boylarında, BLUE'nın paket düşürme yüzdesi RED'e göre çok büyük oranda azdır.

Ancak bu farklılık BLUE'nın daha kötü davranmasından değil, RED'in RTT'nin küçülmesi ile kuyruktaki tıkanıklığa daha kolay tepki verebilir hale gelmesinden kaynaklanmaktadır. **Şekil 3.30**'da darboğaz hattında kullanılan RED kuyruğunun zaman içinde değişimi görülmektedir. **Şekil 3.30a**'da büyük RTT için yapılan ölçümü göstermektedir. RED algoritması kuyruğu kontrol etmekte başarısız olmakta ve bu nedenle kuyruk alt ve üst sınırlar içinde salınmaktadır. Hat zaman içinde boş kaldığından iletilen paket sayısı düşmektedir.



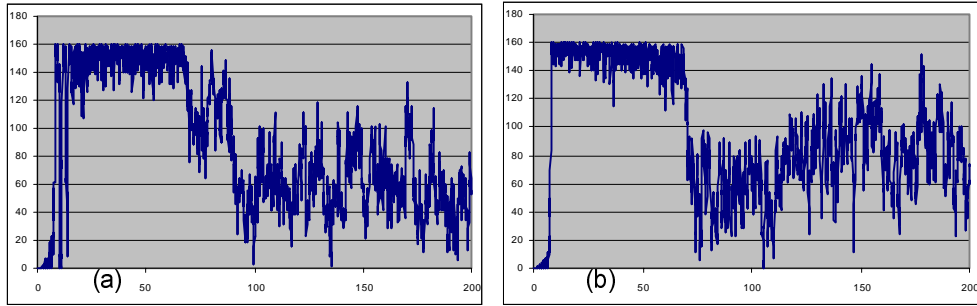
Şekil 3.29 Küçük RTT'ler için RED ve BLUE verimli iletim hızlarının kayıp oranlarının kıyaslanması

Şekil 3.30b'de ise RTT düşürüldüğünde de RED'in yine kuyruk boyunu kontrol edemediği görülmektedir. Ancak bu kez akışların patlama oranları azaldığından kuyruk salınım yapmamaktadır. Kuyruk kontrol edilemediğinden bir DT kuyruk gibi davranmaktadır.



Şekil 3.30 Büyük ve küçük RTT için kuyruk boyu 160 paket iken RED kuyruğu

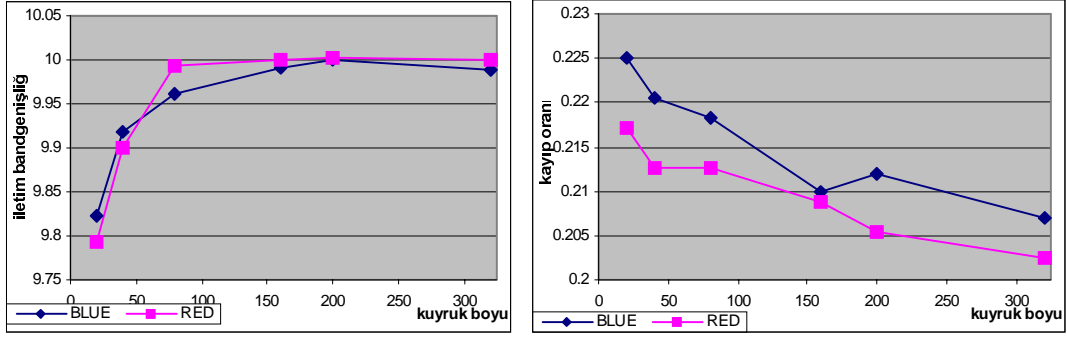
Şekil 3.31'de ise aynı yapıda BLUE kullanıldığında kuyruk boyunun zaman ile değişimi görülmektedir. BLUE hem küçük hem de büyük RTT'li durumlar için kuyruk boyunu kontrol etmekte başarılıdır. Benzetimin ilk 70 saniyesi süresince sürekli yeni akışlar iletme başladığından kuyruk boyu kontrol edilememektedir. Ancak bu süre sonunda BLUE paketleri nasıl işaretlemesi gerektiğini öğrenmekte ve kuyruğu başarılı bir şekilde kontrol eder hale gelmektedir.



Şekil 3.31 Büyük ve küçük RTT için kuyruk boyu 160 iken BLUE kuyruk boyu

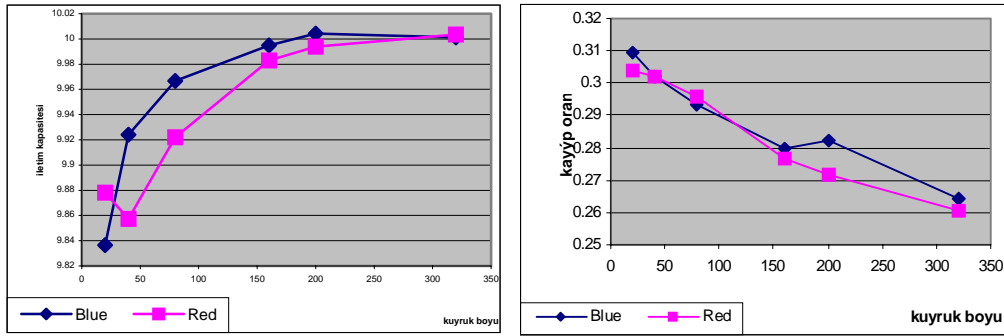
ECN kullanılmadığında ise BLUE RED'e karşı olan üstünlüğünü koruyamamaktadır. BLUE'nın başarısı Feng tarafından da ECN kullanılan bir benzetim ile gösterilmiştir. Ancak ECN kullanılmadığında BLUE RED ile aynı hatta bazen daha kötü sonuçlar vermektedir. Feng tarafından da raporlanılmayan ECN desteksiz BLUE kullanımı için yine **Şekil 3.27**'deki ağ konfigürasyonu kullanılarak benzetim gerçekleştirildi.

Şekil 3.32'de **Şekil 3.28**'dekine benzer şekilde yüksek RTT'ye sahip ağ konfigürasyonu için RED ve BLUE'nun iletim hızları ve paket kayıp oranları görülmektedir. BLUE iletim hızında RED ile yaklaşık aynı sonuçları vermektedir. Her iki kuyruk yönetim yapısı da fazla sayıdaki akış ile büyük kayıp oranlarına neden olurken, BLUE bir miktar daha fazla kayba neden olmaktadır. **Şekil 3.33**'te görüldüğü gibi akışların RTT'si düşürüldüğünde de durum çok değişmemektedir.

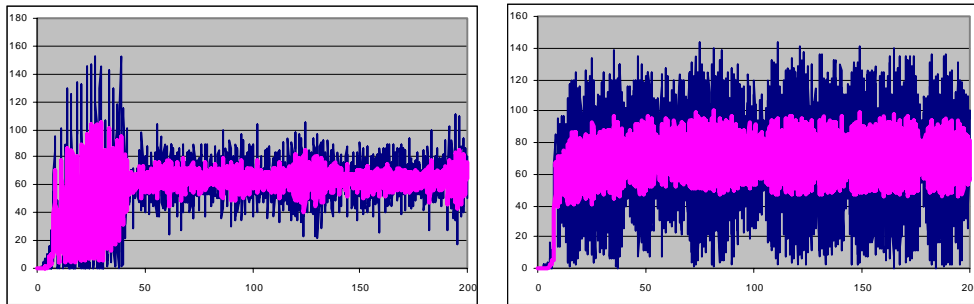


Şekil 3.32 ECN kullanılmadığında yüksek RTT'li ağ yapısında RED ve BLUE iletim hızları ve kayıp oranları

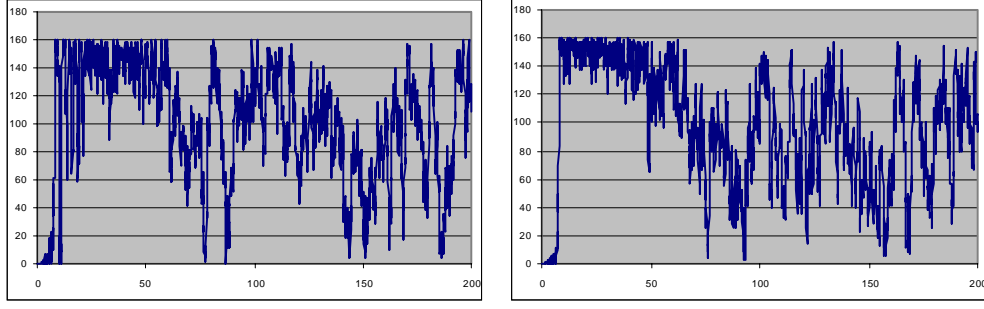
ECN kullanıldığında RED kuyruk boyunu kontrol edememekteydi. ECN kullanılmadığında ise bu sorun çözülmektedir. **Şekil 3.34**'ten RED'in max_p parametresinin daha düşük seçilebileceği görülüyor. Ancak ECN kullanılmadığı durum ile eş koşullarda kıyaslama yapılması için bu parametre yine 1 olarak kullanılmıştır. Bu olasılık düşürüldüğünde RED'in paket düşürme oranı bir miktar daha düşmelidir. **Şekil 3.35**'te BLUE'nın yine kuyruk boyunu kontrol etmede başarılı olduğu görülmektedir.



Şekil 3.33 ECN kullanılmadığında düşük RTT'li ağ yapısında RED ve BLUE iletim hızları ve kayıp oranları



Şekil 3.34 ECN kullanılmadığında büyük ve küçük RTT için kuyruk boyu 160 paket iken RED kuyruğu

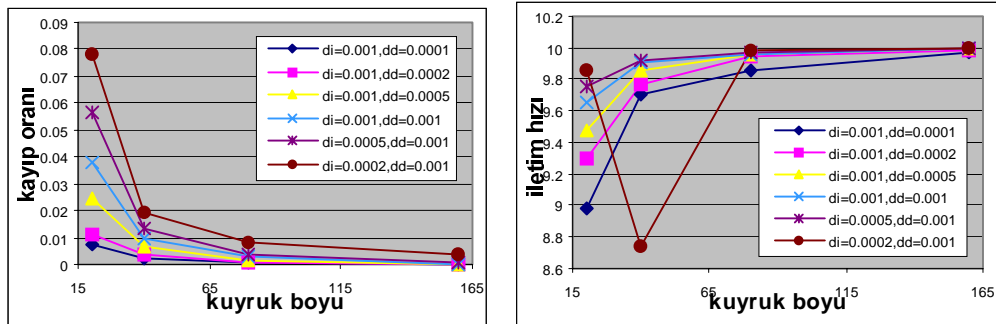


Şekil 3.35 ECN kullanılmadığında büyük ve küçük RTT için kuyruk boyu 160 paket iken BLUE kuyruğu

3.4.5.3 Parametre seçimi

Feng yaptığı benzetimlerde d_i 'yi d_d 'ye göre oldukça büyük seçmiştir. Bunun nedeni işaretleme olasılığının hem gereğinden fazla muhafazakar hem de gereğinden fazla agresif olmasında durumlarında hattın boş kalması ile karşılaşılabilirken, kuyruğun taşmasının sadece işaretleme olasılığının gereğinden az olması durumunda gerçekleşmesidir. Paketler fazla işaretlenildiğinde hattın boş kalacağı açıktır. Gereğinden az işaretlenildiğinde ise kuyruğun taşması sonucunda oluşan topluca paket düşüşleri sonucunda kuyruk boyu sürekli olarak salınır. Bunun sonucunda da hattın boş kalması mümkündür.

BLUE'nın d_i ve d_d parametrelerinin başarımına etkisini incelemek için Şekil 3.27'deki ağ yapısı üzerinde bu parametrelerin değişik değerleri için benzetim yapılarak kayıp oranları ve verimli iletim oranları ölçüldü. Feng d_i ve d_d parametrelerini birbirlerinin belirli katlarında seçmesi taban alınarak d_i/d_d oranının 10, 5, 2, 1, 0,5 ve 0,2 olduğu durumlarda büyük olan parametre 0,001, 0,003, 0,006, 0,010, 0,030, 0,050 yapılarak benzetimler gerçekleştirildi.



Şekil 3.36 BLUE'nın 1000 aktif akış için değişik kuyruk boylarında değişik parametrelere tepkisi

Sonuçları EK C’de verilen benzetimler sonucunda parametrelerin binde birler seviyesinde olmasının daha olumlu sonuçlar verdiği görülmüştür. Tüm benzetimlerde büyük parametrenin 0,001 olduğu sonuçlar **Şekil 3.36**’da toplu olarak görülmektedir. Bu sonuçlar incelendiğinde kullanılan benzetim ağı için $d_t = 0,0005$ ve $d_d = 0,001$ seçilmesinin en uygun sonuçları verdiği gözlemlenildi.

3.5 Açık Tıkanıklık Bildirimi

TCP’nin tıkanıklık denetimi yapıları ağı kara kutu kabul ederek ağın kabul edebileceği trafik miktarını anlamak için sürekli olarak iletim hızlarını arttırmalar. İletim hızının artması ile ağda tıkanıklık oluşur ve paket düşürülür. Ağı kara kutu kabul ederek tıkanıklığı paket kaybı ile algılamak gecikmeye ve paketlerin düşürülmesine karşı çok hassas olmayan uygulamalar için uygun olsa da ses ve görüntü verisi transferi, web’de dolaşma veya telnet gibi etkileşimli uygulamalar için uygun değildir.

TCP’nin bu şekilde iletim yapması DT tipindeki kuyrukların taşması ile paket kaybına neden olur. Aktif kuyruk yönetimi yapıları tıkanıklığı kuyruk taşmadan farkederek ve üç düğümleri haberdar edebilir. Bunun yanında kuyruklar dolma aşamasına gelmediğinden ortalama gecikme de düşecektir.

Aktif kuyruk yönetimi yapıları tıkanıklığı uç düğümlere bildirmek için paketleri düşürebileceği gibi paketlerin belirli alanlarını iletim hattında tıkanıklık oluştuğunu belirtmek için işaretleyebilir. Bu tip bir yapıya açık tıkanıklık bildirimi (ECN, Explicit Congestion Notification) ismi verilir [68].

ECN gereksiz paket düşürülmesini engellemesinin yanında tıkanıklığın çok daha çabuk bir şekilde farkedilmesini sağlar. ECN kullanılmadığında tıkanıklık ya 3 çift paket alındısı ile veya RTO süresinin aşılması ile farkedilir. ECN ile ise en geç bir RTT sonrasında tıkanıklık oluştuğu bilgisi kaynak düğüme iletilir.

ECN’den önceden de SQ mesajları DECbit yapısı ile tıkanıklık kaynaklara açık bir şekilde haber verilmekteydi.

3.5.1 IP’de yapılan değişiklikler

ECN desteği için IP paket başlığında iki bit kullanılmaktadır. ECT (ECN Capable Transport) biti uç düğümlerin iletim protokollerinin ECN desteklediğini ifade ederken, yönlendiriciler tarafından doldurulan CE (Congestion Experienced) biti ise iletim

hattında tıkanıklığın olduğunu anlatır. ECT ve CE için IP paket başlığındaki TOS alanının 6 ve 7 numaralı bitleri kullanılır.

ECN destekleyen bir iletim protokolü CE biti işaretlenmiş bir paket gelmesine, yolda tek bir kayıp olduğundaki gibi iletim penceresini yarılayarak tepki vermelidir.

ECN bütün Internet'e eş zamanlı olarak kurulamayacağından tıkanıklık uzun bir süre hem ECN ile hem de paket düşürülmesi ile kaynak düğümlere haber verilecektir. Böyle bir yapıda CE biti çekilmiş bir pakete normal paket düşürülmesinden farklı bir tepki vermesi durumunda akışlara karşı adaletsiz davranılmış olacaktır. Bu nedenle paket düşürülmesinin algılanması ve CE biti çekilmiş bir paketin alınmasına kaynak düğümün iletim seviyesi protokolleri aynı tepkiyi vermelidir.

RED gibi aktif kuyruk yönetimi yapıları koşturan yönlendiriciler uç düğüme tıkanıklığı bildirmek istediklerinde paket düşürmeyi seçebileceği gibi uç düğümler ECN destekliyor ise (ECT biti çekili ise) CE bitini işaretleyerek paketi düşürmeden de iletebilir.

CE biti çekili bir paket alan bir yönlendirici paket üzerinde bir işlem yapmadan iletimi gerçekleştirir. Kuyruklar tamamen dolu olduğu için paketin düşürülmesi gerekiyor ise paket CE biti çekili olsa da düşürülür.

3.5.2 TCP'de yapılan değişiklikler

ECN IP protokolü dışından üst katmandaki iletim protokolü tarafından da desteklenilmelidir. Çünkü tıkanıklığa bu katmanda tepki verilmektedir ve bağlantı kurulurken her iki düğümünde ECN desteklediği sınanmalıdır.

TCP için ECN 3 yeni mekanizmaya ihtiyaç duyar:

1. Bağlantı kurulurken her iki tarafında ECN desteklediğinin sınanması
2. Hedef düğümün CE biti çekilmiş bir paketi aldığını gönderen tarafa bildirebilmesi için ECN-Echo bayrağı
3. Kaynak düğüm tarafından tıkanıklığa tepki verildiğini anlatmak için kullanılan CWR (Congestion Window Reduced) bayrağı

TCP iletim öncesinde bağlantı kurarken gönderdiği SYN paketinde CWR ve ECN-Echo bayraklarını çekili yollayarak ECN desteklediğini karşı düğüme iletir. Hedef

düğüm ECN destekliyor ise kaynak düğüme gönderdiği SYN-ACK paketi ile SYN alındısı gönderirken sadece ECN-echo bayrağını çeker. Bu karşılıklı işaretleşme sonucunda iki taraf da ECN desteklerinden emin olurlar.

ECN destekleyen kaynak TCP protokolü gönderdiği paketlerin IP başlığındaki ECT bitini çekerek iletim yapar. Kaynak ECN-echo biti çekili bir paket alındısı alırsa yol üzerinde tıkanıklık oluştuğunu anlayarak buna tepki verir. Kaynak düğüm buna her pencerede en fazla bir kez olmak üzere tıkanıklık penceresi (*cwnd*) ve *ssthresh* değerini yarılayarak cevap verir. Aynı pencere içinde alacağı ECN-echo bayrağı çekili yeni paketler için yeni paketler gönderilir.

Hedef düğüm CE biti çekili bir paket aldığı anda buna karşılık yolladığı paket alındılarında ECN-echo bitini çeker. Alındıların kaybolabilme tehlikesi nedeni ile tek bir CE biti çekili paket için bir seri paket alındısının ECN-echo biti çekilerek gönderilir. Bu işlem CWR bayrağı çekili bir paket gelen dek devam eder.

ECN destekleyen bir kaynak hangi nedenle (RTO süresinin aşılması, çift paket alındısı alınması veya ECN-echo bayrağına tepki olarak) olursa olsun tıkanıklık penceresini azaltıldığında bunu karşı tarafa CWR bayrağını çekerek haber verir.

3.5.3 ECN başarımı

Floyd [69]'da yaptığı benzetimlerle ECN'in gereksiz paket kaybını önleyerek hem uçtan uca iletim hızında hem de karşılaşılan gecikme için DT ve ECN kullanılmayan RED yapılarından daha iyi olduğu gösterilmiştir.

4 DS YAPISININ GERÇEKLENMESİ

Farklılaştırılmış Hizmetler yapısı ölçeklenebilir bir şekilde kaliteli hizmet sunulmasını sağlayacak bir altyapı oluşturmaktadır. Bu bölümde daha önceden mimarisi ve temel hizmetleri açıklanan DS yapısını destekleyen bir yönlendiricinin yapısı açıklanacaktır.

Bölüm 4.1'de bir DS yönlendiricisinin sahip olması gereken temel birimler incelenecektir. Bölüm 4.2'de tek bir kuyruk üzerinden AF hizmeti sunmada kullanılan bir kuyruk yönetim mekanizması açıklanarak başarıyı benzetimler ile incelenecektir. Bölüm 4.3'de ise bölüm 4.1 ve bölüm 4.2'de açıklanan yapılar ışığında bir ağda istenilen yapıdaki hizmetlerin ne şekilde verilebileceği gösterilecektir. Bölüm 4.4'de DS yapısının AF hizmetinin başarıyı incelenecektir.

4.1 Örnek bir DS yönlendiricisi

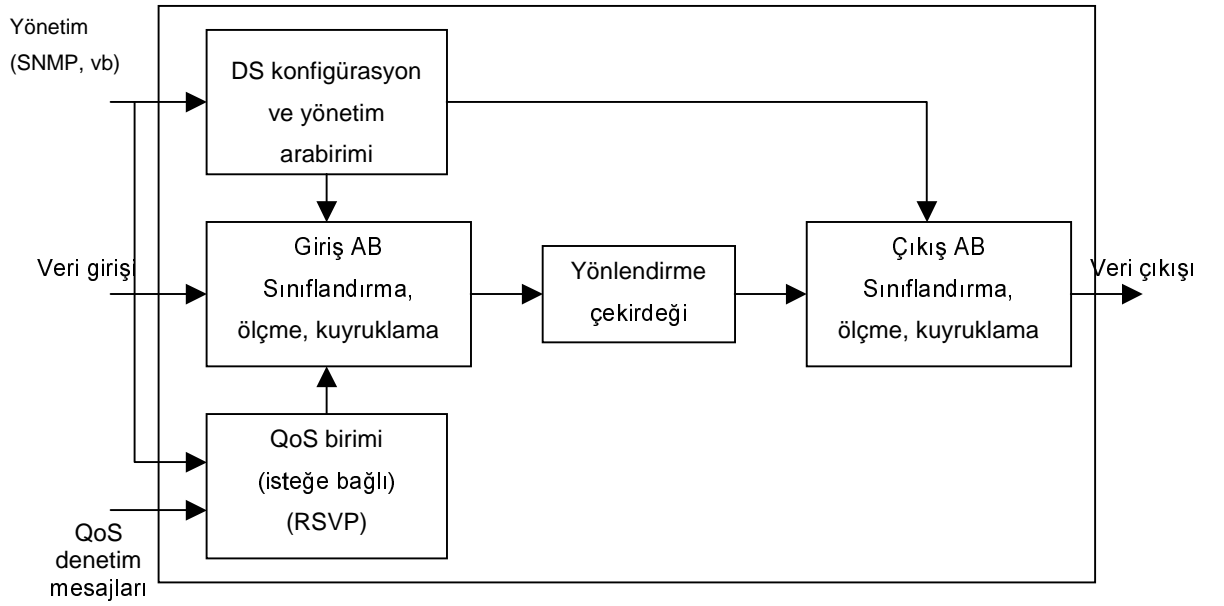
Şekil 4.1'de bir DS yönlendiricinin temel blokları görülmektedir [70]. Bir yönlendiricide genellikle bunların hepsi bulunmamaktadır ancak şekilde tüm olası bloklar gösterilmiştir. Yönlendiriciye gelen veri temel olarak giriş arabirimi, yönlendirme bloğu ve çıkış arabirimlerinden geçmektedir. Normalde yönlendirme birimine çok sayıda giriş ve çıkış arabirimi bağlıdır. Giriş ve çıkış arabirimleri sınıflandırma, ölçme ve kuyruklama birimlerinden oluşmaktadır ancak bunların tümünün iki blokta da olması gerekmez.

Yönetim birimi ağ yöneticisinin SNMP ve benzeri protokoller ile düğümün işleyiş parametrelerini değiştirebilmesine ve diğer istatistiksel verilerin alınmasını sağlar. QoS birimi ise isteğe bağlıdır ve belirli akış gruplarına kaynak ayrılmasını veya ayrılan kaynakların serbest bırakılmasını sağlar.

Giriş veya çıkış arabirimlerinde bulunabilen sınıflandırıcılar tek girişli çok çıkışlı elemanlardır ve paketlerin başlıklarına göre hangi çıkışına aktarılacağına karar vermekle yükümlüdür. IP adresleri ve port numaralarının kontrol edildiği MF sınıflandırma ve sadece DS alanının kontrol edildiği BA sınıflandırma yanında IP

başlığındaki belirli bitlere veya alt katmana ilişkin belirli alanlara göre de sınıflandırma yapılabilmektedir.

DS servisi sunan İSS gelen trafiğin anlık profiline göre verilen hizmeti değiştirmek isteyebilir. Bunu sağlayabilmek için belirli sınırların dışındaki paketlerin başka şekilde hizmet alabilmesi için trafik ölçücüler kullanılır. Ölçücüler tek girişli çok çıkışlı birimlerdir ve değişik trafik seviyeleri değişik çıkışlarına yönlendirilir. Bu yapıda ölçücü, gelen trafiğin hızını ölçer ve verilen bir değer ile kıyaslar. İletim hızı bu eşik değeri altında ise profile uygun aksi halde profile uygun değildir.



Şekil 4.1 DS yönlendiricisinin temel blokları [70]

AF PHB'sinde yeşil, sarı ve kırmızı olarak isimlendirilen üç seviyeli bir ölçücü kullanılmaktadır. Yeşil seviye profile uygunluğu temsil ederken, sarı seviye kısmen uygunluğu, kırmızı seviye ise profile uygun olmamayı temsil etmektedir [71]. Bu değişik seviyeler değişik kuyruklara alınmasını tetikler.

Ölçücülere örnek olarak ortalama hız, jeton kovası veya çok seviyeli jeton seviyesi ölçücüleri gösterilebilir. Ortalama hız ölçücüsünde gelen trafiğin belirli bir zaman aralığında ortalaması eşik değerler ile kıyaslanır. Buna göre daha karmaşık olan jeton kovası ölçücüsünde trafiğin geliş hızı ve patlama miktarı parametre olarak verilir. Kovaya belirtilen hızla eklenen jetonlar kullanılmadıkça kova boyutunun (patlama oranı) belirlediği miktarda biriktirilir. Kovada jeton var ise gelen trafik profil içi aksi halde profil dışıdır. Jeton kovası ölçücüsünün daha karmaşık bir şekli ise çok seviyeli jeton kovasıdır. İki jeton kovalı bir örnekte ilk jeton kovasında jeton

kalmadığında diğer jeton kovaına başvurulur. Böylece 3 seviyeli bir ölçücü elde edilir. [71, 72]'de örneđi verilen bu ölçücü AF PHB için kullanılabilir.

Sınıflandırıcı ve ölçücülerden alınan bilgiler ile işaretleyici, kontrollü ve normal düşürücüler ve kuyruklar kontrol edilmektedir. İşaretleyiciler paketlerin DS alanına belli bir değeri atamada kullanılır. Normal paket düşürücüler paketlerin tümünü düşürmekte iken kontrollü düşürücüler ise uç düşümlere tıkanıklık bilgisi verebilmek için RED gibi kontrollü bir şekilde paket düşürölmesini sağlar.

Çizelgeleyici girişlerinden gelen paketlerin iletim zamanlamasını belirli bir disipline göre düzenler. Kullanılabilecek çeşitli hizmet disiplinleri şunlar olabilir: önce gelen önce hizmet alır, mutlak öncelik, iletim sınırlı öncelik, ağırlıklı bandgenişliđi paylaşımı, vb.

4.2 RIO

Clark ve Fang [73]'de paketlere farklı seviyelerde hizmet verebilmek için “beklenen kapasite” isimli yapıyı önermişlerdir. Bu yapıda her kullanıcı için bir servis profili belirlenir ve kullanıcının ađa giren paketleri bu profilin kullanıcıya izin verdiđi ölçüdeki paketini daha iyi hizmet alabilecek şekilde işaretlenir. Profilin daha iyi hizmet alması için işaretlediđi paketler profil içi (in), bađıl olarak daha kötü hizmet alacak paketler de profil dışı (out) olarak isimlendirilir. Ađın içinde ise yönlendiriciler paketlerin sınıflarını ayırderek paketleri tiplerine göre düşürürler.

Clark ve Fang bu yapıyı oluşturabilmek için DS'e benzer şekilde ađ sınırlarında işaretleyiciler ve ađ içindeki yönlendirici kuyruklarında kořmak üzere RIO (RED with In and Out) algoritmasını geliřtirmişlerdir.

RIO [73], RED'in temel prensiplerini kullanmaktadır ve bu yüzden RED'in tüm özelliklerini taşımaktadır. RIO, RED'den farklı olarak tıkanıklık durumunda profil dışı (out) paketleri profil içi (in) paketlerden ayırt edebilmektedir.

4.2.1 RIO algoritması

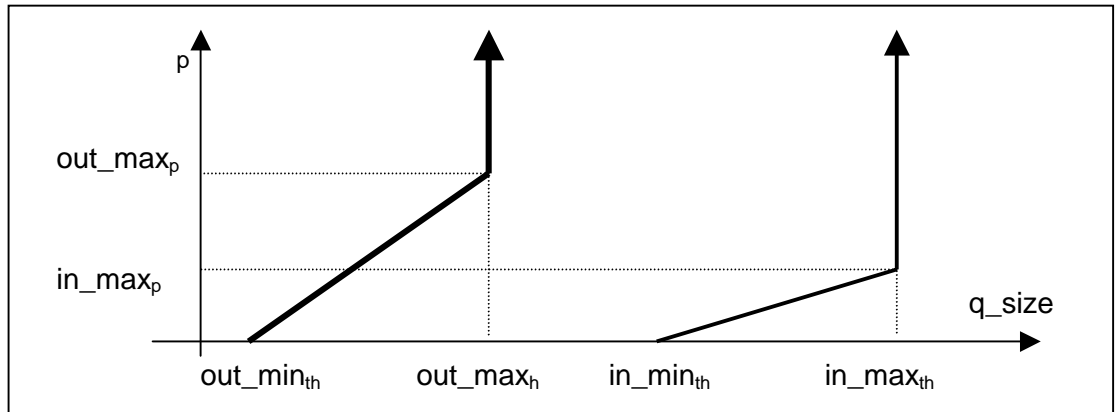
RIO paketleri sınıflarına göre farklı olasılıklarla düşürebilmek için tek bir kuyruk için biri profil içi paketler için diđer profil dışı paketler için olmak üzere iki ayrı RED algoritması kořturmaktadır.

RIO, RED ile birebir aynı yapıyı kullanmaktadır ancak parametreleri profil içi paketler için ayrı ve profil dışı paketler için ayrı olmak üzere iki gruptur. Yönlendirici gelen her paketin profil içi ve dışı olduğunun kontrolünü yapar.

Paket profil içi ise, tüm paketleri içeren ortalama kuyruk boyu hesaplanır. Bu ortalama kuyruk boyu in_min_{th} altında ise paket işaretlenilmez. in_max_{th} ise paket kesinlikle işaretlenilir. Bu iki değer arasında ise kuyruk boyu ile doğru orantılı bir olasılıkla işaretlenilir.

Gelen paket profil dışı ise bu kez sadece profil dışı paketleri içeren ortalama kuyruk boyu hesaplanır ve paket düşürme işlemi buna göre yapılır. Avg_{out} , out_min_{th} ve out_max_{th} ile kıyaslanır ve olasılıklar bu parametrelere göre hesaplanır.

RED'de paket düşürme olasılığı hesaplanılırken kullanılan max_p olasılığı da RIO için profil içi ve profil dışı paketler için farklıdır ve hesaplamalarda gelen paketin cinsine göre uygun max_p parametresi kullanılır.

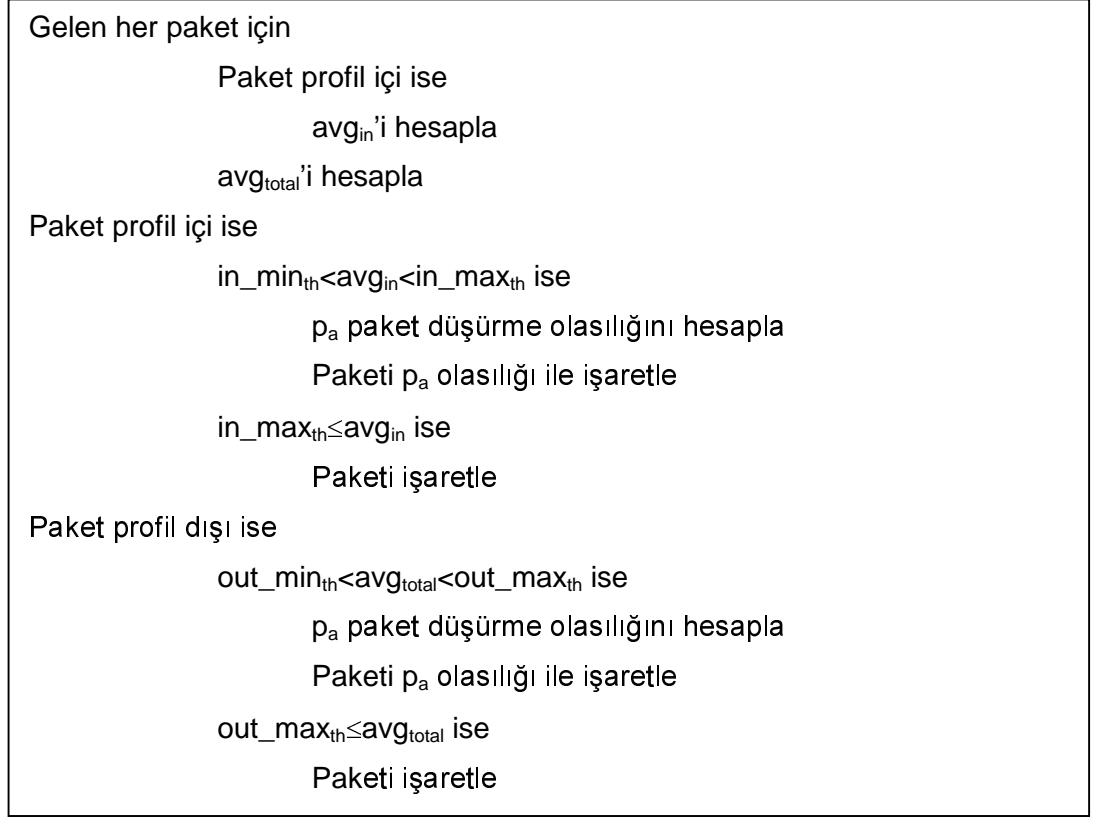


Şekil 4.2 RIO parametreleri

Profil dışı ve profil içi paketlerin farklılaştırılması RIO parametrelerinin uygun seçilmesi ile yapılmaktadır. RIO algoritmasının profil dışı paketleri düşürmede daha agresif olması için öncelikle profil dışı paketler profil içi paketlere göre çok daha önceden düşürülmeye başlanılır. Bu out_max_{th} 'in in_min_{th} 'den küçük tutulması ile sağlanır. Buna ek olarak profil dışı paketlerin işaretlenme olasılığının üst sınırı, profil içi paketlerinkine göre daha fazladır.

RIO kullanan bir yönlendirici tıkanıklık başladığında profil dışı paketleri düşürmeye başlar. Tıkanıklık uzun sürdüğünde profil dışı paketlerin hepsini düşürmeye başlar. RIO, tıkanıklığın daha da artması ile tıkanıklığı kontrol edebilmek için profil içi paketleri de düşürmeye başlar. İyi tasarlanmış bir ağ yapısı ve trafik profili ile bu durum oluşmamalıdır.

Profil içi paketler için paket düşürme olasılığı belirlenirken in_min_{th} ve in_max_{th} parametreleri tüm kuyruğun boyu ile değil, kuyruk içindeki profil içi paketlerin miktarı ile kıyaslanılmaktadır. Bu sayede kuyrukta ne kadar profil içi paket istenildiği belirlenebilmektedir.



Şekil 4.3 RIO algoritması

Şekil 4.3’nda detaylı olarak verilen RIO algoritmasına dikkat edilirse profil dışı paketlerin düşürülme olasılığının hesaplanılmasında kuyruktaki toplam profil dışı paket sayısı değil, toplam kuyruk boyu parametre olarak kullanılmaktadır. Bunun yerine kuyruk içindeki toplam profil dışı paket miktarı kullanılsaydı out_min_{th} , out_max_{th} ve out_max_p parametrelerinin seçimi çok kolay olmazdı. Çünkü kuyrukta toplam ne kadar profil dışı paket istenildiği kolay belirlenemez. Kuyrukta ancak fazla yer varsa profil dışı paket bulunmalıdır. Diğer bir alternatif ise hesaplamalarda kuyruktaki profil içi paket sayısını kullanmak ve profil içi paket miktarı az iken daha fazla profil dışı paketin kuyruğa girmesine izin vermektir. Paket düşürme olasılığı hesaplanılırken de kuyruktaki profil içi paket sayısı ile ters orantı kurulabilir. Ancak bu işlemler ancak kuyrukta fazla sayıda profil içi paket bulunmakta ise anlamlı sonuçlar verecektir. Aksi halde az sayıdaki profil içi paket için hesaplamalar hassas olamaz. Bunlar gözönüne alındığında, en uygun olanın tüm kuyruk boyunun

parametre olarak kullanılması ve kuyrukta en fazla ne kadar profil dışı paket bulunmasına izin verileceğinin belirlenmesi olduğu görülür.

4.2.2 RIO'nun başarımı

RIO'nın davranışı gözlemlemek ve başarımını ölçmek için **Şekil 4.4**'teki ağ yapısı kullanıldı. 33Mbps'lik bir darboğaz hattını paylaşan 10 kaynağın ilk 5 tanesi 1Mbps, diğer 5 tanesi de 5Mbps trafiği profil içi olarak iletecek şekilde ayarlandı.

200 saniye süren benzetimin ilk 70 saniyesinde rastlantısal olarak başlatılan kaynaklar sonsuz uzunlukta bir dosyanın iletimini yapmaktadırlar ve değişik RTT'ye sahiptirler. 90-190 saniye aralığı için her kaynağının iletim hızı ölçülmüştür.

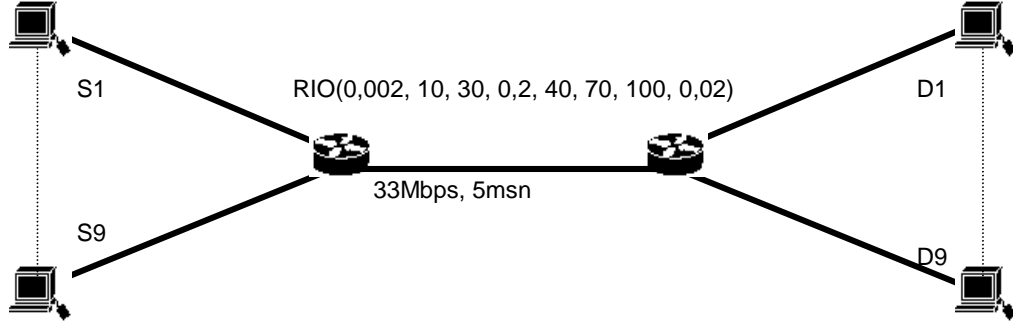
Darboğaz hattında RED ve RIO kullanılarak yapılan ölçümler **Tablo 4.1**'de görülmektedir.

Tablo 4.1 RED ve RIO iletim hızları

| Bağlantı | RTT (msn) | RED (Kbps) | Profil (Kbps) | RIO (Kbps) |
|----------|-----------|---------------|------------------|---------------|
| 0 | 20 | 6454 | 5000 | 5300 |
| 1 | 20 | 6242 | 1000 | 2696 |
| 2 | 40 | 3447 | 5000 | 4298 |
| 3 | 40 | 3403 | 1000 | 1727 |
| 4 | 60 | 2330 | 5000 | 4001 |
| 5 | 60 | 2275 | 1000 | 1325 |
| 6 | 80 | 1766 | 5000 | 3661 |
| 7 | 80 | 1713 | 1000 | 1188 |
| 8 | 100 | 1298 | 5000 | 3577 |
| 9 | 100 | 1500 | 1000 | 1070 |

TCP pencere tabanlı iletim yapmaktadır ve tıkanıklığı farketmediğinde tıkanıklık penceresini yarılamaktadır. Bundan sonra tıkanıklık önleme aşamasında iletim

gerçekleştiren TCP, tıkanıklık penceresini gelen her paket alındısı ile $1/cwnd$ kadar artırır. Bunun sonucunda tıkanıklık penceresi her RTT'de bir paket artar. Bu nedenle RTT'si daha fazla olan akışın eski hızına ulaşması daha zor olmaktadır. TCP iletim hızının RTT'ye olan bu bağımlılığı nedeni ile farklı RTT'ye sahip akışlar farklı iletim hızına ulaşır.



Şekil 4.4 RIO test ağı yapısı

Açıklandığı gibi RED kullanıldığında farklı RTT'ye sahip akışlar çok farklı hızlara ulaşmaktadır. Aynı RTT'ye sahip olan akışların iletim hızlarında da farklılıklar görülmektedir. Bu TCP-Reno'nun penceresinin çok salınımlı olmasından kaynaklanmaktadır.

RIO kullanıldığında elde edilen hızlar ise aynı tablonu son sütununda görülmektedir. İletim hızı RIO ile de RTT'ye bağlı olarak değişmektedir ancak bu değişim RED'e göre çok daha azdır. Bunun yanında istenildiği gibi farklı seviyelerde iletim hızı da kontrollü bir şekilde sağlanmıştır.

Tablo 4.2 ECN kullanıldığında RED ve RIO iletim hızları

| Bağlantı | RTT (msn) | RED (Kbps) | Profil (Kbps) | RIO (Kbps) |
|----------|-----------|---------------|------------------|---------------|
| 0 | 20 | 6412 | 5000 | 5832 |
| 1 | 20 | 6136 | 1000 | 3078 |
| 2 | 40 | 3596 | 5000 | 4625 |
| 3 | 40 | 3253 | 1000 | 1735 |
| 4 | 60 | 2317 | 5000 | 4218 |
| 5 | 60 | 2399 | 1000 | 1364 |

| | | | | |
|---|-----|------|------|------|
| 6 | 80 | 1784 | 5000 | 3857 |
| 7 | 80 | 1814 | 1000 | 1180 |
| 8 | 100 | 1418 | 5000 | 3766 |
| 9 | 100 | 1459 | 1000 | 1093 |

Tablo 4.2'de ise aynı testlerde ECN kullanıldığında ortaya çıkan durum görülmektedir.

4.3 DS yönlendiricisi ayarları

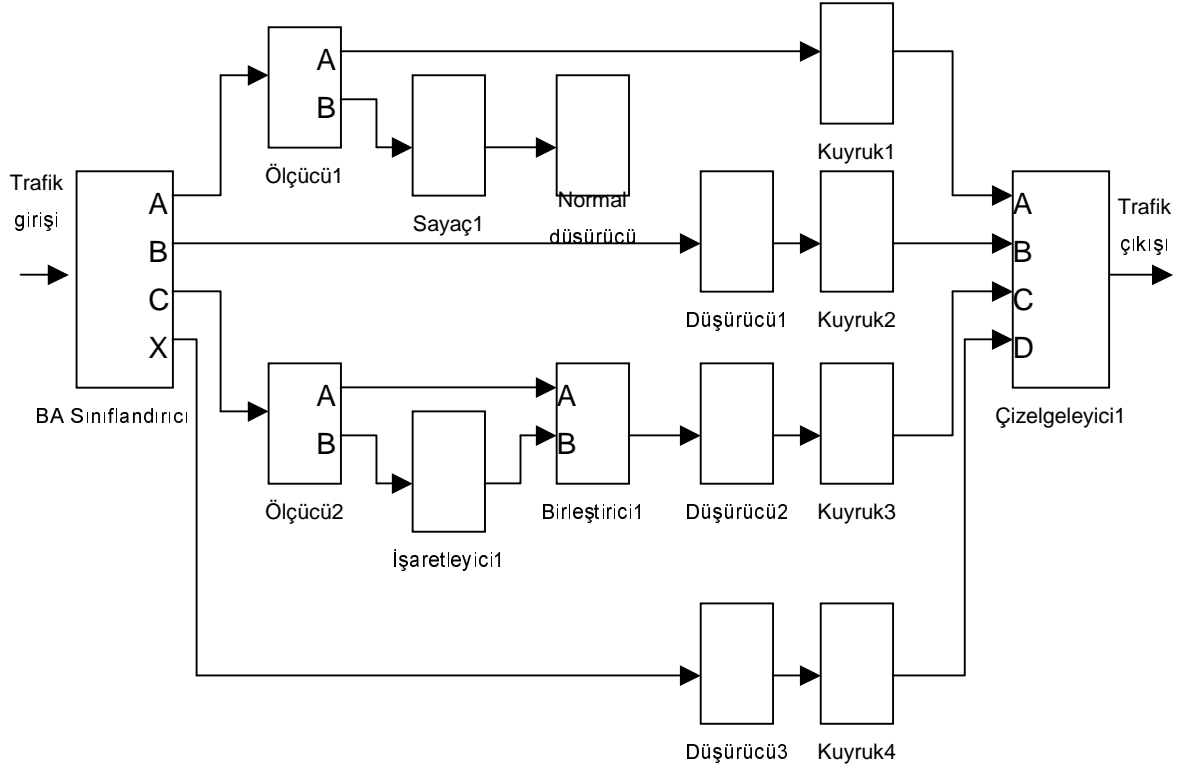
Bu bölümde verilen özelliklerdeki hizmetleri ağ üzerinde sunmak için DS yönlendiricisinin temel bloklarının ne şekilde seçilmesi gerektiği ve ayarların ne şekilde olması gerektiği gösterilmektedir. Müşteri ve İSS (İnternet Servis Sağlayıcı) arasında yapılan bir anlaşma (Service Level Agreement, SLA) ile İSS ağı içerisinde müşterinin trafiğine verilecek olan hizmetin niteliği (Service Level Specification, SLS) tanımlanır. DS tarafından bir DS alanı içinde sunulan hizmet kalitesinin uçtan uca sunulabilmesi, DS destekleyen ağlar arasında yapılan SLS tanımlamaları ile mümkün olmaktadır. Bu çeşit bir tanımlama aşağıdaki yapıda olabilir:

Tablo 4.3 Örnek bir SLS

| DS Alanı | PHB | Profil | Davranış |
|-----------|------|---------|--|
| 001001 | EF | Profil1 | Profil dışı trafiği düşür |
| 001100 | AF11 | Profil2 | Profile uygunlaştı, dolu olduğunda sondan düşür. |
| 001101 | AF21 | Profil3 | Profil dışı paketlerin DS alanını 001000 olarak işaretle, dolu olduğunda sondan düşür. |
| Diğerleri | BE | Yok | RED tipi paket düşür |

Bu hizmet tanımlamasına göre müşteri 001001 olarak işaretlenmiş paketleri Profil1'e uygun oldukça EF PHB'sine göre hizmet alacaktır. Profil dışında kalan paketler ise düşürülecektir. Düşürülen paket müşteriye daha fazla bandgenişiği satın almak için

ikna etme amacıyla sayılabilir. 001100 olarak işaretlenmiş gelen paketler profile uydurularak yönlendirme işlemi gerçekleştirilecektir. 001101 DS kodu ile gelen trafiğin ise Profil3'e uymayan paketleri daha kötü hizmet almak üzere 001000 olarak işaretlenecektir.



Şekil 4.5 Örnek hizmeti sunmak için gerekli yönlendirici yapısı

Şekil 4.5'te belirtilen SLS'i sağlamak için gereken trafik düzenleme blokları görülmektedir. Sınıflandırma aşamasında tek bir BA sınıflandırıcı kullanılmaktadır. Paketleri IP başlığındaki DS alanına göre sınıflandıran bu birimin 3 temel çıkışı bulunmaktadır (A, B, C). Diğer sınıflara uymayan paketler ise X çıkışına yönlendirilmektedir.

001001 paketlerinin profile uygunluğunu kontrol etmek için kullanılan ölçücünün profil dışı olarak ayırdığı paketler düşürülmektedir. Ancak müşteriye ek bandgenişiği alması için ikna etmede kullanılabileceği düşünülerek, düşürülen paketler sayılmaktadır. Profil1'e uyumlu paketler ise DT kullanılan bir kuyruğa gönderilir. Çıkışta kullanılan çizelgeleyicinin konfigürasyonu doğru yapıldığında bu kuyruktan kesinlikle paket düşürülmez. Çünkü gelen trafik miktarı ancak profilin belirlediği kadardır.

001100 olarak işaretli paket doğrudan DT kullanılan bir kuyruğa gönderilmektedir. Bu kuyruğun parametreleri ile trafiğin profile uygunlaştırılması sağlanabilir. 001101

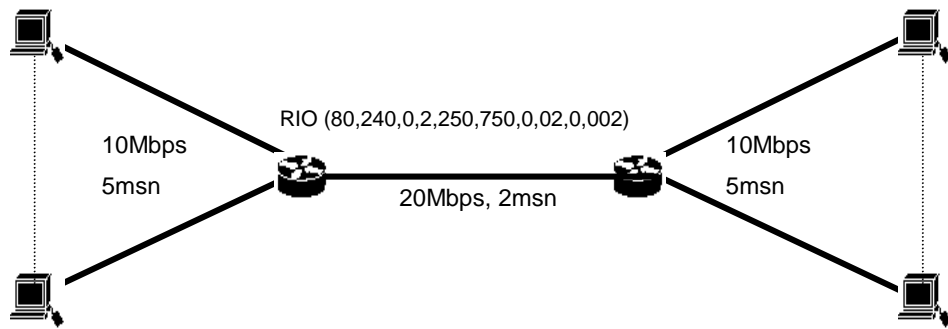
paketleri ise sınıflandırıcı çıkışında bir ölçücü ile profile uyan ve uymayan olarak ayrılmaktadır. Profile uymayan paketlerin hizmet kalitesi düşecek şekilde işaretlenilmek üzere işaretleyiciye gönderilir. DS alanı işareti değiştirilen ve değiştirilmeyen paketlerin tümü üzerinde RIO benzeri bir algoritma koştan bir kuyruğa gönderilir.

Tüm sistemin çıkışında bulunan çizelgeleyici ise trafik sınıflarının uygun oranda hizmet almasını sağlar. Burada WRR tipi bir çizelgeleyici kullanılarak kuyruk ağırlıkları SLS'e uygun olarak verilebilir.

4.4 Benzetimler ile AF başarımı

Ibanez tarafından yapılan benzetimde AF PHB'nin trafik akışlarına kesin bir garanti sağlamaktan uzak olduğu görülüyor [74]. AF ancak akışlara bağlı olarak garantiler sunabilmektedir.

Şekil 4.6'daki gibi Ibanez'in benzetim ağına uygun olarak kurulan ağda FTP ve CBR kaynakları ile AF trafiği koşturuldu. 200 saniye koşturulan benzetimin ilk 70 saniyesinde rastlantısal olarak başlatılan 40 kaynak karşı uçtaki 40 hedef ile konuşmaktadır. 90 ve 190 saniyeleri arasında akışların elde ettiği iletim hızı ölçüldü. Akışların RTT'si 24msn'dir. 10Mbps ile darboğaz oluşturan iletim hattına bağlanan kaynak ve hedefler 20Mbps'lik bir hattı paylaşmaktadır.



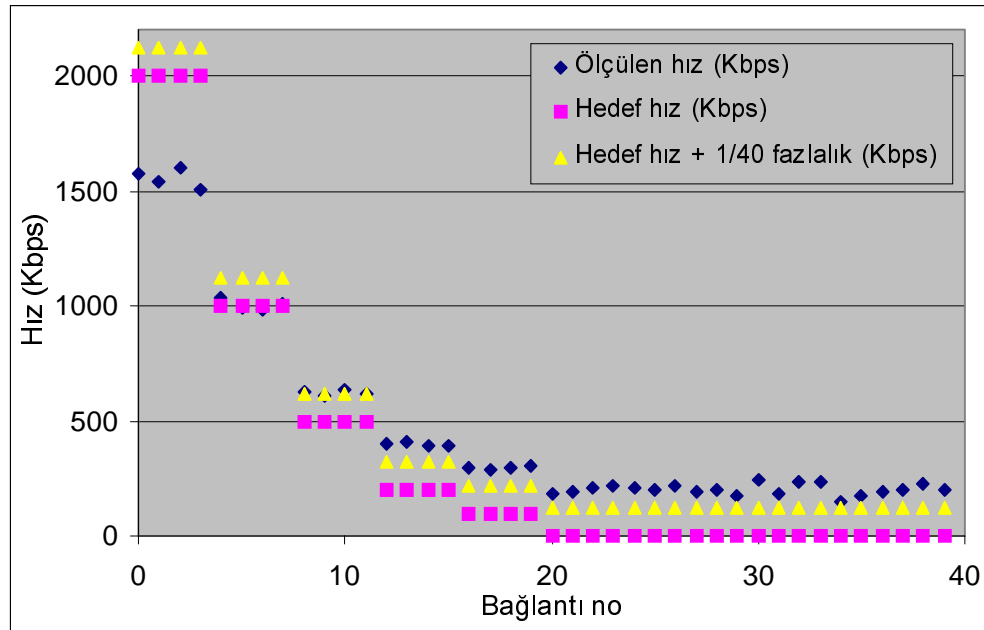
Şekil 4.6 AF benzetim ağı

Darboğaz hattı AF kuyruğunda RIO kullanılmıştır. [75]'te Ibanez tarafından yapılan benzetimde RIO parametrelerinin sağlıklı seçilmediği belirtildiğinden, kendi benzetimimizde RIO parametreleri [74]'te kullanılan parametrelerden farklı seçilmiştir. Min_{th} , max_{th} , max_p , profil dışı trafik için 80, 240, 0,2 ve profil içi trafik için de 250, 750, 0,02 seçilmiştir. Max_{th} , min_{th} oranı [61]'de önerildiği gibi 3 olarak

seçilmiştir. Profil dışı ve içi trafik için max_p ise [73]'te yapılan benzetimlerdeki şekilde seçilmiştir.

İlk 4 kaynağın iletim hedefi 2Mbps iken ikinci dördlünün 1Mbps, sonrakinin 0,5Mbps'dir. Diğer iki dördlünün hedef iletim oranı ise 0,2 ve 0,1Mbps'dir. Bu şekilde AF trafik toplamı 15,2Mbps'dir. Son 20 kaynağı ise BE iletim yapacak şekilde konfigüre edilmiştir. İbanez, hem BE hem de AF paketlerini tek bir RIO kuyruğunda toplanıldığı bir konfigürasyon kullanmıştır. Bu yapı ile yapılan benzetimde İbanez'in sonuçlarına benzer sonuçlar elde edilmiştir.

Şekil 4.7'de İbanez'in konfigürasyonuna benzer şekilde yapılan benzetim sonuçları görülmektedir. Kareler her akışın hedeflenen iletim hızını, üçgenler ise buna artan bandgenişliğinin 40'ta birinin eklenmiş halini göstermektedir. Buradan da görüldüğü gibi 20-39 arasındaki akışlar 0Mbps hedef değere sahipken, adil bir dağıtım ile 120Kbps'lik bir hıza ulaşmalıdır. Dörtgenler ise ölçülen iletim hızını göstermektedir.

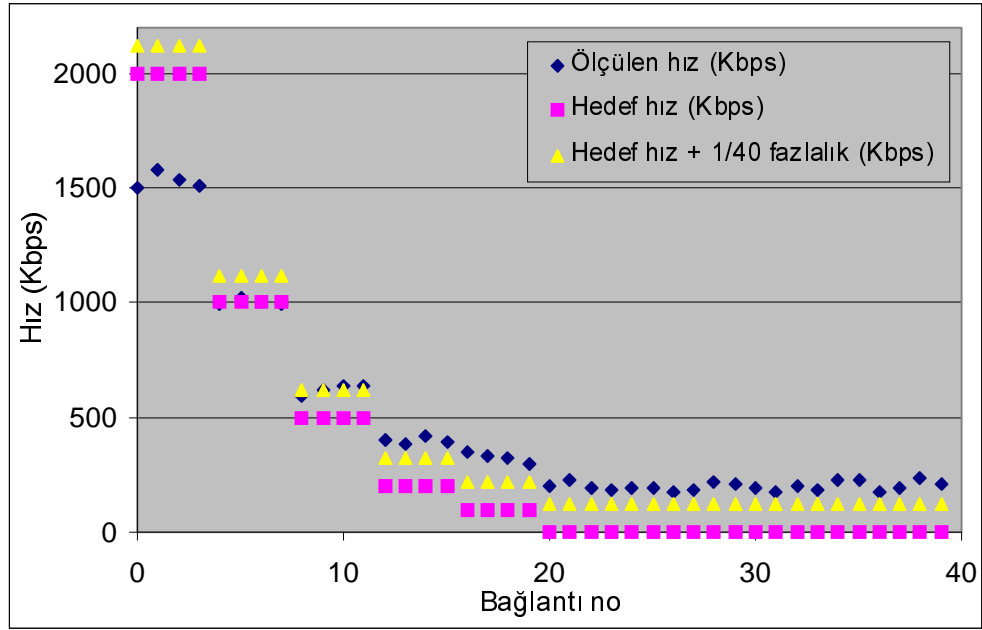


Şekil 4.7 20Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları

Benzetim sonucunda ancak hedef iletim hızı düşük olan bağlantıların hedeflerine ulaşabildiği görülmektedir. Hedeflenen iletim hızının artması ile iletim hızı da artmaktadır ancak bu artış beklenen ölçüde olmamaktadır. Örnek olarak 0 numaralı bağlantının hedeflenen hızı 4 numaralı bağlantının iki katı iken, elde edilen iletim hızı ancak 1,5 katıdır. Bunun nedeni TCP'nin tıkanıklık penceresinin davranışidir. TCP'nin iletim hızını tıkanıklık penceresi sınırlamaktadır. 0 numaralı kaynağın

tıkanıklık penceresi 4 numaralı kaynağinkinden fazladır. Bu nedenle tıkanıklık durumunda tıkanıklık pencereleri azaltıldığında, sıfırlandığında veya yarıldığında, daha hızlı olan kaynağın eski hızına ulaşması daha uzun zaman almaktadır.

Bu benzetimde sonsuz uzunlukta bir dosyanın iletimini yapan FTP kaynakları kullanılmıştır. Ancak Internet'te genellikle kısa süreli iletim yapan akışlar aktiftir. Aynı benzetim 2msn açık, 3msn kapalı iletim yapan exponansiyel iletim yapan trafik kaynakları kullanılarak tekrarlandığında elde edilen sonuçlar **Şekil 4.8**'de görülmektedir. Bu durumda da benzer sonuçlar elde edilmiştir.

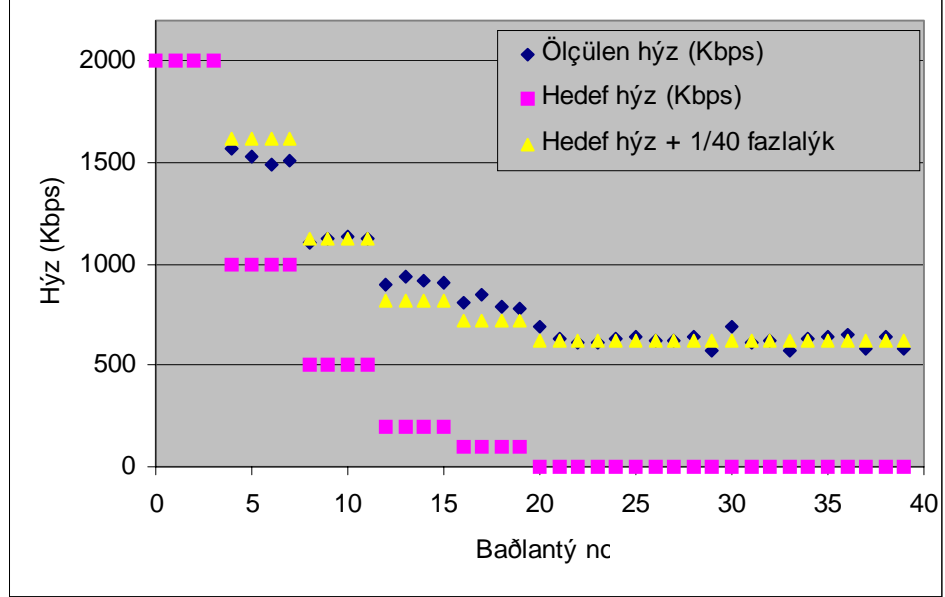


Şekil 4.8 20Mbps darboğaz üzerinden 20 kısa süreli AF bağlantısı benzetim sonuçları

[74]'teki benzetim hakkında yapılan başka bir eleştiri de AF trafiğinin darboğaz hattının kapasitesinin çok büyük bir oranını (%76) kullanacak şekilde konfigüre edilmiş olmasıdır [76] [77]. Bu nedenle Ibanez'in konfigürasyonuna benzer şekilde toplam AF trafiğin %76'sını alacak şekilde yapılan benzetim yanında, darboğaz hattının kapasitesi ağa giren trafik miktarını arttırmadan iki katına çıkartılarak da bir başka benzetim yapılmıştır. Bu şekilde AF trafiği darboğaz hattının %38'i oranında olmaktadır.

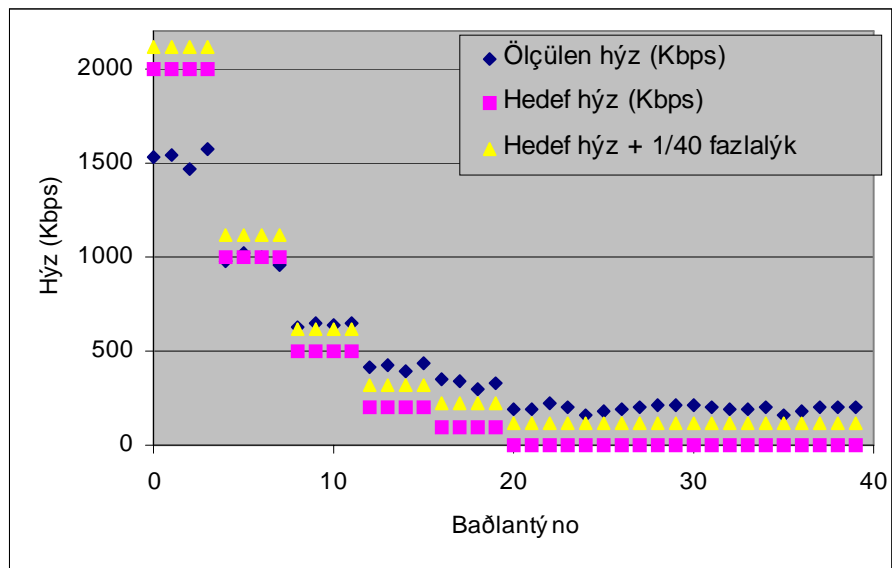
AF trafiğinin toplam trafik içindeki oranı azaltıldığında AF kaynakları **Şekil 4.9**'da görüldüğü gibi hedeflenen hıza ulaşmaktadır. Ancak yine de iletim hızı beklendiği ölçüde artmamıştır. Bunun nedeni de TCP'nın tıkanıklık penceresidir. TCP paket kaybında tıkanıklık penceresi yarılayarak iletim hızını azaltır. Aynı anda paketi

düşürülen farklı büyüklükte tıkanıklık pencerelerine sahip akışlardan büyük pencere boyuna sahip olanın eski hızına ulaşması daha fazla zaman alacaktır. Çünkü tıkanıklık sonrasında iletim hızı doğrusal olarak arttırılmaktadır.



Şekil 4.9 40Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları

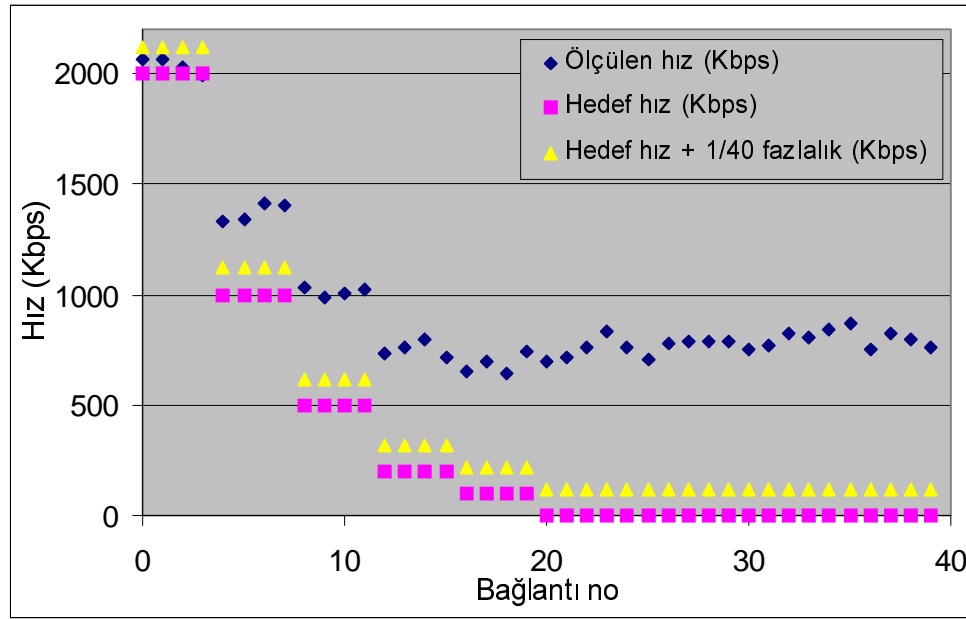
Ibanez benzetimde AF be BE paketlerini aynı RIO kuyruğunda tutmaktadır. Ancak genel yaklaşım AF ve BE paketler için ayrı kuyrukları ağırlıklı round-robin (Weighted Round Robin, WRR) yapısıyla çıkışa bağlamaktır. Ibanez'in benzetimi bu yapı ile tekrarlanıldığında da benzer sonuçlar elde edildi.



Şekil 4.10 20Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları (WRR, AF %80, BE %20)

WRR, AF kuyruğuna %80 BE kuyruğuna ise %20 ağırlıkla hizmet verecek şekilde konfigüre edilmiştir. Bu sayede 15,2Kbps toplam hedef hızına sahip AF kaynaklarına 16Mbps'lik bir kaynak ayrılmış olur. **Şekil 4.10**'da da görüldüğü gibi önceki benzetimlerden farklı bir sonuç elde edilmemiştir.

AF trafiğinin hedeflenen iletim hızına ulaşması ancak toplam hedeflenen iletim hızından daha yüksek bir bandgenişliği kaynağı AF trafiğine ayrıldığında gerçekleşir. **Şekil 4.11**'de darboğaz hattın iletim hızının 40Mbps'e çıkartılması ile AF'e ayrılan bandgenişliğinin 24Mbps'e çıkartılması sonucunda elde edilen sonuçlar görülmektedir.



Şekil 4.11 40Mbps darboğaz üzerinden 20 AF FTP bağlantısı benzetim sonuçları (WRR, AF %60, BE %40)

Bu benzetimler sonucunda Ibanez'in vardığı sonuçların doğru olduğu görülmüştür. AF, net bir iletim hızı garantisi vermek için uygun değildir. AF ancak bağlı öncelikler vermede başarılı olabilmektedir. AF sınıfına ayrılan kaynak yüksek ise bu miktarda kaynağın trafik akışlarına sağlanması mümkün olamamaktadır. AF sınıfının taahhüt edilen bandgenişliğini alabilmesi ancak taahhüt edilen limitin düşük olması durumunda mümkündür.

RIO tabanlı AF'nin hedeflenen hızları yüksek olan bağlantıların hedeflenen hızlarına ulaşamaması sorununu çözmek için [78]'de çeşitli mekanizmalar önerilmiştir. Bunlardan ilki profil dışı olarak işaretlenen her paket için işaretleyicinin kaynak TCP'ye bir bilgi göndermesi ve kaynağın buna tepki olarak tıkanıklık penceresini bir

azaltmasıdır. Bu yapının beklendiği şekilde yüksek iletim hızına hedefine sahip olan akışların hedeflerine daha fazla yaklaşmasını sağlamaktadır. Ancak bu yapı TCP'nin fazlada kalan bandgenişliğini kullanmasını da engellemektedir.

Bir başka mekanizma ise paketlerin, akışların profil içi ve profil dışı olarak işaretlenmesi yanında iletim hedeflerini göstermek üzere işaretlenilmesidir. Paketlerin iki seviyeli olarak değil de 3 seviyeli olarak işaretlenilmesi de bir diğer yapıdır. Bu yapıda işaretleme birimi akışın hem kısa vadede hem de uzun vadede iletim hızını hesaplamaktadır. Uzun vadede hedeflenen iletim hızı üzerine çıkılmasına neden olan paketler OUT_OUT olarak işaretlenirken, sadece kısa vadede hedeflenen hızın üzerine çıkılmasına neden olan paketler de IN_OUT olarak işaretlenilmekte ve yönlendiricilerde de bu yapıya uygun olarak düşürme işlemi gerçekleştirilmektedir.

Bunların dışındaki diğer değişiklik ise TCP'nin değiştirilmesini gerektirmektedir. Bu yapıda TCP hedef penceresi ve fazla kapasite pencereleri tutmakta ve gönderdiği her paketin profil içi veya dışı olduğu bilgisini tutmaktadır. Düşürülen paketin profil içi veya profil dışı olmasına göre bu iki pencere güncellenilmekte ve buna uygun şekilde iletim yapılmaktadır.

5 BIO

Aktif kuyruk yönetimi mekanizmaları içinde üzerinde en çok çalışma yapılan ve kullanılan algoritma RED'dir. RED'in yönlendiricilerde kullanılması da IETF tarafından da önerilmektedir. RED kullanıldığında DT tipi kuyrukların neden olduğu patlamalı akışlara karşı yapılan adaletsizlik gibi sorunlar engellenmekte ve kuyruk boyu kontrol edilebilmektedir.

Akışlara farklı kalitelere hizmetler sunmada kullanılan DS mimarisinde yönlendiricilerde RED ve RIO tipi kuyruklar kullanılmaktadır. BE tipi akışların bulunduğu kuyruk üzerinde RED koşturulurken, AF hizmetini sunmak için ise RED'in değiştirilmiş bir hali olan ve tek kuyruk ile hizmet farklılaştırması yapabilen RIO kuyrukları kullanılmaktadır. RED mekanizması TCP'nin tıkanıklık denetimi yapılarıyla birlikte tıkanıklığı önlemede kullanılırken, RIO ise tek kuyruk üzerinden farklı hizmetler sunabilmeyi sağlamaktadır.

Yeni bir çalışma olan BLUE, aktif akış sayısının fazla olduğu yönlendiricilerde RED'e göre daha iyi bir başarımla elde etmektedir. BLUE'nın kuyruk boyunu kontrol etmekte ve düşük kayıp oranı ve yüksek iletim verimi sağlamadaki başarısı yaptığımız benzetimlerde ve [67]'de gösterilmiştir.

Çok sayıda aktif akışın bulunduğu DS yönlendiricilerinde RED yerine BLUE kullanıldığında daha kısa kuyruklar kullanarak işlem yapmak mümkün olur.

Bu tez çalışmasında, AF kuyrukları için kullanılan RIO algoritması yerine kullanmak üzere, RIO'ya benzer şekilde BIO (BLUE with In and Out) algoritmasını önermekteyiz. Profil içi ve profil dışı paketler için aynı kuyruk üzerinde iki ayrı BLUE algoritmasının koştugu BIO yapısı ile farklı öncelikteki AF paketlerinin bulunduğu kuyruk boyu çok daha kısa olabilir. BIO çok sayıda AF bağlantısının aktif olduğu DS yönlendiricilerinde kuyruk boyunu daha başarılı bir şekilde kontrol ederek ECN desteği ile paket kayıp oranının daha düşük olmasını sağlayacaktır.

Bu bölümde öncelikle önerilen BIO algoritması detayları verilmektedir.

5.1 Algoritma

BIO algoritmasında profil içi ve dışı paketler için ayrı ayrı olacak şekilde iki BLUE algoritması koşturmaktadır. Profil içi ve dışı paketler için 2 ayrı paket işaretleme olasılığı tutulmaktadır ve kuyruğun anlık değişimine göre paket işaretleme olasılığı güncellenmektedir.

Profil içi paketler için kuyruk taşması nedeni ile paket kaybı olduğunda profil içi paket işaretleme olasılığı artırılırken, fazla paket düşürülmesi nedeni ile kuyruk boş kaldığında bu olasılık düşürülür. Paket işaretleme olasılığının artırılma ve düşürülme katsayıları kullanıcı tarafından belirlenen iki parametredir. Paket işaretleme olasılığının çok sık değiştirilerek kararsızlığa neden olmasını engellemek için paket işaretleme olasılığının güncellenmesi belirli bir zaman periyodu içinde sadece bir kez yapılmaktadır.

```
Paket kaybında veya  $Q_{len} > \max_{th_{in}}$  olduğunda:
    if ((now – last_in_increment) > in_hold_time )
         $p_{in,m} = p_{in,m} + d_{in,i}$ ;
        last_in_increment = now;
 $Q_{len} > \max_{th_{out}}$  olduğunda:
    if ((now – last_out_increment) > out_hold_time )
         $p_{out,m} = p_{out,m} + d_{out,i}$ ;
        last_out_increment = now;
Link boş kaldığında
    if ((now – last_in_decrement) > in_hold_time )
         $p_{in,m} = p_{in,m} - d_{in,d}$ ;
        last_in_decrement = now;
    if ((now – last_out_decrement) > out_hold_time )
         $p_{out,m} = p_{out,m} - d_{out,d}$ ;
        last_out_decrement = now;
Gelen her yeni pakette:
    Paket profil içi ise
        Paketi  $p_{in,m}$  olasılığı ile işaretle;
    Paket profil dışı ise
        Paketi  $p_{out,m}$  olasılığı ile işaretle;
```

Şekil 5.1 BIO algoritması

Profil dışı paketler için de benzer şekilde paket işaretleme olasılığı hesaplanılır. Ancak bu olasılığın artırılması kuyruk taşması durumunda değil profil dışı paketler için bir parametre ile belirlenen üst eşik değerinin aşılması ile yapılmaktadır. Profil dışı paketler için paket düşürme olasılığının güncellenme katsayıları da kullanıcı tarafından belirlenen parametrelerdir.

Kuyruğa bir profil dışı bir paket geldiğinde profil dışı paket işaretleme olasılığı ile işaretlenirken, profil içi bir paket geldiğinde de profil içi paket işaretleme olasılığı ile işaretleme yapılır.

5.2 BIO Başarımı

BIO algoritmasının başarımını ölçmek için RIO başarım testlerinde kullanılan ağ yapısı kullanıldı. **Şekil 4.4**'te yapısı görülen ağda 10 adet kaynak düğüm 10 adet hedefe bir darboğaz hat üzerinden bağlanılmıştır. Darboğaz hatta RIO ve BIO kuyrukları kullanılarak ölçümler gerçekleştirilmiştir. RIO parametreleri RIO başarım testlerindeki ile aynı, BIO parametreleri ise BLUE başarım testlerindeki ile aynı seçilmiştir. Düğüm çiftlerinin farklı RTT'lere ($RTT_i = 2 \cdot (i/2 + 1) \cdot 20 + 2$ msn, i =düğüm çifti numarası) sahip olmasının sağlandığı ağda 5 kaynak düğümün hedef hızı 5Mbps iken diğer 5 kaynağın hedef hızı da 1Mbps olarak belirlenmiştir. Darboğaz hattın giriş yönlendiricisinin giriş arabirimindeki işaretleyiciler paketleri bu profillerin içinde kaldıkça profil içi olarak işaretlemektedirler. Profile uyumsuz paketler ise profil dışı olarak işaretlenilmektedir.

BLUE algoritması yönlendirici üzerindeki aktif akış sayısının fazla olması durumunda daha başarılı olduğundan, BLUE algoritmasını temel alan BIO algoritmasının başarım ölçümlerinde de fazla sayıda akışın aktif olması gerekmektedir. Bunu sağlayabilmek için her düğüm üzerinde değişik sayıda akışın aktif olması sağlandı. Testlerde her düğümde 1, 10 ve 100 akışın aktif olduğu durumlar için ölçümler gerçekleştirildi. Tüm benzetimlerde ECN desteği aktive edildi.

Trafiğe uygulanan profil akış bazında değil akış topluluğu bazında çalışmaktadır. Bir kaynak düğümünden gelen tüm akışlar tek bir profil içinde kalacak şekilde işaretleme gerçekleştirilmektedir. Örnek vermek gerekirse, her düğümde 100 akışın aktif olduğu durum için 5Mbps'lık ve 1Mbps'lık trafik profillerini 100'er akış paylaşmaktadır.

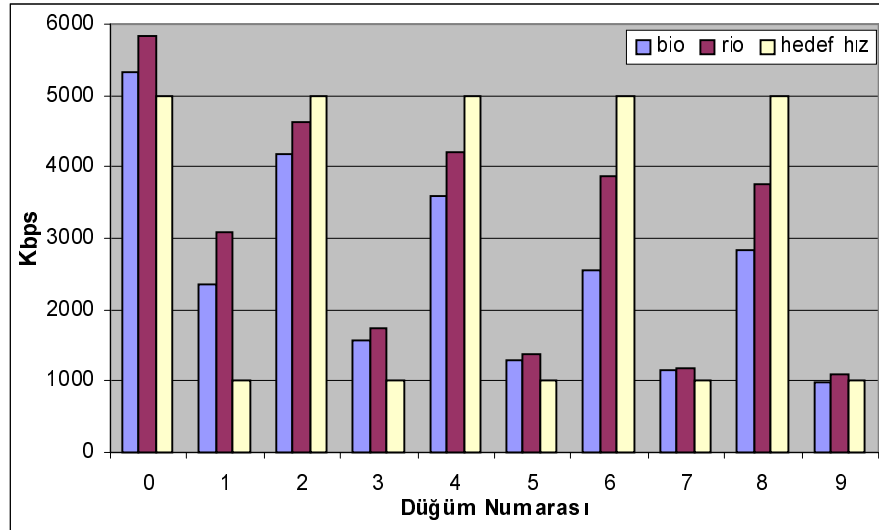
Her akışın sonsuz uzunluktaki bir dosyanın iletimini yaptığı FTP oturumları koşturduğu benzetimde ilk 70 saniye içinde tüm akışlar rastlantısal olarak iletme

başlatılarak hat kapasitesi kullanımı ve paket kayıp oranı ölçümleri 90-190 saniyeleri aralığında gerçekleştirilmiştir.

5.2.1 10 düğüm üzerinde 1'er aktif akış

Yapılan ilk benzetimde RIO başarımlar benzetimi ile aynı şekilde her düğüm üzerinde sadece tek akışın aktif olması sağlandı. Darboğaz hat üzerindeki kuyrukta RIO ve BIO aktive edilerek paket kayıp oranı ve hat kullanımı ölçüldü. Bu ölçümlerin yanında algoritmaların kuyruk kontrolünü izleyebilmek için darboğaz hattaki AF kuyruk boyu da gözlenildi.

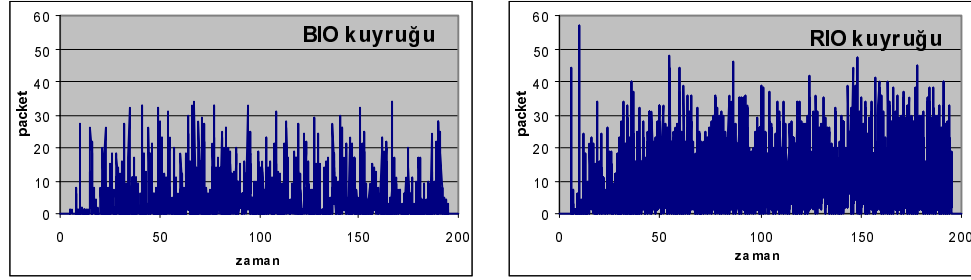
Şekil 5.2'de 10 düğüm üzerinde birer FTP akışının aktive edilmesi durumunda BIO ve RIO kuyruklarının kullanılması ile elde edilen başarılı iletim hızları görülmektedir. RTT arttıkça, başarılı iletim hızında TCP iletim hızının RTT'ye bağımlılığından kaynaklanan bir düşüş görülmektedir. Bu durum RIO benzetimlerinde incelenilmiştir.



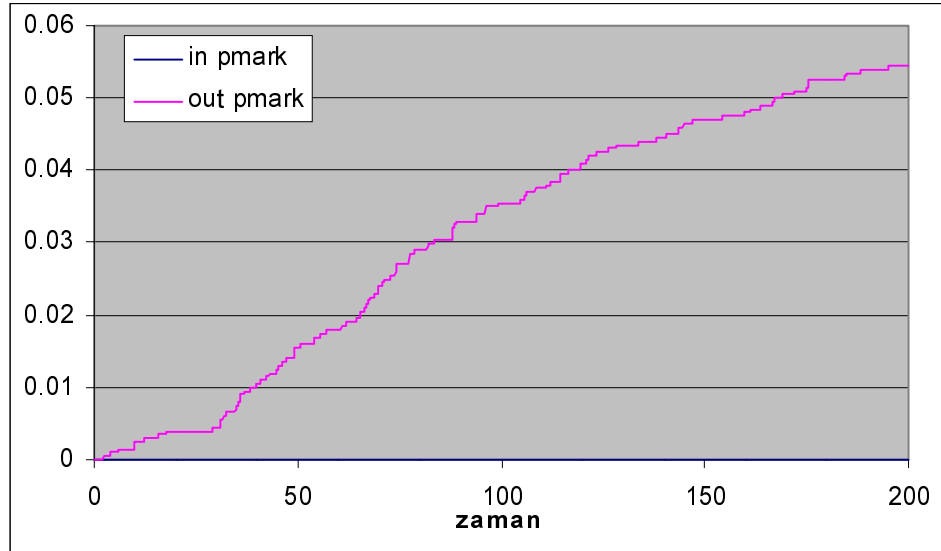
Şekil 5.2 Her düğümde 1 akış aktif olduğu durum için BIO ve RIO başarılı iletim hızları

Bunun dışında tüm akışlar için BIO, RIO'dan daha kötü bir başarımlar göstermektedir. Bunun nedeni ise BIO'nun paket işaretleme olasılığını otomatik olarak güncelleme çabasıdır. **Şekil 5.3'**te görüldüğü gibi BIO kuyruğu ortalaması, RIO kuyruğuna oranla daha düşüktür. Bu durum BIO'nun profil dışı paketleri gereğinden fazla işaretlemesinden kaynaklanmaktadır. BIO, kuyruk boş kaldığında profil dışı paket işaretleme olasılığını ($p_{out,m}$) azaltırken, kuyruk boyunun belirli bir değeri ($maxth_{out}$) geçmesi durumunda da $p_{out,m}$ olasılığı artırılmaktadır. Profil içi paketler nedeni ile kuyruk boyu sürekli olarak $maxth_{out}$ değeri üzerinde dolaşmakta ve bu da $p_{out,m}$

olasılığının sık sık artırılmasına neden olmaktadır. Ancak yine çok sayıda gelen profil içi paketler nedeni ile kuyruk çok az boş kaldığından $p_{out,m}$ olasılığı azaltılamamaktadır. $p_{out,m}$ olasılığının sürekli olarak arttığı **Şekil 5.4**'te de görülmektedir. Bu şekilde de görüldüğü gibi profil içi paket işaretleme olasılığı da "0" olarak kalmaktadır. Profil dışı paketlerin agresif işaretlenmesi ve trafiğin yeterince yoğun olamaması kuyruğun taşmasını engellemektedir.



Şekil 5.3 Her düğümde tek FTP aktif olduğu durum için BIO ve RIO kuyrukları



Şekil 5.4 Her düğümde tek FTP aktif olduğu durum için BIO paket işaretleme olasılığı değişimi

BIO'nun profil dışı paketleri gereğinden fazla işaretlemesi sonucunda RIO'ya oranla daha fazla paket kaybına neden olduğu **Tablo 5.1**'de görülmektedir.

Tablo 5.1 Her düğümde tek FTP aktif olduğu durum için BIO ve RIO paket kayıp oranları

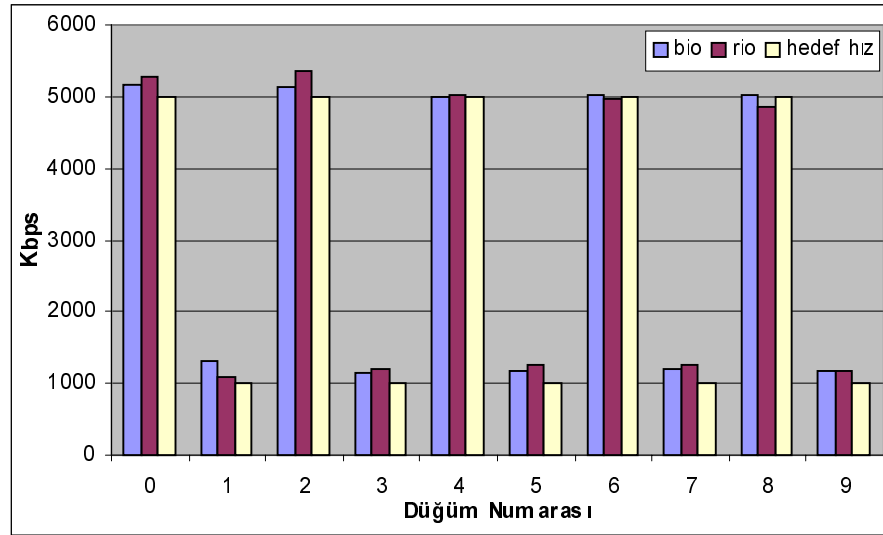
| | RIO | BIO |
|-------------------|----------|----------|
| Paket kayıp oranı | 0,000149 | 0,001134 |

Tüm bunlar gözönüne alındığında paket işaretleme olasılığının otomatik olarak ayarlanıldığı BLUE ve ARED gibi yapıların birden fazla öncelikte trafiğin tek bir kuyrukta toplandığı ve değişik olasılıklar ile işaretleme yapıldığı RIO tipi yapılar içinde kullanımının uygun olmadığı ortaya çıkmaktadır.

5.2.2 10 düğüm üzerinde 10'ar aktif akış

Aktif akış sayısının fazla olduğu durumlarda RED kuyruk boyunu kontrol etmekte başarısız olmaktadır. RIO ve BIO'nun agresif trafik altındaki başarımlarını ölçmek için düğümler üzerindeki aktif akış sayısı artırılarak ölçümler tekrarlanıldı.

Şekil 5.5'te her düğüm üzerinde onar FTP oturumu aktif iken düğümlerin başarılı iletim hızları görülmektedir. Bu benzetimde de BIO, RIO'ya göre daha kötü bir başarımlar elde etmektedir.

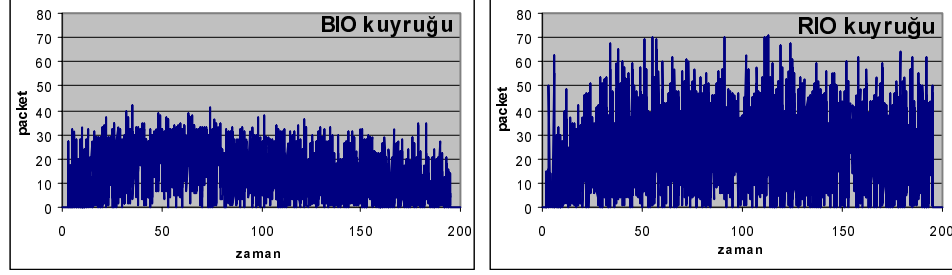


Şekil 5.5 Her düğümde 1 akış aktif olduğu durum için BIO ve RIO başarılı iletim hızları

Şekil 5.6'da görüldüğü gibi BIO kuyruk ortalaması RIO'nun kuyruk ortalamasına göre çok daha düşüktür. RIO kuyruğu sınırlarına dek kullandığı halde BIO ancak kuyruğun yarısını kullanabilmektedir.

BIO'nun kuyruğu verimsiz kullanması başarılı iletim hızlarının düşük olmasına neden olmaktadır. RIO kuyruğu daha fazla kullansa da kuyruğun salınımlı yapısı nedeni ile hat kapasitesi tam olarak kullanılamamaktadır. RIO'nun ortalama kuyruk boyu daha fazla olduğundan ve kuyruk salınımlı olarak hareket ettiğinden, RIO'da kuyruk

sınırlarını aşma nedeni ile oluşan paket kayıpları BIO'ya oranla çok daha fazladır. BIO ve RIO için paket kayıp oranları **Tablo 5.2**'de görülmektedir.

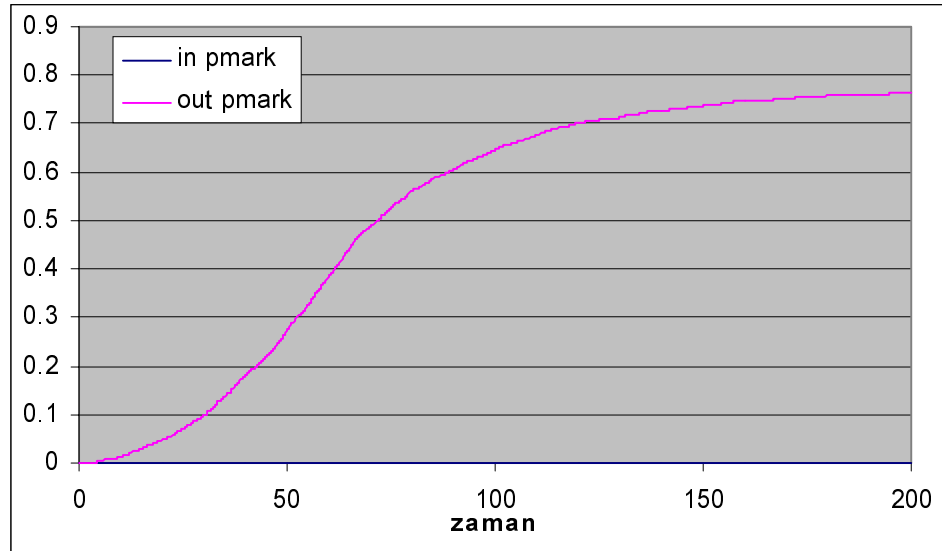


Şekil 5.6 Her düğümde 10 FTP oturumu aktif olduğu durum için BIO ve RIO kuyrukları

Tablo 5.2 Her düğümde 10 FTP aktif olduğu durum için BIO ve RIO paket kayıp oranları

| | RIO | BIO |
|--------------------------|---------|----------|
| Paket kayıp oranı | 0,02501 | 0,010538 |

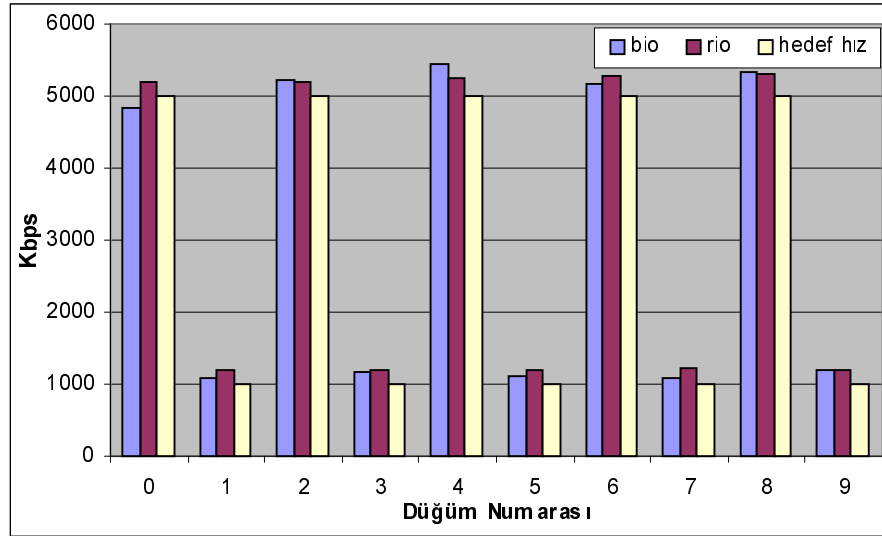
BIO'nun profil dışı paket işaretleme olasılığı ise yapılan ilk benzetimdeki gibi yine gereğinden fazla arttırılmaktadır. **Şekil 5.7**'de görüldüğü gibi benzetim sonlarında bu olasılık tepe değeri olan 1 değerine yaklaşmaktadır.



Şekil 5.7 Her düğümde 10 FTP aktif olduğu durum için BIO paket işaretleme olasılığı değişimi

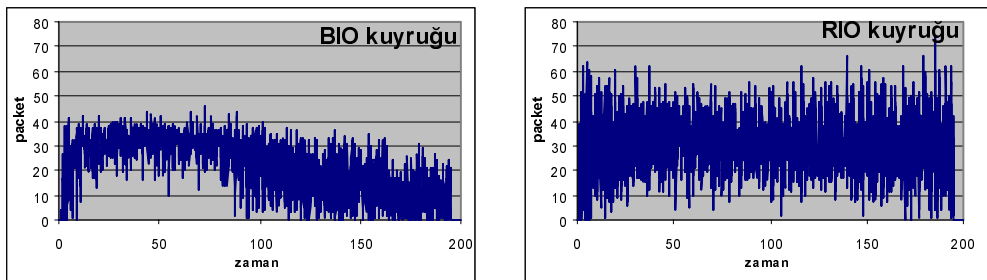
5.2.3 10 düğüm üzerinde 100'er aktif akış

BIO'nun daha agresif bir trafik altındaki davranışını gözlemek için benzetim ağındaki her düğüm üzerinde 100'er adet FTP oturumu aktive edildi. BIO toplamda 1000 akışın aktif olduğu durumda da RIO'ya üstün gelememektedir. **Şekil 5.8**'de görüldüğü gibi BIO başarılı iletim hızı RIO'nun başarılı iletim hızından daha düşüktür.



Şekil 5.8 Her düğümde 100 akış aktif olduğu durum için BIO ve RIO başarılı iletim hızları

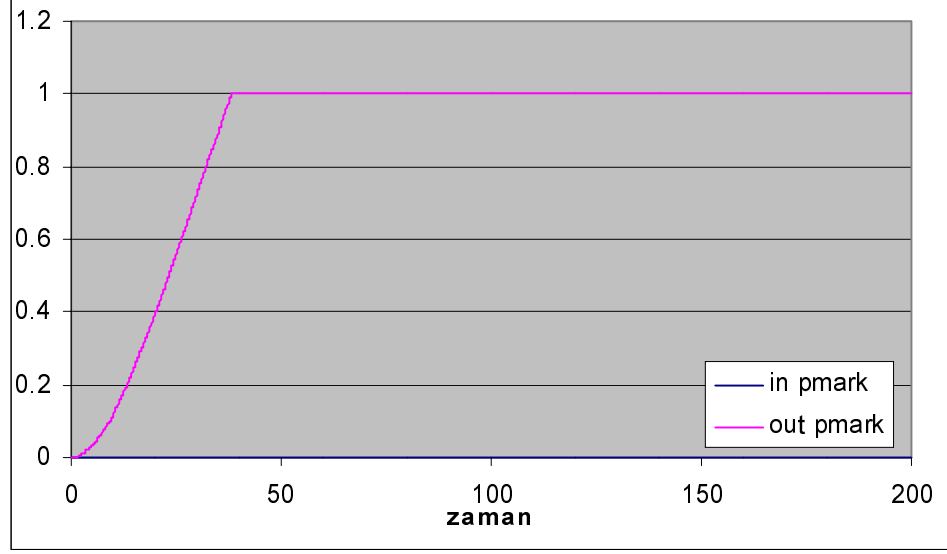
Diğer benzetimlerde olduğu gibi BIO kuyruk ortalamasının, RIO kuyruk ortalamasından daha düşük olduğu **Şekil 5.9**'da görülmektedir.



Şekil 5.9 Her düğümde 100 FTP oturumu aktif olduğu durum için BIO ve RIO kuyrukları

BIO'nun, profil dışı paket işaretleme olasılığını bu benzetimde çok daha hızlı bir şekilde arttırarak bu olasılığı kısa sürede tepe değer olan 1 değerine ulaştırdığından

görülmektedir. Bu nedenle 1000 akışın oluşturduğu yoğun trafik altında bile BIO kuyruk boyu alt sınırlarda dolaşmaktadır.



Şekil 5.10 Her düğümde 10 FTP aktif olduğu durum için BIO paket işaretleme olasılığı değişimi

5.2.4 Analiz

BIO, BLUE yapısını temel alarak tek kuyruk üzerinden değişik önceliklerdeki paketler hizmet verme amacına yönelik olarak tasarlanılmıştır. Yönlendirici üzerinde çok sayıda akışın aktif olduğu durumlarda BLUE'nun RED' karşı üstün gelmesinin faydalarının BIO yapısında da gözlenilmesi beklenilmekteydi.

Ancak seviyeli kuyruk yapısının özellikleri BLUE'nun faydalarının BIO'da tam olarak gözlenebilmesini engellemiştir. İki seviyeli kuyruk yapısında profil dışı paketler, profil içi paketlerden çok önceden düşürülmeye başlanıldığından akışlar RED kuyruk yapısında olduğu kadar agresif davranmamaktadır. Bu durum BIO testlerinde toplam 1000 akışın aktif olduğu benzetimde açık olarak görülmektedir. **Şekil 3.30** ve **Şekil 5.9**'da görüldüğü gibi, 1000 akış aktif olduğunda RED kuyruğu kontrolden çıkarken; RIO kullanıldığında kuyruk genelde çok daha kısa kalmaktadır. Bu, benzetimlerimizde BLUE'nun faydalarının tam olarak görülmesini engellemektedir.

Bunun yanında BIO'nun diğer bir sorunu ise normalde faydalı olan paket işaretleme olasılığının dinamik olarak ayarlanmasıdır. BIO yapısında profil içi ve profil dışı paketler için ayrı paket düşürme olasılıkları bulunur ve kuyruk boyunun ayrı eşik değerlerine ulaşması ile bu olasılıklar güncellenir.

$\max_{th_{out}}$, $\min_{th_{in}}$ değerinden küçük seçilir ve kuyruk boyu $\max_{th_{out}}$ değerini aştığında profil dışı paket işaretleme olasılığı artırılır. Bu olasılığın artması ile daha fazla işaretlenen profil dışı paketlerin miktarı toplam trafik içinde azalsa da profil içi paketler gelmeye devam etmektedir. Gelen profil içi paketler kuyruk boyunu arttırdığından, profil dışı paket işaretleme olasılığı sürekli arttırılmaktadır. Yönlendirici üzerindeki trafik yoğunluğuna bağlı belirli bir süre sonra tepe değerine ulaşan bu olasılık nedeni ile gelen her profil dışı paket işaretlenilmektedir. Bu da iletim hızının düşmesine ve kuyruğun zaman zaman boş kalmasına neden olmaktadır.

Bu soruna çözüm olarak profil dışı paketler için ayrı bir paket işaretleme olasılığı tutmak ve zaman içinde bunu güncellemek yerine, profil dışı paketlerin profil içi paket işaretleme olasılığının belirli bir katı olasılıkla işaretlemesi yolu kullanılabilir. Bu yapıda profil dışı paketlerin kuyrukta bulunabilecekleri üst sınır yine profil dışı paketlerinkinden farklı seçilmeli ve kuyruk boyu bu değeri aştığında tüm profil dışı paketler düşürülmelidir.

5.3 BIO için kullanılan NS konfigürasyonu

Çalışmalarımızda yaptığımız benzetimlerde NS'in 2.1b5 versiyonu ihtiyaçlara göre değiştirilerek kullanıldı. Feng tarafından NS1 için geliştirilen BLUE kodu [79] esas alınarak *ns2.1b5* için tekrar yazılan BLUE kodu yanında, NS paketinde bulunmayan DS desteği için Sean Murphy tarafından geliştirilen modüller temel alınarak oluşturulan DS modülleri kullanıldı [80].

Murphy'nin DS modülleri DS ağı girişlerinde trafiğin işaretlenmesini ve şekillendirilmesini gerçekleyen bir birim ve gelen trafiğe düğüm tarafından farklı seviyelerde hizmetler verilmesini sağlayacak çizelgeleyici ve RED, RIO kuyruklarını içermektedir.

DS yapısında BLUE ve BIO kullanımının testi için Murphy'nin DS kodunun bazı bölümleri değiştirildi. Murphy'nin kodu ECN desteklemediğinden Murphy'nin kendi geliştirdiği tüm kuyruk yapıları yerine geliştirilen yeni kuyruk kodları kullanıldı. NS paketindeki standart RED ve DropTail kodları DS yapısına uyumlu hale gelmesi için değiştirildi. Standart NS dağıtımında varolmayan RIO desteği için James Scott ve Yun Wang tarafından geliştirilen kodun [81] gerekli kısımları DS yapısına uyumlu hale getirilerek kullanıldı. Daha önceden *ns2.1b5*e uyarlanan BLUE kodu da DS yapısına entegre edilerek, BIO için yeni kod geliştirildi.

6 SONUÇ

Internet'in son yıllarda üssel olarak büyümesi ile TCP/IP protokolündeki eksiklikler açığa çıkmaktadır. TCP'nin her kapasitedeki hatlar üzerinde başarılı bir şekilde çalışmasını sağlayan sıkışma denetimi mekanizmalarının temelleri 1986 yıllarındaki ilk sorunlar üzerine yapılan çalışmalar ile atıldı. Günümüzde IETF tarafından Internet'in bugünlere başarılı bir şekilde gelmesini sağlayan bu yapılara ek olarak aktif kuyruk yönetimi mekanizmalarının kullanılması da önerilmektedir.

Karşılaşılan en önemli sorunlardan biri, yönlendirici üzerinde çok sayıda akış aktif durumda ise RED'in kuyruk boyunu kontrol etmede çok başarılı olamamasıdır. ECN ile birlikte kullanıldığında iletim hızını arttırırken paket kayıp oranını da düşüren BLUE aktif kuyruk yönetim mekanizması RED'e alternatif olarak öne sürülmektedir.

IP ağlarının sıkışma yanındaki bir diğer sorunu da yeni ağ uygulamalarının farklı hizmet alma beklentisinin karşılanmasıdır. IETF tarafından 1994 yılında bu soruna çözüm amacına yönelik olarak başlatılan Tümüleşik Hizmetler mimarisi, yapısındaki ölçeklenebilirlik sorunları nedeni yeterince yaygınlaşmamıştır. Tümüleşik Hizmetler yapısının ölçeklenebilirlik sorunları üzerine, IETF'de son yıllarda yeni bir yapı olan Farklılaştırılmış Hizmetler üzerinde çalışılmaktadır.

Tümüleşik Hizmetler yapısı IP ağları üzerinde ölçeklenebilir ve farklı hizmetler sunmak için gerekli yapıları tanımlarken ağ üzerindeki düğümlerin bu hizmetleri destekleyebilmesi için RED aktif kuyruk yönetimi yapısının değiştirilmiş bir hali olan RIO kuyruk yapısı kullanılmaktadır.

Bu tez çalışmasında üzerinde yoğun olarak çalışılan aktif kuyruk yönetimi mekanizmaları detaylı bir şekilde incelenerek yapılan benzetimler ile performansları incelendi. Üzerinde en çok çalışılan aktif kuyruk yönetim mekanizması olan RED'in yararları ve yapısındaki sorunlar incelenerek bu sorunlara çözüm sağlayan ARED ve diğer mekanizmalar incelendi. Benzetimlerimizde birçok araştırmada aktif olarak kullanılan ve USC/ISI, XEROX PARC, LBLN, UCB kurumlarından oluşan bir grup tarafından ortaklaşa olarak C++ ve OTCL dillerinde geliştirilen ve kaynak kod olarak dağıtılan NS isimli ağ benzetim aracı kullanılmıştır.

Ayrıca, henüz çok yeni olan ve üzerinde hiçbir değerlendirme çalışması yapılmamış bulunan BLUE algoritması gerçekleştirilen benzetim kodu ile detaylı bir şekilde incelenerek alternatifi olan RED ile kıyaslanılmıştır.

Yapılan benzetimler ve analizler sonucunda temel olarak aşağıdaki sonuçlar çıkarılmıştır:

1. DT tipi kuyruklara karşı sıkışma önleme ve kuyruk boyunu kontrol etmede başarılı bulunan RED aktif kuyruk yönetim mekanizmasının parametre seçimi ağ ve trafik yapısı ve akış sayısı gibi birçok etkene bağlı olarak yapılmalıdır.
2. RED mekanizması üzerine bir iyileştirme olan ARED, RED'in parametrelerini akış sayısına ve trafik yüküne göre dinamik olarak ayarlayarak parametre seçimi işlemini bir miktar kolaylaştırmaktadır.
3. Yeni geliştirilen BLUE aktif kuyruk yönetim yapısı ECN ile birlikte kullanıldığında RED'e karşı düşük paket kayıp oranı ve yüksek iletim hızı ile daha üstün gelmektedir. Ancak şu an ECN'in geniş çaplı bir kullanımı sözkonusu değildir. ECN kullanılmadığında ise toplam kayıp ve iletim hızı konularında BLUE'nun RED' bir üstünlüğü kalmamaktadır.
4. RIO tipi kuyruk yapısı ile tek bir kuyruk üzerinde değişik seviyelerde hizmet vermek mümkün olsa da, Farklılaştırılmış Hizmetler yapısındaki AF hizmeti için RIO kullanıldığında hedeflenen hızı yüksek olan akışların beklenen hıza ulaşmaları mümkün olamamaktadır. Yüksek hedef hızı olan akışlar ancak kuyruğa fazla kapasite ayırımı ile mümkün olabilmektedir.

Bu tez çalışmasında RIO'nun, parametre seçimi ve aktif akış sayısı fazla olduğunda kuyruk boyunu kontrol etmeye ilişkin sorunlarına karşılık, BLUE algoritmasını temel alan BIO algoritması geliştirilmiştir. Geliştirdiğimiz BIO algoritmasının kullanılması ile çok sayıda akışın aktif olduğu yönlendiricilerde kuyruk boyunun daha iyi kontrol edilmesi ve paket düşürme oranının azaltılması amaçlanmıştır. Ancak paketlerin profil içi ve dışı olarak sınıflandırıldığı ve buna göre hizmet verildiği yapılarda yönlendirici üzerindeki trafiğin normaldeki kadar agresif olmaması nedeni ile geliştirilen yapının faydaları gözlenememiştir. Bunun yanında profil dışı paket işaretleme olasılığının, profil içi paketler nedeni ile sürekli artarak 1 değerine ulaşması paketlerin gereğinden fazla işaretlenilmesine neden olmuştur. Bu hat kapasitesinin verimsiz kullanılması sonucunu doğurmuştur.

Gelecekte bu tez çalışması üzerine yapılabilecek ek çalışma olarak BIO yapısında profil dışı paketlerin gereğinden fazla işaretlenilmesini önleme amacına yönelik olarak önerdiğimiz değişikliğin gerçekleştirilerek, değiştirilmiş BIO'nun başarımının daha agresif bir trafik altında benzetimler ve gerçek sistemler ile ölçülmesi gösterilebilir. Bunun yanında hedef hızı yüksek olan akışların beklenenden daha az hizmet almasını engellemek için [78] ile RIO için önerilen yapıların BIO yapısındaki başarımlarını incelenebilir.

KAYNAKLAR

- [1] **Internet Software Consortium**, 2000, Internet Domain Survey, <http://www.isc.org/ds/host-count-history.html>
- [2] NS: Network Simulator, <http://www-mash.cs.berkeley.edu/ns/>
- [3] REAL Network Simulator, <http://www.cs.cornell.edu/skeshav/real/overview.html>
- [4] VINT (Virtual InterNetwork Testbed) Project, <http://netweb.usc.edu/vint/>
- [5] DARPA, <http://www.darpa.mil/>
- [6] **Stevens, R.**, 1994, TCP/IP Illustrated: the protocols, *Addison-Wesley Publishing Company, Inc.*, Massachusetts
- [7] **Ferguson, P., Huston, G.**, 1998, Quality of Service, Delivering QoS on the Internet and in Corporate Networks, ISBN 0-471-24358-2, *John Wiley & Sons, Inc.*, New York
- [8] **Cisco Documentation**, Quality of Service (QoS) Networking, http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/qos.htm
- [9] **Ferguson, P., Huston, G.**, 1998, Quality of Service in the Internet: Fact, Fiction, or Compromise, <http://www.employees.org:80/~ferguson/inet-qos.ps>
- [10] **Floyd, S., Jacobson, V.**, 1995, Link-sharing and Resource Management Models for Packet Networks, *IEEE/ACM Transactions on Networking*, Vol. 3 No 4, <http://www.aciri.org/floyd/papers/link.pdf>
- [11] **Keshhav, S.**, 1997, An Engineering Approach to Computer Networking, pp 234-236, *Addison-Wesley*
- [12] **IETF**, Integrated Servies working group, <http://www.ietf.org/html.charters/intserv-charter.html>
- [13] **Braden, R., Clark, D., Shenker, S.**, 1994, Integrated Services in the Internet Architecture: an Overview, Request for Comments: 1633
- [14] **Shenker, S., Partridge, C., Guerin, R.**, 1997, Specification of Guaranteed Quality of Service, Request for Comments: 2212
- [15] **Shenker, S., Partridge, C., Guerin, R.**, 1997, Specification of the Controlled-Load Network Element Service, Request for Comments: 2211

- [16] **IETF**, Integrated Services over Specific Link Layers (issll) working group, <http://www.ietf.org/html.charters/issll-charter.html>
- [17] **Garrett, M., Borden, M.**, 1998, Interoperation of Controlled-Load Service and Guaranteed Service with ATM, Request for Comments 2381
- [18] **Crawley, E., Berger, L., Berson, S., Baker, F., Borden, M., Kawczyk, J.**, 1998, A Framework for Integrated Services and RSVP over ATM, Request for Comments 2382
- [19] **Bernet, Y., Binder, J., Blake, S., Carlson, M., Carpenter, B., Keshav, S., Davies, E., Ohlman, B., Verma, D., Wang, Z., Weiss, W.**, 1999, A Framework for Differentiated Services, Draft-ietf-diffserv-framework-02.txt,
- [20] **Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.**, 1998, An Architecture for Differentiated Services, Request for Comments: 2475
- [21] **Information Sciences Institute University of Southern California**, 1981, INTERNET PROTOCOL, Request for Comments: 791
- [22] **Deering, S., Hinden, R.**, 1998, Internet Protocol, Version 6 (IPv6) Specification, Request for Comments: 2460
- [23] **Bless, B., Wehrle, K.**, 1999, A Lower Than Best-Effort Per-Hop Behaviour, Internet Draft: draft-bless-diffserv-lbe-phb-00.txt
- [24] **Diederich, J., Zitterbart, M.**, 1999, An Expedited Forwarding with Dropping PHB, Internet Draft: draft-dieder-diffserv-phb-efd-00.txt
- [25] **Kilikki, K., Ruutu, J.**, 1999, Interoperability PHB Group, Internet Draft: draft-kilikki-diffserv-interoperability-00.txt
- [26] **Caret, S., Hodgkinson, T., O'Neill, A., Mortimore, D.**, 1998, A Bounded-Delay service for the internet, Internet Draft: draft-carter-bounded-delay-00.txt
- [27] **Heinanen, J., Baker, F., Weiss, W., Wroclawski, J.**, 1999, Assured Forwarding PHB Group, Request for Comments: 2597
- [28] **Jacobson, V., Nichols, K., Poduri, K.**, 1999, An Expedited Forwarding PHB, Request for Comments: 2598
- [29] **Nichols, K., Blake, S., Baker, F., Black, D.**, 1998, Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, Request for Comments: 2474
- [30] **Nichols, K., Jacobson, V., Zhang, L.**, 1999, A Two-bit Differentiated Services Architecture for the Internet, Internet Draft: draft-nichols-diff-svc-arch-02
- [31] **Stoica, I., ve diğ erleri**, 1999, Per Hop Behaviours Based on Dynamic Packet States, Internet Draft: draft-stoica-diffserv-dps-00.txt

- [32] **Stoica, I., Zhang, H.**, 1999, Providing Guaranteed Services Without Per Flow Management, *Proceedings of ACM SIGCOMM'99*, BOSTON, MA, pp. 81-94, <http://redriver.cmcl.cs.cmu.edu/~hzhang-ftp/SIGCOM99.pdf>
- [33] **Köhler, S., Schafer, U.**, 1999, Performance Comparison of Different Class-and-Drop Treatment of Data and Acknowledgements in DiffServ IP Networks, *University of Würzburg Institute of Computer Science Research Report Series, Report No 237*, <http://www-info3.informatik.uni-wuerzburg.de/TR/tr237.ps.gz>
- [34] **Kim, H., Leland, W., Thomson, S.**, Evaluation of Bandwidth Assurance Service using RED for Internet Service Differentiation, *Bell Communications Research*, <ftp://ftp.bellcore.com/pub/world/hkim/assured.ps.Z> veya <ftp://ftp.telcordia.com/pub/world/hkim/assured.ps.Z>
- [35] **Braden, R., Zhang, L., Berson, S., Herzog, S., Jamin, S.**, 1997, Resource ReSerVation Protokol (RSVP) – Verison 1 Functional Specification, Request for Comments: 2205
- [36] **Nagle, J.**, 1984, Congestion Control in IP/TCP Internetworks, Request For Comments: 896
- [37] **Jacobson, V.**, 1988, Congestion Avoidance and Control, *Proceedings of SIGCOMM '88*, Stanford, CA, ACM, <http://www-nrg.ee.lbl.gov/papers/congavoid.pdf>
- [38] **Braden, R.**, 1989, Requirements for Internet Hosts -- Communication Layers, Request For Comments: 1122
- [39] **Stevens, W.**, 1997, TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms, Request For Comments: 2001
- [40] **Jain, R.**, 1990, Congestion Control in Computer Networks: Trends and Issues, *IEEE Network*, pp. 24-30., http://www.cis.ohio-state.edu/~jain/papers/cong_tre.htm
- [41] **Mankin, A., Ramakrishnan, K.**, 1991, Gateway Congestion Control Survey, Request For Comments: 1254
- [42] **Yang, C., Reddy, A.**, 1995, A taxonomy for congestion control algorithms in packet switching networks., *IEEE Network Magazine*, Vol 9, Number 5, <http://www.comsoc.org/pubs/surveys/yang/yang-orig.html>
- [43] **Braden, B. ve diğerleri**, 1998, Recommendations on Queue Management and Congestion Avoidance in the Internet, Request For Comments: 2309
- [44] **Floyd, S.**, 2000, Congestion Control Principles, Internet Draft: draft-ietf-floyd-cong-03.txt
- [45] **Floyd, S., Fall, K.**, 1999, **Promoting the Use of End-to-End Congestion Control in the Internet**, *IEEE/ACM Transactions on Networking*, <http://www.aciri.org/floyd/papers/collapse.may99.pdf>

- [46] **Defense Advanced Research Projects Agency Information Processing Techniques Office**, 1981, TRANSMISSION CONTROL PROTOCOL, Request For Comments: 793
- [47] **Jacobson, V.**, 1990, Modified TCP Congestion Control Algorithm, end2end-interest mailing list, <ftp://ftp.ee.lbl.gov/email/vanj.90apr30.txt>
- [48] **Allman, M., Paxson, V., Stevens, W.**, 1999, TCP Congestion Control, Request For Comments: 2581
- [49] **Floyd, S., Jacobson, V.**, 1993, Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Transactions on Networking*, <http://www.aciri.org/floyd/papers/early.pdf>
- [50] **Postel, J.**, 1981, INTERNET CONTROL MESSAGE PROTOCOL, Request For Comments: 792, September 1981
- [51] **Prue, W., Postel, J.**, 1987, Something a Host Could Do with Source Quench: The Source Quench Introduced Delay (SQULD), Request For Comments: 1016
- [52] **Baker, F.**, 1995, Requirements for IP Version 4 Routers, Request For Comments: 1812
- [53] **Salim**, 1998, Backward ECN for the Internet Protocol, Internet Draft: draft-salim-jhsbnns-ecn-00.txt
- [54] **Ramakrishnan, K.K., and Jain, R.**, 1990, A binary feedback scheme for congestion avoidance in computer networks, *ACM Transactions on Computer Systems*, V.8, N.2, pp. 152-181
- [55] **Rosolen, V., Bonaventure, O., and Leduc, G.**, 1999, A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic, *ACM Computer Communication Review*, and Impact of cell discard strategies on TCP/IP in ATM UBR networks, *Proc. of the 6th Workshop on Performance Modelling and Evaluation of ATM Networks (IFIP ATM'98)* Ilkley, UK, July 98.
- [56] **Floyd, S.**, 2000, Recommendation on using the "gentle_" variant of RED, <http://www.aciri.org/floyd/red/gentle.html>
- [57] **Labrador, M., Banerjee, S.**, 1999, Packet Dropping Policies for ATM and IP Networks, *IEEE Communications Surveys*, vol 2, no 3, <http://www.comsoc.org/pubs/surveys>
- [58] **May, M., Bonald, T., Bolot, J.**, 2000, Analytic Evaluation of RED Performance, INRIA, Sophia-Antipolis, France, <http://www-sop.inria.fr/rodeo/personnel/mmay/papers/infocom00.ps>
- [59] **Feng, W., Kandlur, D., Shin, K.**, 1999, A Self-configuring RED Gateway, INFOCOM '99, <http://www.eecs.umich.edu/~wuchang/work/CSE-TR-349-97.ps.Z>
- [60] **Feng, W.**, 1996, Internet e-mail, http://www.eecs.umich.edu/~wuchang/red/misc/dynamic_red.txt

- [61] **Floyd, S.**, 1997, RED: Discussions of Setting Parameters, email message, <http://www.aciri.org/floyd/REDparameters.txt>
- [62] **Jacobson, V.**, 1998, Notes on using RED for Queue Management and Congestion Avoidance, *NANOG 13*, Dearborn, MI
- [63] **Doran, S.**, 1998, EBONE Interface Graphs, <http://adm.ebone.net/~smd/red-1.html>
- [64] **May, M., Bolot, J., Diot, C., Lyles, B.**, 1999, Reasons not to deploy RED, INRIA Sophia-Antipolis, ENSIM, Sprintlabs, <http://www-sop.inria.fr/rodeo/personnel/mmay/may-red.html> <http://www-sop.inria.fr/rodeo/personnel/mmay/papers/slides-iwqos-sprint.ps>
- [65] **Jacobson, V., Nichols, K., Poduri, K.**, 1999, RED in a Different Light Draft Version on June 1, 1999, http://www.cisco.com/public/cons/red_light.pdf http://www.cs.ucla.edu/classes/fall99/cs219/papers/red_light.pdf.gz
- [66] **Anjum, F., Tassiulas, L.**, 1999, Balanced-RED: An Algorithm to Achieve Fairness in the Internet, *Center for Satellite and Hybrid Communication Networks Technical Research Report*
- [67] **Feng, W., Kandlur, D., Saha, D., Shin, K.**, 2000, BLUE: A new class of active queue management, <http://www.eecs.umich.edu/~wuchang/blue/>
- [68] **Ramakrishnan, K., Floyd, S.**, 1999, A Proposal to add Explicit Congestion Notification (ECN) to IP, Request for Comments: 2481
- [69] **Floyd, S.**, 1994, TCP and Explicit Congestion Notification, *ACM Computer Communication Review*, V. 24 N. 5, p. 10-23, ftp://ftp.ee.lbl.gov/papers/tcp_ecn.4.ps.Z
- [70] **Bernet, Y., Smith, A., Blake, S., Grossman, D.**, 2000, A Conceptual Model for Diffserv Routers, Internet Draft: draft-ietf-diffserv-model-03.txt
- [71] **Heinanen, J., Guerin, R.**, 1999, A Single Rate Three Color Marker, Request for Comments: 2697
- [72] **Heinanen, J., Guerin, R.**, 1999, A Two Rate Three Color Marker, Request for Comments: 2698
- [73] **Clark, D., Fang, W.**, Explicit Allocation of Best Effort Packet Delivery Service, <http://diffserv.lcs.mit.edu/Papers/exp-alloc-ddc-wf.pdf>
- [74] **Ibanez, J., Nichols, K.**, 1998, Preliminary Simulation Evaluation of an Assured Service, Internet Draft: draft-ibanez-diffserv-assured-eval-00.txt, <http://ec.eurecom.fr/~ibanez/draft-ibanez-diffserv-assured-eval-00.pdf>
- [75] **Feng, W.**, Email on DiffServ WG, 1998, <http://www-nrg.ee.lbl.gov/diff-serv-arch/msg02184.html>
- [76] **Heinanen, J.**, Email on DiffServ WG, 1998, <http://www-nrg.ee.lbl.gov/diff-serv-arch/msg02176.html>

- [77] **Medina, O.**, Email on DiffServ WG, 1998, <http://www-nrg.ee.lbl.gov/diff-serv-arch/msg02174.html>
- [78] **Yeom, I., Reddy, N.**, 1999, "Realizing throughput guarantees in a differentiated services network," *Proceedings of ICMCS*, <http://dropzone.tamu.edu/~ikjun/diff-serv.ps>
- [79] **Feng, W.**, BLUE code for NS1, <http://www.eecs.umich.edu/~wuchang/blue/ns-blue.tgz>
- [80] **Murphy, S.**, Diffserv additions to ns-2, <http://www.teltec.dcu.ie/~murphys/ns-work/diffserv/index.html>
- [81] **Scott, J., Wang, J.**, RIO code for ns2.1b6, <http://www.aciri.org/ns/rio-ns2-1b6.tar.gz>
- [82] **Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.**, 1996, RTP: A Transport Protocol for Real-Time Applications, Request for Comments: 1889

EK A NS AĞ BENZETİM ARACI

Benzetim çalışmalarımız için kendimize özgü bir araç geliştirmek yerine bu tip çalışmalarda yaygın olarak kullanılan **NS** [2] isimli ağ benzetim aracı kullanıldı. NS değişik ağ araştırmalarında kullanılmış ve halen kullanılmakta olan bir benzetim ortamıdır. NS TCP, yönlendirme ve çoklu iletim (multicast) konularının benzetimi için C++ ve TCL ile yazılmış modüllerden oluşmaktadır.

1989'da REAL [3] benzetim aracının üzerine geliştirilmeye başlanan NS zaman içinde büyük değişikliklere uğramıştır. NS şu an VINT [4] (Virtual InterNetwork Testbed) projesinin bir parçası olarak USC/ISI (University of Southern California, Information Sciences Institute), XEROX PARC (XEROX Palo Alto Research Center), LBNL (Lawrence Berkeley National Laboratory), UCB (University of California, Berkeley) kurumlarından oluşan bir grup tarafından ortaklaşa geliştirilmektedir. VINT, mevcut ve gelecekteki protokollerin benzetimine olanak sağlayacak bir benzetim ortamı geliştirme amacına yönelik DARPA [5] tarafından desteklenen bir projedir.

Çeşitli RFC'ler, RED, RIO, ECN, BLUE ve DS benzetimleri gibi birçok çalışmada kullanılan NS halen üzerinde çalışılan ve değişik gruplarca hem kullanılan hem de desteklenen bir benzetim aracıdır.

6.1 Yapısı

NS'in temel işlevlerinin gerçekleştirildiği bölüm C++ ile nesne tabanlı bir yapıda yazılmıştır. Kullanıcılar benzetim senaryosunu OTCL (Object TCL) kullanarak yazmaktadırlar. Bunu sağlayabilmek için C++ kodlarının bir de OTCL arabirimi bulunmaktadır. Tüm NS kodu C++ ve OTCL ile yazılmıştır.

Benzetim Simulator isimli sınıf tarafından yönetilmektedir. Temel işlevler ise Node, Link ve Agent sınıfları ve bunların alt sınıfları tarafından yapılmaktadır.

Node sınıfı ağ üzerindeki bir düğümü temsil etmektedir. Uç nokta düğümü ve yönlendirici olabilen Node sınıfı, paket alıp-verme ve yönlendirme yeteneklerine sahiptir. Düğümler Link sınıfı ile birbirine bağlanmaktadır ve tek-yönlü, çift-yönlü ve CBQ hatları kullanılabilir. Düğümlerin çıkış arabirimlerinde bulunan kuyruklar Queue sınıfları ile temsil edilirler ve DT, RED, FQ, SFQ, DRR, CBQ kuyruk tipleri standart olarak desteklenilmektedir.

Benzetimi asıl yürüten sınıf ise Agent sınıfıdır ve bu sınıf iletim seviyesi protokolünü işleten süreçler olarak düşünülebilir. Agent sınıfının alt sınıfları ve işlevleri şu şekilde özetlenebilir:

İletim seviyesi protokollerini gerçekleyen bu sınıfların üzerinde ise değişik uygulamaları emüle eden Application sınıfı bulunmaktadır. Bu sınıfın telnet, FTP, Traffic alt sınıfları bulunmaktadır. Traffic sınıfı ise exponensiyel trafik üreten

Traffic/Expoo, pareto dağılımına göre trafik üreten Traffic/Pareto, sabit bir trafik üreten Traffic/CBR ve gerçek ağ trafiğini üreten Traffic/Trace alt sınıfları bulunmaktadır.

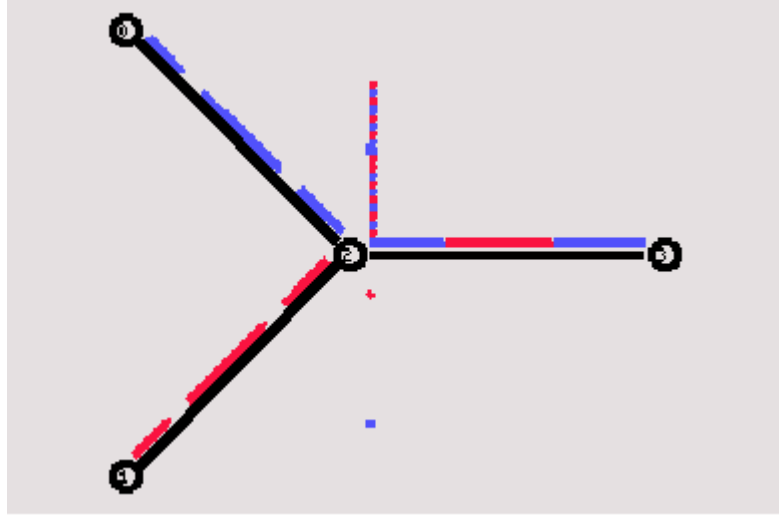
Tablo A.1 NS benzetim aracı temel objeleri

| Agent alt-sınıfı | İşlevi |
|-----------------------|---|
| Null | Sadece gelen trafiği alır |
| LossMonitor | Düşülen paketlerin iletileceği hedef sınıfı |
| TCP | TCP iletim protokolünün değişik versiyonlarını gerçekleyen alt sınıflar |
| TCP/FullTcp | |
| TCP/Reno | |
| TCP/NewReno | |
| TCP/Vegas | |
| TCP/Sack1 | |
| TCP/Fack | |
| TCPSink | |
| TCPSink/DelAck | |
| TCPSink/Sack1/ | TCP sınıfları tarafından gelen paketleri alarak bunlar için paket alındıkları üreten alt sınıflar |
| DelAck | |
| UDP | UDP protokolünü gerçekleyen alt sınıf |
| RTP | RTP protokolünü gerçekleyen alt sınıf |
| Session/RTP | |
| RTCP | |
| SRM | |

UNIX ve LINUX sistemlerinde geliştirilen ns Windows sistemlerinde de çalışabilmektedir. Kaynak kod olarak dağıtılan ns, diğer araştırmacıların da istedikleri yerleri değiştirebilmelerine ve yeni yapılar eklemelerine olanak tanımaktadır. VINT projesi tarafından geliştirilen standart kısım dışında diğer geliştiriciler tarafından eklenen modüller de tüm kullanıcıların kullanımına sunulmaktadır [2].

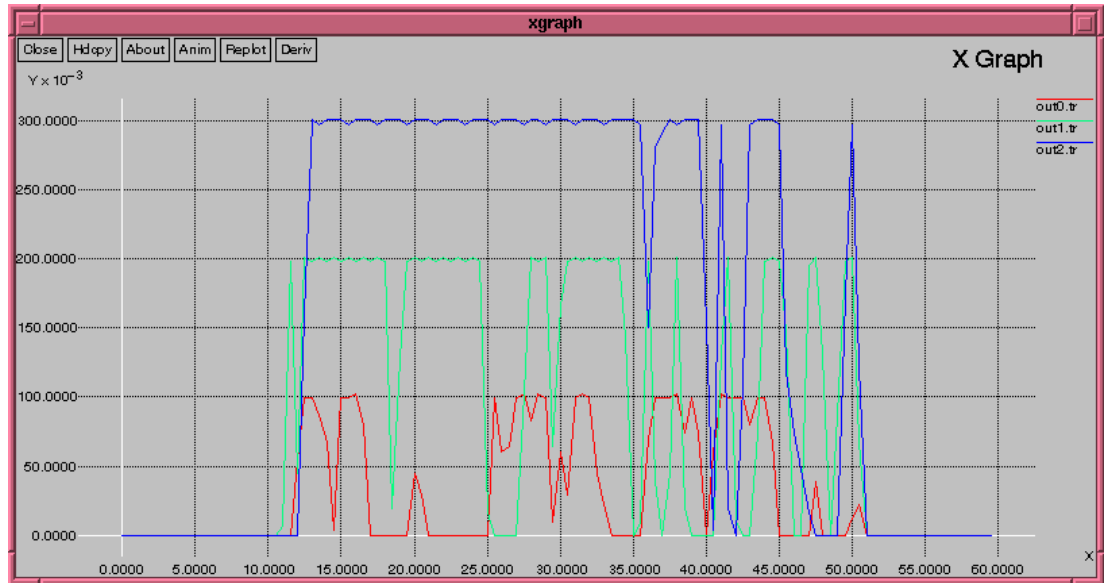
Temel ağ bileşenlerinin dışında benzetimden veri toplayabilmek ve bunların görüntülenmesi için de destek bulunmaktadır. Veri toplama amacına yönelik kuyruk ve akış izleme sınıfları bulunmaktadır. NS'in olay tabanlı yapısı sayesinde belirli zamanlarda belirli verileri istenilen dosyalarda toplamak mümkündür.

Toplanan verilerin kullanıcı tarafından izlenebilmesi ve yorumlanabilmesi için NS paketinde iki temel araç bulunmaktadır. Bunlardan ilki nam'dır (network animator). Bu araç tüm benzetim süresince otomatik olarak toplanan veriler sayesinde, benzetimi kullanıcıya tekrar oynatır.



Şekil A.1 nam örnek ekran görüntüsü

Şekil A.1'de nam'ın basit bir örnek ağ için koşturulduğundaki ekran görüntüsü bulunuyor. Şekildeki daireler kullanıcı tarafından tanımlanan düğümleri gösterirken, çizgiler de hatları göstermektedir. Bu benzetimde 0 ve 1 numaralı düğümler 2 numaralı düğüm üzerinden 3 numaralı düğüm ile haberleşmektedir. Hatlar üzerindeki parçalı çizgiler paketleri göstermektedir. Bu örnek için farklı kaynaktan gelen paketlere farklı renkler verilmiştir. 2 numaralı düğüm yanındaki yükselti ise 3 numaralı düğümüne giden hattın kuyruğunu göstermektedir. Dikkat edildiğinde kuyruktan düşürülen paketler de görülebilir.



Şekil A.2 Örnek bir xgraph ekranı

Nam, benzetimin genel gidişini görmek ve bazı basit yapıların işleyişini kontrol etmek için kullanılabilir. Ancak tam olarak faydalı istatistiksel bilgiler elde edilemez. Ayrıca, benzetim süresi uzadıkça nam'ın kullanması için ns tarafından hazırlanan dosya çok büyüyebilmektedir.

Toplanan kuyruk boyları, paket kayıpları gibi verilerin kullanıcı tarafından görüntülenmesi için ise xgraph isimli araç bulunmaktadır. Xgraph tarafından giriş olarak alınan ASCII dosyalar, benzetim sırasında periyodik olarak güncellenir.

Şekil A.2'de farklı hızlarda iletim yapan 3 CBR kaynağının elde ettikleri iletim hızları görülmektedir. Yatayda zaman gösterilirken, dikeyde kuyruk boyu gösterilmektedir. Xgraph, bu şekilde veri kıyaslama ve görme amaçlı kullanım için oldukça uygun bir araçtır.

EK B GECİKME DEĞİŞİMİ HESABI

Gecikme değişimini, bazıları belirli bir zaman aralığındaki paketlerin iletim gecikmelerinin en üst ve en alt değerlerinin farkı olarak, bazıları da ardışıl paketlerin iletimleri sırasında oluşan gecikmelerinin farkının üst sınırı olarak algılamaktadır.

i numaralı paketin kaynaktan gönderiliş zamanının $gönderiliş_i$ ve bu paketin hedef düğüme varış zamanının da $varış_i$ olduğu düşünülürse, gecikme değişiminin ilk tanımı şu şekilde formülize edilebilir:

$$J = \max(varış_i - gönderiliş_i) - \min(varış_i - gönderiliş_i) \quad \text{Denklem B.1}$$

İkinci tanımın formülü ise aşağıdaki gibidir.

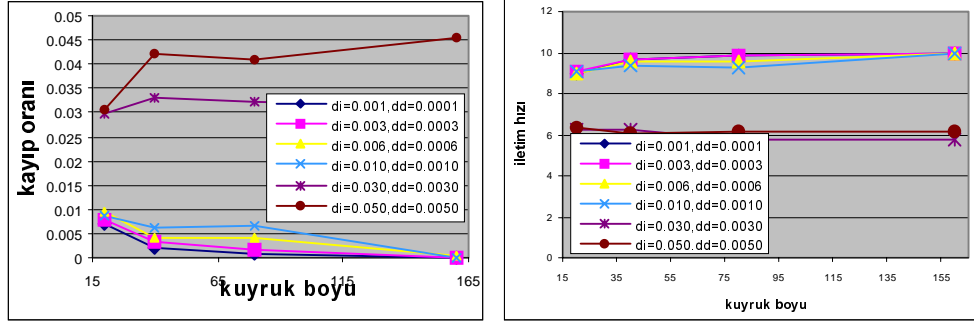
$$J = \max(| (varış_i - gönderiliş_i) - (varış_{i-1} - gönderiliş_{i-1}) |) \quad \text{Denklem B.2}$$

Yapılan benzetimlerdeki gecikme değişimini ölçmek için [82]'de verilen tanım kullanılmıştır. Bu tanıma göre i numaralı paket geldiğinde gecikme değişimi şu şekilde hesaplanır:

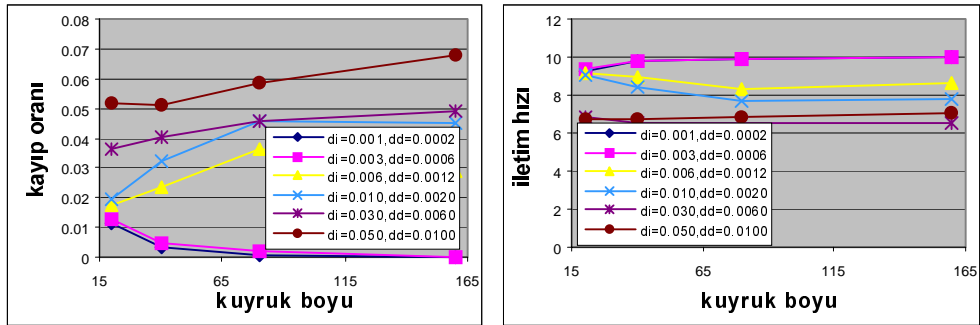
$$D_{i,j} = (varış_j - gönderiliş_j) - (varış_i - gönderiliş_i) \quad \text{Denklem B.3}$$

$$J_i = 15/16 * J_{i-1} + 1/16 * |D_{i-1,i}| \quad \text{Denklem B.4}$$

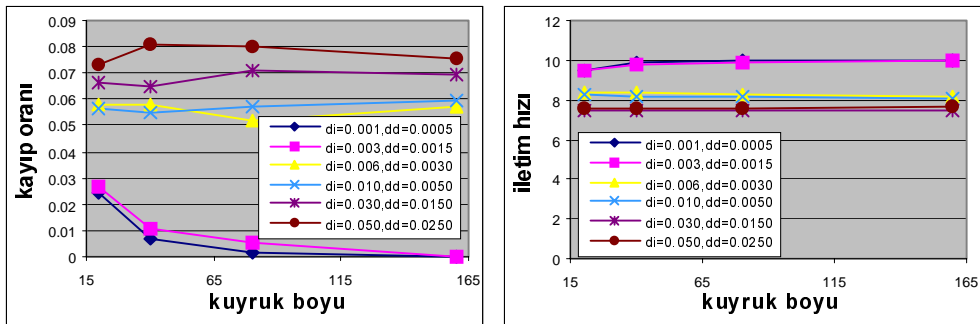
EK C BLUE PARAMETRE ANALİZİ İÇİN GERÇEKLEŞTİRİLEN BENZETİM SONUÇLARI



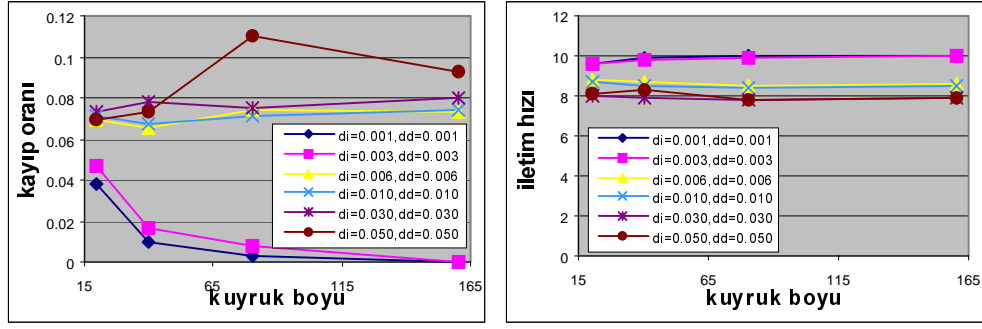
Şekil C.1 d_i/d_d oranı 10 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı



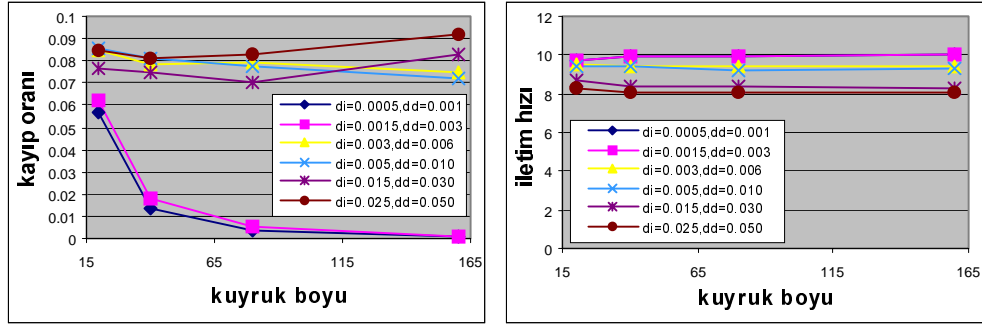
Şekil C.2 d_i/d_d oranı 5 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı



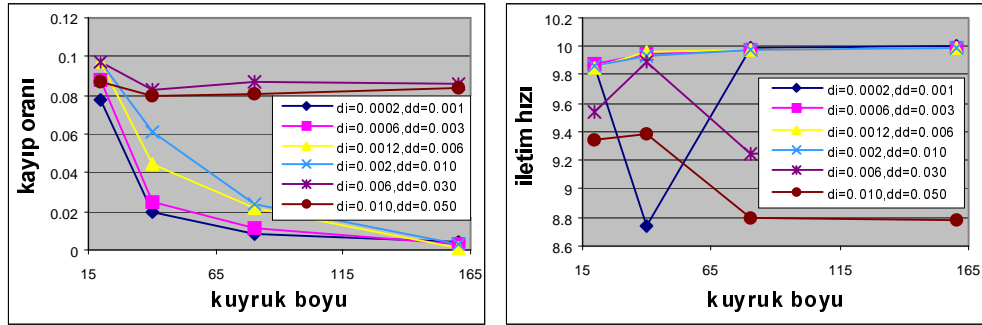
Şekil C.3 d_i/d_d oranı 2 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı



Şekil C.4 d_i/d_d oranı 1 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı



Şekil C.5 d_i/d_d oranı 0,5 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı



Şekil C.6 d_i/d_d oranı 0,2 iken değişik parametre seviyeleri için BLUE kayıp oranı ve iletim hızı

ÖZGEÇMİŞ

1974 yılında Eskişehir'de doğan Bahri Okuroğlu, 1992 yılında Haydarpaşa Anadolu Meslek Lisesi Elektronik Bölümü'nü bitirerek girdiği İstanbul Teknik Üniversitesi Kontrol ve Bilgisayar Mühendisliği Bölümü'nden 1996 yılında lisans derecesi aldı.

1996 yılından bu yana Netaş – Nortel Networks Araştırma-Geliştirme Bölümü'nde ses santrallerine uzaktan erişim amaçlı iletişim sistemleri geliştirilmesi ve ağ cihazları üzerindeki gerçek zamanlı işletim sistemleri konularında çalışan Bahri Okuroğlu, iletişim protokolleri mimarisi ve uyarlaması, UNIX tabanlı gerçek zamanlı işletim sistemleri ve çeşitli dillerde programlama konularında tecrübe sahibidir.

- [1] **Internet Software Consortium**, 2000, Internet Domain Survey, <http://www.isc.org/ds/host-count-history.html>
- [2] NS: Network Simulator, <http://www-mash.cs.berkeley.edu/ns/>
- [3] REAL Network Simulator, <http://www.cs.cornell.edu/skeshav/real/overview.html>
- [4] VINT (Virtual InterNetwork Testbed) Project, <http://netweb.usc.edu/vint/>
- [5] DARPA, <http://www.darpa.mil/>
- [6] **Stevens, R.**, 1994, TCP/IP Illustrated: the protocols, *Addison-Wesley Publishing Company, Inc.*, Massachusetts
- [7] **Ferguson, P., Huston, G.**, 1998, Quality of Service, Delivering QoS on the Internet and in Corporate Networks, ISBN 0-471-24358-2, *John Wiley & Sons, Inc.*, New York
- [8] **Cisco Documentation**, Quality of Service (QoS) Networking, http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/qos.htm
- [9] **Ferguson, P., Huston, G.**, 1998, Quality of Service in the Internet: Fact, Fiction, or Compromise, <http://www.employees.org:80/~ferguson/inet-qos.ps>
- [10] **Floyd, S., Jacobson, V.**, 1995, Link-sharing and Resource Management Models for Packet Networks, *IEEE/ACM Transactions on Networking*, Vol. 3 No 4, <http://www.aciri.org/floyd/papers/link.pdf>
- [11] **Keshhav, S.**, 1997, An Engineering Approach to Computer Networking, pp 234-236, *Addison-Wesley*
- [12] **IETF**, Integrated Services working group, <http://www.ietf.org/html.charters/intserv-charter.html>
- [13] **Braden, R., Clark, D., Shenker, S.**, 1994, Integrated Services in the Internet Architecture: an Overview, Request for Comments: 1633
- [14] **Shenker, S., Partridge, C., Guerin, R.**, 1997, Specification of Guaranteed Quality of Service, Request for Comments: 2212

- [15] **Shenker, S., Partridge, C., Guerin, R.**, 1997, Specification of the Controlled-Load Network Element Service, Request for Comments: 2211
- [16] **IETF**, Integrated Services over Specific Link Layers (issll) working group, <http://www.ietf.org/html.charters/issll-charter.html>
- [17] **Garrett, M., Borden, M.**, 1998, Interoperation of Controlled-Load Service and Guaranteed Service with ATM, Request for Comments 2381
- [18] **Crawley, E., Berger, L., Berson, S., Baker, F., Borden, M., Kawczyk, J.**, 1998, A Framework for Integrated Services and RSVP over ATM, Request for Comments 2382
- [19] **Bernet, Y., Binder, J., Blake, S., Carlson, M., Carpenter, B., Keshav, S., Davies, E., Ohlman, B., Verma, D., Wang, Z., Weiss, W.**, 1999, A Framework for Differentiated Services, Draft-ietf-diffserv-framework-02.txt,
- [20] **Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.**, 1998, An Architecture for Differentiated Services, Request for Comments: 2475
- [21] **Information Sciences Institute University of Southern California**, 1981, INTERNET PROTOCOL, Request for Comments: 791
- [22] **Deering, S., Hinden, R.**, 1998, Internet Protocol, Version 6 (IPv6) Specification, Request for Comments: 2460
- [23] **Bless, B., Wehrle, K.**, 1999, A Lower Than Best-Effort Per-Hop Behaviour, Internet Draft: draft-bless-diffserv-lbe-phb-00.txt
- [24] **Diederich, J., Zitterbart, M.**, 1999, An Expedited Forwarding with Dropping PHB, Internet Draft: draft-dieder-diffserv-phb-efd-00.txt
- [25] **Kilikki, K., Ruutu, J.**, 1999, Interoperability PHB Group, Internet Draft: draft-kilikki-diffserv-interoperability-00.txt
- [26] **Caret, S., Hodgkinson, T., O'Neill, A., Mortimore, D.**, 1998, A Bounded-Delay service for the internet, Internet Draft: draft-carter-bounded-delay-00.txt

- [27] **Heinanen, J., Baker, F., Weiss, W., Wroclawski, J.**, 1999, Assured Forwarding PHB Group, Request for Comments: 2597
- [28] **Jacobson, V., Nichols, K., Poduri, K.**, 1999, An Expedited Forwarding PHB, Request for Comments: 2598
- [29] **Nichols, K., Blake, S., Baker, F., Black, D.**, 1998, Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, Request for Comments: 2474
- [30] **Nichols, K., Jacobson, V., Zhang, L.**, 1999, A Two-bit Differentiated Services Architecture for the Internet, Internet Draft: draft-nichols-diff-svc-arch-02
- [31] **Stoica, I., ve diğerleri**, 1999, Per Hop Behaviours Based on Dynamic Packet States, Internet Draft: draft-stoica-diffserv-dps-00.txt
- [32] **Stoica, I., Zhang, H.**, 1999, Providing Guaranteed Services Without Per Flow Management, *Proceedings of ACM SIGCOMM'99*, BOSTON, MA, pp. 81-94, <http://redriver.cmcl.cs.cmu.edu/~hzhang-ftp/SIGCOM99.pdf>
- [33] **Köhler, S., Schafer, U.**, 1999, Performance Comparison of Different Class-and-Drop Treatment of Data and Acknowledgements in DiffServ IP Networks, *University of Würzburg Institute of Computer Science Research Report Series, Report No 237*, <http://www-info3.informatik.uni-wuerzburg.de/TR/tr237.ps.gz>
- [34] **Kim, H., Leland, W., Thomson, S.**, Evaluation of Bandwidth Assurance Service using RED for Internet Service Differentiation, *Bell Communications Research*, <ftp://ftp.bellcore.com/pub/world/hkim/assured.ps.Z> veya <ftp://ftp.telcordia.com/pub/world/hkim/assured.ps.Z>
- [35] **Braden, R., Zhang, L., Berson, S., Herzog, S., Jamin, S.**, 1997, Resource ReSerVation Protokol (RSVP) – Verison 1 Functional Specification, Request for Comments: 2205
- [36] **Nagle, J.**, 1984, Congestion Control in IP/TCP Internetworks, Request For Comments: 896

- [37] **Jacobson, V.**, 1988, Congestion Avoidance and Control, Proceedings of SIGCOMM '88, Stanford, CA, ACM, <http://www-nrg.ee.lbl.gov/papers/congavoid.pdf>
- [38] **Braden, R.**, 1989, Requirements for Internet Hosts -- Communication Layers, Request For Comments: 1122
- [39] **Stevens, W.**, 1997, TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms, Request For Comments: 2001
- [40] **Jain, R.**, 1990, Congestion Control in Computer Networks: Trends and Issues, IEEE Network, pp. 24-30., http://www.cis.ohio-state.edu/~jain/papers/cong_tre.htm
- [41] **Mankin, A., Ramakrishnan, K.**, 1991, Gateway Congestion Control Survey, Request For Comments: 1254
- [42] **Yang, C., Reddy, A.**, 1995, A taxonomy for congestion control algorithms in packet switching networks., *IEEE Network Magazine*, Vol 9, Number 5, <http://www.comsoc.org/pubs/surveys/yang/yang-orig.html>
- [43] **Braden, B. ve diğerleri**, 1998, Recommendations on Queue Management and Congestion Avoidance in the Internet, Request For Comments: 2309
- [44] **Floyd, S.**, 2000, Congestion Control Principles, Internet Draft: draft-ietf-floyd-cong-03.txt
- [45] **Floyd, S., Fall, K.**, 1999, **Promoting the Use of End-to-End Congestion Control in the Internet**, *IEEE/ACM Transactions on Networking*, <http://www.aciri.org/floyd/papers/collapse.may99.pdf>
- [46] **Defense Advanced Research Projects Agency Information Processing Techniques Office**, 1981, TRANSMISSION CONTROL PROTOCOL, Request For Comments: 793
- [47] **Jacobson, V.**, 1990, Modified TCP Congestion Control Algorithm, end2end-interest mailing list, <ftp://ftp.ee.lbl.gov/email/vanj.90apr30.txt>

- [48] **Allman, M., Paxson, V., Stevens, W.**, 1999, TCP Congestion Control, Request For Comments: 2581
- [49] **Floyd, S., Jacobson, V.**, 1993, Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Transactions on Networking*, <http://www.aciri.org/floyd/papers/early.pdf>
- [50] **Postel, J.**, 1981, INTERNET CONTROL MESSAGE PROTOCOL, Request For Comments: 792, September 1981
- [51] **Prue, W., Postel, J.**, 1987, Something a Host Could Do with Source Quench: The Source Quench Introduced Delay (SQID), Request For Comments: 1016
- [52] **Baker, F.**, 1995, Requirements for IP Version 4 Routers, Request For Comments: 1812
- [53] **Salim**, 1998, Backward ECN for the Internet Protocol, Internet Draft: draft-salim-jhsbnns-ecn-00.txt
- [54] **Ramakrishnan, K.K., and Jain, R.**, 1990, A binary feedback scheme for congestion avoidance in computer networks, *ACM Transactions on Computer Systems*, V.8, N.2, pp. 152-181
- [55] **Rosolen, V., Bonaventure, O., and Leduc, G.**, 1999, A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic, *ACM Computer Communication Review*, and Impact of cell discard strategies on TCP/IP in ATM UBR networks, *Proc. of the 6th Workshop on Performance Modelling and Evaluation of ATM Networks (IFIP ATM'98)* Ilkley, UK, July 98.
- [56] **Floyd, S.**, 2000, Recommendation on using the "gentle_" variant of RED, <http://www.aciri.org/floyd/red/gentle.html>
- [57] **Labrador, M., Banerjee, S.**, 1999, Packet Dropping Policies for ATM and IP Networks, *IEEE Communications Surveys*, vol 2, no 3, <http://www.comsoc.org/pubs/surveys>

- [58] **May, M., Bonald, T., Bolot, J.**, 2000, Analytic Evaluation of RED Performance, INRIA, Sophia-Antipolis, France, <http://www-sop.inria.fr/rodeo/personnel/mmay/papers/infocom00.ps>
- [59] **Feng, W., Kandlur, D., Shin, K.**, 1999, A Self-configuring RED Gateway, INFOCOM '99, <http://www.eecs.umich.edu/~wuchang/work/CSE-TR-349-97.ps.Z>
- [60] **Feng, W.**, 1996, Internet e-mail, http://www.eecs.umich.edu/~wuchang/red/misc/dynamic_red.txt
- [61] **Floyd, S.**, 1997, RED: Discussions of Setting Parameters, email message, <http://www.aciri.org/floyd/REDparameters.txt>
- [62] **Jacobson, V.**, 1998, Notes on using RED for Queue Management and Congestion Avoidance, *NANOG 13*, Dearborn, MI
- [63] **Doran, S.**, 1998, EBONE Interface Graphs, <http://adm.ebone.net/~smd/red-1.html>
- [64] **May, M., Bolot, J., Diot, C., Lyles, B.**, 1999, Reasons not to deploy RED, INRIA Sophia-Antipolis, ENSIM, Sprintlabs, <http://www-sop.inria.fr/rodeo/personnel/mmay/may-red.html> <http://www-sop.inria.fr/rodeo/personnel/mmay/papers/slides-iwqos-sprint.ps>
- [65] **Jacobson, V., Nichols, K., Poduri, K.**, 1999, RED in a Different Light Draft Version on June 1, 1999, http://www.cisco.com/public/cons/red_light.pdf veya http://www.cs.ucla.edu/classes/fall99/cs219/papers/red_light.pdf.gz
- [66] **Anjum, F., Tassiulas, L.**, 1999, Balanced-RED: An Algorithm to Achieve Fairness in the Internet, *Center for Satellite and Hybrid Communication Networks Technical Research Report*
- [67] **Feng, W., Kandlur, D., Saha, D., Shin, K.**, 2000, BLUE: A new class of active queue management, <http://www.eecs.umich.edu/~wuchang/blue/>
- [68] **Ramakrishnan, K., Floyd, S.**, 1999, A Proposal to add Explicit Congestion Notification (ECN) to IP, Request for Comments: 2481

- [69] **Floyd, S.**, 1994, TCP and Explicit Congestion Notification, *ACM Computer Communication Review*, V. 24 N. 5, p. 10-23, ftp://ftp.ee.lbl.gov/papers/tcp_ecn.4.ps.Z
- [70] **Bernet, Y., Smith, A., Blake, S., Grossman, D.**, 2000, A Conceptual Model for Diffserv Routers, Internet Draft: draft-ietf-diffserv-model-03.txt
- [71] **Heinanen, J., Guerin, R.**, 1999, A Single Rate Three Color Marker, Request for Comments: 2697
- [72] **Heinanen, J., Guerin, R.**, 1999, A Two Rate Three Color Marker, Request for Comments: 2698
- [73] **Clark, D., Fang, W.**, Explicit Allocation of Best Effort Packet Delivery Service, <http://diffserv.lcs.mit.edu/Papers/exp-alloc-ddc-wf.pdf>
- [74] **Ibanez, J., Nichols, K.**, 1998, Preliminary Simulation Evaluation of an Assured Service, Internet Draft: draft-ibanez-diffserv-assured-eval-00.txt, <http://ec.eurecom.fr/~ibanez/draft-ibanez-diffserv-assured-eval-00.pdf>
- [75] **Feng, W.**, Email on DiffServ WG, 1998, <http://www-nrg.ee.lbl.gov/diff-serv-arch/msg02184.html>
- [76] **Heinanen, J.**, Email on DiffServ WG, 1998, <http://www-nrg.ee.lbl.gov/diff-serv-arch/msg02176.html>
- [77] **Medina, O.**, Email on DiffServ WG, 1998, <http://www-nrg.ee.lbl.gov/diff-serv-arch/msg02174.html>
- [78] **Yeom, I., Reddy, N.**, 1999, "Realizing throughput guarantees in a differentiated services network," *Proceedings of ICMCS*, <http://dropzone.tamu.edu/~ikjun/diff-serv.ps>
- [79] **Feng, W.**, BLUE code for NS1, <http://www.eecs.umich.edu/~wuchang/blue/ns-blue.tgz>
- [80] **Murphy, S.**, Diffserv additions to ns-2, <http://www.teltec.dcu.ie/~murphys/ns-work/diffserv/index.html>

[81] **Scott, J., Wang, J.**, RIO code for ns2.1b6, <http://www.aciri.org/ns/rio-ns2-1b6.tar.gz>

[82] **Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.**, 1996, RTP: A Transport Protocol for Real-Time Applications, Request for Comments: 1889