E. Ü. FEN BİLİMLERİ ENSTİTÜSÜ

EGE ÜNİVERSİTESİ

YÜKSEK LİSANS TEZİ

GÖREVDEŞ AĞLARDA GECİKME METRİĞİ TABANLI HİYERARŞİK KATMAN GELİŞTİRME ÜZERİNE BİR ÇALIŞMA

Sercan DEMİRCİ

Tez Danışmanı: Prof. Dr. Turhan TUNALI

Uluslararası Bilgisayar Anabilim Dalı

Bilim Dalı Kodu: 619.02.04
Sunuş Tarihi: 13.08.2010

Bornova-İZMİR
2010

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

(YÜKSEK LİSANS TEZİ)

GÖREVDEŞ AĞLARDA GECİKME METRİĞİ TABANLI HİYERARŞİK KATMAN GELİŞTİRME ÜZERİNE BİR ÇALIŞMA

Sercan DEMİRCİ

Tez Danışmanı: Prof. Dr. Turhan Tunalı

Uluslararası Bilgisayar Anabilim Dalı

Bilim Dalı Kodu: 619.02.04
Sunuş Tarihi: 13.08.2010

Bornova-İZMİR
2010

Sercan DEMİRCİ tarafından YÜKSEK LİSANS tezi olarak sunulan "Görevdeş Ağlarda Gecikme Metriği Tabanlı Hiyerarşik Katman Geliştirme Üzerine Bir Çalışma" başlıklı bu çalışma E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliği ile E.Ü. Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi'nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve 13/08/2010 tarihinde yapılan tez savunma sınavında aday oybirliği/oyçokluğu ile başarılı bulunmuştur.

Jüri Üyeleri:		İmza
Jüri Başkanı	: Prof. Dr. E. Turhan TUNALI	.....
Raportör Üye	: Yrd. Doç. Dr. Hasan BULUT	.....
Üye	: Prof. Dr. M. Emin DALKILIÇ	.....

ÖZET

GÖREVDEŞ AĞLARDA GECİKME METRİĞİ TABANLI HİYERARŞİK KATMAN GELİŞTİRME ÜZERİNE BİR ÇALIŞMA

DEMİRCİ, Sercan

Yüksek Lisans Tezi, Uluslararası Bilgisayar Enstitüsü
Tez Yöneticisi: Prof. Dr. Turhan TUNALI
Ağustos 2010, 95 sayfa

Günümüzde Internet üzerinden yapılan ve herhangi bir merkezi yapıya veya hiyerarşik kontrole ihtiyaç duymayan işlemler ve iletişim ortamları, klasik dağıtık sistemlere göre büyük ölçüde daha karmaşık hale gelmişlerdir. Bu nedenle gelişmekte olan görevdeş ağlara olan ilgi artmıştır ve görevdeş ağlar üzerinden yüksek kalitede video akışlandırma yapmak olanaklı hale gelmiştir.

Bu çalışmada, üstüne bindirmeli ağ yapıları (overlay networks) incelenmiş ve görevdeş ağlar için yeni bir mimari geliştirilmiştir. Önerilen metot, sistemin kolay yönetilebilir olması için eşlerin hiyerarşik bir yapı içerisinde kümelenmesine ve verimli bir içerik iletimi için bu hiyerarşik yapı üzerine kurulacak olan çoklu gönderim ağacı temeline dayanmaktadır. Tasarlanan üstüne bindirmeli ağ yapısında, eşlerin kümelenmesi için kullanılan mekanizma, eşlerin birbirleri ile ölçtükleri paket dolaşım süresidir. Bu sayede ağ topolojisi üzerinde birbirine yakın olan eşlerin aynı küme içerisinde bulunması hedeflenmiştir.

Önerilen üstüne bindirmeli ağ yapısı, Internet test ortamı olan PlanetLab üzerinde gerçeklenip kümeleme performansına bakılmıştır. Alınan test sonuçları, önerilen hiyerarşik üstüne bindirmeli ağ yapısının, emsal düz yapılara göre daha iyi bir kümeleme yaptığını göstermiştir.

Anahtar sözcükler: görevdeş ağlar, çoklu gönderim ağacı, üstüne bindirmeli ağ, paket dolaşım süresi.

ABSTRACT

A STUDY ON DEVELOPING DELAY BASED HIERARCHICAL OVERLAY IN PEER-TO-PEER NETWORKS

DEMİRCİ, Sercan

MSc in International Computer Institute
Supervisor: Prof. Dr. Turhan TUNALI
August 2010, 95 pages

In state of the art Internet, computing and communications environments are significantly more complex than classical distributed systems, lacking any centralized organization or hierarchical control. There has been much interest in emerging peer-to-peer networks and over these networks high quality video streaming became feasible.

In this study, methods for constructing overlay networks are examined and a novel system is developed and implemented on PlanetLab peer-to-peer platform. Proposed method organizes peers into a hierarchy of clusters for easy management and efficient content transmission. Round-trip-time (RTT) is the metric used for clustering of peers in the proposed overlay network. By this way, proposed architecture identifies clusters of peers that are near to each other in the network topology.

The overlay network that we proposed is implemented on PlanetLab which is Internet test environment for clustering performance. Test results show that our proposed hierarchical overlay network yields better results than other flat architectures.

Keywords: peer-to-peer networks, multicast tree, overlay network, round-trip-time.

TEŞEKKÜR

Öncelikle bu tez konusu üzerinde bana çalışma imkanı sunan ve ayrıca yüksek lisans eğitimim süresince danışmanım olan Prof. Dr. Turhan TUNALI' ya deneyimleri, bilgileri ve önerileriyle araştırma ve geliştirmeye yönlendirmesi ve sağladığı kaynaklar ile destek olmasından dolayı teşekkürü bir borç bilirim. Ayrıca tez süresince her türlü fikir ve desteklerinden dolayı proje yürütücümüz olan Yrd. Doç. Dr. Hasan BULUT'a, projedeki arkadaşlarım Araştırma Görevlisi ve doktora öğrencisi olan Müge Fesci SAYIT'a, Asil YARDIMCI'ya ve Yağız KAYMAK'a; öte yandan bana yüksek lisans eğitimi yapma imkanı sunan Uluslararası Bilgisayar Enstitüsü'ne, "108M411" numaralı "Görevdeş Ağlarda Ölçeklenebilir Gruplandırma ve Video Akışlandırma" isimli proje kapsamında yüksek lisans çalışmamı maddi olarak destekleyen Türkiye Bilimsel ve Teknolojik Araştırma Kurumu'na (TÜBİTAK) ve tez çalışması boyunca yanımda olan aileme teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

İÇİNDEKİLER (devam)

İÇİNDEKİLER (devam)

ŞEKİLLER DİZİNİ

Şekil Sayfa

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

ÇİZELGELER DİZİNİ

Çizelge Sayfa

KISALTMALAR

Kısaltmalar	Açıklama
CDF	Cumulative Distribution Function
DHT	Distributed Hash Tables
HTTP	Hypertext Transfer Protocol
P2P	Peer-to-Peer
RON	Resilient Overlay Network
RP	Rendezvous Point
RTT	Round-Trip-Time
SON	Semantic Overlay Network
TTL	Time-To-Live
UDP	User Datagram Protocol

1. GİRİŞ

Son yıllarda iletişim teknolojilerinin maddi boyutunun azalması ve bu iletişim teknolojilerinden özellikle Internet kullanımının yaygınlaşması, teknoloji kullanıcılarının daha performans gerektiren işlemleri daha hızlı bir şekilde gerçekleştirmek istemelerine sebep olmuştur. Kullanıcıların Internet dünyasındaki ihtiyaçlarının artışı, hizmet veren sistemler tarafından karşılanmak zorundadır. Ayrıca hizmeti alan kullanıcı da aldığı hizmetin kesintiye uğramadan devam etmesini istemektedir. Bu da merkezi sistemler tarafından kontrol edilen kullanıcılar arasındaki veri iletişimini sıkıntıya sokabilmektedir. Kullanıcıların merkezi sistemlere ihtiyaç duymadan performans gerektiren dosya paylaşımı veya medya akışlandırması gibi işlemleri verimli şekilde yapmak istemeleri günümüzde görevdeş (Peer-To-Peer, P2P) ağlara olan ilgiyi arttırmıştır. P2P ağların bugün geldiği nokta, merkezi sistemlerde yaşanılan darboğaz sorununun aşılmasına yardımcı olmuştur. P2P ağların, sistem üzerindeki tüm kaynakları bir araya topladığını ve bu bilgileri paylaşanları da ağ içinde bağımsız kullanıcılar haline getirdiğini söyleyebiliriz.

Tez konusu kapsamında öncelikli olarak, ağdaki gereksiz mesajlaşma trafiğini azaltmak ve performans artışı sağlamak amacıyla gecikme bakımından yakın olan eşlerin aynı kümelerde toplanması sağlanarak kümeleme yapılmıştır. Kümeleme yaparken ve bu kümeler ile hiyerarşik ağaç yapısı oluşturulurken, paket dolaşım süresinden (Round-Trip-Time, RTT) yararlanılmıştır. Kümeleme işlemi sırasında, birbirleri arasındaki RTT değerleri belirli bir eşik değerinin altında kalan eşler aynı kümelerde toplanırlar. Bu sayede RTT değerleri birbirine yakın olan eşlerin, oluşturulan hiyerarşik üst katmanda birbirlerine erişimleri sağlanmış olur. Ayrıca ağ üzerinde paket yönlendirme ve akışlandırma işlemini daha verimli yapmak için oluşturulan bu kümeler, hiyerarşik bir biçimde yerleştirilerek ağaç yapısına sahip bir üst katman oluşturulmuştur. Bu sayede her paketin izleyeceği yol kesin olarak belirlenerek gereksiz ağ trafiğinin önüne geçilmiş olur.

Tez kapsamında hedeflenen, görevdeş ağlar üzerinde video akışlandırması olduğundan, öncelikli olarak sisteme dahil olan eşlerin doğru bir şekilde,

kendilerine RTT değeri bakımından en yakın olan kümelere girmelerini en kısa sürede, az mesajlaşma yaparak ve en önemlisi de kısıtlı bilgiye sahip olarak sağlamaktır. Bunu sağlayabilmek için oluşturulan hiyerarşik katmanlı ağaç yapısı üzerinde çalışan RTT değerini baz alan dağıtık bir algoritma önerilmiştir. Önerdiğimiz dağıtık hiyerarşik üstüne bindirmeli ağ yapısının performans kriterleri olan kümelerin çapı yani küme içerisinde birbirine en uzak olan iki eş arasındaki RTT değeri, küme içerisinde küme liderine en uzak olan düğümlerin RTT değerleri, küme içerisindeki düğümlerin birbirleriyle olan gecikme değerleri, kümeler arasındaki RTT değerleri, kümelerin doluluk oranı, eşlerin kümelere ne kadar sürede dahil oldukları, kümeleme hassasiyeti ve kümeleme doğruluğu sonuçları önerilen algoritmanın iyi bir kümeleme yaptığını ortaya çıkarmıştır.

Tezin bundan sonraki bölümleri şu şekilde düzenlenmiştir. İkinci bölümde, konu hakkındaki literatür çalışmalarına değinilecektir. Üçüncü bölümde ise geliştirilen ve çalıştırılan sistemin tasarım notları açıklanacaktır. Daha sonraki bölümde düzenlenen algoritmanın sonuçları, referans algoritma ile TÜBİTAK'ın "108M411" numaralı ve "Görevdeş Ağlarda Ölçeklenebilir Gruplandırma ve Video Akışlandırma" isimli proje dahilinde kullanılan PlanetLab Internet test ortamında kümeleme hassasiyeti ve kümeleme doğruluğu karşılaştırılacaktır ve sonuç bölümünde ise çalışma sonunda elde edilen veriler yorumlanacaktır.

2. LİTERATÜR ÖZETİ

Bu bölümde öncelikli olarak görevdeş ağlar ile ilgili temel kavramlar ve bu kavramların özellikleri açıklanacaktır. Daha sonra görevdeş ağların çeşitleri ile çalışma prensipleri anlatılacaktır. Son olarak literatürde bulunan, görevdeş ağlar üzerine medya akışlandırması için tasarlanmış olan üstüne bindirmeli ağlardan ve bu ağların eksik yanlarından bahsedilecektir.

2.1 İlgili Kavramlar

Görevdeş ağlar üzerine kurulmuş olan medya akışlandırması için tasarlanan üstüne bindirmeli ağ sistemlerinin anlaşılabilmesi için, öncelikle görevdeş ağların tanımına, özelliklerine, gelişim sürecine ve sınıflandırılmasına bakılacaktır. Öte yandan tezde kullanılacak olan video akışlandırma yolu için oluşturulacak olan çoklu gönderim (multicast) ağaçlarının uygulanması için oluşturulan üstüne bindirmeli mimariye değinilecektir. Son olarak literatürde bulunan görevdeş ağlar üzerinde kullanılan üstüne bindirmeli ağ yapıları açıklanacaktır.

2.2 Görevdeş Ağların Tanımı ve Özellikleri

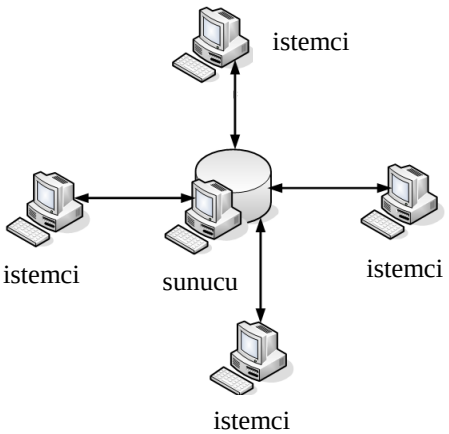
Görevdeş ağları, herhangi bir merkezi otoritenin yardımına gerek duyulmadan eşlerin kaynak paylaşmak için birbirleriyle iletişim kurması olarak tanımlayabiliriz. Eşler arasındaki kaynak paylaşımı dağıtık protokoller ile yapılmaktadır. Dünya üzerinde çok fazla sayıda eş bulunduğu için bu dağıtık protokollerin verimli şekilde çalışabilmesi için en önemli etkenlerden biri ölçeklenebilirliktir.

Görevdeş ağlar diğer Internet uygulamalarından farklı olarak merkezi sunucular yerine çok fazla sayıda olan uç kullanıcı (end user) ile veri paylaşmaya meğillidirler. Ayrıca görevdeş ağlar, istemci-sunucu modellerinde yaşanabilecek olan darboğaz sorununu çözebilmek için karmaşık algoritmalara ihtiyaç duyarlar (Pourebrahimi et al., 2005).

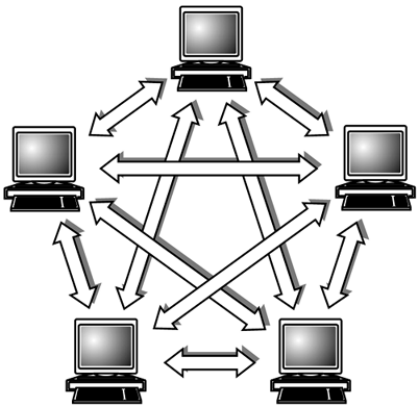
Dağıtık mimari uygulamaları geliştirilirken çoğunlukla istemci-sunucu modelleri göz önüne alınmaktadır. Oysa ki dağıtık mimari uygulamaları için P2P modeli de söz konusudur. P2P modelinde istemci ve sunucu arasında fark yoktur ve alt yapı hazırlıkları diğer modellere göre biraz daha karmaşıktır. Programlama zorluğu nedeni ile geliştiriciler zaman zaman bu modelden kaçınırlar. Oysa ki günümüzde P2P üzerine kurulu olan pek çok sistem yer almaktadır. P2P uygulamalarına örnek olarak, anlık mesajlaşma (instant messaging), dosya paylaşımı (file sharing), oyunlar, görsel ve sesli veri aktarımı ve veri kopyalama (data replication) gösterilebilir.

Görevdeş ağların önemli karakteristiklerini özetlemek gerekirse; kendi kendilerine organize olabilirler (self-organized) ve ortamın dinamik olarak devamlı değişmesinden dolayı uyarlanabilirdirler (adaptive). Eşler serbestçe sisteme katılabilir ve sistemden ayrılabilirler. Dağıtık bir yapıya sahip olmaları ve merkezi bir sunucuya ihtiyaç duymamalarından dolayı hatalara karşı toleranslı (fault tolerant) ve yük bakımından dengelidirler (load-balanced) (Li and Wu, 2005).

Şekil 2.1'de örnek bir istemci-sunucu modeli, Şekil 2.2'de ise örnek bir görevdeş ağ modeli gösterilmiştir. İstemci-sunucu modelinde tüm istemciler hizmet almak için sunucuya bağlıdırlar. Görevdeş ağ modelinde ise eşler hizmet almak için merkezi bir sunucuya ihtiyaç duymazlar. Bunun yerine birbirlerine hizmet sunarlar.



Şekil 2.1. Örnek bir istemci-sunucu modeli.



Şekil 2.2. Örnek bir görevdeş ağ modeli.

Görevdeş ağların, istemci-sunucu gibi geleneksel ağlara nazaran pek çok güçlü yanı vardır. Görevdeş ağlar, dağıtık olmalarından dolayı hatalara karşı dayanıklıdır. Bu ağlarda merkezi sunucunun bulunmaması tek nokta hatasını (single point of failure) ortadan kaldırır. Görevdeş ağlardaki eş sayısı birkaç yüz olabileceği gibi milyonlara da ulaşabilmektedir. Bu da görevdeş ağların ne kadar esnek bir yapıya sahip olabileceğini gösterir. Görevdeş ağların, dezavantajlarını ise şöyle sıralayabiliriz. Merkezi bir sunucunun olmayışı ağdaki eşlerin organizasyonunu zorlaştırır. Çok sayıda eşin bulunduğu görevdeş ağlarda kaynakların tespit edilmesinin gittikçe zorlaşması ve yönetimin zorluğu ağın ölçeklenebilirliğini sıkıntıya sokar.

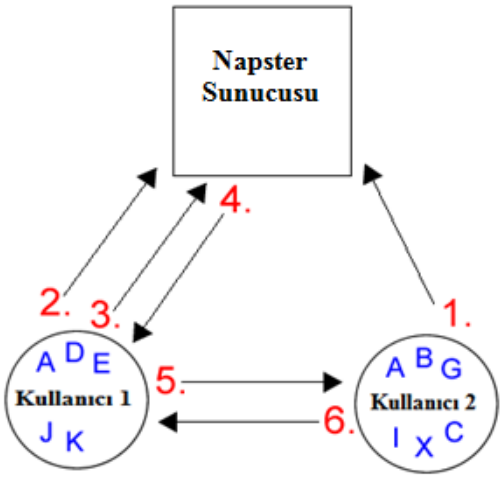
2.3 Görevdeş Ağların Gelişim Süreci

Tezin bu kısmında görevdeş ağların tarih içinde geçirdiği değişime ve çalışma prensiplerine değinilecektir. Sırasıyla birinci nesil görevdeş ağlar, ikinci nesil görevdeş ağlar, üçüncü nesil görevdeş ağlar ve son nesil görevdeş ağlar anlatılacaktır.

2.3.1 Birinci nesil görevdeş ağlar

Günümüzde P2P protokolünün popüler bir uygulama olmasının temelleri, 1999 yılında Shawn Fanning tarafından ortaya çıkartılan "Napster" sistemine dayanmaktadır (Lua et al., 2005). Öncelikle amacı müzik dosyalarının rahat paylaşılması olan uygulamanın çalışma mantığı gayet basittir. Sistem, ağdaki eşlerin sahip oldukları müzik dosyalarının indekslendiği merkezi bir sunucuya sahiptir. Arama yapan bir kullanıcının isteği, protokole ait kontrol paketleri kullanılarak merkezi sunucuya ulaşır, sunucu aranan dosya ismini, tuttuğu listede arar. Eğer dosyayı paylaşıma açmış bir kullanıcı var ise, listeden elde edilen sonuca göre bu kullanıcının IP adresi, istemciye yine kontrol paketleri kullanılarak iletilir. Bir sonraki aşamada, istemci IP adresini bildiği kullanıcıdan istediği dosyayı veri paketleri kullanarak indirmeye başlar. Napster, yapısal olmayan (unstructured) fakat merkezi görevdeş ağ kategorisinde bulunur. Napster'ın çalışma mekanizması Şekil 2.3'te gösterilmiştir (Wang and Li, 2003). Şekilde, 2 numaralı kullanıcı Napster sunucusuna bağlanır ve paylaşacağı

dosyaların listesini sunucuya gönderir. 1 numaralı kullanıcı aynı şekilde Napster sunucusuna bağlanır ve paylaşacağı dosyaların listesini sunucuya gönderir. Daha sonra 1 numaralı kullanıcı, Napster sunucusunda B dosyasını arar. Napster sunucusu, B dosyasını içeren kullanıcıların bilgisini geri döndürür. 1 numaralı kullanıcı, B dosyasını içeren 2 numaralı kullanıcıdan B dosyasını talep eder. Son olarak 2 numaralı kullanıcı, B dosyasını 1 numaralı kullanıcıya gönderir.

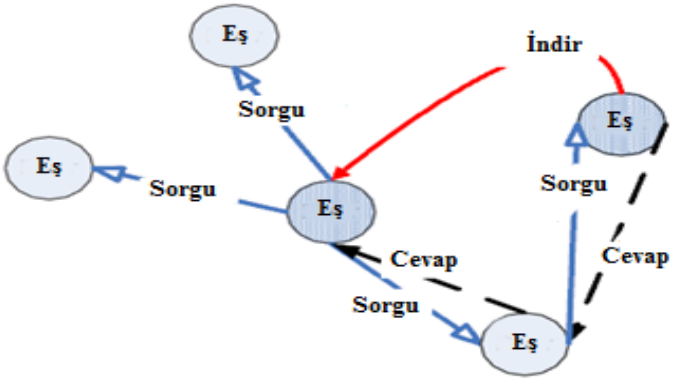


Şekil 2.3. Napster müzik paylaşım servisinin çalışma mekanizması.

2.3.2 İkinci nesil görevdeş ağlar

İkinci nesil P2P protokolü olarak anılan "Gnutella" temel olarak, dosya paylaşımı için kullanılan açık kaynak kodlu grup üyeliği ve arama protokollerini kullanan ilk merkezi olmayan dosya paylaşım ağıdır (Ripeanu et al., 2002). Görevdeş ağlar arasındaki kategorilerde, yapısal olmayan ve tamamen merkezi olmayan (decentralized) bir görevdeş ağ olarak sınıflandırılabilir. Gnutella'nın tasarım amaçları arasında dinamik bir ortamda çalışabilme yeteneği, performans, ölçeklenebilirlik, güvenilirlik ve anonimlik yer almaktadır (Ripeanu et al., 2002). Dosya araması yapan istemcinin sorgusu, ilk olarak doğrudan bağlı olduğu sunuculara aktarılır. Sorguyu alan sunucu, kendisinin ve kendisine doğrudan bağlı diğer kullanıcıların paylaşıma açtığı dosya listesinde arama yapar. Bir sonuca ulaşıldığında, dosya sahibinin IP adresi istemciye gönderilir. Eğer arama başarısız olursa sorgu, sunucu tarafından kendisine doğrudan bağlı olan tüm kullanıcılara aktarılır. Daha önce yönlendiricilerde (router) kullanılan ve dedikodu (gossipping) olarak anılan bu teknik sayesinde sorgu, ağ içinde kademe kademe ilerleyebilir. Sorgunun bir sunucudan kendisine doğrudan bağlı sunuculara aktarılma işlemi

sırasında istemci tarafından belirlenen yaşam süresi (time-to-live, TTL) değeri bir kademe düşürülür. Bu değer sıfıra ulaştığında sorgu sona erdirilerek sonsuz döngünün önüne geçilir. Arama sırasında her kullanıcının hem istemci hem de sunucu şeklinde davranıyor olması P2P iletişiminin geleneksel sunucu-istemci iletişiminden en önemli farkıdır. Napster'da bu fark sadece veri paketleri iletişiminde ortaya çıksada Gnutella ile birlikte kontrol paketleri iletişimi de tüm kullanıcılar arasında yapılmaya başlanır. Gnutella, trafiğin belirlenmesi ve engellenmesi konusunda Napster'a göre çok başarılı olsa da, sorgu sürecinin verimsizliği sebebiyle başarısız olmuştur (Ripeanu et al., 2002). Gnutella'nın çalışma mekanizması Şekil 2.4'te gösterilmiştir.

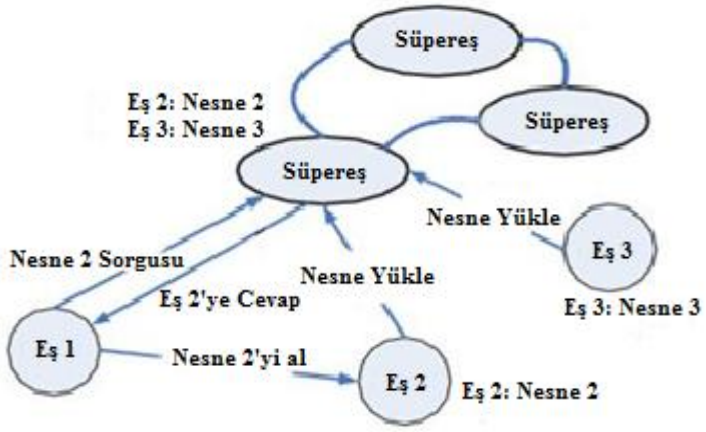


Şekil 2.4. Gnutella servisinin çalışma mekanizması (Lua et al., 2005 – Figure 8).

2.3.3 Üçüncü nesil görevdeş ağlar

Üçüncü nesil P2P protokolü olan "FastTrack" yapısal olmayan karma (hybrid) bir görevdeş ağ protokolüdür (Liang et al., 2006). Arama yapmayı verimli hale getirme amacıyla ağda paylaşılan dosyaları indeksleme amaçlı çalışan ve sadece dosya sağlayan birçok sunucunun hizmet vermesi temeline dayanmaktadır. FastTrack ölçeklenebilirliği arttırmak için süpereş (super-peer) adı verilen eşleri kullanır. Süpereşler, sistemdeki sıradan eşlerden farklı olarak daha fazla göreve sahiptir. Süpereşler bant genişliği, depolama alanı ve hesaplama yeteneği bakımından diğer eşlerden daha fazla kapasiteye sahip olduğu gibi arama işlemlerinin gerçekleştirilmesi için üstveri (metadata) depolamaya da gönüllüdürler (Lua et al., 2005). Kullanıcı, bağlanacağı süpereşi seçerek

sorgusunu başlatır. Sorguyu alan sunucu kendisine bağlı kullanıcılarda dosya var ise hemen cevabı istemciye döndürür, aksi durumda sadece diğer süpereşlerle iletişime geçip sorguyu onlara yönlendirir (Liang et al., 2006). FastTrack'in yapısı ve arama mekanizması Şekil 2.5'te gösterilmiştir.

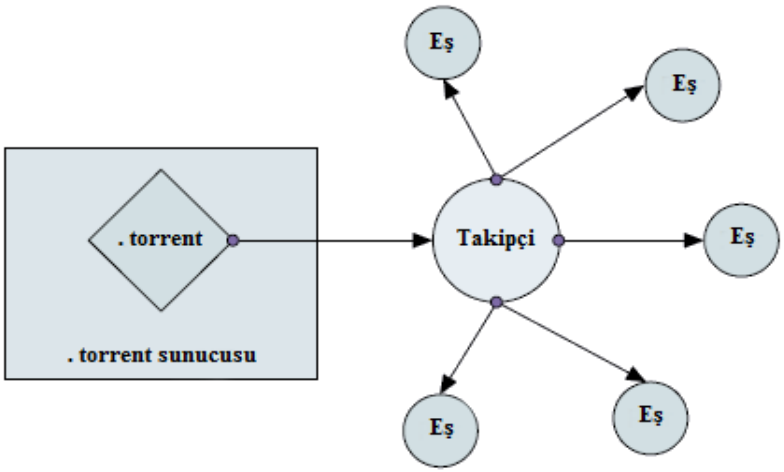


Şekil 2.5. FastTrack yapısı ve arama mekanizması (Lua et al., 2005 – Figure 9).

2.3.4 Son nesil görevdeş ağlar

Son nesil P2P protokolü olan "BitTorrent" merkezi bir görevdeş ağ protokolüdür (Lua et al., 2005). Diğer P2P sistemlerinden çok farklı bir yöntem izlemektedir. Bu sistemi diğer indirme yöntemlerinden ayıran en önemli fark, dosyayı çekme hızını dosyaları çeken kişilerin ve dosyanın popülaritesinin yönlendiriyor olmasıdır. Yani bir dosya ne kadar popülerse o kadar çok kişi o dosyayı çeker, ne kadar çok kişi o dosyayı çekiyorsa o dosyanın eş sayısı o kadar artar ve ne kadar çok eş varsa dosyayı çekme hızınız o derece artar. Normalde Internet üzerinden bir dosya indirmek istediğinizde bilgisayarınız dosyanın bulunduğu sunucuya bir istek yollar. Dosya sunucuda belirli bir yerdedir ve sunucu bilgisayarınıza o dosyanın adresini gönderir. Daha sonra da dosyayı indirmeye başlarsınız. BitTorrent sisteminde ise bir dosya indirmek istediğinizde öncelikle indirmek istediğiniz dosya ile ilgili bilgilerin bulunduğu *.torrent* dosyasını bir sunucudan indirirsiniz. Bu dosyada indirebileceğiniz dosyaların isimleri, boyutları, nereden indirilecekleri gibi bilgiler yer alır. Bu sistemde indireceğiniz dosya tek bir yerde değildir. Bir dosyayı aynı anda birçok kişiden

indirirsiniz. Aslında BitTorrent sisteminin çalışma prensibi çok basittir. İlk olarak belirlenen ana sunucu paylaşılacak dosyayı küçük boyutlu parçalara (fragments) ayırır. Bu parçalar daha sonra takipçi (tracker) dosyasını çeken bütün kullanıcılara dağıtılır. Takipçi, bir sunucu üzerinde çalışan ve sunucuya yüklenmiş paylaşım dosyaları içindeki bilgileri kullanarak kullanıcılar arasındaki veri alış-verişini kontrol eden ve yönlendiren bir yapıdır. Eşler önce takipçiye bağlanır ve oradan aldığı diğer kullanıcı adresleri ile onlara bağlanıp karşılıklı dosya transferi yaparlar (Lua et al., 2005). BitTorrent'in yapısı Şekil 2.6'da gösterilmiştir.



Şekil 2.6. BitTorrent mimarisi (Lua et al., 2005 – Figure 10).

Günümüzde son nesil P2P uygulamaları arasında "LittleShoot" ve "NextShare" yer almaktadır. LittleShoot, kullanıcıların verilerini dağıtık bir şekilde paylaşmaz sadece dağıtık hesaplama yapar. LittleShoot ile kullanıcı bilgisayarına bir hipermetin aktarma iletişim protokolü (Hypertext Transfer Protocol, HTTP) sunucusu kurulur. Kullanıcıların, paylaşmak istedikleri dosyaları LittleShoot ile kurulan bu HTTP sunucusuna yüklemelerine gerek yoktur. LittleShoot P2P uygulamasının BitTorrent sisteminden farklı yönleri bulunmaktadır. BitTorrent protokolünde herhangi bir güvenlik duvarından geçiş bulunmamaktadır. LittleShoot protokolünde güvenlik duvarından geçiş vardır. BitTorrent, HTTP ile uyumlu çalışamaz. Bu nedenle BitTorrent uygulamalarında geçersiz kaynak dosyaları bulunabilmektedir. LittleShoot protokolü ise HTTP ile doğrudan uyumlu olarak çalışabilmektedir. BitTorrent sistemlerinde takipçi kullanılarak dosya paylaşımı yapmak sıkıntı yaratmaktadır. LittleShoot uygulaması, kullanıcılar arasında veri paylaşımı yapmak için güvenli kullanıcı

veri bloğu iletişim protokolü (User Datagram Protocol, UDP) mekanizmasını kullanmaktadır. LittleShoot, merkezi arama işlemi yapar. Bu sayede düşük gecikme ile kullanıcıların arama sorgularına kısa sürede cevap verilir (LittleShoot, 2010).

NextShare dosya paylaşım protokolü ile görevdeş ağlarda canlı olarak medya akışlandırma teknolojisi gerçeklenmiştir. Bu yeni teknoloji ile tüm kullanıcılar Internet üzerinden diğer milyonlarca Internet kullanıcısına erişerek medya akışı yapabilmektedirler. Protokolde anahtar noktayı bant genişliği kullanımı oluşturmaktadır. NextShare teknolojisi ile tek bir kullanıcının harcadığı bant genişliği miktarına eşit bant genişliği ile milyonlarca kullanıcıya hizmet verilebilmektedir. Yapılan medya akışlandırma işlemi BitTorrent uygulamalarından farklı olarak tamamen dağıtık olarak yapılmaktadır. NextShare teknolojisi ile Internet üzerinden verimli şekilde canlı olarak yayınlar izlenebilmektedir (NextShare, 2010).

2.4 Görevdeş Ağların Sınıflandırılması

Görevdeş ağları temel olarak iki farklı özelliğe göre sınıflandırabiliriz. Birinci özellik, görevdeş ağın merkezi bir yapı kullanıp kullanmamasıdır (Lv et al., 2002). İkincisi ise, görevdeş ağın yapısal olup olmamasıdır. Öncelikle merkezi olan ve merkezi olmayan görevdeş ağları inceleyelim.

2.4.1 Merkezi görevdeş ağlar

Bu tip görevdeş ağlarda merkezi bir sunucu mevcuttur. Merkezi sunucu, sistemde var olan tüm eşlerin ve eşler üzerindeki dosyaların listesini tutar. Merkezi sunucu bulunduran bu gibi ağlarda yönetim basittir. Ancak tüm sorguların gerçekleştiği tek bir merkezi sunucunun olması, ağda darboğaz oluşmasına ya da sunucuda tek nokta hatası meydana gelmesine sebep olabilir. Bu gibi ağların en klasik örneği Napster'dır (Lv et al., 2002).

2.4.2 Merkezi olmayan görevdeş ağlar

Merkezi bir sunucunun olmayışı tek nokta hatası olasılığını ortadan kaldırır. Eşlerin verilerinin listesi, yine eşlerin kendi üzerinde tutularak veriler dağıtık olarak listelenir. Öte yandan merkezi bir yapının olmadığı bu ağlarda tüm işlemler eşlerin ortak çalışmasıyla halledilir. Bu ağlara Gnutella (Lv et al., 2002), Freenet (Clarke et al., 2000), CAN (Ratnasamy et al., 2001) ve Chord (Stoica et al., 2001) örnek olarak verilebilir.

Şimdi de görevdeş ağları yapısal olup olmamasına göre inceleyelim.

2.4.3 Yapısal olmayan görevdeş ağlar

Yapısal olmayan görevdeş ağlarda eşler, verilerin yerini tespit etmek için tüme gönderim (broadcast) ile bütün komşularına mesaj gönderir. Bu tip ağlarda oluşturulan topoloji ve verilerin ağ üzerindeki yeri hakkında herhangi bir sınırlama bulunmamaktadır. Ancak bu ağlardaki eş sayısının çok artması ağdaki mesaj trafiğini fazlasıyla arttırır. Bu da sistemin ölçeklenebilirliği önündeki en büyük engeldir. Yapısal olmayan görevdeş ağların oluşturulmasının ve yönetiminin kolaylığı bu ağları çekici kılmaktadır. Bu ağlara örnek olarak Gnutella (Lv et al., 2002), KaZaA (Good and Krekelberg, 2003) ve FastTrack (Lv et al., 2002) verilebilir.

2.4.4 Yapısal olan görevdeş ağlar

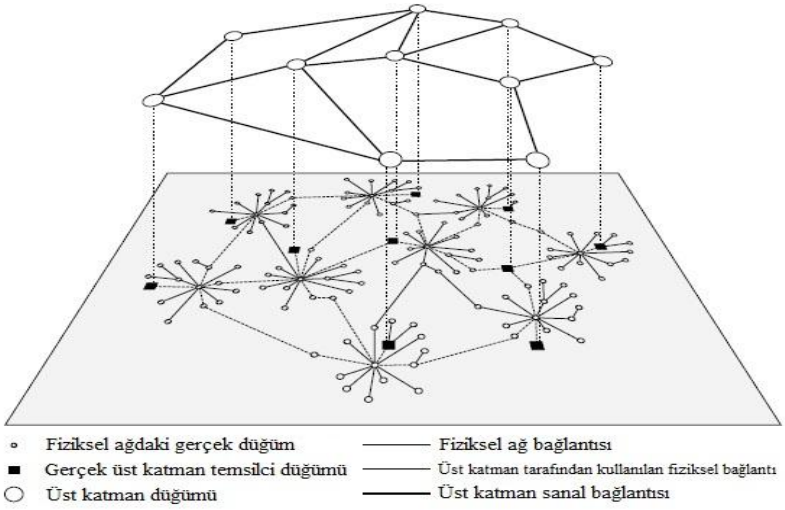
Yapısal (structured) ağlardaki topoloji ve verilerin ağ üzerindeki yerleri kesin kurallara bağlıdır. Dağıtık hesaplama tabloları (distributed hash tables, DHT), bu tip ağlarda sıklıkla kullanılmaktadır. DHT sayesinde ağ üzerindeki verilerin yerlerinin tespiti, yapısal olmayan ağlardaki verilerin yerlerinin tespitine göre çok daha hızlı olmaktadır. Veri tespiti hızlı olmasına rağmen bu ağların oluşturulması ve sürdürülebilirliği (maintenance) maliyetlidir. Bu ağlara CAN (Ratnasamy et al., 2001), Chord (Stoica et al., 2001) ve Tapestry (Zhao et al., 2004) örnek olarak verilebilir.

2.4.5 Karma görevdeş ağlar

Karma yapılarda eşlerin heterojenliğinden (heterogeneity) yararlanılır. Bu tip yapılarda hesaplama gücü, depolama alanı ve bant genişliği gibi özellikleri diğer eşlerden yüksek olan eşler, genellikle süper eş adı verilen yetkin eşler olarak görevlendirilirler. Süper eşler kendilerine bağlı eşler üzerindeki verilerin listesini tutarak aramanın daha verimli gerçekleşmesini sağlar ve merkezi sunucu zorunluluğunu ortadan kaldırır. Bu tip ağlara örnek olarak Gnutella 0.6 (Lu and Callan, 2003) ve KaZaA (Good and Krekelberg, 2003) verilebilir.

2.5 Üstüne Bindirmeli Ağlar

Üstüne bindirmeli ağ, fiziksel bir ağın üzerine kurulmuş sanal (virtual) bir bilgisayar ağıdır. Üstüne bindirmeli ağlardaki bağlantılar fiziksel ağdaki birden fazla bağlantının birleşiminden oluşabilir. Görevdeş ağlar, Internet üzerine kurulmuş bir üstüne bindirmeli ağ örneğidir. Üstüne bindirmeli ağ oluşturmanın belli başlı avantajları mevcuttur. Fiziksel ağların üzerine oluşturulan üstüne bindirmeli ağlardaki bağlantıların esnekliği sayesinde, fiziksel ağlarda gerçekleştirilmesi zor olan arama gibi işlemlerin kolaylıkla yapılması sağlanır. Azaltılmış (pruned) bağlantılar sayesinde ağ trafiği daha verimli yapılarak sistem performansı arttırılır. Üstüne bindirmeli ağlar sayesinde dosya paylaşımı ya da medya akışlandırması gibi uygulamalar kolaylıkla gerçeklenir. Bu avantajların yanı sıra dezavantajlar da mevcuttur. Üstüne bindirmeli ağlar, her türlü Internet kullanıcısına açık olduğundan güvenlik ve gizlilik (privacy) gibi sorunlar oluşabilmektedir. Üstüne bindirmeli ağları oluşturmak ve sürdürülebilirliğini sağlamak fazladan maliyet getirir. Şekil 2.7'de örnek bir üstüne bindirmeli ağ yapısı gösterilmiştir (Amir et al., 2002).



Şekil 2.7. Örnek bir üstüne bindirmeli ağ yapısı (Amir et al., 2002 – Figure 1).

Üst katman ağlarına baktığımızda göze çarpan ilk iki mimari dirençli üstüne bindirmeli ağ (resilient overlay network, RON) (Andersen et al., 2001) ve anlamsal üstüne bindirmeli ağdır (semantic overlay network, SON) (Crespo and Molina, 2002).

Dirençli üstüne bindirmeli ağ, üst katmanda bulunan yollarda kesinti olduğunu ve performansta yaşanan düşüşleri saniyeler içerisinde fark edip belirleyebilen ve bu durumun üstesinden gelebilen ilk geniş alan (wide area) üstüne bindirmeli ağdır. RON, var olan Internet yapısının altında yatan yönlendirme protokolünün üzerine kurulmuştur. RON yapısına dahil olan düğümler (node) Internet üzerindeki yönlendirme yollarının fonksiyonelliğini ve kalitesini monitör etmek için kullanılır. Bu düğümler ayrıca yönlendirme metriklerini optimize ederek gelen yönlendirme paketlerinin doğrudan Internet üzerinden mi yoksa RON yapısı üzerinden mi gideceğine karar verirler (Andersen et al., 2001).

RON yapısının ilk amacı altta yatan fiziksel ağ yollarını kullanarak birbiriyle iletişim kurmak isteyen düğümlerin grup oluşturmasını sağlamaktır. RON, ısrarlı sorgulama (aggressively probing) ve monitörleme yaparak kendi ağına dahil olan düğümler arasındaki bağlantıları fark eder. Eğer altta yatan Internet yönlendirme yolu en iyi yol ise, bu bulunan yol kullanılır ve başka hiçbir RON düğümü iletim (forwarding) yolu ile ilişkilendirilmez. Eğer altta yatan

Internet yönlendirme yolu en iyi yol değil ise, RON yönlendirme yolu olarak kendi ağı içerisinde bulunan RON düğümlerini kullanır (Andersen et al., 2001).

RON düğümleri, yönlendirme yolunun kalitesini sağlamak için yol metrikleri olan gecikme ve paket kayıp oranı (packet loss rate) değerlerini dikkate alarak birbirlerine iletim yolu ile ilgili tablolarını gönderirler. Her bir RON düğümü aktif sorgulama deneylerinin kombinasyonunu gözlemleyerek yol metriği için kendisine veri sağlar (Andersen et al., 2001).

RON sisteminin ikinci amacı, bilinen sistemlere göre yönlendirme ve yol seçimlerini dağıtık uygulamalar ile daha sıkı (tightly) bir şekilde yaparak birleştirmektir. Bu sayede birçok dağıtık uygulama ile RON yapısını kullanmış olur (Andersen et al., 2001).

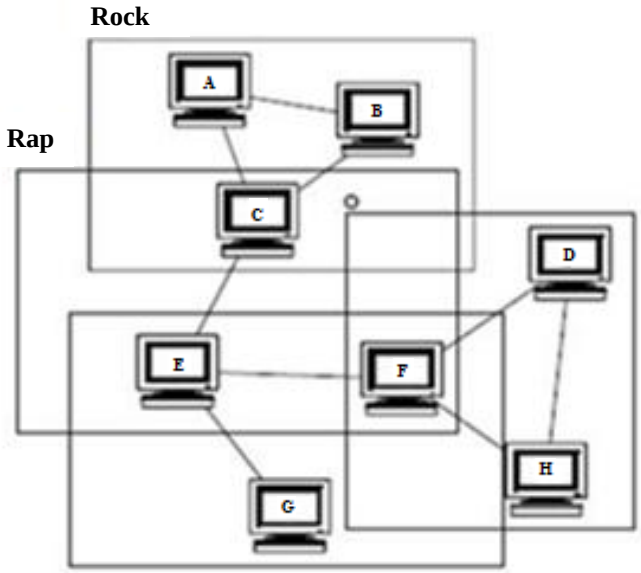
RON mimarisinin üçüncü amacı, verimli yönlendirme politikalarının (policy) gerçeklenmesi için bir taslak (framework) sağlamaktır. Örneğin, iletişim sırasında kullanılan paketlerin, kategorilere göre sınıflandırılmasının gerçeklenmesi taslak olabilir (Andersen et al., 2001).

RON yapısında yönlendirme politikasını açıklamak gerekirse; paket RON sistemine girdiği zaman ilk önce sınıflandırılır ve etiketlenir (tag). Bu etiketin amacı, paketin her bir RON yönlendiricisindeki yönlendirme tablolarına uygun olarak işlenmesidir. Herhangi bir metrik için RON düğümünden diğer tüm düğümlere yönlendirme için gerekli olan en kısa mesafe hesaplanır. Bu sayede iletim yolu belirlenmiş ve bir sonraki paket için yönlendirme tabloları oluşturulmuş olur. Şekil 2.8’de dirençli üstüne bindirmeli ağ mimarisinde yönlendirme işlemi gösterilmiştir.

Şekil 2.8. Dirençli üstüne bindirmeli ağ mimarisinde yönlendirme işlemi (Andersen et al., 2001 – Figure 3).

Anlamsal üstüne bindirmeli ağlardaki eşler arasındaki bağlantı, eşlerin sahip oldukları içeriklere göre yapılır. Örneğin, Jazz dosyasına sahip olan düğümler birbirine bağlanır. Böylece anlamsal olarak ilişkili düğümlerin oluşturduğu anlamsal üstüne bindirmeli ağ oluşur. Sorgular ilgili SON yapıları üzerinden iletilir (Crespo and Molina, 2002).

P2P sistemlerinde dökümanlar eşler tarafından tutulur. Fakat SON sistemlerinde dökümanlar yerine düğümler sınıflandırılmaktadır. SON mimarisi, anlamsal olarak benzer içeriğe sahip olan düğümlerin aynı kümelerde toplanması esasına dayanmaktadır. Bu sistemler, arama performansını arttıran esnek yapılardır. Bu ağlarda arama mesajları uygun kümelere yönlendirilerek arama sorgusunu alan kümelerde uyuşan veri bulma olasılığı artar ve aranan dosyaların hızlıca bulunması sağlanarak, aranan içerikle ilişkisi olmayan düğümlerdeki arama yükünün azaltılması sağlanmış olur. Örnek bir anlamsal üstüne bindirmeli ağ mimarisi Şekil 2.9'da gösterilmiştir.



Şekil 2.9. Örnek bir anlamsal ağ mimarisi (Crespo and Molina, 2002 – Figure 1).

Şekil 2.9'da A dan H ye sekiz adet düğüm görülmektedir. A, B ve C düğümlerinin hepsi Rock müziklerine sahiptir. Bu yüzden bu düğümler aynı içeriğe sahip olduklarından dolayı birbirleriyle iletişim kurabilirler. Benzer şekilde C, E ve F düğümleri de Rap müziklerine sahiptir ve bu yüzden onlar da birbirleriyle iletişim kurabilirler. Kurulan bu SON içerisinde hangi düğümün hangi düğüm ile iletişime geçeceği konusunda bir mecburiyet yoktur. Örneğin Şekil 2.9'da Rap SON sisteminde yer alan C düğümünün aynı SON içerisinde yer alan F düğümü ile bağlanması gerekmez. Ayrıca düğümler birden fazla SON sistemine dahil olabilirler. Şekil 2.9'da C düğümü hem Rap hem de Rock SON sistemine dahildir. SON sisteminde sorgular, ait oldukları SON içerisinde işlenir ve cevaplanır. Böylece sorgu için geçen süre azaltılmış olur (Crespo and Molina, 2002).

SON sistemini oluştururken karşılaşılan en önemli zorluk, sorguları ve düğümleri içeriklerine göre sınıflamaktır. Örneğin, bir düğümün birkaç tane Rock müziği ile ilgili dökümanı varsa bu düğümün yüzlerce Rock müziği bulunan SON yapısına eklenip eklenmemesi gerektiği önemli bir sıkıntıdır. Ayrıca sorguya cevap aranırken hangi SON yapısının, ilgili sorguya cevap vermek için seçileceği de yine ayrı bir zorluk olarak karşımıza çıkmaktadır (Crespo and Molina, 2002).

Yukarıda bahsedilen RON ve SON dışında daha birçok üst katman ağ yapısı mevcuttur. Genellikle uygulama katmanında oluşturulan ve eşlerin kümelenmesi mantığına dayanan bu ağlardan en önemlileri NARADA (Chu et al., 2002), NICE (Banerjee et al., 2002), ZIGZAG (Tran et al., 2004), mOVERLAY (Zhang et al., 2004), NEMO (Birrer and Bustamante, 2005) ve CANICULA (Chen, 2006)'dır. Şimdi bu yapıları ayrıntılı olarak inceleyelim.

2.5.1 Narada üstüne bindirmeli ağ modeli

Çoklu gönderim ile ilişkili fonksiyonel işlemlerin gerçeklenmesinde IP kullanımı çok yaygın olan bir protokol olmuştur. Fakat daha sonraları IP çoklu gönderim sisteminin ölçeklenebilirlik, ağ yönetimi, konuşlanma (deployment) ve üst seviye fonksiyonel işlemler (örneğin hata, akış ve tıkanıklık (congestion) kontrolleri gibi) için sorun teşkil ettiği ortaya çıkmıştır (Chu et al., 2002).

Bahsi geçen NARADA yapısında IP çoklu gönderim sistemi yerine uç sistem çoklu gönderimi (End System Multicast) önerilmiştir. Bu uç sistem çoklu gönderim yapısı bütün çoklu gönderim ile ilişkili işlemleri (örneğin üyelik yönetimi (membership management) ve paket kopyalanması (packet replication)) gerçekler. Bununla birlikte uç sistem çoklu gönderim sistemi, fiziksel hat (physical link) üzerinde paketlerin kopyalarının taşınmasına ve IP çoklu gönderim sistemine göre gecikmenin daha fazla olmasına neden olmaktadır (Chu et al., 2002).

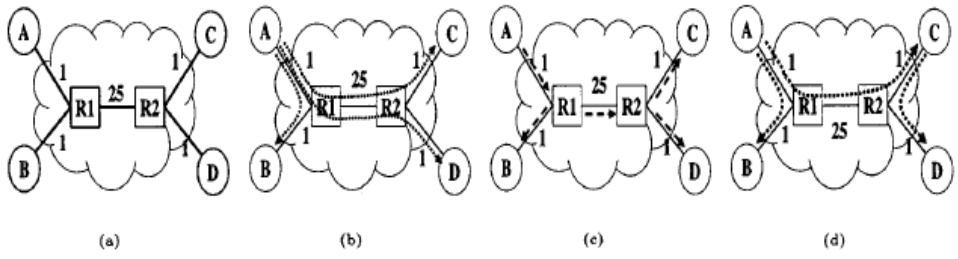
NARADA protokolü, tamamen dağıtık bir yapı kullanarak uç sistemlerin, üstüne bindirmeli yapılar içerisine kendilerini organize etmelerini sağlar. Ayrıca uç sistemler, ağ dinamiklerini ve uygulama seviyesindeki performansları dikkate alarak üstüne bindirmeli ağ yapılarının verimliliğini optimize etmeye çalışırlar (Chu et al., 2002).

NARADA protokol mimarisi tasarlanırken IP katmanına ek olarak hangi yeni özelliklerin eklenmesi gerektiği önemli bir parametre olarak ortaya çıkmıştır. Çoklu gönderim ve servis kalitesi (quality of service) IP katmanına eklenmesi gereken iki çok önemli özellik olarak ortaya konulmuştur. Servis kalitesi, yalnızca

uç sistemler tarafından sağlanamayabilirken, IP katmanı tarafından desteklenmektedir. Ama bu durum çoklu gönderim için geçerli değildir. Uç sistemler için çoklu gönderim servislerini, IP servislerinin üstüne gerçeklemek olası çözümlerdir (Chu et al., 2002).

Protokol yapısında önemli olan başka bir nokta da, çoklu gönderim servislerinin IP katmanında veya uç sistemlerde gerçeklenmesine karar verilmesidir. Ayrıca bu çoklu gönderim sistemlerinin üst katmanlarda mı yoksa daha yüksek performans için alt katmanlarda mı gerçeklenmesi konusu da yine önemli tasarım parametreleri olarak karşımıza çıkmaktadır (Saltzer et al., 1984).

Uç sistem yapılarının birçok avantajlarının yanında IP çoklu gönderim sistemine alternatif olabilmesi için çözülmesi gereken sorunları bulunmaktadır. Örneğin, üstüne bindirmeli ağ yaklaşımı IP çoklu günderim sisteminde olduğu gibi verimli şekilde gerçeklenememektedir. Aynı fiziksel hat üzerinde gereksiz yere gezinmeden üstüne bindirmeli yapılar arasında paket iletişimi yapmak imkansızdır. Bu nedenle fiziksel hat üzerinde gereksiz trafik üretilmiş olunur ki bu da istenmeyen bir durumdur. Ayrıca uç sistemler arasında iletişim kurmak bir uç sistemden diğer uç sisteme gezinmeyi gerektirir, yine bu da gecikmenin artması demektir (Chu et al., 2002). Şekil 2.10'da IP çoklu gönderim, tek yönlü gönderim (unicast) ve uç sistem çoklu gönderim yapıları arasındaki fark gösterilmiştir.



Şekil 2.10. IP çoklu gönderim, tek yönlü gönderim ve uç sistem çoklu gönderim yapıları (Chu et al., 2002 – Figure 1).

Şekil 2.10 (a) da örnek bir fiziksel topoloji görülmektedir. Şekilde R1 ve R2 yönlendiricileri; A, B, C ve D uç sistemleri temsil etmektedir. Uç sistemler ve yönlendiriciler arasındaki hatların gecikme değerleri de şekilde görülmektedir. R1 ve R2 arasındaki hattın çok maliyetli bir hat olduğunu şekilden anlamaktayız.

Diğer hatların ise maliyetinin çok düşük olduğu yine şekilde gözükmektedir. Varsayalım ki A uç sistemi, diğer bütün uç sistemlere veri göndermek istiyor olsun.

Şekil 2.10 (b) de tek yönlü gönderim gösterilmiştir. Tek yönlü gönderimde A kaynağına yakın olan A ile R1 arasındaki hat üzerinde fazla trafik görülmektedir. Bu hat üzerinde verinin üç adet kopyası taşınmaktadır. R1 ile R2 arasındaki hat üzerinde de aynı verinin iki adet kopyası taşınmaktadır. R1 ile R2 arasındaki hattın maliyeti çok yüksek olduğundan bu hat üzerinde verinin iki adet kopyasını taşımak maliyeti çok arttıracaktır. Bu istenmeyen bir durum olduğundan tek yönlü gönderim için büyük bir dezavantaj teşkil etmektedir.

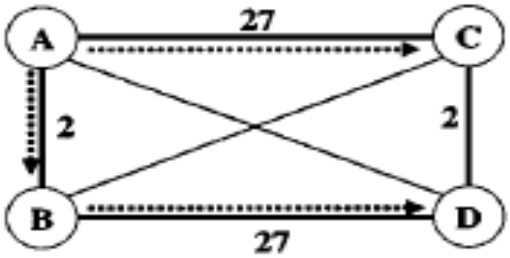
Şekil 2.10 (c) de IP çoklu gönderim sistemi gösterilmiştir. Bu gönderim şeklinde R1 ile R2 yönlendiricisi arasındaki hat üzerinde fazla trafik engellenmiştir. Verinin sadece bir adet kopyası herhangi bir fiziksel hat üzerinden geçmektedir. Ayrıca tüm alıcıların (recipient) veriyi alması için geçen gecikme değeri, A uç sisteminin doğrudan tek yönlü gönderim sistemini kullanması sonucu ortaya çıkan gecikme değeri ile aynı olmaktadır. Bu da toplam geçen gecikme süresinin IP çoklu gönderim sisteminde çok daha az olduğunu göstermektedir.

Şekil 2.10 (d) de uç sistem çoklu gönderim sistemi gösterilmiştir. Tek yönlü gönderime göre A kaynağı ile R1 yönlendiricisi arasındaki hat üzerinde taşınan verinin kopya sayısı azaltılmıştır. R1 ile R2 yönlendiricisi arasındaki maliyeti yüksek olan hat üzerinde verinin sadece bir adet kopyası taşınmaktadır. IP çoklu gönderim sistemi ile karşılaştırmak gerekirse, A kaynağı ile R1 arasındaki hat üzerinde verinin iki adet kopyası taşınmaktadır ve A kaynağı ile D uç sistemi arasındaki gecikme de artmıştır.

NARADA sisteminin tasarım özellikleri; uç sistemlerin tamamen dağıtık bir şekilde oluşturulması, grup üyeliklerinin dinamik olarak değişmesine karşı dayanıklı olması, yapının oluşması sonucu ağ yapısından ortaya çıkan ağaç yapılarının verimli olması, oluşan üstüne bindirmeli yapıları birbirine bağlayan fiziksel hatlar üzerindeki paket trafiğinin minimal tutulması, oluşturulan üstüne bindirmeli yapıdaki uç sistemlerin ölçeklenebilir şekilde ağ bilgisini elde

edebilmesini sağlayan bir mekanizma içermesi ve oluşturulacak olan üstüne bindirmeli yapının Internet yol karakteristiklerine (örneğin; bant genişliği ve gecikme gibi) uyum sağlayabilmesi şeklinde açıklanabilir (Chu et al., 2002).

NARADA yapısı sonucunda ortaya çıkan ağaç yapıları iki aşamada gerçekleşir. İlk aşamada, örgü (mesh) adı verilen bağlı çizge (connected graph) yapıları oluşturulur. İkinci aşamada, oluşturulan bu örgülerin kapsayan ağaçları (spanning tree) meydana getirilir. Her oluşturulan ağaç, iyi bilinen yönlendirme algoritmaları kullanılarak kaynak tarafından yönetilir (Chu et al., 2002). Şekil 2.11'de A kaynağı tarafından yönetilen ve NARADA yapısının fiziksel topoloji için oluşturduğu örgü yapısı gösterilmiştir.



Şekil 2.11. Narada yapısında örnek bir örgü tabanlı yaklaşım (Chu et al., 2002 – Figure 2).

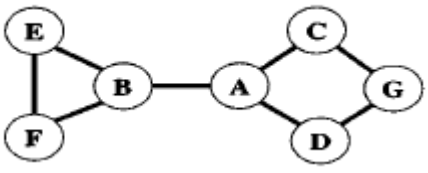
Örgü tabanlı yaklaşımlar, çok kaynaklı (multi source) uygulamalar tarafından desteklenir. Böylece tek kaynaklı ve tek paylaşımlı ağaç yapılarının tek hata noktası sorunlarının önüne geçilmiş olur. Ayrıca tek bir kaynak için birden çok üstüne bindirmeli ağaç oluşturmak da ayrı bir alternatif olabilir (Chu et al., 2002).

Örgü yapısı içerisindeki üstüne bindirmeli hatlar kullanılarak veri göndermek için ağaçlar oluşturulur. Bu önemlidir, çünkü kaliteli ağaç yapıları oluşturmak için iyi örgü yapıları kurulmalıdır. İyi bir örgü yapısının iki özelliği bulunmaktadır; ilk özellik herhangi bir üyeler arasındaki yolun kalitesinin, herhangi bir üyeler arasındaki tek yönlü gönderim yolunun kalitesi ile karşılaştırılabilir olmasıdır, ikinci özellik ise örgü içerisindeki her üyenin sınırlı sayıda komşusu olmasıdır (Chu et al., 2002). Burada yol kalitesinden kastedilen gecikme ve bant genişliği gibi metriklerdir. Komşu sayısını sınırlayarak örgü

yapısı üzerinde çalışan yönlendirme algoritmalarının maliyeti kontrol altına alınır (Chu et al., 2002).

NARADA yapısındaki grup yönetimlerinin nasıl yapıldığını özetleyelim. Yeni bir üye sisteme katılacağı zaman, öncelikle bu üyenin bütün üyelerin bulunduğu grupların listesini dışarıdan tüm yapıya sahip bir mekanizmadan aldığı varsayılmaktadır. Çalışmada bu mekanizmanın nasıl çalıştığına değinilmemiştir. Ayrıca tüm yapıyı bilen bu mekanizmanın olması, tamamen dağıtık olarak anlatılan yapının tümüyle dağıtık olmadığı anlamına gelmektedir. Daha sonra sisteme katılmak isteyen üye dışarıdan elde ettiği tüm grup listesi içerisinden rastgele gruplar seçer ve o gruplara katılmak istediğini bildiren bir mesaj gönderir. Sisteme dahil olmak isteyen üye, gönderdiği mesaja cevap alıncaya kadar bu işlem devam eder. Grupların hangisinden daha önce cevap alırsa o gruba girer. Anlatılan bu yapı düz bir yapıdır. Hiyerarşik bir yapı olmadığından, sisteme dahil olmak isteyen üye çok fazla sayıda sisteme girmek istediğini bildiren mesaj gönderir, bu da mesaj karmaşıklığının artmasına neden olmaktadır. Ayrıca gecikme metriği dikkate alınmadığı için grup içerisindeki üyeler arasındaki mesafeler çok fazla olabilir. Bu da sistem sonucu oluşan grupların dengesiz dağılım göstereceği anlamına gelmektedir. Sistemin hiyerarşik şekilde tasarlanmamış olması da ölçeklenebilirlik açısından verimsiz olduğunu göstermektedir (Chu et al., 2002).

Grup içerisindeki üyeler gruplarını terk edebilirler. Gruptan ayrılma işlemi haberli veya habersiz olarak iki şekilde yapılmaktadır. Şekil 2.12'de NARADA yapısında örnek bir üstüne bindirmeli topoloji gösterilmiştir.



Şekil 2.12. Narada yapısında örnek bir üstüne bindirmeli topoloji örneği (Chu et al., 2002 – Figure 4).

Şekildeki topoloji üzerinden gruptan ayrılma işlemini açıklamak gerekirse, varsayalım ki C habersiz bir şekilde gruptan ayrılsın. Örgü yapısı içerisinde

bulunan ve komşuları olan A ve G'nin belirli aralıklarla C'den aldığı mesaj duracaktır. A ve G birbirlerinden bağımsız olarak C' ye fazladan trafik oluşmasına neden olan mesajlar göndereceklerdir. Eğer C'den herhangi bir cevap alamazlar ise C'nin sistemden ayrıldığını düşünecekler ve bu bilgiyi örgü içerisindeki diğer üyelere göndereceklerdir. Burada kullanılan algoritmada da mesaj karmaşıklığı son derece yüksektir. Üye sayısının çok fazla olduğunu düşünürsek gereksiz olarak gönderilen kontrol mesajlarının sistem üzerinde oluşturacağı mesaj karmaşıklığı son derece yüksek olacaktır (Chu et al., 2002).

Örgü yapısında bölünmeler meydana gelebilmektedir. Yine Şekil 2.12 üzerinden yola çıkarsak; varsayalım ki A üyesi habersiz bir şekilde sistemden ayrılsın. Bunun sonucunda örgü yapısı ikiye bölünecektir. Üyeler ilk önce bölünmeyi tespit etmeye çalışacaklardır. Bölünme tespit edildikten sonra, bölünmenin meydana geldiği bölgeye en az bir tane üstüne bindirmeli hat eklenecektir (Chu et al., 2002).

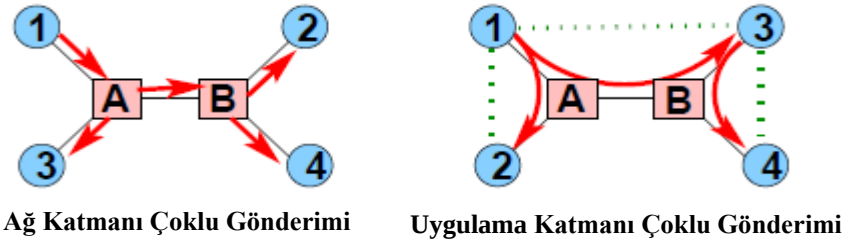
2.5.2 Nice üstüne bindirmeli ağ modeli

NICE, düşük bant genişliğine sahip ve geniş alıcı kümelerinin veri akışlandırma uygulamaları için tasarlanmış yeni bir ölçeklenebilir uygulama seviyesi çoklu gönderim (application-layer multicast) protokolüdür. Önerilen protokol, uygulama seviyesi çoklu gönderim düğümlerinin hiyerarşik olarak kümelenmesine ve değişik sayıda veri gönderim ağaçlarının (data delivery trees) desteklenmesi temellerine dayanmaktadır (Banerjee et al., 2002).

Çoklu gönderim, veri aktarım uygulamaları içerisinde paket göndermek için verimli bir tekniktir. Ayrıca ölçeklenebilir çok kullanıcılı uygulamalar için kullanışlı bir sistemdir. Uygulama seviyesi çoklu gönderim protokolleri, ağın alt yapısını (infrastructure) değiştirmezler. Bunun yerine yalnızca uç kullanıcılarda çoklu gönderim iletiminin fonksiyonelliğini gerçeklerler. Bu tip uygulama seviyesi çoklu gönderim protokollerinin, ticari içerik dağıtım (content-distribution) ağlarının verimli şekilde gerçeklenmesi için kullanımı artmıştır (Pendarakis et al., 2001). NICE protokolünün tasarım amacı, geniş veri akışlandırma uygulamalarını düşük kontrol maliyeti ile gerçeklemek ve düşük

gecikmeli ağaçlar üretmektir. Ayrıca bu protokolü yüksek bant genişliği gereksinimi duyan uygulamalar için kullanmak da mümkündür (Banerjee et al., 2002).

Uygulama seviyesi çoklu gönderim tekniğinin ana fikri Şekil 2.13'te gösterilmiştir. Bilinen çoklu gönderim yapısında veri paketlerinin kopyalanması ağ içerisindeki yönlendiricilerde yapılırken, uygulama seviyesi çoklu gönderim sistemlerinde uç kullanıcılarda yapılmaktadır. Mantıksal olarak uç kullanıcılar üstüne bindirmeli ağ oluştururlar. Uygulama seviyesi çoklu gönderim sisteminin amacı veri transferi için oluşturulan üstüne bindirmeli yapının verimliliğini ve sürdürülebilirliğini sağlamaktır. Uygulama seviyesi çoklu gönderim protokolleri, aynı hat üzerinden özdeş (identical) paketleri göndermek zorundadırlar fakat bu sistem, bilinen çoklu gönderim sisteminden daha verimsizdir (Banerjee et al., 2002).

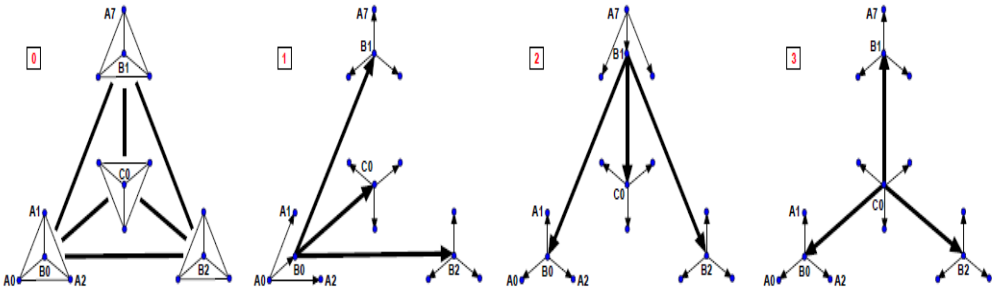


Şekil 2.13. Ağ ve uygulama seviyesi çoklu gönderim sistemleri (Banerjee et al., 2002 – Figure 1).

Genel olarak uygulama seviyesi çoklu gönderim protokollerini değerlendirirken; veri gönderim yolunun kalitesi, oluşturulan üstüne bindirmeli yapının hatalara karşı direnci ve kurulan sistemin kontrol maliyetine bakılmaktadır. NICE yapısında hiyerarşik olarak kontrol topolojisi oluşturulur. Oluşturulan bu hiyerarşik sistem yapısaldır ve ek bir yönlendirme hesaplarına gerek duyulmaz (Banerjee et al., 2002).

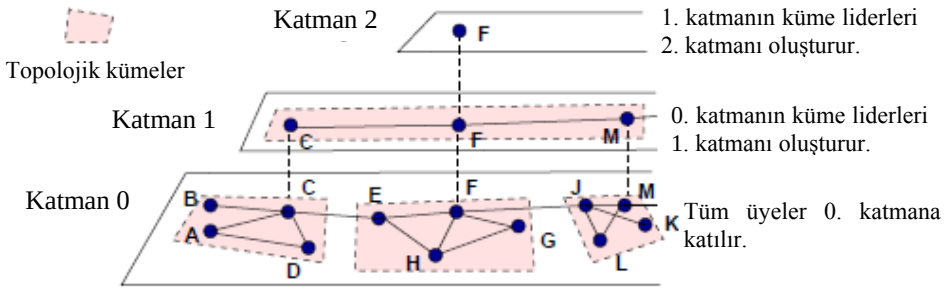
NICE protokolü, uç kullanıcılar kümesini hiyerarşik bir yapı içerisine yerleştirir. Hiyerarşik olan bu yapı çoklu gönderim tekniğini kullanan üstüne bindirmeli veri yollarını tanımlamaktadır. Hiyerarşik yapı sayesinde sistemin ölçeklenebilirliği sağlanmış olur. Her üye hiyerarşide en alt seviyede yer almaktadır. Hiyerarşideki her üye kendisine yakın olan üyelerin durum bilgilerini

tutmaktadır ve bu üyeler grup içerisindeki diğer üyeler hakkında sınırlı bilgiye sahiptir. NICE protokolünde gruplama için üyeler arasındaki uçtan uca gecikme metriği kullanılmıştır (Banerjee et al., 2002). NICE yapısında iki katmanlı hiyerarşi için kontrol ve veri gönderim yolları Şekil 2.14'te gösterilmiştir.



Şekil 2.14. Nice protokolünde iki katmanlı hiyerarşi için kontrol ve veri gönderim yolları (Banerjee et al., 2002 – Figure 3).

Katmanlar, en alt katmana sıfır değeri verilerek üst katmanlara doğru sırayla artacak şekilde numaralandırılırlar. En alt katman L_0 olarak temsil edilmektedir. Üyeler her katman için kümelere bölünmüşlerdir. Her kümenin içereceği üye sayısı k ve 3k-1 ($k \in Z^+$) arasında olmalıdır. Protokolün başlangıç aşamasında k değeri sabit bir değer olarak seçilmektedir ve protokolün çalışması esnasında değişmemektedir (Banerjee et al., 2002). Ayrıca her kümenin bir küme lideri bulunmaktadır. Protokol dağıtık olarak küme liderini kümenin merkezi olacak şekilde seçmektedir. Burada merkezi ifadesinden, küme içerisindeki diğer üyelere olan toplam gecikme mesafesi en az olan üyenin lider olarak seçilmesi kastedilmektedir. Küme liderini kümenin merkezinde seçmenin önemi, sisteme katılmak isteyen yeni üyenin hiyerarşi üzerinde kendisine gecikme olarak en yakın kümeyi hızlı bir şekilde bulmasıdır. Sistemdeki bütün üyeler en alt katman olan L_0'da yer almaktadır. Kullanılan kümeleme protokolü, L_0'da bulunan bütün üyeleri kümelere ayırmıştır (Banerjee et al., 2002). L_0'da bulunan tüm kümelerin küme liderleri bir üst katman olan L_1'de yer almaktadırlar. Şekil 2.15'te açıklanan hiyerarşik yerleşim gösterilmiştir.



Şekil 2.15. Nice protokolünde örnek bir hiyerarşik yerleşim (Banerjee et al., 2002 – Figure 2).

Şekil 2.15'te L_0'da yer alan kümeler [ABCD], [EFGH] ve [JKLM]' dir. Varsayalım ki C, F ve M bulundukları kümelerin küme liderleri olsun. Bu lider üyeler bir üst hiyerarşik katman olan L_1'de yeni bir küme oluştururlar, bu küme [CFM] kümesidir. F üyesi de L_1 katmanında bulunan kümenin merkezinde olduğu düşünüldüğünde küme lideri olarak seçilmiştir. Farklı katmanlarda bulunan üyelerin bu sistem içerisinde sahip olması gereken özellikler vardır. Bu özellikler;

- bir üye herhangi bir katmanda sadece bir kümeye aittir,
- eğer bir üye herhangi bir L_i katmanındaki bir kümede yer alıyorsa, mutlaka $L_{0,.....,L_{i-1}}$ katmanlarında da küme içerisinde yer almak zorundadır,
- eğer bir üye herhangi bir L_i katmanında yer almıyorsa, $j > i$ koşulunu sağlayan L_j katmanında bulunmamalıdır,
- her kümenin üye eleman sayısı k ile 3k-1 arasında olmalıdır ve küme lideri kümenin merkezinde bulunmalıdır,
- sistemde en çok $\log_k N$ (N, küme üyelerinin sayısı) katman bulunmalıdır ve en üst katmanın sadece bir üyesi olmalıdır

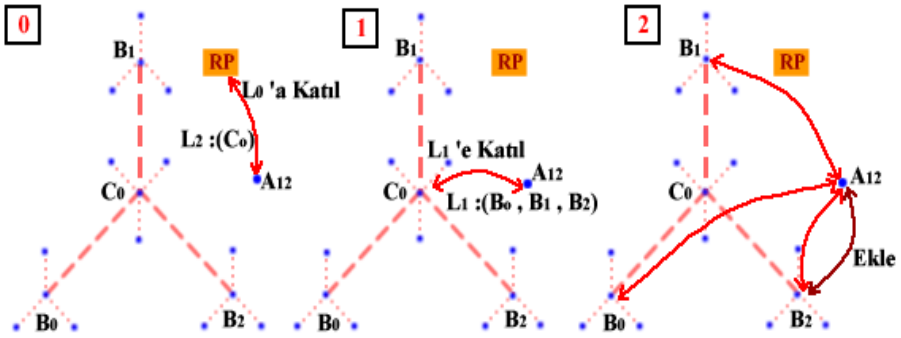
şeklinde sıralanır (Banerjee et al., 2002).

NICE protokolünü biraz daha ayrıntılı olarak inceleyelim. Sistemde öncelikle özel bir üyenin olduğu varsayılmaktadır. Sisteme dahil olmak isteyen üyelerin bildiği bu özel üyenin ismi randevu noktasıdır (rendezvous point, RP).

Protokole girmek isteyen her üye önce RP ile temasa geçmektedir. RP üyesi, hiyerarşik yapıdaki en üst kümenin lideri olan üyedir (Banerjee et al., 2002).

NICE protokolü kendi içerisinde üç ana kısımdan oluşmaktadır; yeni bir üyenin kümeye dahil olması, belirli aralıklarla yapılan kümelerin sürdürülebilirliğinin sağlanması ve küme liderlerinin sistemden ayrılmasına karşı sistemin kendini tekrar toparlaması olarak özetlenmektedir (Banerjee et al., 2002).

Yeni bir üyenin çoklu gönderim kümesine katılması, en alt katman olan L_0'da bulunan küme ile ilişkilendirilmesi demektir. Yeni bir üyenin sisteme katılması Şekil 2.16'da gösterilmiştir. Şekilde sisteme dahil olmak isteyen A_{12} üyesi ilk önce RP noktası ile temasa geçerek en üst katmanda bulunan kümenin lideri olan üyenin bilgisini alır. Şekilde en üst katman olan L_2'de sadece C_0 bulunmaktadır. A_{12} üyesi C_0 üyesi ile iletişime geçerek C_0 üyesinden alt katman olan L_1'deki diğer üyeler olan (B_0, B_1, B_2) bilgisini alır. A_{12} üyesi, her bir (B_0, B_1, B_2) üyesine sisteme katılmak için sorgu göndererek hangi üyenin kendisine en yakın olduğunu belirlemeye çalışır. Bu işlemler tekrarlamalı şekilde yeni üyenin en alt katman olan L_0'da kendine en yakın kümeyi bulup, o kümeye katılıncaya kadar devam eder (Banerjee et al., 2002).



Şekil 2.16. Nice protokolünde yeni bir üyenin çoklu gönderim kümesine katılması (Banerjee et al., 2002 – Figure 5).

Protokolde önemli olan başka bir nokta da kümelerin merkezlerinde bulunan küme liderlerinin hiyerarşik düzendeki yerleşimleridir. Üyelerin kümelere katılması veya bulundukları kümelerden ayrılmasıyla kümelerin gecikme değerlerinde dinamik olarak değişimler meydana gelecektir. Bu da belirli

aralıklarla küme lideri seçimi işleminin yapılması gerekliliğini ortaya koyar. Eğer küme lideri seçimi işlemi sonunda küme lideri değişirse bir üst katmanda bulunan daha önceki küme liderinin bir üst katmandan çıkarılması gerekmektedir. Çünkü eski lider artık normal bir üye olmuştur. Yeni seçilen küme lideri ise bir üst katmana katılacaktır (Banerjee et al., 2002).

NICE hiyerarşik yapısında üyelerin devamlı sisteme giriş çıkış yapmasından dolayı belirli aralıklarla sistemin sürdürülebilirliğinin sağlanması gerekmektedir. Belirli aralıklarla yapılan işlemler küme bölünmesi (cluster split), küme birleşmesi (cluster merge) ve küme içerisinden üyelerin ayrılması (host departure) şeklinde açıklanabilir.

Öncelikle küme bölünmesi durumunu inceleyelim. Küme bölünmesi, küme liderinin kontrolünde olan bir durumdur. Küme lideri belirli aralıklarla kümenin eleman sayısını kontrol eder. Eğer küme lideri, kümenin eleman sayısının belirlenen 3k-1 değerini aştığını tespit ederse küme bölünme mekanizmasını çalıştırır ve kümeyi iki eşit parçaya böler. Yapılan bu küme bölünme işlemi küme içerisinde bulunan üyeler arasındaki ikili gecikme ölçüm değerlerine bakılarak ve çizge kuramlarında kullanılan polinom zamanda çözülemeyen zorlayıcı K-merkez problemine benzer şekilde küme bölünmesi sonucu oluşan yeni kümenin maksimum yarı çapını minimize ederek yapılmaktadır. Kullanılan yöntem bir yaklaşım stratejisidir. Küme bölünmesi sonucu oluşan iki yeni kümenin maksimum yarıçaplarını minimize etmek temel amaçtır. Aynı zamanda bölünme sonucu kümenin iç dinamiği değiştiği için küme lideri seçimi yapılır. Küme liderinin seçimi yapıldıktan sonra yeni oluşan kümenin lideri bir üst katmana katılır (Banerjee et al., 2002).

Küme bölünmesine benzer şekilde küme birleşmesi de küme lideri tarafından kontrol edilir. Küme içerisindeki eleman sayısı k değerinin altında kalırsa küme lideri tarafından küme birleşimi mekanizması çalıştırılır. Birleşmenin meydana geleceği küme lideri bir üst katmanda bulunduğu küme içerisindeki diğer üyeler ile gecikme mesafesini bulmak için sorgular gönderir. Kendisine gecikme olarak en yakın olan üyenin L_0'da bulunan kümesine birleşmek istediğini bildiren mesajı kendisine en yakın olan üye liderine gönderir

ve küme birleşimi işlemi tamamlanır. Küme birleşiminin yaşandığı küme lideri kendisini bir üst katmandan siler ve L_0 katmanındaki yeni kümesine katılır (Banerjee et al., 2002).

Küme içerisinden bir üyenin ayrılması şu şekilde olmaktadır. Küme içerisinden ayrılmak isteyen bir H üyesi sistemde bulunduğu tüm kümelere sistemden ayrılmak istediğini belirten bir mesaj gönderir. Bu barışçıl bir ayrılmadır. H üyesi haber vermeden de sistemden ayrılabilir. Bu durumu fark eden küme lideri ilgili üyeyi listesinden çıkartır. Eğer H üyesi küme lideri ise ve kümeden habersiz şekilde ayrıldıysa bu durum yeni bir liderin seçilmesi gerekliliği durumunu tetikler. Küme içerisindeki her üye birbirinden bağımsız olarak yeni küme liderini belirler. Çoklu liderler birbirleri ile uzlaşarak tek bir küme lideri seçerler (Banerjee et al., 2002).

2.5.3 Zigzag üstüne bindirmeli ağ modeli

ZIGZAG yaklaşımında, eşlerin hiyerarşik bir yapıda kümelenmesi hedeflenmektedir. Oluşturulan bu hiyerarşik düzene yönetimsel (administrative) organizasyon denilmektedir. Yönetimsel organizasyon sayesinde, sistemin kolay idare edilebilmesi sağlanır ve verimli şekilde veri iletişimi için çoklu gönderim ağaçları oluşturulur. Protokolde oluşturulan çoklu gönderim ağaçlarının yüksekliği kullanıcıların logaritmik değerine eşittir ve eşlerin komşu sayıları da sabit bir değer ile sınırlandırılmıştır. Bu sayede ağ üzerindeki darboğazdan kaçınarak veri gönderim yolu üzerindeki zıplama (hop) sayısı azaltılır ve uçtan uca gecikme küçük tutulur. ZIGZAG yapısında akışlandırma ortamlarında karşılaşılan sorunlara, P2P sistemlerinin uygulanabilirliği incelenmiştir (Tran et al., 2004).

P2P akışlandırma mimarisinde, gönderim ağaçları kaynak tarafından başlatılır ve sadece alıcıları içerir. Alıcılardan bazıları içeriği kaynaktan doğrudan alırlarken bazıları da akışlandırma yolu üzerindeki alıcılardan alırlar. Bu yaklaşım büyük ölçekli akışlandırma sistemlerinde kritik sorunları ortaya çıkarır. Sorunlardan ilki medya kaynağında darboğaz oluşmasıdır. İkinci sorun ise fazladan alınan hizmetlerin maliyetidir. Son sorun ise kullanılan Internet yapısı

üzerinde IP çoklu gönderim sisteminin kullanılmasının çok mümkün olmayışıdır (Tran et al., 2004).

Verimli bir P2P akışlandırma sistemi kurmanın bir takım zorlukları bulunmaktadır. Bu zorluklar;

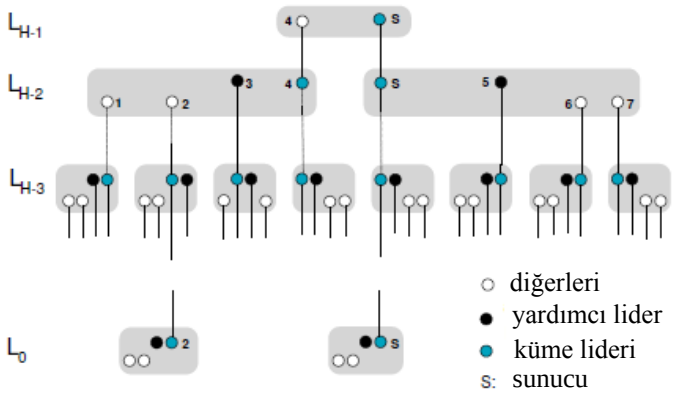
- medya içeriği, kaynaktan alıcıya giderken birçok ara alıcıdan geçtiği için kaynak ve alıcı arasındaki uçtan uca gecikme değeri gereğinden fazla olmaktadır. Gecikme değerini azaltmak için oluşturulan ağacın yüksekliğinin az olması ve sisteme katılma işleminin hızlı bir şekilde gerçekleşmesi gerekmektedir,
- alıcıların sisteme karşı davranışları kararsızdır. Yani sisteme giriş ve çıkış konusunda serbest hareket ederler. Bu yüzden sağlanan akışlandırma servisinin hatalara karşı dayanıklı olarak tasarlanması gerekmektedir,
- alıcılar, P2P ağının verimliliğini arttırmak ve sistemin sürdürülebilirliğini sağlamak için bazı veri yapıları bilgilerini saklayarak bu bilgileri diğer alıcılar ile değiş tokuş yaparlar. Bu işlemin sisteme getireceği kontrol maliyeti, ağ kaynaklarının gereğinden fazla şekilde tüketilmemesi için düşük tutulmalıdır

şeklinde sıralanabilir (Tran et al., 2004).

ZIGZAG protokolü bazı kurallar kullanarak, alıcıları hiyerarşik çoklu gönderim ağacı üzerinde kümeleme tekniğini kullanır. Bu kurallara bağlanabilirlik (connectivity) kuralları denilmektedir. Protokolde her kümenin bir küme lideri ve yardımcı lideri bulunmaktadır. Küme lideri, küme içerisindeki üyelikleri monitörleme işinden sorumludur. Yardımcı lider ise medya içeriğini küme üyelerine iletmekle görevlidir. Protokolde, küme liderinin veya yardımcı liderin sistemden ayrılması durumunda sistemin çalışabilirliğinin devamı sağlanmıştır. Şimdiye kadar olan kısımları özetlersek, tasarlanan protokolün arzu edilen bazı özellikleri sağladığı görülür. Bu özelliklerden birincisi, sistemdeki alıcıların sayısını N ile temsil edersek N sayısının ne kadar büyüdüğünün bir önemi yoktur, sadece çoklu gönderim ağacı üzerindeki düğümlerin komşu sayıları sabit bir değer

ile sınırlandırılmalıdır. İkinci özellik çoklu gönderim ağacının yüksekliği O(logN) ile sınırlandırılmalıdır. Üçüncü özellik hatalara karşı dayanıklılık yerel olarak yapılmalıdır. Dördüncü özellik protokolün kontrol maliyeti düşük olmalıdır. Son özellik ise sisteme katılma mekanizması hızlı olmalıdır ve sistemin sürdürülebilirliğinin sağlanması için gereken maliyet düşük ve alıcıların sayısından bağımsız olmalıdır şeklinde özetlenebilir (Tran et al., 2004).

ZIGZAG protokolünde iki önemli birim bulunmaktadır. Bu birimler eşlerin birbirleri arasında mantıksal ilişkilerini temsil eden yönetimsel organizasyon ve fiziksel ilişkilerini temsil eden çoklu gönderim ağacıdır. Öncelikle yönetimsel organizasyonun nasıl oluştuğuna, daha sonra oluşturulan bu organizasyon üzerine kurulan çoklu gönderim ağacının nasıl kurulacağına bakalım. Yönetimsel organizasyon, sistem içerisindeki eşlerin yönetimini sağlamak için kullanılan bir yapıdır. Şekil 2.17'de eşlerin yönetimsel organizasyonu gösterilmiştir.

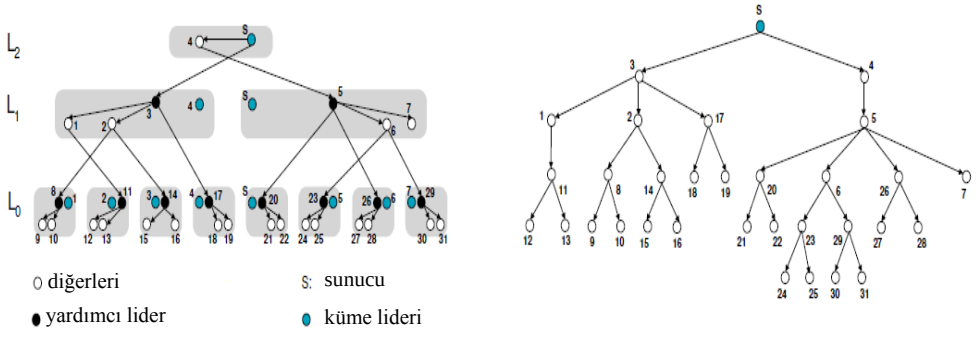


Şekil 2.17. Zigzag protokolünde eşlerin yönetimsel organizasyonu (Tran et al., 2004 – Figure 1).

Şekil 2.17 incelenecek olursa, en üst katman olan H – 1 katmanında S sunucusu hem küme lideri hem de yardımcı liderdir. H – 2 katmanındaki kümede bulunan 3 numaralı eş, ilgili kümenin yardımcı lideri ve aynı zamanda H – 3 katmanındaki kümenin lideridir. 4 numaralı eş, H – 2 katmanında bulunan kümenin lideridir ve aynı zamanda H -1 katmanındaki kümenin de bir üyesidir.

Çoklu gönderim ağacı, yönetimsel organizasyon üzerine kurulmaktadır. Çoklu gönderim ağacı oluştururken eşlerin sisteme katılması, sistemden ayrılması

ve sistemin optimizasyonu gibi mekanizmaların takip etmesi gereken bağlanabilirlik kuralları vardır (Tran et al., 2004). Yönetimsel organizasyon üzerine kurulan çoklu gönderim ağacı Şekil 2.18'de gösterilmiştir.



(a) Yönetimsel organizasyon üzerine kurulan çoklu gönderim ağacı

(b) Çoklu gönderim ağacının farklı açıdan görünüşü

Şekil 2.18. Zigzag protokolünde yönetimsel organizasyon üzerine kurulan çoklu gönderim ağacı (Tran et al., 2004 – Figure 2).

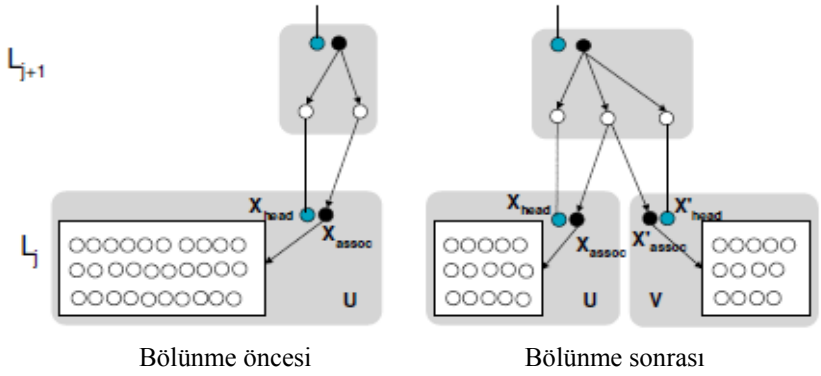
Protokole yeni bir üyenin katılması şu şekilde olmaktadır. Sisteme katılmak isteyen P üyesi, sunucuya sisteme katılmak istediğini bildiren bir istek mesajı gönderir. Eğer yönetimsel organizasyon yapısında tek bir katman varsa, P sunucuya bağlanır. Eğer sistemde birden çok katman varsa istek mesajı, çoklu gönderim ağacı yolu üzerinden yeni üyenin kendisine uygun kümeyi buluncaya kadar iletilir. Yeni üye bu işlemler esnasında gecikme metriğini kullanarak en alt katmanda kendisine en yakın kümeye girmiş olur (Tran et al., 2004).

Tasarlanan protokolde sisteme yeni üyelerin katılacağı gibi sistemden ayrılmak isteyen üyeler de olabilir. Sistemden ayrılmak isteyen üye haberli veya habersiz şekilde sistemden ayrılabilir. Varsayalım ki sistemden ayrılmak isteyen X üyesi sistemde en üst katmanda bulunsun. X üyesinin en üst katmanda bulunması, X üyesinin aynı zamanda ara katmanlarda ve en alt katmanda da bulunduğu sonucunu ortaya çıkarır. X üyesinin sistemden ayrıldıktan sonra yapılması gereken işlemler; ilk olarak ebeveynin X üyesi ile olan bağlantıyı silmesi, ikinci olarak X üyesinin çocuklarının medya içeriğini alabilmek için yeni bir ebeveyne ihtiyaç duyması, üçüncü olarak X üyesi eğer ara katmanlarda küme lideri ise yeni bir küme lideri seçilmesi ve son olarak eğer X üyesi en üst

katmanda yardımcı lider ise yeni bir yardımcı liderin seçilmesi şeklindedir (Tran et al., 2004).

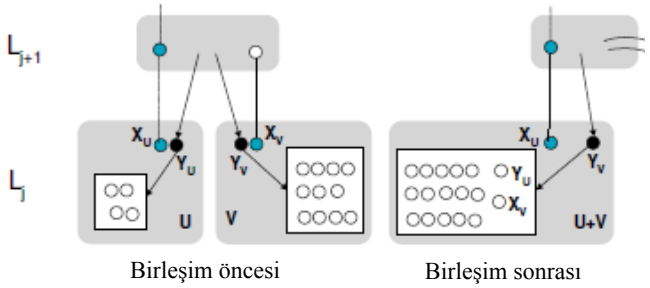
Sisteme giren ve çıkan üyelerin olması, bazı kümelerin eleman sayısının belirtilen k ve 3k aralığının altına düşmesine veya üstüne çıkmasına neden olmaktadır. Eğer kümenin eleman sayısı belirtilen aralığın üstüne çıkarsa ilgili kümenin bölünerek yeni kümelerin oluşması, eğer kümenin eleman sayısı belirtilen aralığın altına düşerse ilgili kümenin kendisine uygun olan başka küme ile birleşmesi gerekmektedir (Tran et al., 2004).

Protokolde küme bölünme işlemi şu şekilde yapılmaktadır. Varsayalım ki j katmanında bulunan, küme lideri X_{head}, yardımcı lideri X_{assoc} ve diğer üyeleri X_1, ..., X_n olan bir U kümesinin eleman sayısının belirtilen aralığı aştığını düşünelim. Amaç yeni bir V kümesi oluşturarak U kümesinden bazı üyeleri V kümesine geçirmektir (Tran et al., 2004). Şekil 2.19'da ana fikir gösterilmiştir.



Şekil 2.19. Zigzag protokolünde örnek bir küme bölünmesi (Tran et al., 2004 – Figure 4).

Protokolde küme birleşimi işlemi, üst katmanlardan alt katmanlara doğru yapılmaktadır. Diğer bir ifadeyle üst katmandaki bir kümenin eleman sayısı minimum eşik değer olan k'nin altına düşerse ilgili kümede birleşim işlemi, alt katmanda bulunan kümeye göre daha önce yapılacaktır (Tran et al., 2004). Şekil 2.20'de örnek bir küme birleşimi gösterilmiştir.



Şekil 2.20. Zigzag protokolünde örnek bir küme birleşimi (Tran et al., 2004 – Figure 5).

2.5.4 mOverlay üstüne bindirmeli ağ modeli

mOVERLAY protokolünün uygulanması sonucu oluşturulan üstüne bindirmeli yapı, uç kullanıcılar arasındaki iletişim maliyetini azaltmalı ve mesajın hedefe ulaşmasını az bir maliyetle garanti etmelidir. Bölgesel olarak birbirine yakın olan kullanıcılar ile üstüne bindirmeli yapı oluşturmak için dinamik işaretleme (dynamic landmark) tekniğinden yararlanılmıştır. Tasarlanan üstüne bindirmeli yapı üzerinde yeni bir kullanıcının sisteme dahil olması için verimli bir yerleştirme mekanizması kullanılır (Zhang et al., 2004).

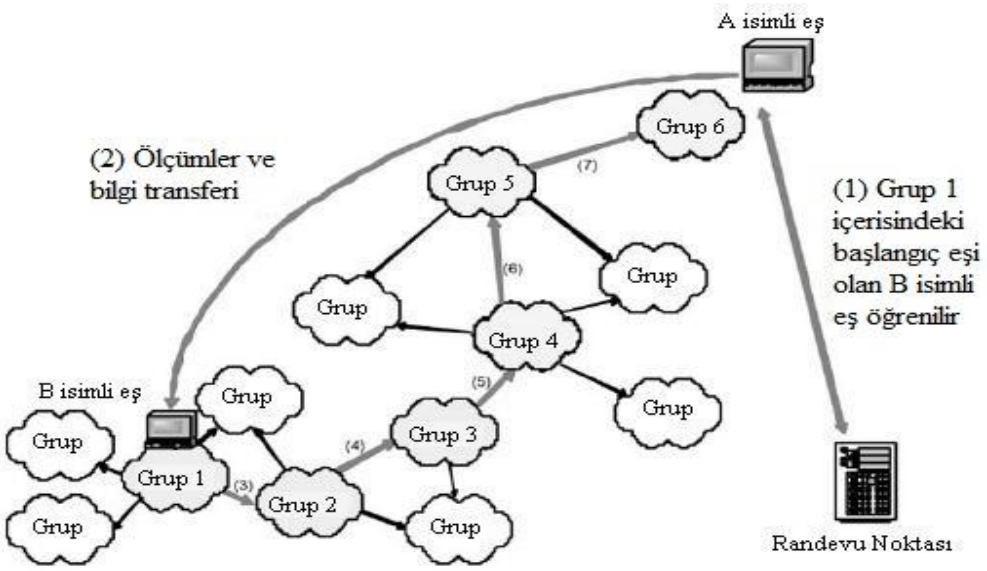
Oluşturulan üstüne bindirmeli yapı üzerinde odaklanılması gereken çok önemli iki nokta bulunmaktadır. Bunlar verimlilik ve ölçeklenebilirliktir. Verimlilikten kastedilen üstüne bindirmeli yapı üzerinde yapılan iletişimin maliyetinin çok fazla olmamasıdır. Bu prensibe bağlı olarak üstüne bindirmeli yapı üzerindeki komşu kullanıcılar birbirlerine yakın olmalıdırlar. Ölçeklenebilirlikten kastedilen ise sistemin kullanıcı sayısı ve veri transferi arttıkça, üstüne bindirmeli yapının yönetilebilir olmasıdır. Üstüne bindirmeli ağ yapısının sürdürülebilirlik maliyeti olabildiğince az olmalıdır. Üstüne bindirmeli yapı, dağıtık bir şekilde gerçeklenmelidir. Sonuç olarak bu gözlemlerden şu çıkıyor ki; bölgesel veya ağdaki yakınlık, üstüne bindirmeli bir sistem tasarlamanın en önemli karakteristik özellikleridir. Çünkü yakınlık, tasarlanan üstüne bindirmeli yapının verimliliğini ve ölçeklenebilirliğini etkilemektedir (Zhang et al., 2004).

Protokolde birbirine yakın olan kullanıcıların gruplanarak üstüne bindirmeli yapı oluşturulur. Gruplama mekanizması sabit bir işaretlemeye dayanmamaktadır. Önerilen gruplama tekniği ölçeklenebilir ve hatalara karşı dayanıklıdır. Ayrıca gruplama yaparak sistemin sürdürülebilirlik maliyeti önemli derecede azaltılır (Zhang et al., 2004).

Önerilen mimari iki seviyeli hiyerarşik ağ yapısından oluşmaktadır. Üst seviyeler gruplardan, alt seviye ise gruplar içerisindeki kullanıcılardan oluşmaktadır. Bilgi, grup içerisindeki tüm kullanıcılar arasında paylaşılır. Bölgesel yakınlık tabanlı yaklaşımı açıklamadan önce gruplama mekanizmasına değinelim. Grup, birbirine yakın olan kullanıcıların kümesine denir. Alt sistemde kullanılan ağ üzerindeki herhangi bir P pozisyonunu düşünürsek, eğer P ile A kullanıcısı arasındaki mesafe ile P ile B kullanıcısı arasındaki mesafe birbirine eşit ise A ve B kullanıcısı aynı gruptadır. Burada değinilen iki kullanıcı arasındaki mesafe ağ gecikmesi, RTT veya iki kullanıcı arasındaki hat üzerindeki minimum bant genişliği olarak ifade edilebilir. Bir grup diğer birçok grup ile mesaj alış-verişi yapabilir. Yeni bir kullanıcının hangi gruba dahil olacağını belirlemek için gruplama kriteri kullanılmaktadır. Gruplama kriteri şu şekilde işlemektedir; sisteme katılmak isteyen yeni bir Q kullanıcısının A grubunun komşu grupları ile mesafesi, A grubu ile A grubunun komşuları arasındaki mesafeye eşit ise Q kullanıcısı A grubuna aittir. Burada grup komşuları dinamik işaretleyiciler olarak sisteme etki ederler. Yakın olan grubu işaretleyici olarak seçeriz böylece uzakta bulunan bir grup ile mesafe ölçümü yapılmamış olur. Dinamik işaretlemenin bir diğer avantajı ise, ölçüm ve veri transferi için gerekli olan maliyetin düşürülmesidir. Dinamik işaretleme, birçok yararlı uygulamalar için ve üstüne bindirmeli yapıların toplam sürdürülebilirlik maliyetlerini azaltmak açısından önemli derecede rol oynar (Zhang et al., 2004).

Yeni bir üyenin gecikme metriğini dikkate alarak kendine yakın bir gruba yerleşimini inceleyelim. Öncelikle, protokolde üstüne bindirmeli ağ yapısındaki bazı kullanıcıları bilen bir RP kullanıcısı bulunmaktadır. Sisteme katılmak isteyen tüm kullanıcılar RP kullanıcısının nerede olduğunu bilirler. RP, genel olarak ya bir makinedir ya da makineler kümesidir. RP, sisteme katılmak isteyen yeni kullanıcı için başlangıç noktasıdır. İlk basamakta sisteme katılmak isteyen yeni

kullanıcı, RP ile temasa geçerek sistemde bulunan birçok kullanıcı bilgisini alır. Bu kullanıcılara başlatma kullanıcıları (boot hosts) denilmektedir. Bu kullanıcıların görevi, sisteme katılmak isteyen yeni kullanıcıyı yakın gruba yönlendirmektir (Zhang et al., 2004). Şekil 2.21'de yeni bir A kullanıcısının RP ile temasa geçerek gruba yerleşmesi gösterilmiştir.



Şekil 2.21. mOverlay protokolünde yeni bir kullanıcının sisteme katılması (Zhang et al., 2004 – Figure 2).

Şekilden de anlaşılacağı üzere RP, A kullanıcısının 1. grupta bulunan B başlatma kullanıcısı ile temasa geçmesi gerektiğini bildirir. Daha sonra A kullanıcısı B kullanıcısı ile mesafesini ölçer. Aynı zamanda B kullanıcısı kendi bulunduğu grup olan 1 numaralı grubun bütün komşu grupları bilgisini A kullanıcısına gönderir. Bütün bu komşu grupları bir listede tutulur, bu listeye aday grup (candidate group) listesi denilmektedir. A kullanıcısı sırayla aday grup listesi içerisinden grupları seçer ve seçtiği her grup ile kendi mesafesini ölçer. Kendisine en yakın olan grubu belirler. Eğer daha önce açıklanan gruplama kriteri ile karşılaşılırsa A kullanıcısı 1 numaralı gruba girer ve işlem tamamlanır. Gruplama kriteri ile karşılaşılmamışsa A kullanıcısına en yakın olan grup ile o gruba olan mesafe olan D_{min} kaydedilir. Şekilde 2 numaralı grup A kullanıcısına en yakın gruptur. 2 numaralı grup içerisindeki başlatma kullanıcısı bütün komşu gruplarının bilgisini A kullanıcısına gönderir. A kullanıcısı, 2 numaralı grubun tüm komşu grupları ile mesafesini ölçer. Tekrar gruplama kriteri kontrol edilir. Eğer gruplama kriteri ile karşılaşılmışsa A kullanıcısı 2 numaralı gruba girer ve

işlem tamamlanır. Eğer gruplama kriteri ile karşılaşılmamışsa 2 numaralı grubun komşuları arasında A kullanıcısı ile olan en kısa mesafe A kullanıcısına söylenir. Varsayalım ki A kullanıcısına, 2 numaralı grubun komşuları arasında en kısa mesafesi olan grup 3 olsun. 3 numaralı gruba olan mesafe ile daha önce kaydedilen D_{min} karşılaştırılır. Eğer D_{min} daha büyük ise; D_{min} değerinin yerine yeni ölçülen değer kaydedilir ve 3 numaralı grup A kullanıcısına en yakın grup olarak kaydedilir. Eğer D_{min} daha küçük ise 2 numaralı grup A kullanıcısına en yakın grup olarak kalmaya devam eder. Aynı prosedür 3 numaralı grupta başlar ve gruplama kriteri ile karşılaşılıncaya kadar devam eder. Eğer bütün gruplar gezinilip gruplama kriteri ile karşılaşılmazsa o an için kaydedilen grup ve D_{min} en yakın grup olarak seçilir. Şekilde A kullanıcısı 6 numaralı gruba katılmıştır. Eğer bütün grupları gezmek çok zaman alıyorsa yerleşme algoritmasının ne zaman bitirilmesi gerektiğini ayarlamak gerekli olur (Zhang et al., 2004).

Üstüne bindirmeli yapının oluşturulması ve sürdürülebilirliğinin sağlanması için birkaç protokol vardır. Bu protokoller bilinen yapısal olmayan üstüne bindirmeli yapılardaki protokollere benzerdir. Var olan protokoller üzerinde birkaç değişiklik yaparak ölçeklenebilirlik ve hatalara karşı dayanıklılık açısından daha verimli bir üstüne bindirmeli yapı oluşturulur. Sırasıyla değişiklik yapılan protokolleri inceleyelim. İlk olarak yeni grup oluşturma protokolünü ele alalım. Üstüne bindirmeli ağ yapısı oluşturmanın ilk aşaması, sistemde az sayıda grup varken sıklıkla yeni gruplar oluşturmaktır. Yeni bir grup oluşturmanın diğer bir koşulu ise yeni bir kullanıcının gruplama kriterine uygun olarak kendisine yakın bir grup bulamamasıdır. Yeni bir grup oluşturmanın prosedürü basittir. Daha belirli bir şekilde ifade edecek olursak kullanıcı, grup numarası ile yeni grubunu ilan eder. Aynı zamanda ilan ettiği yeni grubun M tane komşu gruplarını bulur. M tane komşu grupları bulmak için, ilk olarak RP mekanizmasından aldığı birkaç başlatma kullanıcılarını kullanarak yerleşim protokolünü tekrar eder. Yeni kullanıcıya yakın birkaç grup sağlanır. İkinci olarak gruba katılma protokolüne değinelim. Normal durumda, eğer kullanıcı kendisine uygun olan grupta yer alıyorsa, yeni kullanıcı ilgili gruba girer ve grup içerisindeki birkaç kullanıcı ile doğrudan iletişime geçer. Yeni kullanıcı, grubun sürdürülebilirliğini sağlamak için grubun kullanıcılarından gerekli bilgileri alabilir. Üçüncü olarak incelenen protokol bilgi paylaşım protokolüdür. Bilgi paylaşma, grup işlemleri için önemli

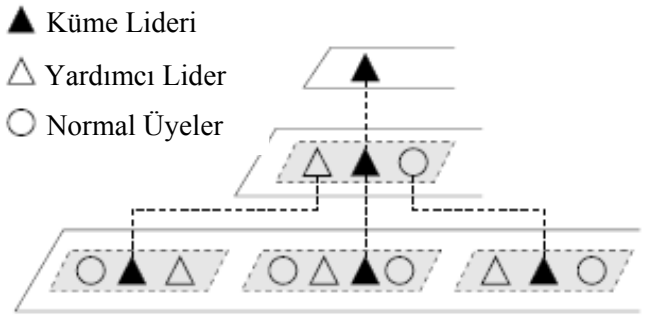
bir bileşendir. Aynı grup içerisinde yer alan kullanıcılar benzer ağ karakteristiklerine sahiptir yani A kullanıcısından belirli bir X grubuna olan mesafe, A kullanıcısının bulunduğu grubun tüm kullanıcılarının X grubuna olan uzaklığı gibi dikkate alınır. A kullanıcısı ile X grubu arasındaki ölçüm bilgisi, A kullanıcısının bulunduğu grubun tüm diğer kullanıcılarına gönderilir. Oluşturulan üstüne bindirmeli yapının doğasına bağlı olarak, grup içerisinde paylaşılan bilgi selbasma tekniği kullanılarak gönderilir. Dördüncü olarak incelenen protokol bilgi güncelleme protokolüdür. Bilgi güncelleme protokolünde grup içerisindeki her kullanıcı, bilgi güncelleme için dağıtık bir algoritma çalıştırır. Güncelleme işlemini tamamlamak için yerel kullanıcının önbelleği kullanılır. Son olarak kullanıcının sistemden ayrılma protokolüne değinelim. Oluşturulan mimaride, kullanıcı üstüne bindirmeli ağı terk ederken yapması gereken herhangi bir işlem yoktur. İstediği zaman sistemi terk edebilir. Ancak sistem performansını arttırmak için, kullanıcı sistemi terk ettiği zaman grup liderini bilgilendirmesi gerekmektedir (Zhang et al., 2004).

2.5.5 Nemo üstüne bindirmeli ağ modeli

Çoklu gönderim, grup iletişimi kullanan uygulamaları (örneğin sesli ve görüntülü konferans, içerik dağıtımı ve medya akışlandırma) desteklemek için verimli bir mekanizmadır. Çoklu gönderim tekniği ile ağ üzerindeki gereksiz trafiğin önüne geçilir (Birrer and Bustamante, 2005).

Tasarlanan NEMO protokolü, iki önemli tekniğe dayanmaktadır. Birinci teknik, yardımcı liderler (co-leaders) kullanarak bağımlılıkları ortadan kaldırmaktır. İkinci teknik ise olumsuz teyit alındı (negative acknowledgement) mesajı tetiklenerek kayıp paketleri tespit etmektir. NEMO tasarımı, basitliği ve minimum seviyede bağımlılığı kullanan bir protokoldür. Protokolün amacı, belirli gecikme sınırları içerisinde yüksek gönderim oranı gerektiren uygulamaların, mükemmel güvenilirliğe ihtiyaç duymadan verimli şekilde çalışmasını sağlamaktır. Yardımcı liderler, sistemde hata tespit edilinceye kadar verinin iletiminden sorumludurlar. Olasılıksal yaklaşım kullanarak oluşturulan üstüne bindirmeli yapının sürdürülebilirlik maliyeti azaltılır (Birrer and Bustamante, 2005).

Tasarlanan protokol, çoklu gönderim için eşlerin ağ yakınlığına göre kümelenmeleri sistemine dayanmaktadır. Bütün eşler en alt katmanın üyesidir. Kümelerin eş sayıları d ve $3d - 1$ ($d \in Z^+$) arasında değişmektedir. Protokolün başlangıç aşamasında d değeri sabit bir değer olarak seçilmektedir ve protokolün çalışması esnasında değişmemektedir (Birrer and Bustamante, 2005). Kümelerin her birinde küme lideri seçilir ve bu küme lideri üst katmanlara doğru taşınır. Tek bir düğüme bağlılığı engellemek için küme lideri, yardımcı liderlerin sayısını düzenleyerek kendi takımını (crew) oluşturur. Bu işlem bütün eşlerin kümeler içerisine yerleşinceye kadar devam eder. Mantıksal hiyerarşi kurularak eşlerin birden çok kümede bulunması sağlanır (Birrer and Bustamante, 2005). Şekil 2.22'de NEMO protokolünün mantıksal yapısı gösterilmiştir.



Şekil 2.22. Nemo protokolünün mantıksal yapısı (Birrer and Bustamante, 2005 – Figure 1).

NEMO yapısında grupların yönetimini ve kümelerin dinamikliğini inceleyelim. Sisteme yeni bir eş katılacağı zaman ilk önce iyi bilinen bir düğüm olan RP ile iletişime geçilir. RP, kendisiyle iletişime geçen yeni eşe, en üst katmandaki üyelerin bilgisini gönderir. Bu bilgiyi alan yeni eş, tek tek üyeler ile ağ yakınlığına bakar ve kendisine en yakın olan üyeyi belirler. Kendisine en yakın olan üyeden, alt seviyede bulunan üyelerin bilgisini alır. Bu işlemler yeni eşin en alt seviyede kendisine en uygun kümeyi buluncaya kadar tekrarlanır (Birrer and Bustamante, 2005).

Protokol içerisinde, üyeler sistemden haberli veya habersiz şekilde ayrılabilirler. Normal küme üyeleri, bulundukları kümenin liderini bilgilendirerek kümeden ayrılabilirler. Bu üyelerin herhangi bir sorumluluğu yoktur. Küme lideri, sistemden ayrılmadan önce bulunduğu tüm kümeler için yerine geçecek bir lider

üye seçer. Tasarlanan protokol, habersiz şekilde sistemden ayrılmaları tespit etmek için, küme içerisindeki üyelere belirli aralıklarla mesaj gönderir. Eğer küme içerisindeki üyelerden mesaja cevap gelmez ise ilgili üyenin sistemi terk ettiği düşünülür. Sistemdeki dinamik değişimlere değinmek gerekirse, her üye belirli aralıklarla bir üst katmandaki küme liderleriyle olan ağ yakınlıklarına bakar ve eğer kendisine daha yakın başka bir küme lideri tespit ederse, yakın olan küme liderinin bulunduğu kümeye geçer (Birrer and Bustamante, 2005).

Küme içerisindeki üyeliklerin devamlı olarak değişmesinden dolayı, kümeler büyüyüp küçülür. Küme içerisindeki üye sayısı belirlenen değerden az ise küme birleşimi gerçekleştirilir. Eğer küme içerisindeki eleman sayısı belirlenen değerden fazla ise küme bölünmesi gerçekleştirilir. Küme bölünmesi için genetik algoritma kullanılmıştır. Küme birleşimi ve küme bölünmesi işlemleri, küme lideri tarafından kontrol edilir. Küme bölünmesi sonucu oluşan iki yeni kümenin eleman sayısı en az 3d / 2 kadardır. Küme bölünmesi, oluşan iki yeni kümenin çapı minimum olacak şekilde gerçekleştirilir (Birrer and Bustamante, 2005).

Küme bölünmesi ve küme birleşimi işlemleri her bir giriş çıkış işlemlerinde kontrol edilir. Bu kontrol de küme eleman sayısına bakılarak yapılmaktadır. NEMO protokolü, sistemin sürdürülebilirliğini sağlamak için küme bölünmesi ve küme birleşimi işlemlerini olasılıksal yaklaşım kullanarak yapar. Yani kümenin eleman sayısı, belirtilen değeri aştığında her zaman küme bölünmesi gerçekleştirilmez (Birrer and Bustamante, 2005).

2.5.6 Canicula üstüne bindirmeli ağ modeli

Karma üstüne bindirmeli ağlar; yapısal üstüne bindirmeli yapıların basitliği ve ölçeklenebilirliği, yapısal olmayan üstüne bindirmeli yapıların esnekliği özelliklerinin birleşiminden oluşurlar (Ruixiong et al., 2005). Tasarlanan CANICULA protokolü, gelişmiş karma üstüne bindirmeli ağları kullanır (Chen, 2006).

Karma üstüne bindirmeli ağlarda, düğüm mesafesi tahmin sistemi tam doğru olarak çalışmaz. Bu protokolde, üçgenlere bölme yaklaşımı (triangulated heuristic) kullanılarak, ağ mesafesi tahminlemesi yapılmıştır (Ng and Zhang, 2002). Daha sonra tahmin edilen ağ yakınlığına göre düğümlerin yapısal kümeler içerisine yerleştirilmesi sağlanmıştır (Chen, 2006).

Tasarlanan sistemde düğümün ağdaki konumunu belirlemek için basit koordinat sistemi kullanılmıştır. Bu koordinat sisteminin basitliği, her bir düğümün sistem üzerinde iyi bilinen işaretlenmiş olan düğümlere olan RTT mesafesini ölçmesi sistemine dayanmaktadır. Ağ koordinat bilgisi alındıktan sonra iki düğüm arasındaki mesafe tahmin edilir. İki düğüm arasındaki mesafe bilgisi alındıktan sonra düğümler kümelenir.

Üstüne bindirmeli yapıya katılmak isteyen yeni düğüm ilk önce ağ üzerindeki koordinatlarını tanımlar. Daha sonra üçgenlere bölme yaklaşımı kullanılarak en yakın küme bulunur. Tasarlanan yapıda yeni düğüm eğer kendisine en yakın kümeyi bulamazsa veya en yakın kümeye olan mesafesi belirlenen eşik değerinden büyükse, yeni düğüm omurga (backbone) ağına katılır ve koordinatlarını sisteme bildirir. Eğer yeni düğüm kendisine en yakın kümeyi bulursa o kümeye katılır. Protokolün dağıtık şekilde tasarlanmasından dolayı bütün düğümler arasında uçtan uca ikili ölçüm yapılmasına gerek yoktur. Bu şekilde büyük ölçekli ağlarda maliyet azaltılmış ve ölçeklenebilirlik sağlanmış olur (Chen, 2006).

Literatürde yer alan NARADA, NICE, ZIGZAG, mOVERLAY, NEMO ve CANICULA üstüne bindirmeli yapılarının birçok eksik yönleri bulunmaktadır. Sırasıyla bu eksik yönleri inceleyelim.

NARADA yapısında grup içerisindeki üye sayısının sınırlandırılması sistemde bulunmamaktadır. Gruplarda fazla sayıda üye varsa ilgili grubun bölünmesinin grup dengesini sağlayacağı ve grup içi mesajlaşmayı azaltacağı düşünülmemiştir. Sistem kurulduktan sonra grup içerisinde az sayıda üye barındıran grupların birleştirilmesi konusuna da değinilmemiştir. Grup içerisinde herhangi bir lider üye seçimi yapılması konusu da incelenmemiştir. Son olarak

kümeleme hassasiyeti ve kümeleme doğruluğuna bakılmamıştır. Tasarlanan protokol, daha küçük boyutlu üstüne bindirmeli yapılar için uygun olup, sistem daha büyüdüğünde ölçeklenebilirlik sorunları ortaya çıkabilir.

NICE yapısında küme bölünmesi için kullanılan yaklaşım algoritmasının polinom zamanda çözülemeyen bir problemden yola çıkılarak yapıldığı anlatılmaktadır. Kullanılan bu yaklaşım, mesaj karmaşıklığının çok fazla olduğunu ortaya çıkarmaktadır. Sisteme dahil olmak isteyen bir üyenin hangi kümeye daha yakın olduğunu belirlemek için en üst katmandan en alt katmana kadar sürekli mesajlaşma ile gecikme mesafesini ölçmesi gerekmektedir. Sistemdeki üye sayısı çok fazla olduğu durumlarda mesaj karmaşıklığı çok yüksek olacaktır. Küme birleşimi yöntemi sonunda kümenin iç dinamiği değiştiğinden küme lideri seçiminin yeniden yapılması gerekmektedir. Küme lideri sistemden ayrıldığında küme içerisindeki tüm üyeler birbirleriyle mesajlaşma yaparak birbirlerinden bağımsız çoklu liderler seçmektedir. Küme lideri sistemden ayrıldığında sürekli yeniden küme lideri seçimi algoritmasını çalıştırmak maliyet açısından istenmeyen bir durumdur. Son olarak önerdikleri protokolün yapısal olması da görevdeş ağlar için çok uygun değildir. Çünkü görevdeş ağlarda ağın yapısı dinamik olarak sürekli değişmektedir. Sistemin kümeleme hassasiyetine ve kümeleme doğruluğuna bakılmamıştır.

ZIGZAG yapısında bazı küme bölünmesi ve küme birleşimi için kullanılan yaklaşımların çok çeşitli durumlarda farklı şekillerde çalışması dağıtık şekilde tasarlanan yapı hakkında bazı eksiklikler olduğunu göstermektedir. Küme içerisinde herhangi bir üyenin küme lideri seçilebileceği anlaşılmaktadır. Küme liderinin, kümenin merkezinde olması beklenmektedir. Benzer şekilde yardımcı liderin de rastgele bir üyenin yerine merkeze yakın olan ikinci üyenin seçilmesi küme içerisinde diğer üyelere yakınlık açısından önemlidir. Çünkü yardımcı liderin bu sistemdeki görevi, küme içerisindeki diğer üyelere medya akışını sağlamak olduğu için yardımcı liderin, küme içerisindeki diğer üyelere yakın olması beklenmektedir. Küme birleşiminde kullanılan yaklaşımda sadece küme içerisindeki eleman sayısının eşik değerden az olup olmadığına bakılmıştır. Küme birleşiminin meydana geleceği iki kümenin liderleri arasındaki gecikme parametresi dikkate alınmamıştır. Yani birbirine gecikme metriği olarak uzak olan

üyeler, kümelerin birleşmesi sonucu aynı küme içerisine girmektedirler. Bu da kümeleme açısından birbirine gecikme olarak yakın olan üyelerin aynı kümeye girmeleri gerektiği prensibine aykırı bir durum olarak ortaya çıkmaktadır. Küme içerisinden küme lideri ayrıldığı zaman küme içerisinde yeniden küme lideri seçilmektedir. Sisteme giriş çıkış sayısının fazla olduğu P2P sistemlerinde her seferinde yeniden küme lideri seçimi işlemini başlatmak maliyet açısından istenmeyen bir durumdur. Bunun yerine küme içerisindeki üyelik ilişkilerini bilen başka bir üye yani yedek lider seçilebilir. Son olarak tasarlanan yapının kümeleme hassasiyetine ve kümeleme doğruluğuna bakılmamıştır.

mOVERLAY yapısında öncelikle oluşturulan yapının düz bir yapı olması sisteme sürekli giriş çıkış yapılan P2P ağlar için çok uygun değildir. Yeni bir kullanıcı sisteme girerken kendisine en yakın grubu bulmak için birçok grubu gezmektedir. Bu gezme işlemi mesaj karmaşıklığının artmasına neden olmaktadır. Aynı zamanda gruplama kriterinin ne zaman bitirilmesi gerektiği de önemli bir sıkıntı olarak karşımıza çıkmaktadır. Kullanılan gruplama kriterindeki çok büyük bir sorun ise birbirine çok uzak olan iki kullanıcının girmek istedikleri gruba sadece mesafeleri aynı olduğu için ilgili grupta yer almalarıdır. Grup içerisindeki kullanıcıların birbirine uzak olması grup içerisindeki iletişimin verimliliğini düşürecektir. Grup içerisindeki kullanıcıların sayısı arttıkça grup bölünmesi işlemi düşünülmemiştir. Aynı şekilde grup içerisindeki kullanıcı sayısı az ise grupların birleştirilmesi için bir protokol tasarlanmamıştır. Grup içerisinde lider seçiminin nasıl yapıldığı konusu belli değildir. Herhangi bir kullanıcı liderlik görevinden sorumludur. Lider olan kullanıcının grubun merkezinde seçilmesi bu tip sistemlerde çok önemlidir. Çünkü merkezde seçilen lider, grup içerisindeki kullanıcıların sürdürülebilirliğinden sorumlu olduğu için, grup içi iletişim maliyetinin azaltılmasını sağlar.

NEMO yapısında küme liderlerinin nasıl seçildiğine değinilmemiştir. Küme liderinin sistemden ayrılması durumunda yerine geçecek olan liderin seçilme kriterinin ne olduğu açıklanmamıştır. Küme bölünmesi için kullanılan genetik algoritmanın karmaşıklığı $O(cn^2)$ olarak belirtilmektedir. Bu da kullanılan algoritmanın maliyetinin çok yüksek olduğunu göstermektedir. Küme birleşiminde sadece kümelerin eleman sayıları dikkate alınmıştır. Oysa ki küme

birleşiminin yaşanacağı iki kümenin liderleri arasındaki mesafe de dikkate alınmalıdır. Eğer mesafe dikkate alınmazsa birbirine uzak olan üyeler aynı küme içerisinde bulunacaklardır. Son olarak sistemin kümeleme hassasiyetine ve kümeleme doğruluğuna bakılmamıştır.

CANICULA yapısında öncelikle kurulan üstüne bindirmeli mimarinin yapısal olması P2P ağlar için çok uygun değildir. Ağ üzerindeki mesafeyi tahmin etmek için kullanılan üçgenlere bölme yaklaşımının sisteme getireceği ek yük dikkate alınmamıştır. Ağ üzerinde iyi bilinen düğümlerin işaretlenerek o düğümlere olan RTT mesafesi ölçümü yapılmıştır. Fakat P2P sistemlerde sisteme katılan düğümlerin sistemdeki kullanıcılar hakkında bilgisi yoktur. Düğümler kısıtlı bilgi kullanarak sisteme dahil olmaktadırlar. Oluşturulan kümeleme yapısı hiyerarşik değildir. Kümelerin içerisindeki düğüm sayılarının fazla veya az olması durumunda kümelerde yapılabilecek bölünme ve birleşim işlemleri düşünülmemiştir. Küme lideri kavramı yoktur. Sistemin kümeleme hassasiyetine ve kümeleme doğruluğuna bakılmamıştır.

Literatürdeki NARADA, NICE, ZIGZAG, mOVERLAY, NEMO ve CANICULA üstüne bindirmeli yapıları arasındaki farklar Tablo 2.1'de verilmiştir.

	Narada	Nice	Zigzag	mOverlay	Nemo	Canicula
Yapı	*Hiyerarşik değil ve yapısal*	*Hiyerarşik ve yapısal*	*Hiyerarşik ve yapısal*	*Hiyerarşik değil ve yapısal*	*Hiyerarşik ve yapısal*	*Hiyerarşik değil ve yapısal*
Lider Seçimi	*Yok*	*Var (lider, merkezde)*	*Var (herhangi bir üye)*	*Var (herhangi bir üye)*	*Var (herhangi bir üye)*	*Yok*
Küme Bölünmesi	*Yok*	*Var*	*Var*	*Yok*	*Var*	*Yok*
Küme Birleşmesi	*Yok*	*Var*	*Var*	*Yok*	*Var*	*Yok*
Üyenin ayrılması	*Var (haberli-habersiz)*	*Var (haberli-habersiz)*	*Var (haberli-habersiz)*	*Var (haberli-habersiz)*	*Var(haberli-habersiz)*	*Yok*

Tablo 2.1. Literatürdeki üstüne bindirmeli yapılar arasındaki farklar tablosu.

3. TASARLANAN ÜSTÜNE BİNDİRMELİ HİYERARŞİK KÜMELEME YAPISI

Tezin bu bölümünde daha önceki literatür özeti bölümünde açıklanan üstüne bindirmeli yapıların eksik kalan kısımlarından yola çıkılarak geliştirilen sistemin detayları anlatılacaktır. Sırasıyla ilk önce hiyerarşik mantıksal ağ mimarisine, ikinci olarak tasarlanan ağ mimarisi içerisindeki dağıtık kümeleme mantığına ve ölçeklenebilirliği sağlamak için gerekli olan mekanizmalara ve son olarak da sistemin hatalara karşı nasıl dirençli hale getirildiğine değinilecektir.

3.1 Hiyerarşik Mantıksal Ağ Oluşumu

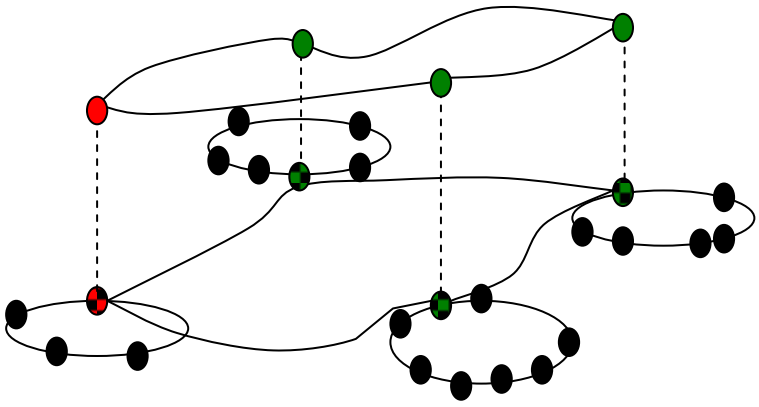
Görevdeş ağ üzerinde oluşturulacak olan hiyerarşik mantıksal ağ, video aranmasında ve akışlandırmasında büyük önem taşımaktadır. P2P çalışacak bir video akışlandırma sistemi için en çok çalışılan yöntem, çoklu gönderim ağaçları oluşturarak kaynak aracılığı ile videonun istenen düğümlere ulaştırılmasıdır. Bu tip yöntemde, akışlandırma esnasında kaynak ya da hedef ile kaynak arasındaki herhangi bir düğümün herhangi bir şekilde oluşan hatadan dolayı, sistemden ayrılmasıyla oluşabilecek aksaklıkları engellemek için, aynı düğümleri içeren birden fazla çoklu gönderim ağacı kullanan yapılar bulunmaktadır (Sayıt et al., 2009).

Ağaç yapısının kullanıldığı sistemlerde kaynak görevini gören düğümlerin sistemden bir hata sonucu ayrılması ya da fazla sayıda düğümün bu düğümle iletişime geçmesi problem yaratabilmektedir. Ağaç üzerinden video akışlandırmanın getirdiği bu tip dezavantajlar için öne sürülen bir diğer yöntem kümelerin kullanılmasıdır. Kümeler kullanılarak gerçekleştirilen video akışlandırma sistemlerinde, düğümlerin topolojik konumlarına, sistemde bulunan düğüm sayısına ya da komşuluk durumlarına göre farklı yollarla kümeleme yapılmaktadır.

Tasarlanan yapıda kümeleme yapmak için gecikme ölçütü göz önüne alınmıştır. mOverlay yapısı, iki düğüm arası gecikme ölçütüne dayandırılan ve basitliği açısından öne çıkan bir sistemdir. Bu yapı basitliği açısından öne

çıkmasına karşın algoritmada belirsiz noktalar bulunmaktadır. Literatürdeki mOverlay yapısının en önemli eksikliklerinden birisi hiyerarşik bir katman yapısı kullanmamasıdır.

Tasarlanan yapı, TÜBİTAK'ın "108M411" numaralı "Görevdeş Ağlarda Ölçeklenebilir Gruplandırma ve Video Akışlandırma" isimli projesi kapsamında, birbirine gecikme ölçütü olarak yakın düğümlerden oluşan kümeler ve her küme liderinin oluşturduğu üst hiyerarşik seviyedeki kümelerden oluşmaktadır. Önerilen yapıda her düğüm ve her lider farklı hiyerarşik kümede yer alsa da aynı algoritmayı çalıştırmaktadır. Sistemde en fazla N adet düğümden oluşan kümelerde, bir düğüm lider görevini görmektedir. Birbirine komşu M adet kümenin liderleri ise hiyerarşik yapıda bir üst kümeyi oluştururlar ve içlerinden bir tanesi liderlerin lideri olarak görev yapmaktadır. Yeni bir hiyerarşik katmanın sisteme eklenmesi için bazı koşulların gerçekleşmesi gerekmektedir. Bu koşullar; bir katmanda bulunan küme içerisindeki düğüm sayısının belirli bir değeri aşması ve ikincisi ise bir katmanda en az bir küme olduğu zaman, bu kümeye katılmak için gecikme ölçümü yapan düğümün bu kümeye giremeyip ayrı bir küme oluşturması şeklinde açıklanabilir. Şekil 3.1'de iki katmanlı örnek bir hiyerarşik yapı gösterilmiştir. Şekilde yeşil renkte görünen düğümler en alt katmanda bulunan kümelerin lider düğümlerini temsil etmektedir. Yeşil düğümler, yani en alt hiyerarşik katmanın lider düğümleri, bir üst katmanda bir küme oluşturmaktadır. Kırmızı renkte belirtilen lider bu üst kümenin lideridir. Bu yapıda, kırmızı renkli düğüm hem en alt seviyedeki hiyerarşik katmanda hem de bir üst seviyedeki hiyerarşik katmanda lider görevindedir (Sayıt, 2010).



Şekil 3.1. İki katmanlı örnek bir hiyerarşik yapı.

Tasarlanan bu hiyerarşik yapı, literatürde bulunan NICE, ZIGZAG ve NEMO yapılarında oluşturulan yapıya benzemektedir. Ancak farklı yönleri vardır. Literatürdeki bu yapılarda küme içerisinde bulunan üyeler, küme içerisindeki diğer üyeleri bilmektedir. Önerilen yapıda ise küme içerisindeki üyeler sadece küme liderini bilmektedirler, bu da önerdiğimiz yapıdaki üyelerin daha sınırlı bilgiye sahip oldukları anlamına gelmektedir. Küme içerisindeki üyelerin birbirleri hakkında bilgiye sahip olması dağıtık sistemler açısından çok uygun değildir. Üyelerin küme içerisinde birbirlerinden haberdar olmaları, mesaj karmaşıklığını azaltır fakat dağıtık olarak çalışması uygun olan görevdeş ağlar için çok uygun değildir.

3.2 Hiyerarşik Yapının Devamlılığını Sağlayan Protokoller

Tasarlanan dağıtık P2P sisteminin, değişen düğüm yapısı ve sayısına bağlı olarak, dinamik ve periyodik aralıklarla kendini düzenlemesi gerekmektedir. Bu sistemi yapılandıran ve devamlılığını sağlayan protokoller bu bölümde açıklanacaktır.

3.2.1 Sisteme yeni bir düğümün katılması

Sisteme katılmak isteyen bir düğüm ilk olarak, adresini her düğümün bildiği bir sunucuya başvurur. Bu sunucuya randevu noktası adı verilir. Sistemde henüz bir küme yoksa sisteme katılmak istediğini bildiren düğüm, randevu noktasından, tutulan bir küme bilgisi olmadığı cevabını alır. Bu düğüm, en alt katmanda küme lideri olarak sisteme katılmış olur ve randevu noktasına küme numarasını bildirir. Eğer sistemde çok katmanlı bir yapı kurulmuş ise sisteme katılmak için randevu noktasına başvuran düğüm, randevu noktasından en üst seviyede bulunan küme liderinin bilgisini alır. Sisteme katılmak isteyen düğüm, en üst seviyedeki küme liderinin bulunduğu küme içerisindeki tüm temsilci küme liderleri ile RTT mesafesini ölçer. Bu mesafelerden en yakın üst lider belirlendikten sonra sisteme girecek olan düğüm bir alt katmanda hangi kümeye dahil olacağını belirlemiş olur. Algoritma kendini her katmanda tekrar ederek, her seviyede sisteme katılacak olan düğüm, ilgili katmandaki temsilci küme liderlerinin bulunduğu küme liderleri arasında ölçüm yapar ve kendisine RTT mesafesi olarak en yakın

kümeyi seçerek alt katmana geçer. Bu algoritma, sisteme dahil olacak düğümün fiziksel olarak en alt seviyede katılacağı kümenin belirlenmesi ile tamamlanmış olur. Randevu noktasının herkes tarafından bilinen bir düğüm olması ve her düğümün sisteme katılırken randevu noktasına başvurması randevu noktasında darboğaz oluşmasına neden olabilir. Bu yüzden tasarlanan sistem test edilirken düğümlerin randevu noktasına aynı anda değil belirli bir zaman sonra başvurmasını sağladık. Ayrıca randevu noktalarını birden fazla seçerek yani yedek randevu noktaları belirleyerek, randevu noktasında oluşabilecek tek nokta hatasının önüne geçmiş olduk. Düğümün herhangi bir kümeye dahil olması için göz önünde tutulan kriter RTT gecikme ölçütü ve katılacağı kümenin bulunduğu seviyenin eşik değeridir. Yeni bir düğümün sisteme katılması için tasarlanan yapı, literatürde bulunan NICE ve NEMO yapıları ile benzerdir. Tek fark bu yapılarda randevu noktası, en üst hiyerarşide bulunan küme içerisindeki tüm düğüm listesini bilmektedir. Önerilen yapıda ise randevu noktası, en üst hiyerarşide bulunan küme liderini bilmektedir. Sisteme katılmak isteyen yeni bir düğümün çalıştırdığı algoritma Şekil 3.2'de verilmiştir.

```
Join_system()
//Bu rutin, sisteme ilk kez giriş yapan düğüm tarafından çalıştırılır

send_msg(RP, JOIN); //RP: Randevu noktası
Leader1 = recv_msg(RP); //en üst hiyerarşik katmanın lider adresi

if (first_node_in_the_system)
        construct_new_hierarcy_layer(Leader1=me, Cluster=C1);
else {
        for all hierarchy layers Hi i=n:max hierarchy layer to 0
                {
                send_msg(Leaderi);
                Ci_Node_List = recv_msg(Leaderi);
                measure_RTT(C1_Node_List);
                Leaderi = min_RTT(Ci_Node_List);
                }
        join_cluster(Ci); //en alt katmandaki en yakın küme
        Ci_Node_List = recv_msg(Leaderi);
        measure_RTT(Ci_Node_List); //ölçüm değerleri tablosu oluşturulur
    }
```

Şekil 3.2. Sisteme katılmak isteyen yeni bir düğümün çalıştırdığı algoritma.

Şekil 3.3'te küme lideri tarafından, kümeye yeni bir düğüm girdiğinde çalıştırılan algoritma verilmiştir.

```
Add_new_node_to_cluster():
//Bu rutin, küme lideri tarafından, kümeye yeni bir düğüm girdiğinde  çalıştırılır

k: my_cluster_id;
send_msg(node, Ck_Node_List);

//küme bölünmesi
if (no_of_cluster_members_exceeds_threshold){
        send_measure_msg(Furthest_Node_List);//RTT  mesafesine göre en
                                                    uzak üç düğüme gönderilir.
        (1)RTT_for_nodei = recv_msg(Furthest_Node_List);
        (2)furthest = choose_furthest_node_having_max_RTT();
        send_msg(furthest,  CLUSTER_PARTITIONING);
        for all nodei ∈ Ck_Node_List
                send_msg(nodei ,JOIN_MY_CLUSTER);

        Ck_Node_List = ∅; //küme listesi sıfırlanır

        new_node = source(recv_msg(ACCEPT));
        Ck_Node_List = Ck_Node_List ∪ new_node;
        send_msg(new_node,  Ck_Node_List);
}

if (upper_hierarchical_layer_does_not_exist){
        construct_new_hierarcy_layer(Leadern=me,  Cluster=Cn);
        Cn_Node_List = Cn_Node_List ∪ furthest;
}
```

(1) Küme içindeki lider, kendi kümesi içinde bulunan diğer düğümler ile RTT cinsinden uzaklığını ölçer ve küme liderine en uzak olan üç düğüm belirlenir.

(2) Bu mesajı alan üç düğüm, birbirleri ile RTT mesafelerini ölçerek sonuçları küme liderine iletirler. Lider aldığı bu mesajlar içerisindeki RTT mesafelerini büyükten küçüğe doğru sıralar. Sıralama sonucunda aralarında en uzak mesafenin olduğu iki düğümden küme liderine en uzak olan düğüm, küme bölünmesi sonucu oluşacak olan yeni kümenin lideri olur.

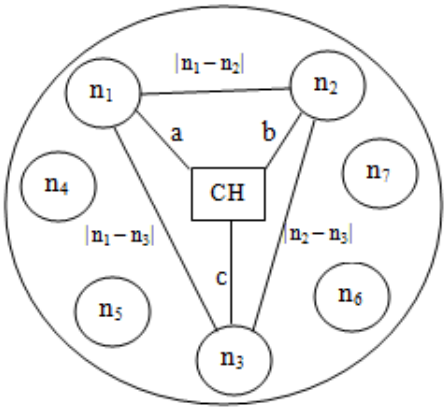
Şekil 3.3. Küme lideri tarafından kümeye yeni bir düğüm girdiğinde çalıştırılan algoritma.

3.2.2 Küme bölünmesi

Küme içerisindeki düğüm sayısı daha önceden belirlenmiş bir eşik değeri geçerse, küme lideri bunu farkeder ve küme bölünme algoritmasını çalıştırır. Bu algoritma çalışmaya başladığı zaman ilk olarak eşik değerinin geçildiği küme içerisindeki lider, kendi kümesi içerisinde bulunan diğer düğümler ile RTT uzaklığını ölçer ve küme liderine en uzak olan üç düğüm belirlenir. Daha sonra küme lideri, kendisine en uzak olan bu üç düğüme kendi aralarında RTT mesafelerini ölçmeleri gerektiğini bildirir. Bu mesajı alan üç düğüm, birbirleri ile RTT mesafelerini ölçerek sonuçları küme liderine iletirler. Lider aldığı bu mesajlar içerisindeki RTT mesafelerini büyükten küçüğe doğru sıralar. Sıralama sonucunda aralarında en uzak mesafenin olduğu iki düğümden küme liderine en uzak olan düğüm, küme bölünmesi sonucu oluşacak olan yeni kümenin lideri olur. Diğer düğüm ise, küme içerisindeki bahsedilen üç düğüm dışındaki diğer düğümlerin bulundukları kümede kalıp kalmayacaklarını belirlemek için seçilir. Bu düğüme, küme bölünmesi için kullanılan referans düğüm adı verilir. Diğer üyeler gibi sıradan bir düğümdür. Bu işlem sonucunda küme lideri, küme

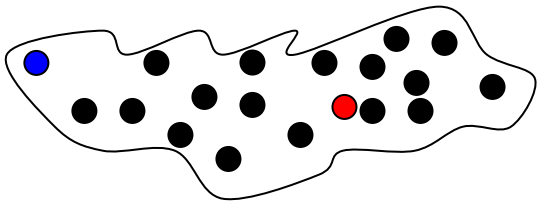
içerisindeki tüm diğer düğümlere, hem yeni kümenin küme lideri ile hem de bulundukları kümenin referans düğümü ile RTT mesafesi ölçmeleri gerektiğini bildirir. Bu mesajı alan düğümler hem yeni küme lideri hem de referans düğüm ile RTT mesafelerini ölçerler. Ölçüm sonucunda ölçüm gerçekleştiren düğüm, eğer referans düğüme daha yakın ise bulunduğu kümede kalmaya devam eder aksi taktirde yeni küme liderine daha yakın ise oluşacak yeni kümeye katılmaya karar verir ve bu kararını eski küme liderine bildirir. Ayrıca yeni bir küme oluştuğu için de bu küme lideri bir üst katmanda temsil edilecektir. Ölçülen RTT değerlerinin eşit çıkması durumunda ise düğüm bulunduğu kümede kalmaya devam eder. Şekil 3.4'te önerilen yapı üzerinde küme bölünmesi için örnek bir koşul gösterilmiştir. Şekilde ölçülen RTT değerlerinin sıralamasının $c > a > b$ ve $|n_1 - n_3| > |n_2 - n_3| > |n_1 - n_2|$ şeklinde olduğunu varsayarsak, yeni oluşacak küme lideri n_3 ve küme bölünmesinin meydana geldiği kümenin referans düğümü n_1 olacaktır. Şekilde CH, küme liderini; n_1, n_2, n_3, n_4, n_5, n_6 ve n_7 ise küme içerisindeki düğümleri temsil etmektedir.

Literatürde bulunan NICE, ZIGZAG ve NEMO yapılarında tasarlanan küme bölünme algoritmalarının karmaşık olmasından dolayı yeni bir küme bölünme algoritması tasarlanmıştır. NICE protokolünde polinom zamanda çözülemeyen bir problem olan K merkez probleminden yola çıkılarak küme bölünmesi yapılmıştır. Bu da NICE yapısında çalıştırılan küme bölünme algoritmasının mesaj karmaşıklığının yüksek olduğunu göstermektedir. ZIGZAG yapısında ise küme bölünmesi için kullanılan protokolde, küme içerisindeki üyelerin RTT değerleri azalan sıraya göre sıralanır. Sıralama sonucu oluşan liste ikiye bölünür. Listenin bir yarısı bulundukları kümede kalır, diğer yarısı ise diğer kümeye geçerler. Kullanılan bu küme bölünmesi yaklaşımı da verimli bir küme bölünmesi oluşturmamaktadır ve üyelerin küme içerisinde RTT ölçmek için devamlı mesajlaşmalarına neden olmaktadır. Bu da mesaj karmaşıklığının her küme bölünmesinde çok fazla olacağını göstermektedir. NEMO yapısında kullanılan küme bölünme yaklaşımında kullanılan genetik algoritmanın karmaşıklığı $O(cn^2)$ olarak belirtilmektedir. Bu da kullanılan algoritmanın maliyetinin çok yüksek olduğunu göstermektedir. Tez kapsamında önerilen yapıda ise küme içerisindeki üyelerin sayısı ne olursa olsun, aynı algoritma çalışmaktadır. Küme içerisindeki üyeler arasında ikili RTT ölçümü yapılmayarak mesaj karmaşıklığı azaltılmıştır.



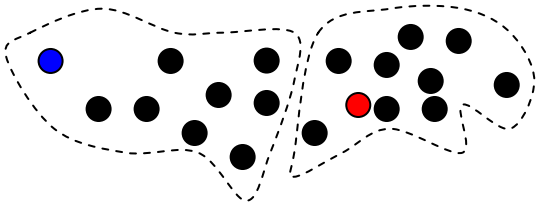
Şekil 3.4. Önerilen yapı üzerinde küme bölünmesi için örnek bir koşul.

Şekil 3.5'te küme eleman sayısı eşik değerini geçen bir küme verilmiştir. Kırmızı renkle belirtilen düğüm küme liderini, mavi renkle belirtilen düğüm ise yeni küme liderini temsil etmektedir.



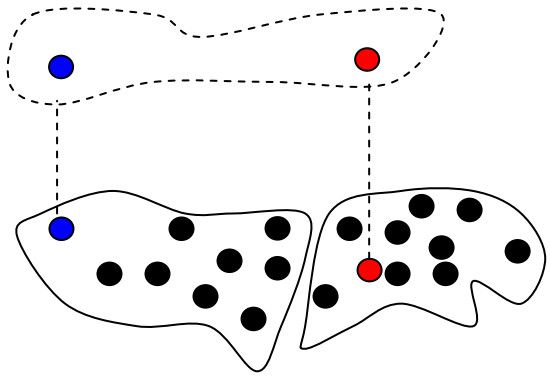
Şekil 3.5. Küme eleman sayısı eşiğini geçen bir küme.

Küme bölünmesi gerçekleştikten sonra oluşan iki yeni küme Şekil 3.6'da gösterilmiştir.



Şekil 3.6. Küme bölünmesinden sonra oluşan iki yeni küme.

Bölünme sonucunda oluşan kümelerin lider düğümleri, bir üst katmanda temsilci düğüm olarak yer alırlar. Bu iki kümenin oluşturduğu bir üst hiyerarşik katman ise Şekil 3.7'de verilmiştir.



Şekil 3.7. Küme bölünmesinden sonra oluşan yeni hiyerarşik katman.

Şekil 3.8'de küme bölünme mesajını alan düğümün çalıştırdığı algoritma verilmiştir.

```
Construct_New_Cluster():
//Bu rutin, CLUSTER_PARTITIONING mesajı alan düğüm tarafından çalıştırılır

k: my_cluster_id;

Ck_Node_List = ∅; //grup listesi sıfırlanır
for all nodei ∈ Ck_Node_List
        send_msg(nodei,JOIN_MY_CLUSTER);
new_node = source(recv_msg(ACCEPT));
Ck_Node_List = Ck_Node_List ∪ new_node;
send_msg(new_node, Ck_Node_List);

if (upper_hierarchical_layer_exist)
        join_cluster(Cupper_layer);
        Cupper_layer_Node_List = recv_msg(Leaderupper_layer);
        measure_RTT(Cupper_layer_Node_List);
```

Şekil 3.8. Küme bölünme mesajını alan düğümün çalıştırdığı algoritma.

Şekil 3.9'da küme içerisindeki düğümlerin, küme bölünmesi sonucu oluşan yeni küme liderinden aldıkları kümeye katılma mesajına karşılık çalıştırdığı algoritma verilmiştir.

```
Join_New_Cluster():
//Bu rutin, JOIN_MY_CLUSTER mesajı alan düğüm tarafından çalıştırılır

k: my_cluster_id;

if (not_invited_before){
        send_msg(source(recv_msg(JOIN_MY_CLUSTER)), ACCEPT);
        Ck_Node_List = recv_msg(Leaderk);
        measure_RTT(Ck_Node_List);
}
```

Şekil 3.9. Küme içerisindeki düğümlerin küme bölünmesi sonucu oluşan yeni küme liderinden aldıkları kümeye katılma mesajına karşılık çalıştırdığı algoritma.

3.2.3 Küme birleşmesi

Küme içerisindeki düğümlerin sayısı belirli bir sayının altına düşerse küme birleştirme algoritması en alt seviyeden en üst seviyeye doğru olacak şekilde çalıştırılır. Küme birleştirme algoritması şu şekilde çalışmaktadır. Çok katmanlı sistem yapısı kurulduktan sonra belirlenen düğüm sayısının altında kalan kümeler için algoritma çalıştırılır. En alt seviyede bulunan kümeler içerisindeki düğüm sayıları daha düzenli dağılım gösterdiği için belirlenen minimum küme düğüm sayısı ile üst katmanlardaki kümenin düğüm sayıları birbirinden farklıdır. Birçok test yapılarak bu değerler en uygun olacak şekilde seçilmiştir. İlk önce birleştirme işlemi için uygun olan kümeler ve kaç tane düğüme sahip oldukları belirlenir. Bu işlem bir üst katmandaki lider (ebeveyn) tarafından yapılır. Daha sonra bu kümeler en az düğüm sayısından en çok düğüm sayısına göre sıralanır. Sıralama sonucu en başta yer alan küme lideri sırayla diğer küme liderleriyle uzaklığını ölçmeye başlar. Gecikme değeri için yine belirlenen bir eşik değeri vardır. Bu değer o an bulunulan seviyeye bağlıdır. Eşik değeri ile karşılaştırıldığında uygun olan kümelerin içerisindeki düğüm sayılarının toplamı da belirlenen eşik değeri aşmıyorsa iki küme birleşme işlemine girer ve birleşir.

Küme birleşimi protokolü literatürde bulunan NICE, ZIGZAG ve NEMO yapılarında karşımıza çıkmaktadır. Tez kapsamında tasarladığımız küme birleşimi protokolü NICE ile aynıdır. Kullanılan küme birleşimi protokolünde hem küme içerisindeki düğüm sayısının eşik değerden az olmasına hem de küme liderleri arasındaki RTT değerlerine bakılarak karar verilmektedir. ZIGZAG ve NEMO yapılarında ise küme birleşimi için kontrol edilen kriter sadece küme içerisindeki düğüm sayılarının eşik değerden az olmasıdır. Sadece küme içerisindeki sayıya bakmak küme birleşimi için verimli değildir. Küme liderleri arasındaki gecikme değerlerinin kontrol edilmemesinden dolayı birbirine gecikme mesafesi olarak uzak olan iki küme sadece düğüm sayıları az olduğu için birleştirilmiş olur. Bu da küme içerisinde yapılacak olan bir medya akışlandırması esnasında birbirine uzak olan düğümlerin kullanılmasına neden olur ve mesajlarda gecikmeler meydana gelir.

3.2.4 Küme lideri seçimi

Küme içerisindeki lider seçimi algoritması, küme içerisindeki düğüm sayısının daha önceden belirlenmiş lider seçimi için kullanılan eşik değerine erişildiğinde çalıştırılmaktadır. Lider, küme içi iletişimde en etkin rol oynayan düğüm olduğu için liderin, kümenin diğer düğümlerine hemen hemen eşit uzaklıkta yani kümenin merkezinde olması gerekmektedir. Lider seçme algoritması şu şekilde çalışmaktadır. Küme içerisindeki tüm düğümlerin (lider düğüm dahil) birbirleri ile olan RTT mesafelerini bildiklerini varsayalım. Her düğüm, küme içerisindeki diğer düğümler ile ölçülmüş olan RTT mesafelerinin toplamını küme liderine bildirir. Küme lideri, diğer düğümlerden aldığı toplam RTT mesafelerini kendi toplam RTT mesafesi ile karşılaştırarak en küçük toplam RTT değerine sahip olan düğümü yeni lider düğüm olarak seçer ve bu düğümü yeni lider olduğuna dair bilgilendirir. Eski lider düğüm, durumunu liderlikten sıradan bir düğüme çevirir ve artık üst katmanda temsil edilmez. Yeni lider düğüm, kendisini temsilci olarak bir üst katmana ekler. Şekil 3.10'da küme lideri seçimi algoritması verilmiştir.

```
Leader_Election():
//Bu rutin küme lideri tarafından belirli aralıklarla çalıştırılır.

calculate_max_RTT_value();
k: my_cluster_id;

for all node_i ∈ C_k_Node_List
        send_msg(node_i, LEADER_ELECTION);
new_leader = node_having_min_max_RTT;
if (new_leader!=me)
        send_msg(Leader_upper_layer, LEAVING);//üst katmandan çıkıyor
```

Şekil 3.10. Küme lideri seçimi algoritması.

Literatürde bulunan NICE, ZIGZAG, mOVERLAY ve NEMO yapılarında da küme lideri seçimi yapılmaktadır. Tez kapsamında tasarlanan küme lideri seçimi protokolü, NICE yapısının kullandığı küme lideri seçimi protokolü ile aynıdır. Küme lideri, kümenin merkezinde yani diğer düğümlere olan RTT mesafesi minimum olan düğüm seçilmektedir. Diğer ZIGZAG, mOVERLAY ve NEMO yapılarında ise küme içerisindeki herhangi bir düğüm küme lideri olarak

seçilmektedir. Küme liderinin merkezde seçilmesi, küme içerisindeki mesaj trafiğinin küme lideri kontrolünde yapıldığı düşünüldüğünde önemli bir noktadır.

Şekil 3.11'de lider seçimi mesajı alan düğüm tarafından çalıştırılan algoritma verilmiştir.

```
Receiving_Leader_Election():
//Bu rutin, LEADER_ELECTION mesajı alan düğüm tarafından çalıştırılır

k: my_cluster_id;
calculate_max_RTT_value();
send_msg(Leader_k, my_max_RTT_value);
```

Şekil 3.11. Küme lideri seçimi mesajını alan düğümün çalıştırdığı algoritma.

3.2.5 Sistemden bir düğümün haberli ayrılması

Sistemden ayrılmak isteyen düğüm, küme liderine mesaj gönderir ve kümeden ayrılır. Küme lideri, kümeden ayrılan düğümü listesinden çıkarır. Eğer küme içerisinde bulunan düğüm sayısı belirli bir eşik değerin altına düşmüş ise küme birleşim algoritması çalıştırılır. Şekil 3.12'de küme liderinin sistemden ayrılmak isteyen düğüm tarafından ayrılma mesajı aldığında çalıştırdığı algoritma verilmiştir. Tez kapsamında tasarlanan bir düğümün sistemden haberli ayrılması protokolü, literatürdeki NARADA, NICE, ZIGZAG, mOVERLAY ve NEMO yapılarında kullanılan protokol ile aynıdır.

```
Receiving_Leaving_Msg():
//Bu rutin, LEAVING mesajı alan lider tarafından çalıştırılır

k: my_cluster_id;
C_k_Node_List = C_k_Node_List – source(recv_msg(LEAVING));
if (no_of_cluster_members_below_threshold)
        join_neighbor_cluster();
```

Şekil 3.12. Küme liderinin düğümden ayrılma mesajı aldığı zaman çalıştırdığı algoritma.

3.2.6 Sistemden bir düğümün habersiz ayrılması

Önerilen yapıda düğümler habersiz bir şekilde bulundukları kümelerden ayrılabilirler. Küme lideri belirli aralıklarla küme içerisinde bulunan düğümlerin kümede bulunup bulunmadığını kontrol eder. Eğer küme lideri, küme içerisindeki bir düğümden cevap alamaz ise o düğümü kendi tuttuğu listeden çıkarır. Tez kapsamında tasarlanan bir düğümün sistemden habersiz ayrılması protokolü, literatürdeki NICE, ZIGZAG, mOVERLAY ve NEMO yapılarında kullanılan protokol ile aynıdır.

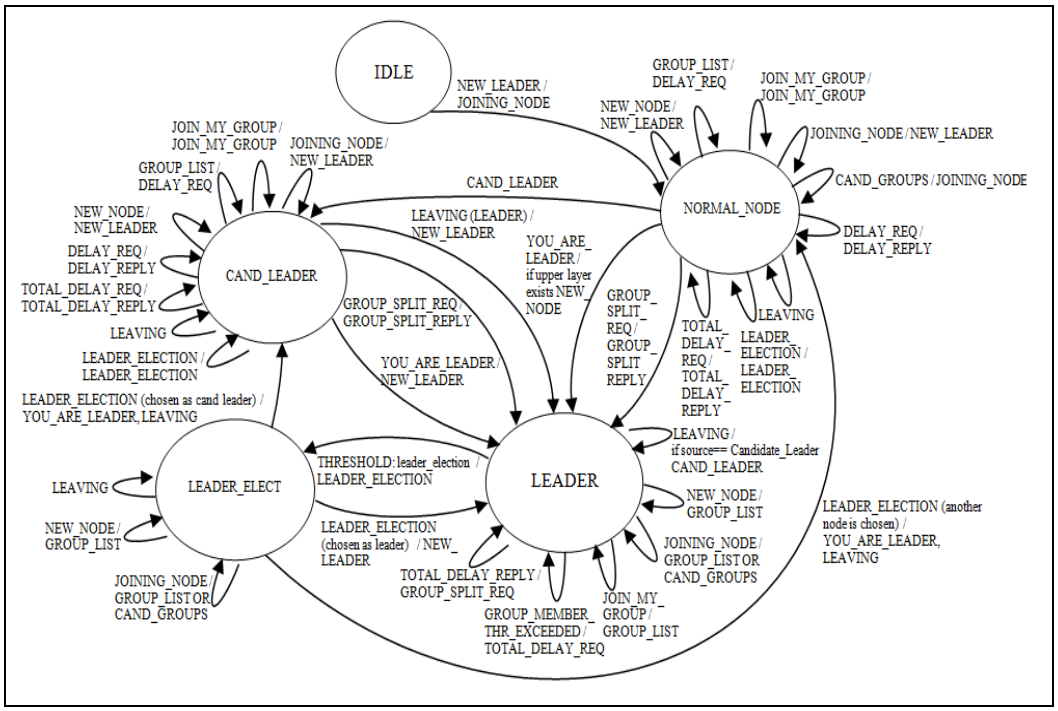
3.2.7 Sistemden küme liderinin haberli ayrılması

Oluşturulan hiyerarşik yapıda küme lideri de sistemden ayrılabilir. Küme liderinin sistemden ayrılması durumunda yeniden küme lideri seçimi algoritmasını çalıştırmak sistem üzerindeki maliyeti arttıracağından bu işlem yedek lider mekanizması kullanılarak daha az maliyet ile yapılır. Yedek küme lideri, küme liderine en yakın düğüm olarak seçilmektedir. Kümeden ayrılmak istediğini bildiren küme lideri, yedek lidere ayrılma mesajı gönderir. Bu mesajı alan yedek lider küme içerisindeki diğer düğümlere kendisinin küme lideri olduğunu bildiren mesaj gönderir. Sistemden ayrılmak isteyen lider, kendisini bir üst katmandan siler. Yedek lider ise kendisini bir üst katmana ekler. Tez kapsamında tasarlanan küme liderinin sistemden ayrılması protokolü, literatürde bulunan NICE, ZIGZAG, mOVERLAY ve NEMO protokollerinden farklıdır. Literatürdeki yapılarda, küme lideri sistemden ayrıldığı zaman sürekli yeni küme lideri seçilmektedir. Küme içerisinde sürekli küme lideri seçimi algoritmasını çalıştırmak mesaj karmaşıklığını arttırmaktadır. Literatürdeki yapılarda küme lideri, sistemden ayrıldığı zaman küme içerisindeki tüm düğümlere sistemden ayrılacağı mesajını göndermektedir. Tez kapsamında önerilen yapıda küme lideri, sadece yedek lidere ayrılacağını göndermektedir. Bu da mesaj karmaşıklığını azaltmaktadır.

3.2.8 Sistemden küme liderinin habersiz ayrılması

Küme liderinin sistemden habersiz ayrılması durumunda devreye girecek olan yedek küme lideridir. Yedek küme liderinin amacı; belirli aralıklarla küme liderinin sistemden ayrılıp ayrılmadığını kontrol etmektir. Küme lideri belirli aralıklarla küme bilgisini yedek lidere gönderir. Yedek küme lideri, liderin sistemden ayrıldığını fark ettiğinde küme içerisinde bulunan diğer düğümlere yeni küme liderinin kendisi olduğu mesajını gönderir. Aynı zamanda önceki küme liderini bir üst katmandan siler ve kendisini hiyerarşide bir üst katmana ekler. Tez kapsamında tasarlanan küme liderinin habersiz ayrılması protokolü, literatürde bulunan diğer yöntemlerden farklıdır. Literatürdeki NICE, ZIGZAG, mOVERLAY ve NEMO yapılarında küme liderinin sistemden ayrıldığını fark eden düğümler aralarında yeniden küme lideri seçimi algoritmasını çalıştırırlar. Bu da düğüm giriş çıkışının devamlı olduğu P2P sistemler için küme liderinin sistemden ayrıldığında her zaman küme lideri algoritmasını çalıştırmak sistem üzerindeki maliyetin artmasına neden olur.

Anlatılan küme devamlılığını sağlayan algoritmalar Şekil 3.13'te sonlu durum makinesi üzerinde gösterilmiştir.



Şekil 3.13. Tasarlanan hiyerarşik kümeleme yapısının sonlu durum makinesi.

Şekilde sonlu durum makinesi verilen sistemde her düğüm IDLE, NORMAL_NODE, LEADER, CAND_LEADER ve LEADER_ELECT durumlarından birine sahiptir. Sistemde kontrol protokolü için çalışan 15 tip mesaj tanımlanmıştır. Sırayla bu mesaj tiplerine ve açıklamalarına değineceğiz.

- *NEW_NODE*: Sistemde var olan bir düğüm, herhangi bir hiyerarşik katmanda yer alan bir kümeye katılacağı zaman bu mesajı o kümenin liderine gönderir.
- *JOINING_NODE:* Sisteme ilk kez giriş yapan düğüm tarafından, kendisine en uygun kümeyi bulmak için herhangi bir hiyerarşik katmanda bulunan küme liderlerine gönderilen mesaj tipidir.
- *CAND_GROUPS:* Sisteme ilk kez giriş yapan bir düğüme küme lideri tarafından gönderilecek olan ve düğümün katılabileceği diğer küme liderlerinin adreslerini tutan mesaj tipidir.
- *NEW_LEADER:* Bu mesaj tipi, bir küme içinde yer alan lider değiştiği zaman ilgili düğümlere yeni lider adresini bildirmek üzere gönderilmektedir.
- *YOU_ARE_LEADER:* Liderlik görevine getirilen düğüme gönderilecek olan mesaj tipidir.
- *GROUP_LIST:* Yeni bir kümeye dahil olan düğüme lider tarafından gönderilecek olan ve o kümenin düğüm adresleri listesini içeren mesaj tipidir.
- *DELAY_REQ:* Bir düğüm listesinde bulunan bir başka düğüm ile arasındaki gecikme değerini bilmiyor ise, bu mesaj tipinde bir mesajı göndererek gecikme hesaplama talebinde bulunur.
- *DELAY_REPLY:* Bir önceki mesaj tipine karşılık olarak gönderilen mesaj tipidir.
- *LEADER_ELECTION:* Lider seçimi algoritması sırasında lider ya da kümeye üye düğümler tarafından gönderilen mesaj tipidir, yeni liderin seçimi için gerekli değerler bu mesaj içerisinde gönderilir.
- *TOTAL_DELAY_REQ:* Lider tarafından gönderilen bu mesaj, lidere en uzak düğümün belirlenmesi amacıyla gönderilir.
- *TOTAL_DELAY_REPLY:* Bir önceki mesaj tipine karşılık olarak gönderilen mesaj tipidir.

- *LEAVING:* Bir düğümün bir kümeden haberli ayrılması sırasında gönderdiği mesaj tipidir.
- *GROUP_SPLIT_REQ:* Lider tarafından lidere en uzak düğüme küme bölünme algoritmasını başlatmak üzere gönderilen mesajdır.
- *GROUP_SPLIT_REPLY:* Düğümlerin küme bölünmesi sırasında kendi kümesine almak üzere diğer düğümlere gönderdiği mesajdır.
- *CAND_LEADER:* Küme içindeki bir düğüme, kendisinin yeni yedek lider olarak seçildiğini belirtmek amacıyla gönderilen mesajdır.

Sisteme ilk kez giriş yapan düğüm, IDLE konumundadır, sunucudan aldığı lider adresi ile birlikte konumunu NORMAL_NODE'a getirir. Bu konumdayken oluşabilecek olaylar, bunlar karşısında oluşacak çıktı ve düğümün geçeceği yeni durum Tablo 3.1'de verilmiştir.

Girdi	**Çıktı**	**Bir Sonraki Durum**
CAND_LEADER	-	*CAND_LEADER*
YOU_ARE_LEADER	*If upper layer exist NEW_NODE*	*LEADER*
GROUP_SPLIT_REQ	*GROUP_SPLIT_REPLY*	*LEADER*
NEW_NODE	*NEW_LEADER*	*NORMAL_NODE*
GROUP_LIST	*DELAY_REQ*	*NORMAL_NODE*
JOIN_MY_GROUP	*JOIN_MY_GROUP*	*NORMAL_NODE*
JOINING_NODE	*NEW_LEADER*	*NORMAL_NODE*
CAND_GROUPS	*JOINING_NODE*	*NORMAL_NODE*
DELAY_REQ	*DELAY_REPLY*	*NORMAL_NODE*
LEADER_ELECTION	*LEADER_ELECTION*	*NORMAL_NODE*
LEAVING	-	*NORMAL_NODE*
TOTAL_DELAY_REQ	*TOTAL_DELAY_REPLY*	*NORMAL_NODE*

Tablo 3.1. NORMAL_NODE durumundaki bir düğüm için durum ve çıktı tablosu.

YOU_ARE_LEADER mesajını alan düğüm, artık bulunduğu kümede yeni liderdir ve eğer bir üst hiyerarşik katman oluşturulmuş ise, bu katmana kayıtlanmak zorundadır. Bu yüzden NEW_NODE tipinde bir mesajı üst katmanın küme liderine gönderir. GROUP_SPLIT_REQ mesajı aldığında, artık kendi oluşturacağı yeni kümenin lideri olacağı için durumunu LEADER olarak değiştirir. NEW_NODE ve JOINING_NODE mesajları alması, bu mesajı atan düğümde, bu düğümün lider olarak kayıtlı olmasından kaynaklanmaktadır. Bu durum daha önceden lider olan ancak liderliği son bulmuş düğümlere, henüz liderliği devam ederken lider adresini almış düğümler tarafından gönderilebilir, bu

durumda bu mesajı gönderen düğüme NEW_LEADER mesajı ile yeni lider adresi bildirilir. Eğer düğüm, GROUP_LIST mesajı almış ise, bir kümeye yeni ilk kez kayıtlanıyor olabilir, küme listesindeki düğümler ile arasındaki gecikmeyi hesaplamak için listedeki düğümlere DELAY_REQ mesajı gönderir. Benzer şekilde DELAY_REQ mesajı aldığında ise, eğer mesajı gönderen düğüm ile arasındaki gecikmeyi henüz hesaplamamışsa, bu düğüm ile arasındaki gecikmeyi hesaplar. JOIN_MY_GROUP mesajı aldığında, eğer bu mesajı alan düğümün kümesine girmeyi kabul ederse, yeni liderine bunu belirtmek üzere JOIN_MY_GROUP mesajını geri gönderir. Eğer düğüm CAND_GROUPS mesajını aldıysa, sisteme ilk kez giriş yaparak aday liderlerin adreslerini almıştır. Bu aday liderlere, kendisine en yakın lideri, dolayısıyla en yakın kümeyi belirlemek üzere JOINING_NODE mesajını, mesaj aldığı katmanın bir alt hiyerarşik katmanına gönderir. Bu durumda JOINING_NODE mesajı, hiyerarşik katmanlar arasında dolaşarak, düğümün en alt katmanda en yakın gruba girmesini sağlar. LEADER_ELECTION mesajını alan NORMAL_NODE konumundaki bir düğüm, listesinde bulunan kendisine en uzak düğüm ile arasındaki gecikmeyi; liderinin küme bölünmesi başlatmadan önceki en uzak düğümü bulmasında kullanılan TOTAL_DELAY_REQ mesajını aldığında komşuları ile kendisi arasındaki ortalama gecikmeyi liderine mesajlar. Bir düğümden LEAVING mesajı aldığında ise, bu düğümü listesinden çıkarır.

LEADER konumunda bulunan bir düğümün alabileceği mesajlar, gerçekleşebilecek girdiler, çıktılar ve liderin bir sonraki durumu Tablo 3.2'de verilmiştir.

Girdi	**Çıktı**	**Bir Sonraki Durum**
NEW_NODE	*GROUP_LIST*	*LEADER*
JOIN_MY_GROUP	*GROUP_LIST*	*LEADER*
JOINING_NODE	*if en alt hiy. katman GROUP_LIST else CAND_GROUPS*	*LEADER*
LEAVING	*if source==Candidate_Leader CAND_LEADER*	*LEADER*
TOTAL_DELAY_REPLY	*GROUP_SPLIT_REQ*	*LEADER*
Grup eleman sayısı eşik değerine ulaştı	*NEIGHBOR_AVG_REQ*	*LEADER*
TIMEOUT:Leader_election()	*LEADER_ELECTION*	*LEADER_ELECT*

Tablo 3.2. LEADER durumundaki bir düğüm için durum ve çıktı tablosu.

Küme lideri, NEW_NODE, JOIN_MY_GROUP ya da en alt hiyerarşik katmanda gelen JOINING_NODE mesajı alırsa bu mesajı atan düğümü küme listesine ekler ve küme listesini düğüme gönderir. JOINING_NODE mesajını alan lider eğer bu mesajı en alt hiyerarşik katmanda almamış ise, sisteme yeni giren düğümün en yakın kümeyi bulabilmesi için o katmanda bulunan düğümlerin listesini düğüme gönderir. LEAVING mesajını alan lider, bu mesaj kümenin yedek liderinden gelmiş ise, kendisine yeni bir yedek lider seçer ve bu seçimi CAND_LEADER mesajı ile bu düğüme bildirir. Küme eleman sayısı eşik değerine ulaştığı zaman, küme bölünme algoritmasını çalıştırmaya başlamadan önce en uzak düğümü seçmek için TOTAL_DELAY_REQ mesajlarını gönderir, aldığı son mesajdan sonra seçtiği gecikme ölçütü olarak en uzak ve yeni kümenin lideri olmaya aday düğüme GROUP_SPLIT_REQ mesajını gönderir. Lider seçiminin yenilenmesi gerektiğini belirleyen zamanlayıcının tetiklenmesi ile lider seçimi algoritması çalışır ve lider, küme elemanlarına seçimin başladığını LEADER_ELECTION mesajı göndererek haber verir.

Yedek lider durumundaki bir düğümün girdi, çıktı ve sonraki durumu Tablo 3.3'te verilmiştir.

Girdi	**Çıktı**	**Bir Sonraki Durum**
YOU_ARE_LEADER	*NEW_LEADER,* *If upper layer exist NEW_NODE*	*LEADER*
GROUP_SPLIT_REQ	*GROUP_SPLIT_REPLY*	*LEADER*
LEAVING	*if source==Leader* *NEW_LEADER, CAND_LEADER*	*if source==Leader* *LEADER*
NEW_NODE	*NEW_LEADER*	*CAND_LEADER*
GROUP_LIST	*DELAY_REQ*	*CAND_LEADER*
JOIN_MY_GROUP	*JOIN_MY_GROUP*	*CAND_LEADER*
JOINING_NODE	*NEW_LEADER*	*CAND_LEADER*
DELAY_REQ	*DELAY_REPLY*	*CAND_LEADER*
LEADER_ELECTION	*LEADER_ELECTION*	*CAND_LEADER*
TOTAL_DELAY_REQ	*TOTAL_DELAY_REPLY*	*CAND_LEADER*

Tablo 3.3. CAND_LEADER durumundaki bir düğüm için durum ve çıktı tablosu.

Yedek lider, kümedeki lider sistemden ayrılırsa lider olarak görev alacak olan düğümdür; ancak bunun dışındaki tüm davranışları NORMAL_NODE durumundaki bir düğüm ile benzerlik göstermektedir. Eğer yedek lider durumundaki bir düğüm, küme liderinden LEAVING mesajı alırsa, kendisini lider

ilan eder ve yeni bir yedek lider seçerek bu düğüme CAND_LEADER mesajı gönderir. Yedek liderin bunun dışındaki tüm davranışları Tablo 3.1 için yapılan açıklamalardan izlenebilir.

Tablo 3.4'te LEADER_ELECT durumundaki bir düğüm için girdi, çıktı ve sonraki durum listesi yer almaktadır.

Girdi	**Çıktı**	**Bir Sonraki Durum**
NEW_NODE	*GROUP_LIST*	*LEADER_ELECT*
JOINING_NODE	*if en alt hiy. katman GROUP_LIST else CAND_GROUPS*	*LEADER_ELECT*
LEAVING	-	*LEADER_ELECT*
LEADER_ELECTION	*if me==Leader CAND_LEADER*	*LEADER*
LEADER_ELECTION	*if me==Candidate_Leader YOU_ARE_LEADER, LEAVING*	*CAND_LEADER*
LEADER_ELECTION	*if me!=Leader || if me!=Candidate_Leader YOU_ARE_LEADER,CAND_LEADER, LEAVING*	*NORMAL_NODE*

Tablo 3.4. LEADER_ELECT durumundaki bir düğüm için durum ve çıktı tablosu.

LEADER_ELECT durumu, liderin lider seçme algoritmasını çalıştırırken geçtiği bir durum olduğu için, NEW_NODE ve JOINING_NODE mesajlarına lider durumunda verdiği tepkileri verir. Ancak LEAVING mesajı aldığında bu mesaj yedek liderden olsa bile yedek lider seçimini lider seçme algoritmasının sonuna kadar erteler. Lider seçimi algoritması, alınan son LEADER_ELECTION mesajı ile sonlanır, seçim sonunda yine kendisi lider çıkarsa, lider durumuna geri döner; eğer yedek lider değişti ise yeni yedek lidere bu durumu mesajlar. Eğer kendisi yedek lider seçilmişse yedek lider konumuna geçer ve yeni lidere YOU_ARE_LEADER mesajı gönderir. YOU_ARE_LEADER mesajı içinde, eğer var ise üst katman grup liderinin adresi belirtilir. Lider seçimi sonunda kendisi ne lider, ne de yedek lider seçilmez ise, yeni lider ve yeni yedek lidere yeni görevlerini bildirir. Lider, CAND_LEADER ya da NORMAL_NODE durumuna geçerse üye olduğu tüm üst hiyerarşik katmanlarda LEAVING mesajı gönderir.

IDLE durumunda olan bir düğüm, NEW_LEADER mesajı alırsa, P2P sisteme giriş yapmak için başvurmuştur ve sistemde bulunan en üst liderin adresi gelen mesaj içindedir. Bu mesajı alan düğüm, durumunu NORMAL_NODE yaparak en üst katman liderine JOINING mesajı gönderir. Eğer bir düğüm

sistemden ya da herhangi bir hiyerarşik katmandan ayrılırsa, ayrıldığı katman veya katmanlardaki durumunu IDLE'a konumlandırır. Bu durum, sonlu durum makinesi ve girdi/çıktı tablolarında gösterilmemiştir.

Sistemde sunucu görevini üstlenen düğüm, kendisine gelen iki tip mesajı kabul etmektedir. Eğer sunucu, NEW_LEADER tipinde bir mesaj alırsa, sistemde bulunan en üst katman liderinden gelen bu mesajın kaynağını, sisteme girecek düğümlere yönlendirmek üzere saklı tutar. JOINING_NODE mesaj tipinde bir mesaj aldığında, elindeki adresi, dolayısıyla o an sistemde bulunan en üst hiyerarşik katmanın lider adresini bu düğüme mesaj olarak iletir. Sunucuda her zaman sadece bir düğüm adresi bulunmaktadır.

Tasarlanan hiyerarşik üstüne bindirmeli yapıda bir düğümün sisteme getireceği kontrol maliyetini hesaplayalım. İlk olarak düğümün, sisteme getireceği maliyeti hesaplamak için kullanılan ifadelerimizi tanımlayalım.

N: Sistemdeki toplam düğüm sayısı
L: Hiyerarşik yapının kurulması sonucu oluşan toplam katman sayısı
min_d: Küme içerisinde olabilecek en az düğüm sayısı
max_d: Küme içerisinde olabilecek en fazla düğüm sayısı

Hiyerarşik yapının oluşması sonucu oluşan toplam katman sayısı (*L*),

$$L \leq \log_{min\_d} N + 1 \qquad \text{(Eş. 3.1)}$$

olacaktır. En kötü durum senaryosunda en üst katmanda bulunan bir M düğümünün küme içerisindeki toplam komşu üye sayısı yani düğüm derecesi,

$$L \times (max\_d - 1) \qquad \text{(Eş. 3.2)}$$

olacaktır. Eş. 3.1'de bulduğumuz eşitsizliği Eş. 3.2'de yerine koyarsak bir düğümün sisteme getireceği kontrol maliyetini aşağıdaki şekilde elde ederiz.

$$(\log_{min\_d} N + 1) \times (max\_d - 1) \cong O(d \log_d N) \qquad \text{(Eş. 3.3)}$$

4. PERFORMANS

Medya akışlandırması için tasarlanan hiyerarşik üstüne bindirmeli kümeleme yapısı PlanetLab üzerinde test edilmiştir. Performans kriterleri olarak kümenin çapına, küme liderine gecikme ölçütüne göre en uzak olan düğüm mesafesine, küme içerisindeki düğümlerin birbirleri arasındaki gecikme mesafesine, kümelerin birbirleriyle olan gecikme mesafesine, kümelerin doluluk oranlarına, düğümlerin kümelere ne kadar sürede dahil olduklarına, kümeleme hassasiyetine (accuracy) ve kümeleme doğruluğuna (correctness) bakılmıştır.

Tasarlanan hiyerarşik kümeleme protokolünün ne kadar etkin olduğunun anlaşılabilmesi için, öncelikli olarak protokolün geliştirildiği test ortamı olan PlanetLab tanıtılacaktır. Daha sonra geliştirilen yapı üzerinde test edilen performans kriterlerine değinilecektir. Son olarak da performans kriterlerine bakılarak alınan test sonuçlarının kümülatif dağılım fonksiyonu (cumulative distribution function, CDF) grafikleri incelenecektir.

4.1 PlanetLab Test Ortamı

PlanetLab test ortamı 2003 yılında kurulmuştur. Gelişmiş akademik enstitülerde ve endüstriyel kuruluşlarda 1000'den fazla araştırmacı PlanetLab test ortamını kullanmaktadır. PlanetLab, yeni ağ servislerinin gelişmesini sağlayan global bir ağ sistemidir. Bu sistem içerisinde araştırma ve geliştirme gerçekleştiren üniversiteler arasında Princeton, Berkeley ve MIT gibi gelişmiş üniversiteler bulunmaktadır. Ayrıca PlanetLab ile ortak çalışmalar yapan endüstriyel kuruluşlar arasında Google ve HP gibi önemli firmalar yer almaktadır. PlanetLab kullanılarak yapılan işlemler arasında, dağıtık depolama (distributed storage), ağ haritalama (network mapping) ve sorgu işleme (query processing) bulunmaktadır. PlanetLab ağı şu an itibariyle 510 noktada 1092 adet düğüm ile hizmet vermektedir. PlanetLab kullanılarak gerçeklenen uygulamaların sonucu, lokal simülasyonlar kullanılarak gerçeklenen uygulamaların sonucundan farklıdır. Çünkü PlanetLab ağı içerisinde kullanılan düğümler birbirine gerçek ağ koşulları ile bağlıdırlar ve alınan test sonuçları gerçeğe yakın çıkmaktadır. PlanetLab düğümlerini kullanarak uygulama geliştirmek, lokal simülasyonda uygulama

geliştirmeye göre daha zordur. Geliştirilen uygulamaların düğümlerde çalıştırılabilmesi için uygulamanın önce düğümlere yüklenmesi gerekmektedir. Ayrıca geliştirilen uygulamanın PlanetLab ortamında çalışabilmesi için dağıtık olarak tasarlanması zorunludur. Dağıtık olarak uygulama çalıştırıldığından dolayı hata ayıklama işlemlerinde sıkıntılar yaşanabilmektedir. 1092 adet düğüm sistemde bulunmasına rağmen belirtilen düğüm sayısına erişmek, Internet bağlantısının kopması ve güvenlik duvarı sebebiyle sıkıntılıdır (PlanetLab, 2010).

4.2 Tasarlanan Hiyerarşik Kümeleme Yapısının Performans Kriterleri

Tezin bu bölümünde görevdeş ağlar için geliştirilen hiyerarşik kümeleme yapısının performans kriterlerine değinilecektir. Sırasıyla incelenecek olan performans kriterleri; küme çapı, küme liderine gecikme ölçütü olarak en uzak olan düğümler mesafesi, küme içerisindeki düğümlerin birbirleriyle olan gecikme mesafesi, kümelerin birbirleriyle olan gecikme mesafesi, kümelerin doluluk oranı, düğümlerin kümelere ne kadar sürede katıldıkları, kümeleme hassasiyeti (accuracy) ve kümeleme doğruluğudur (correctness).

4.2.1 Küme çapı kriteri

Geliştirilen protokol sonucu oluşan kümelerin çapları, bakılması gereken önemli bir kriterdir. Kümenin çapından kastedilen, küme içerisinde birbirleri arasında gecikme değeri en fazla olan iki düğüm arası mesafedir. Bu kriterin incelenmesinin nedeni, görevdeş ağlarda sisteme giren düğüm sayısı arttıkça hiyerarşide en alt katmanda bulunan kümelere giren düğüm sayıları da artacaktır. Dolayısıyla kümelerin boyutları da artacaktır.

4.2.2 Küme liderine uzaklık kriteri

Tasarlanan protokolde bakılması gereken başka bir kriter de küme liderine gecikme ölçütü olarak en uzak olan düğümlerin mesafeleridir. Geliştirilen protokolde küme lideri merkezde yani diğer düğümlere olan toplam gecikme değeri en az olan düğüm seçilmektedir. İncelenen bu kriter ile küme liderine en

uzak düğümleri belirleyerek oluşturulan kümeler hakkında bilgi sahibi oluruz. Küme içerisinde diğer tüm düğümler gerekli bilgileri küme liderinden aldıkları için düğümlerin küme liderine çok uzak olmamaları gerekmektedir.

4.2.3 Küme içerisindeki düğümlerin birbirleriyle olan gecikme mesafeleri kriteri

Önerilen protokol sonucu oluşan kümeler içerisindeki düğümlerin birbirleri arasındaki gecikme değerleri önem taşımaktadır. Kümelerin dengeli bir üye dağılımına sahip olması için küme içerisindeki düğümlerin birbirleriyle olan mesafelerinin çok fazla olmaması gerekmektedir. Küme içerisindeki düğümler arasındaki mesafe küme içerisinde meydana gelen küme bölünmesinde, küme birleşiminde, küme lideri ve yedek lider seçiminde önemli rol oynamaktadır.

4.2.4 Kümelerin birbirleriyle olan gecikme mesafeleri kriteri

Oluşturulan hiyerarşik yapının en alt katmanında fiziksek düğümlerin oluşturduğu kümeler bulunmaktadır. En alt katmanda oluşan bu kümelerin birbirleri arasındaki mesafe önemlidir. Kümeler arasındaki mesafeyi bulmak için, küme liderleri arasındaki mesafe ölçülmüştür. Çünkü küme liderlerini kümenin merkezinde seçtiğimiz için, kümeler arasındaki mesafe hakkında en iyi sonucu bu şekilde alırız. Kümelerin arasındaki mesafelere bakarak oluşan kümelerin birbirlerine ne kadar yakın veya uzak olduğu hakkında bilgi sahibi oluruz. Görevdeş ağlarda sistemin dinamiği giriş çıkışlardan dolayı devamlı değiştiği için, sistemden ayrılan bir düğüm kendisine gecikme olarak daha yakın bir küme liderinin bulunduğu kümeye girebilir. Bu işlem için de iki küme arasındaki gecikme mesafesinden yararlanılır.

4.2.5 Kümelerin doluluk oranı kriteri

Hiyerarşik yapının kurulması sonucu oluşan kümelerdeki düğüm sayılarının dengeli olması gerekmektedir. Kümelerin doluluk oranı kriteri ile kümelerde ortalama kaç tane düğüm olduğuna ve kümelerin doluluk yüzdesine

bakılmaktadır. Ayrıca düğüm sayısı arttıkça oluşan küme sayısının artması da beklenen bir durum olarak ortaya çıkmaktadır.

4.2.6 Düğümlerin kümelere katılma süreleri kriteri

Tasarlanan hiyerarşik yapıda, sisteme katılmak isteyen düğüm en üst hiyerarşide bulunan kümeden başlayarak en alt katmanda bulunan kümeye doğru gecikme ölçümleri yaparak kendisine en yakın mesafeli kümeyi bularak ilgili kümeye katılır. Bu tip tasarlanan hiyerarşik sistemlerde düğümün kümeye yerleşme süresi fazla olmamalıdır. Düğümü olabildiğince çabuk bir şekilde az mesajlaşma yaparak kümeye yerleştirmek hiyerarşik tasarlanan yapılarda önemli bir noktadır. Sisteme katılmak isteyen bir düğümü ne kadar çabuk bir şekilde en alt katmanda bulunan kümeye yerleştirirsek sistemin verimliliği o kadar artar ve sisteme katılmak isteyen bir sonraki düğüm için çok büyük zaman kaybı yaşanmamış olur.

4.2.7 Kümeleme hassasiyeti (accuracy) kriteri

Geliştirilen yapıda önemli başka bir performans kriteri de kümeleme hassasiyetidir (accuracy) (Zhang et al., 2002). Kümeleme hassasiyetinin nasıl hesaplanacağına geçmeden önce bazı ifadeleri tanımlamamız gerekmektedir. Bu ifadeler;

N: Sistemdeki toplam düğüm sayısı
k: sistemdeki toplam küme sayısı
G_i*: i. küme*
$|G_i|$*:* G_i *kümesindeki düğüm sayısı*
L_i*:* G_i *kümesinin lideri*
x: G_i *kümesi içerisindeki bir düğüm*
y: G_j *kümesi içerisindeki bir düğüm*
(x,y): x ve y düğümleri arasındaki gecikme mesafesi

şeklindedir (Zhang et al., 2002). Kümeleme hassasiyeti (accuracy),

$$Kümeleme\ hassasiyeti = 1 - \frac{1}{N'} \sum_{\substack{i,j \in [1...k] \\ i<j}} \sum_{\substack{x \in G_i \\ y \in G_j}} \frac{\left| RTT(x,y) - RTT(L_i, L_j) \right|}{RTT(x,y)} \quad \text{(Eş. 4.1)}$$

şeklinde hesaplanır (Zhang et al., 2002). Eş. 4.1'de N',

$$N' = \sum_{\substack{i,j \in [1...k] \\ i<j}} \left| G_i \right\| G_j \right| \quad \text{(Eş. 4.2)}$$

ile hesaplanır (Zhang et al., 2002). Eş. 4.2'de N' ile tanımlanan ifade, kümeleme hassasiyeti hesaplanacak sistemde bulunan kümelerin içerisindeki düğüm sayılarının birbirleriyle çarpımlarının toplanmasıdır. Eş. 4.1'de küme hassasiyetinin hesaplanması şu şekilde yapılmaktadır; en alt katmanda bulunan bir küme içerisindeki düğüm, en alt katmanda bulunan başka bir küme içerisindeki düğüm ile gecikme mesafesini ölçer. Daha sonra ölçüm yapılan bu iki küme arasındaki küme liderleri arasındaki gecikme değeri ölçülür. İki düğüm arasında ölçülen gecikme değerinden, iki kümenin lideri arasında ölçülen gecikme değeri çıkartılır. Çıkan sonuç, iki düğüm arasında hesaplanan gecikme değerine bölünür. Bir küme içerisindeki her düğüm, diğer küme içerisindeki bütün düğümler ile tek tek bu ölçümleri tekrarlar. İlgili küme ile ölçme işlemi bittikten sonra en alt katmanda bulunan başka bir küme ile ölçme işlemi devam eder. En alt katmanda bulunan tüm kümeler ile bu ölçüm işlemleri tekrarlanır. Kümeleme hassasiyeti sonucunda çıkan yüzdelik oranı ne kadar yüksekse, oluşturulan yapının iyi bir kümeleme yaptığı sonucu ortaya çıkmaktadır (Zhang et al., 2002).

4.2.8 Kümeleme doğruluğu (correctness) kriteri

Görevdeş ağlarda düğümlerin sürekli sistemin dinamiğini değiştirmesinden dolayı, geliştirilen hiyerarşik yapıda düğümlerin yerleri değişebilmektedir. Düğümlerin gecikme ölçütü olarak kendilerine en yakın kümede bulunup bulunmadıklarına bakmak için kullanılan önemli bir kriter de kümeleme doğruluğudur (correctness) (Zhang et al., 2002). Kümeleme doğruluğu,

$$\gamma = \frac{\text{Referans küme liderine olan RTT mesafesi}}{\text{En yakın küme liderine olan RTT mesafesi}} \qquad \text{(Eş. 4.3)}$$

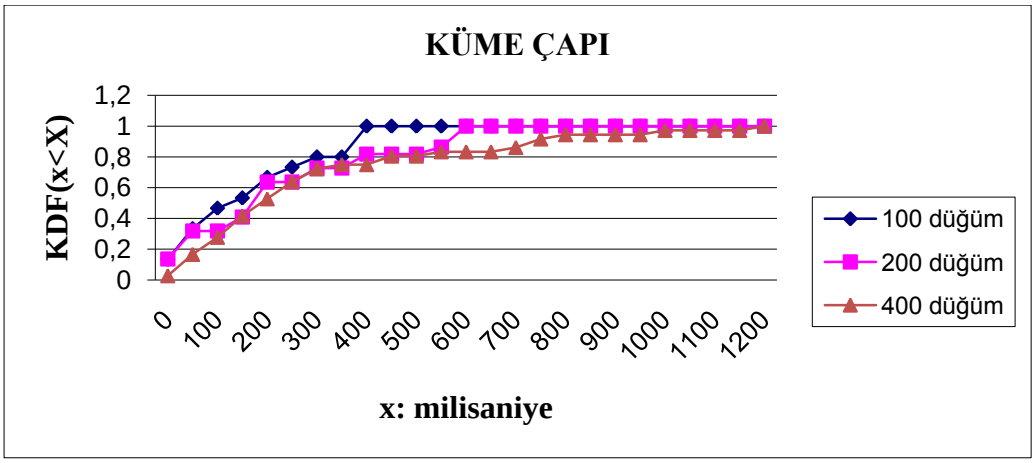
şeklinde hesaplanır (Zhang et al., 2002). Eş. 4.3'te referans küme lideri ile kastedilen düğümün kendi bulunduğu kümenin lideridir. Bir düğümün küme seçimi, o düğümün referans küme lideri ile RTT mesafesinin γ katı aralığında olması durumunda doğru kabul edilir (Zhang et al., 2002).

4.3 PlanetLab Üzerinde Geliştirilen Hiyerarşik Kümeleme Sonuçları

Geliştirilen hiyerarşik kümeleme yapısı test ortamı olan PlanetLab üzerinde 100, 200 ve 400 düğüm ile çalıştırılmıştır. 100 düğüm için küme içerisinde olabilecek en fazla düğüm sayısı 12, 200 düğüm için küme içerisinde olabilecek en fazla düğüm sayısı 15 ve 400 düğüm için küme içerisinde olabilecek en fazla düğüm sayısı 18 olarak ayarlanmıştır. Bu bölümde sırasıyla daha önce açıklanan performans kriterlerinin PlanetLab üzerindeki sonuçlarına değinilecektir. Bu bölümde verilen tüm kümülatif dağılım fonksiyonlarına ait tablolar EK 1 de verilmiştir.

4.3.1 Küme çapı sonuçları

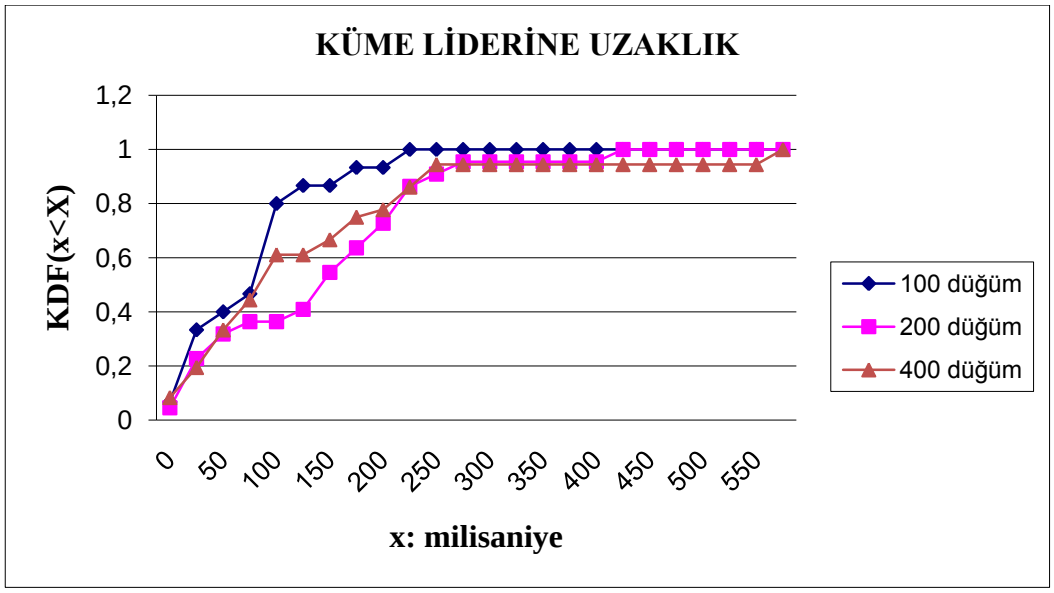
Şekil 4.1'de kümülatif dağılım fonksiyonu %100 olduğu zaman yani düğümlerin tamamı kapsandığı zaman 100 düğüm için küme çapı 400 milisaniye, 200 düğüm için 600 milisaniye ve 400 düğüm için 1200 milisaniye olduğu görülmektedir. Bu beklenen bir sonuçtur. Çünkü düğüm sayısı arttıkça küme içerisine giren düğüm sayısı da artacaktır. Dolayısıyla kümeler daha kalabalıklaşacağı için kümeler büyüyecektir. Kümelerin büyümesi de küme çapını yani küme içerisinde gecikme mesafesi olarak birbirine en uzak olan iki düğüm arası mesafeyi arttıracaktır.



Şekil 4.1. Küme çapı kümülatif dağılım fonksiyonu grafiği.

4.3.2 Küme liderine uzaklık sonuçları

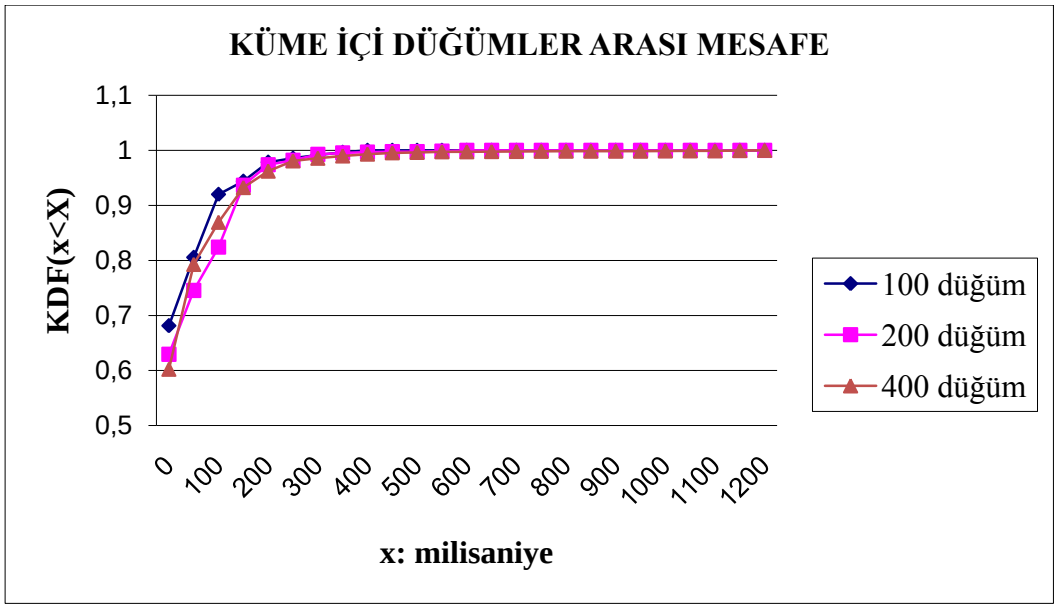
Şekil 4.2'de 100, 200 ve 400 düğüm için küme liderine en uzak düğümlerin kümülatif dağılım fonksiyonu grafiği verilmiştir. Şekilde 100 düğümün tamamı kapsandığında küme liderine olan en uzak düğüm mesafesinin 225 milisaniye, 200 düğümün tamamı kapsandığında küme liderine olan en uzak düğüm mesafesinin 425 milisaniye ve 400 düğümün tamamı kapsandığında küme liderine olan en uzak düğüm mesafesinin 575 milisaniye olduğu anlaşılmaktadır. 400 düğüm için Şekil 4.1'de küme çapı en büyük çıkmıştı. Küme çapının 400 düğümde büyük çıkması Şekil 4.2'de bulunan sonuçları doğrulamaktadır. 400 düğüm için oluşan yapıda düğümler, küme liderine daha uzak bir bölgede bulunmaktadırlar. Bu da kümenin çapının artmasına neden olmaktadır. En iyi sonuç 100 düğüm için çıkmıştır.



Şekil 4.2. Küme içerisinde lidere en uzak olan düğümlerin mesafesinin kümülatif dağılım fonksiyonu grafiği.

4.3.3 Küme içerisindeki düğümlerin birbirleriyle olan gecikme mesafeleri sonuçları

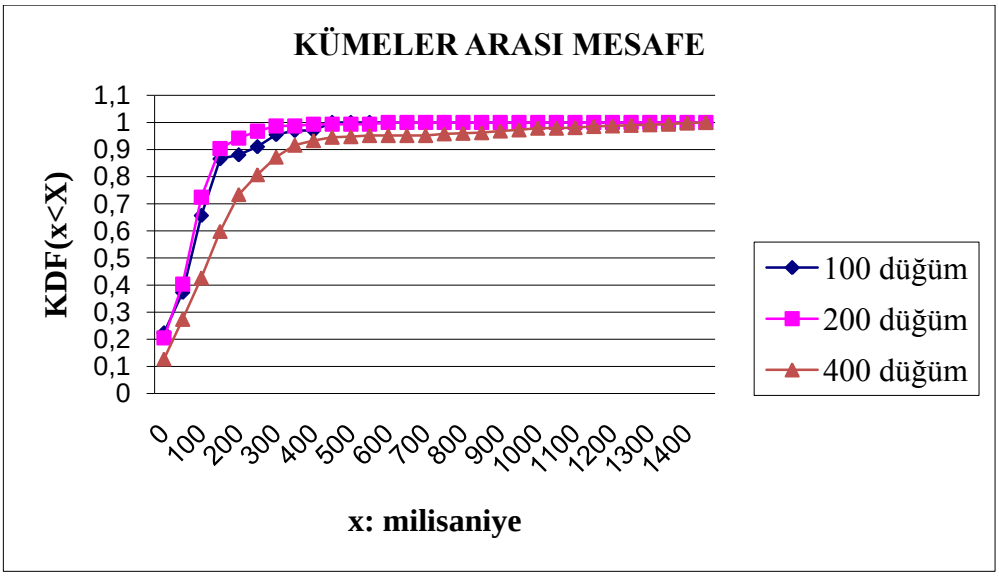
Şekil 4.3'te 100, 200 ve 400 düğüm için her bir küme içerisindeki düğümlerin birbirleriyle olan mesafelerinin kümülatif dağılım fonksiyonu grafiği verilmiştir. Şekilde 100 düğümün tamamı kapsandığında küme içerisindeki düğümlerin birbirleriyle olan gecikme mesafesi 400 milisaniye, 200 düğümün tamamı kapsandığında küme içerisindeki düğümlerin birbirleriyle olan gecikme mesafesi 600 milisaniye ve 400 düğümün tamamı kapsandığında küme içerisindeki düğümlerin birbirleriyle olan gecikme mesafesi 1200 milisaniye olduğu anlaşılmaktadır. Çıkan sonuçlar, Şekil 4.1'de küme çapı grafiği ile uyuşmaktadır. Küme içi düğümler arası mesafe grafiğinde en iyi sonuç 100 düğüm için çıkmıştır. 100 düğüm için oluşan yapıda, düğümlerin birbirine daha yakın olduğu anlaşılmaktadır.



Şekil 4.3. Küme içi düğümler arası gecikme mesafesinin kümülatif dağılım fonksiyonu grafiği.

4.3.4 Kümelerin birbirleriyle olan gecikme mesafeleri sonuçları

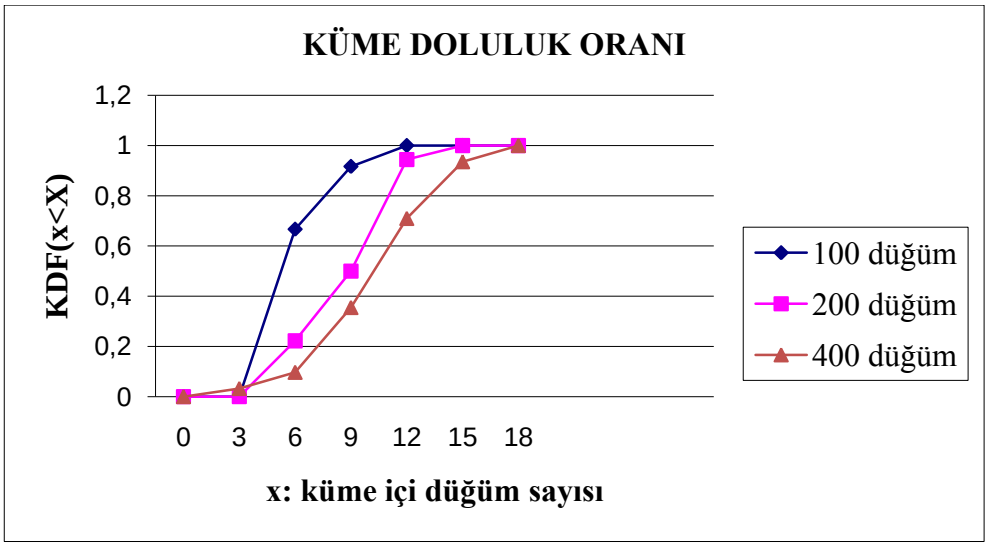
Şekil 4.4'te 100, 200 ve 400 düğüm için kümeler arası mesafenin kümülatif dağılım fonksiyonu grafiği verilmiştir. Kümeler arası mesafe ölçümünden kastedilen, küme liderleri arasındaki gecikme ölçümüdür. Şekilde 100 düğümün tamamı kapsandığında kümeler arası gecikme mesafesinin 450 milisaniye, 200 düğümün tamamı kapsandığında kümeler arası gecikme mesafesinin 600 milisaniye ve 400 düğümün tamamı kapsandığında kümeler arası gecikme mesafesinin 1450 milisaniye olduğu anlaşılmaktadır. 100 düğüm için oluşturulan yapı daha az sayıda küme içerdiği için kümeler arası mesafe en az çıkmıştır. Bu da beklenen bir sonuçtur. Kümeler arası mesafe ölçümünde sadece fiziksel seviye yani en alt seviyedeki kümeler dikkate alınmıştır.



Şekil 4.4. Kümeler arası gecikme mesafesinin kümülatif dağılım fonksiyonu grafiği.

4.3.5 Kümelerin doluluk oranı sonuçları

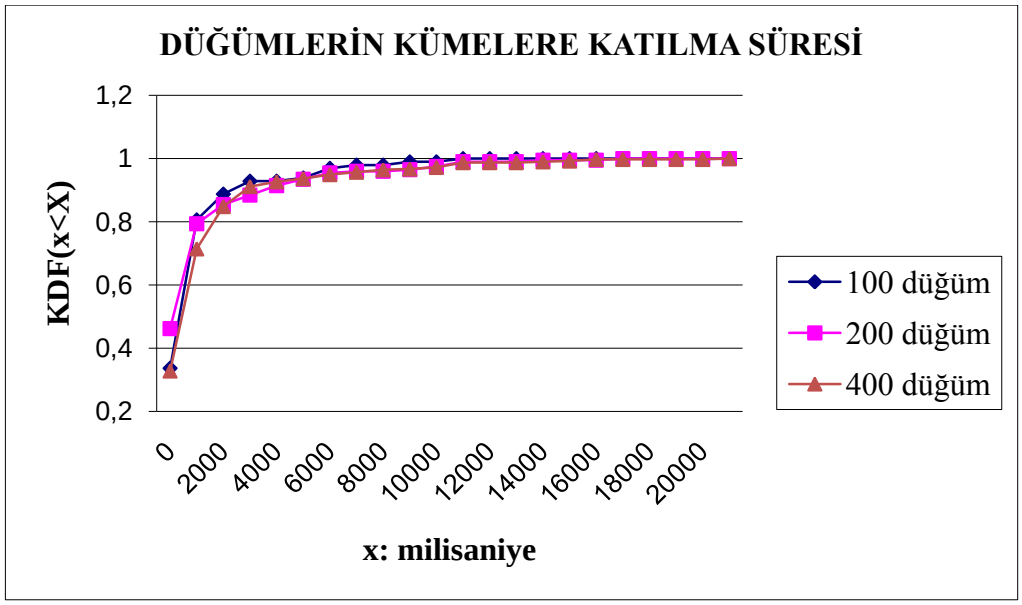
Şekil 4.5'te kümelerin doluluk oranlarının kümülatif dağılım fonksiyonu grafiği verilmiştir. Şekilde 100 düğümün tamamı kapsandığında küme içerisinde olabilecek en fazla düğüm sayısının 12 olduğu, 200 düğümün tamamı kapsandığında küme içerisinde olabilecek en fazla düğüm sayısının 15 olduğu ve 400 düğümün tamamı kapsandığında küme içerisinde olabilecek en fazla düğüm sayısının 18 olduğu görülmektedir. Bu zaten beklenen bir durumdur. Çünkü sistem kurulmadan önce 100 düğüm için küme içi düğüm sayısı 12, 200 düğüm için küme içi düğüm sayısı 15 ve 400 düğüm için küme içi düğüm sayısı 18 olarak ayarlanmıştır. Şekilde dikkat edilmesi gereken önemli sonuç; 100 düğüm için oluşan yapıda kümelerin %91,67'sinde düğüm sayısının 9, 200 düğüm için oluşan yapıda kümelerin %94,44'ünde düğüm sayısının 12 ve 400 düğüm için oluşan yapıda kümelerin %93,55'inde düğüm sayısının 15 olduğu görülmektedir. Bu da oluşan kümelerin dengeli sayıda düğüme sahip olduğunu yani küme bölünmesi ve küme birleşimi işlemlerinin verimli şekilde çalıştığını ortaya koymaktadır.



Şekil 4.5. Kümelerin doluluk oranlarının kümülatif dağılım fonksiyonu grafiği.

4.3.6 Düğümlerin kümelere katılma süreleri sonuçları

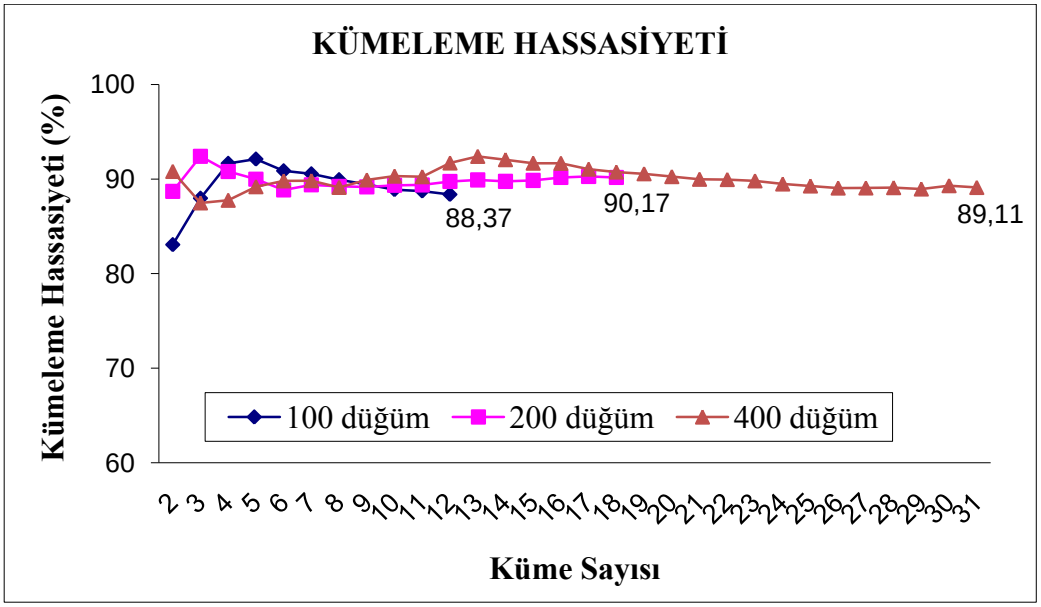
Şekil 4.6'da 100, 200 ve 400 düğüm için düğümlerin kümelere katılma sürelerine bakılmıştır. 100 düğümün tamamı kapsandığında düğümlerin kümelere katılma süresinin 11 saniye, 200 düğümün tamamı kapsandığında düğümlerin kümelere katılma süresinin 17 saniye ve 400 düğümün tamamı kapsandığında düğümlerin kümelere katılma süresinin 21 saniyede olduğu grafikten anlaşılmaktadır. Bu sürelerin normalden fazla çıkmasının nedeni bazı düğümlerin kümelere çok geç katılmalarıdır. 400 düğüm için sürenin fazla olması beklenen bir durumdur. Çünkü 400 düğüm için oluşan üst katmanlarda küme sayısı daha fazla olduğu için sisteme dahil olmak isteyen düğüm, daha fazla sayıda küme lideri ile gecikme mesafesi ölçümü yapmaktadır. Grafik ayrıntılı incelendiğinde 100 düğümün %88,76'sı, 200 düğümün %85,43'ü ve 400 düğümün %84,85'i 2 saniyede kümelerine katılmaktadır.



Şekil 4.6. Düğümlerin kümelere katılma sürelerini gösteren kümülatif dağılım fonksiyonu grafiği.

4.3.7 Kümeleme hassasiyeti (accuracy) sonuçları

Şekil 4.7'de 100, 200 ve 400 düğüm için oluşturulan yapının kümeleme hassasiyeti verilmiştir. 100 düğüm için toplam 12 küme, 200 düğüm için toplam 18 küme ve 400 düğüm için toplam 31 küme oluşmuştur.



Şekil 4.7. Tasarlanan yapıda 100, 200 ve 400 düğüm için kümeleme hassasiyeti grafiği.

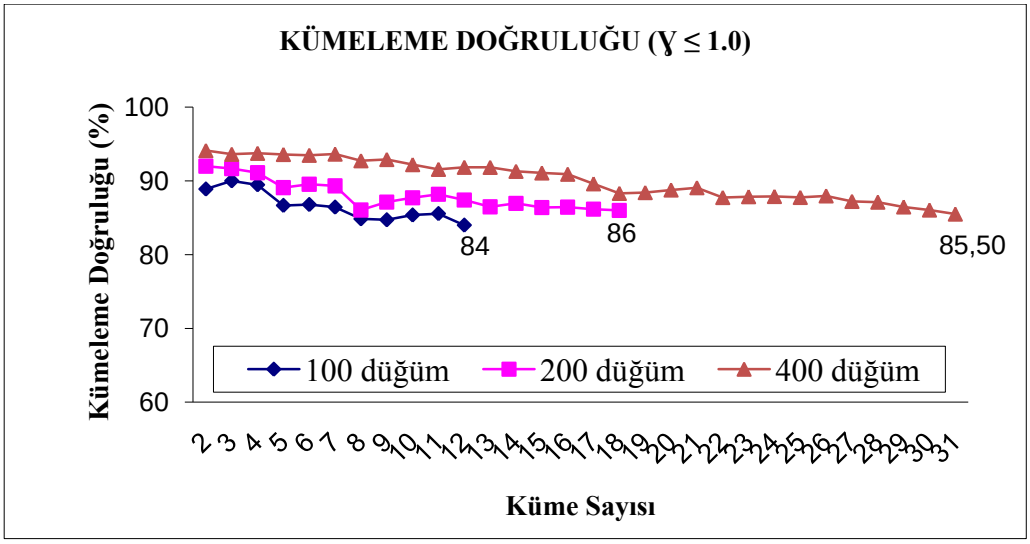
Şekilden de anlaşılacağı üzere 100 düğüm için kümeleme hassasiyeti %88,37, 200 düğüm için kümeleme hassasiyeti %90,17 ve 400 düğüm için kümeleme hassasiyeti %89,11 çıkmıştır. Ölçülen kümeleme hassasiyeti değeri ne kadar yüksek çıkarsa, oluşturulan kümelerin bir o kadar doğru bir şekilde oluşturulduğu ortaya çıkmaktadır. Alınan kümeleme hassasiyeti değerleri gösteriyor ki geliştirilen hiyerarşik kümeleme yapısı iyi bir kümeleme ortaya çıkarmıştır. Literatürdeki mOVERLAY yapısının kümeleme hassasiyeti %70 olarak ölçülmüştür (Zhang et al., 2002). Sonuç olarak geliştirilen yapının test ortamı olan PlanetLab'ta çalıştırılması sonucu oluşan yapının kümeleme hassasiyeti, yerel olarak çalıştırılan mOVERLAY yapısının kümeleme hassasiyetinden daha iyi çıkmıştır.

4.3.8 Kümeleme doğruluğu (correctness) sonuçları

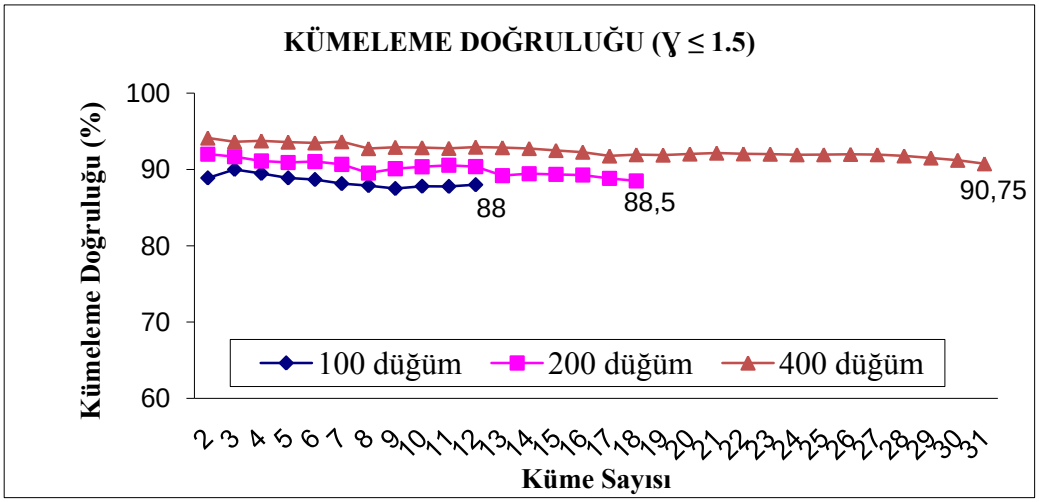
Düğümlerin uygun kümeye yerleşip yerleşmediğini bulmak için, kümeleme doğruluğuna bakılmaktadır (Zhang et al., 2002). Kümeleme doğruluğunu Eş. 4.3'te verilen formül ile buluruz. Bir düğümün küme seçimi, o düğümün referans küme lideri ile RTT mesafesinin γ katı aralığında olması durumunda doğru kabul edilir (Zhang et al., 2002). Tasarlanan hiyerarşik kümeleme yapısında γ = 1.0 olarak kabul ettik. Dolayısıyla $\gamma \leq 1.0$ şartı sağlandığında düğüm, kümeye doğru yerleşmiş olarak kabul edilecektir. Şekil 4.8'de $\gamma \leq 1.0$ şartına bakılarak 100, 200 ve 400 düğüm için oluşturulan yapıda kümeleme doğruluğu verilmiştir. Şekil 4.9'da ise $\gamma \leq 1.5$ şartına bakılarak 100, 200 ve 400 düğüm için oluşturulan yapıda kümeleme doğruluğu verilmiştir.

Şekil 4.8'de 100 düğüm için kümeleme doğruluğu %84, 200 düğüm için kümeleme doğruluğu %86 ve 400 düğüm için kümeleme doğruluğu %85,50 çıkmıştır. Bu değerler ne kadar yüksek çıkarsa, kümeleme doğrulukları da bir o kadar iyi çıkacaktır. Görevdeş ağlarda düğümler sisteme devamlı giriş çıkış yaptıkları düşünüldüğünde Internet ortamında test edilen hiyerarşik yapıda, düğümlerin doğru kümelerde yer aldığını söyleyebiliriz. Literatürde yer alan mOVERLAY yapısının kümeleme doğruluğu $\gamma \leq 1.5$ alınarak test edilmiştir. Geliştirilen hiyerarşik kümeleme yapısında $\gamma \leq 1.0$ alınarak kümeleme doğruluğuna bakılmıştır. mOVERLAY yapısında kümeleme doğruluğu %80

civarında iken geliştirilen ve PlanetLab ortamında test edilen yapıda kümeleme doğruluğu 400 düğüm için düşündüğümüzde %85,50 çıkmıştır. γ parametresi ne kadar yüksek olursa, kümeleme doğruluğu o kadar artar. Şekil 4.9'da mOVERLAY yapısı ile aynı $\gamma \leq 1.5$ parametresi kullanılarak kümeleme doğruluğu gösterilmiştir. Geliştirilen kümeleme yöntemi, $\gamma \leq 1.5$ koşulu ile 400 düğüm için test edildiğinde kümeleme doğruluğu %90,75 çıkmıştır. Sonuç olarak geliştirilen hiyerarşik kümeleme yöntemi literatürde bulunan mOVERLAY yapısından daha iyi bir kümeleme sonucu ortaya çıkarmıştır.



Şekil 4.8. Tasarlanan yapıda 100, 200 ve 400 düğüm için kümeleme doğruluğu grafiği ($\gamma \leq 1.0$).



Şekil 4.9. Tasarlanan yapıda 100, 200 ve 400 düğüm için kümeleme doğruluğu grafiği ($\gamma \leq 1.5$).

Tasarlanan protokolün performans değerleri Tablo 4.1'de verilmiştir.

	100 Düğüm	200 Düğüm	400 Düğüm
Katman Sayısı	*3*	*3*	*3*
Maksimum Küme Kapasitesi	*12*	*15*	*18*
Küme Sayısı	*12*	*18*	*31*
Küme Çapı (Maksimum / Minimum)	*435,21 ms / 27,65 ms*	*606,69 ms / 32,4 ms*	*1213 ms / 36 ms*
Küme Liderine En Uzak Düğüm (Maksimum / Minimum)	*231 ms / 21 ms*	*438 ms / 11 ms*	*587 ms / 14 ms*
Küme İçi Düğümler Arası Uzaklık (Maksimum / Minimum)	*435 ms / 0,11 ms*	*606 ms / 0,098 ms*	*1213 ms / 0,094 ms*
Kümeler Arası Uzaklık (Maksimum / Minimum)	*488 ms / 9 ms*	*603 ms / 0,27 ms*	*1463 ms / 2,31 ms*
Düğümlerin Kümeye Yerleşme Süresi (Maksimum / Minimum)	*11234 ms / 127 ms*	*17152 ms / 185 ms*	*21430 ms/ 216 ms*
Düğümlerin Kümeye Yerleşme Ortalama Süresi	*1,714 sn*	*1,900 sn*	*2,093 sn*
Kümeleme Hassasiyeti (Accuracy)	*% 88,37*	*% 90,17*	*% 89,11*
Kümeleme Doğruluğu ($\chi \leq 1.0$)	*% 84*	*% 86*	*% 85,5*
Kümeleme Doğruluğu ($\chi \leq 1.5$)	*% 88*	*% 88,5*	*% 90,75*

Tablo 4.1. Tasarlanan üstüne bindirmeli hiyerarşik yapının performans kriterleri tablosu.

5. SONUÇ VE ÖNERİLER

Tez çalışmasının öncelikli olarak ilk bölümünde görevdeş ağların genel özelliklerine, gelişim sürecine ve sınıflandırılmasına değinilmiştir. Daha sonra ikinci bölümde üstüne bindirmeli ağların genel yapısı ve literatürde medya akışlandırmasını sağlamak için kümeleme yapan NARADA, NICE, ZIGZAG, mOVERLAY ve NEMO yapıları açıklanmıştır. Üçüncü bölümde ise verimli şekilde medya akışlandırması yapmak için tasarlanan hiyerarşik kümeleme yöntemi detayları ile anlatılmıştır. Son bölümde ise tasarlanan hiyerarşik kümeleme yapısının performansı farklı kriterlere bakılarak incelenmiştir.

Geliştirilen hiyerarşik yapıda kümeleme yapmak için paket dolaşım süresi (RTT) metriği seçilmiştir. Seçilen bu metrik ile paket dolaşım süreleri birbirlerine yakın olan düğümlerin, aynı küme içerisinde bulunmaları hedeflenmiştir. Birbirine yakın olan düğümleri aynı küme içerisinde toplayarak, sistem üzerindeki mesaj trafiği azaltılmış olur. Literatürde bulunan diğer NARADA, NICE, ZIGZAG, mOVERLAY ve NEMO yapıları da kümeleme yapmak için paket dolaşım süresi metriğini kullanmışlardır.

Tez kapsamında hiyerarşik kümeleme yapısı oluşturulurken kullanılan protokollerden bazıları literatürdeki diğer kümeleme yapan sistemler ile benzerlik gösterirken, bazıları da farklılık göstermektedir. Örneğin tasarlanan yapıda, yeni bir düğümün sisteme katılmak amacıyla çalıştırdığı protokol ile literatürde bulunan NICE yapısının aynı işlevi gören protokolü benzerlik göstermektedir. Geliştirilen yapıda küme sürdürülebilirliğini sağlamak için çalıştırılan küme bölünme algoritması, literatürdeki sistemlerin çalıştırdığı küme bölünme algoritmalarından tamamen farklıdır. Ayrıca tasarlanan yapının dirençli hale getirilmesi için çalıştırılan protokoller (örneğin küme liderinin sistemden ayrılması gibi) de literatürde bulunan yapılara göre farklılık göstermektedir.

Tasarlanan hiyerarşik kümeleme yapısında geliştirilen tüm protokoller tamamen dağıtık olarak çalışmaktadır. Örneğin sisteme katılmak isteyen düğüm, sınırlı bilgi ile kendisine en yakın kümeyi bulmaktadır. Kümeleme yapısı için geliştirilen protokollerin sistem üzerinde nasıl çalıştıkları sonlu durum makinesi

ile açıklanmıştır. Ayrıca geliştirilen dağıtık kümeleme protokolünde, en kötü durum senaryosu düşünüldüğünde bir düğümün sisteme getireceği kontrol maliyetinin $O\,(d\,log_d\,N)$ olduğu hesaplanmıştır ve bu maliyetin de kabul edilebilir olduğu görülmektedir.

Literatürde bulunan kümeleme yapıları yerel olarak çalıştırılarak test edilmişlerdir. Tez kapsamında tasarlanan hiyerarşik kümeleme yapısı, test ortamı olan PlanetLab üzerinde 100, 200 ve 400 düğüm için çalıştırılmıştır. Tasarlanan sistemin PlanetLab ortamında çalıştırılması sonucu oluşan yapının kümeleme performansına bakmak için bazı kriterlerden yararlanılmıştır. İncelenen performans kriterleri; küme çapı, küme liderine uzaklık, küme içi düğümler arası gecikme mesafesi, kümelerin birbirleriyle olan gecikme mesafesi, kümelerin doluluk oranları, düğümlerin kümelere yerleşme süreleri, kümeleme hassasiyeti ve kümeleme doğruluğudur. Belirtilen kriterlere bakılarak alınan test sonuçlarına göre küme çapı, küme liderine uzaklık, küme içi düğümler arası gecikme mesafesi, kümelerin birbirleriyle olan gecikme mesafesi değerleri en yüksek 400 düğüm için hesaplanmıştır. Bu da beklenen bir durumdur. Çünkü 400 düğüm için oluşan yapıda fiziksel seviyede daha fazla küme oluşmuştur ve küme içi düğüm sayısı daha fazladır. Kümeleme performansı için bakılan diğer önemli iki kriter ise kümeleme hassasiyeti (accuracy) ve kümeleme doğruluğudur (correctness). Bu iki performans kriteri, 400 düğüm sonucu oluşan yapı üzerinde incelendiğinde kümeleme hassasiyeti %85,5 ve kümeleme doğruluğu da %90,75 çıkmıştır. Literatürdeki mOVERLAY yapısında ise kümeleme hassasiyeti %70 civarında, kümeleme doğruluğu ise %80 civarında çıkmıştır. Tasarlanan yapı sonucu oluşan kümeleme performansı, literatürde bulunan mOVERLAY yapısının kümeleme performansından daha iyi çıkmıştır.

İlerisi için düşünülen, geliştirilen protokolün PlanetLab üzerinde daha fazla sayıda düğüm ile test edilmesi hedeflenmektedir. Tasarlanan yapı üzerinde medya akışlandırması için çoklu gönderim ağaçları oluşturularak literatürdeki diğer medya akışlandırması yapan sistemler ile karşılaştırılması amaçlanmaktadır.

EK 1

Bu bölümde tasarlanan üstüne bindirmeli hiyerarşik kümeleme yapısının performans sonuçlarını gösteren kümülatif dağılım fonksiyonu grafikleri üzerindeki değerler tablolar kullanılarak gösterilecektir.

Ek Açıklamalar-A

Tablo A.1'de küme çapı sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

x (milisaniye)	KDF (x<X) 100 düğüm	KDF (x<X) 200 düğüm	KDF (x<X) 400 düğüm
0	0,133333	0,136364	0,027778
50	0,333333	0,318182	0,166667
100	0,466667	0,318182	0,277778
150	0,533333	0,409091	0,416667
200	0,666667	0,636364	0,527778
250	0,733333	0,636364	0,638889
300	0,8	0,727273	0,722222
350	0,8	0,727273	0,75
400	1	0,818182	0,75
450	1	0,818182	0,805556
500	1	0,818182	0,805556
550	1	0,863636	0,833333
600	1	1	0,833333
650	1	1	0,833333
700	1	1	0,861111
750	1	1	0,916667
800	1	1	0,944445
850	1	1	0,944445
900	1	1	0,944445
950	1	1	0,944445
1000	1	1	0,972222
1050	1	1	0,972222
1100	1	1	0,972222
1150	1	1	0,972222
1200	1	1	1

Tablo A.1. Küme çapı kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.2'de küme liderine uzaklık sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

x (milisaniye)	KDF (x<X) 100 düğüm	KDF (x<X) 200 düğüm	KDF (x<X) 400 düğüm
0	0,066667	0,045455	0,083333
25	0,333333	0,227273	0,194444
50	0,4	0,318182	0,333333
75	0,466667	0,363636	0,444444
100	0,8	0,363636	0,611111
125	0,866667	0,409091	0,611111
150	0,866667	0,545455	0,666667
175	0,933333	0,636364	0,75
200	0,933333	0,727273	0,777778
225	1	0,863636	0,861111
250	1	0,909091	0,944444
275	1	0,954545	0,944444
300	1	0,954545	0,944444
325	1	0,954545	0,944444
350	1	0,954545	0,944444
375	1	0,954545	0,944444
400	1	0,954545	0,944444
425	1	1	0,944444
450	1	1	0,944444
475	1	1	0,944444
500	1	1	0,944444
525	1	1	0,944444
550	1	1	0,944444
575	1	1	1

Tablo A.2. Küme liderine uzaklık kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.3'te küme içi düğümler arası gecikme mesafesi sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

x (milisaniye)	KDF (x<X) 100 düğüm	KDF (x<X) 200 düğüm	KDF (x<X) 400 düğüm
0	0,681384	0,629496	0,602383
50	0,805489	0,745504	0,792815
100	0,920048	0,824191	0,869322
150	0,943914	0,936601	0,932427
200	0,97852	0,973921	0,962211
250	0,98568	0,982014	0,980827
300	0,991647	0,992806	0,985666
350	0,99642	0,995504	0,989762
400	1	0,996853	0,993113
450	1	0,997752	0,995532
500	1	0,997752	0,996277
550	1	0,998651	0,997394
600	1	1	0,997394
650	1	1	0,997766
700	1	1	0,997952
750	1	1	0,998325
800	1	1	0,998511
850	1	1	0,998511
900	1	1	0,998511
950	1	1	0,998511
1000	1	1	0,998883
1050	1	1	0,999069
1100	1	1	0,999069
1150	1	1	0,999628
1200	1	1	1

Tablo A.3. Küme içi düğümler arası gecikme mesafesi kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.4'te kümeler arası gecikme mesafesi sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

x (milisaniye)	KDF (x<X) 100 düğüm	KDF (x<X) 200 düğüm	KDF (x<X) 400 düğüm
0	0,223881	0,205128	0,127615
50	0,373134	0,403846	0,274059
100	0,656716	0,724359	0,426778
150	0,865672	0,903846	0,598326
200	0,880597	0,942308	0,73431
250	0,910448	0,967949	0,807531
300	0,955224	0,98718	0,872385
350	0,970149	0,98718	0,916318
400	0,970149	0,99359	0,933054
450	1	0,99359	0,945607
500	1	0,99359	0,947699
550	1	0,99359	0,951883
600	1	1	0,951883
650	1	1	0,951883
700	1	1	0,951883
750	1	1	0,958159
800	1	1	0,960251
850	1	1	0,962343
900	1	1	0,968619
950	1	1	0,972803
1000	1	1	0,97908
1050	1	1	0,97908
1100	1	1	0,981172
1150	1	1	0,985356
1200	1	1	0,987448
1250	1	1	0,98954
1300	1	1	0,991632
1350	1	1	0,993724
1400	1	1	0,997908
1450	1	1	1

Tablo A.4. Kümeler arası gecikme mesafesi kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.5'te kümelerin doluluk oranını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

x (küme içi düğüm sayısı)	KDF (x<X) 100 düğüm	KDF (x<X) 200 düğüm	KDF (x<X) 400 düğüm
0	0	0	0
3	0	0	0,032258
6	0,666667	0,222222	0,096774
9	0,916667	0,5	0,354839
12	1	0,944444	0,709677
15	1	1	0,935484
18	1	1	1

Tablo A.5. Kümelerin doluluk oranını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.6'da düğümlerin kümelere yerleşme sürelerini gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

x (milisaniye)	KDF (x<X) 100 düğüm	KDF (x<X) 200 düğüm	KDF (x<X) 400 düğüm
0	0,336735	0,462312	0,328283
1000	0,806122	0,79397	0,714646
2000	0,887755	0,854271	0,848485
3000	0,928571	0,884422	0,911616
4000	0,928571	0,914573	0,926768
5000	0,938775	0,934673	0,936869
6000	0,969388	0,954774	0,949495
7000	0,979592	0,959799	0,957071
8000	0,979592	0,959799	0,964646
9000	0,989796	0,964824	0,967172
10000	0,989796	0,974874	0,972222
11000	1	0,98995	0,987374
12000	1	0,98995	0,987374
13000	1	0,98995	0,987374
14000	1	0,994975	0,989899
15000	1	0,994975	0,992424
16000	1	0,994975	0,997475
17000	1	1	0,997475
18000	1	1	0,997475
19000	1	1	0,997475
20000	1	1	0,997475
21000	1	1	1

Tablo A.6. Düğümlerin kümelere yerleşme sürelerini gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.7'de kümeleme hassasiyeti sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

Küme Sayısı	Kümeleme Hassasiyeti (%) - 100 düğüm	Kümeleme Hassasiyeti (%) - 200 düğüm	Kümeleme Hassasiyeti (%) - 400 düğüm
2	83,06	88,7	90,81
3	87,97	92,41	87,48
4	91,66	90,8	87,76
5	92,11	89,99	89,19
6	90,87	88,85	89,81
7	90,55	89,39	89,83
8	89,93	89,23	89,12
9	89,43	89,17	89,91
10	88,88	89,34	90,31
11	88,73	89,38	90,25
12	88,37	89,75	91,7
13		89,91	92,4
14		89,76	92,05
15		89,86	91,68
16		90,17	91,67
17		90,28	91,05
18		90,17	90,74
19			90,56
20			90,26
21			89,98
22			89,95
23			89,81
24			89,48
25			89,27
26			89,05
27			89,06
28			89,09
29			88,95
30			89,3
31			89,11

Tablo A.7. Kümeleme hassasiyeti sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.8'de $\gamma \leq 1.0$ şartına bakılarak kümeleme doğruluğu sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

Küme Sayısı	Kümeleme Doğruluğu (%) - 100 düğüm	Kümeleme Doğruluğu (%) - 200 düğüm	Kümeleme Doğruluğu (%) - 400 düğüm
2	88,89	92	94,12
3	90	91,67	93,62
4	89,47	91,11	93,75
5	86,67	89,09	93,59
6	86,79	89,55	93,48
7	86,44	89,33	93,64
8	84,85	86,05	92,74
9	84,72	87,13	92,91
10	85,37	87,72	92,21
11	85,56	88,19	91,57
12	84	87,41	91,85
13		86,49	91,84
14		86,96	91,3
15		86,39	91,08
16		86,44	90,91
17		86,17	89,61
18		86	88,31
19			88,42
20			88,77
21			89,08
22			87,75
23			87,86
24			87,89
25			87,76
26			87,97
27			87,22
28			87,12
29			86,47
30			86,05
31			85,5

Tablo A.8. $\gamma \leq 1.0$ şartına bakılarak kümeleme doğruluğu sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

Tablo A.9'da $\gamma \leq 1.5$ şartına bakılarak kümeleme doğruluğu sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler verilmiştir.

Küme Sayısı	Kümeleme Doğruluğu (%) - 100 düğüm	Kümeleme Doğruluğu (%) - 200 düğüm	Kümeleme Doğruluğu (%) - 400 düğüm
2	88,89	92	94,12
3	90	91,67	93,62
4	89,47	91,11	93,75
5	88,89	90,91	93,59
6	88,68	91,04	93,48
7	88,14	90,67	93,64
8	87,88	89,53	92,74
9	87,5	90,1	92,91
10	87,8	90,35	92,86
11	87,78	90,55	92,77
12	88	90,37	92,93
13		89,19	92,86
14		89,44	92,75
15		89,35	92,49
16		89,27	92,27
17		88,83	91,77
18		88,5	91,94
19			91,89
20			92,03
21			92,15
22			92,05
23			92,01
24			91,93
25			91,94
26			91,98
27			91,94
28			91,78
29			91,51
30			91,21
31			90,75

Tablo A.9. $\gamma \leq 1.5$ şartına bakılarak kümeleme doğruluğu sonuçlarını gösteren kümülatif dağılım fonksiyonu grafiği üzerindeki değerler tablosu.

KAYNAKLAR DİZİNİ

Amir Y., Awerbuch B., Danilov C. and Stanton J., 2002, Global flow control for wide area overlay networks: a cost-benefit approach, *Proceedings of IEEE Open Architectures and Network Programming (Openarch)*, 155-166pp.

Andersen D., Balakrishnan H., Kaashoek M. and Morris R., 2001, Resilient Overlay Networks, *Proceedings of ACM SOSP*.

Banerjee S., Bhattacharjee B. and Kommareddy C., 2002, Scalable application layer multicast, *Proceedings of ACM SIGCOMM*, 205-217pp.

Birrer S. and Bustamante F. E., 2005, Resilient peer-to-peer multicast without the cost, *Proceedings of MMCN*.

Chen Y., 2006, Canicula: An improved hybrid overlay networks, *Proceedings of the 14th IEEE International Conference on Networks (ICON'06)*.

Chu Y., Rao S. G., Seshan S. and Zhang H., 2002, A case for end system multicast, *IEEE Journal on Selected Areas in Communication (JSAC), Special Issue on Networking Support for Multicast*, 20(8), 1456-1471pp.

Clarke I., Sandberg O., Wiley B. and Hong T. W., 2000, Freenet: A distributed anonymous information storage and retrieval system, *Lecture Notes in Computer Science*.

Crespo A. and Garcia-Molina H., 2002, *Semantic Overlay Networks for P2P Systems*, Technical report, Stanford University.

Good N. S. and Krekelberg A., 2003, Usability and privacy: a study of KaZaA P2P file-sharing, *Proceedings of the conference on Human factors in computing systems*, ACM Press, 137-144pp.

Li X. and Wu J., 2005, Searching Techniques in Peer-to-Peer Networks. Handbook of Theoretical and Algorithmic Aspects of Ad Hoc, Sensor, and Peer-to-Peer Networks, Chapter 37.

Lu J. and Callan J., 2003, Content-Based Retrieval in Hybrid Peer-to-Peer Networks, *Proceedings of the Twelfth International Conference on Information and Knowledge Management (CIKM'03)*.

KAYNAKLAR DİZİNİ (devam)

Liang J., Kumar R. and Ross K. W., 2006, The FastTrack overlay: A measurement study, Computer and Telecommunications Networking, 50(6), 842-858pp.

LittleShoot, *"P2P File Sharing Browser Plugin LittleShoot – BitTorrent - Gnutella", http://www.littleshoot.org/* (Erişim tarihi: 15 Ağustos 2010).

Lua E. K., Crowcroft J., Pias M., Sharma R. and Lim S., 2005, A survey and comparison of peer-to-peer overlay network schemes, *IEEE Communications Surveys & Tutorials, The Electronic Magazine of Orginal Peer-Reviewed Survey Articles*, 7(2), Second Quarter.

Lv Q., Cao P., Cohen E., Li K. and Shenker S., 2002, Search and replication in unstructured peer-to-peer networks, *Proceedings of the 16th ACM International Conference on Supercomputing (ACM ICS'02)*.

NextShare, *"P2P Next", http://p2p-next.org/* (Erişim tarihi: 15 Ağustos 2010).

Ng E. and Zhang H., 2002, Predicting internet network distance with coordinates-based approaches, *Proceedings of IEEE INFOCOMM'02*.

Pendarakis D., Shi S., Verma D. and Waldvogel R., 2001, ALMI: An application level multicast infrastructure, *Proceedings of 3rd Usenix Symposium on Internet Technologies and Systems.*

PlanetLab, *"PlanetLab | An open platform for developing, deploying, and accessing planetary-scale services", http://www.planet-lab.org/* (Erişim tarihi: 15 Ağustos 2010).

Pourebrahimi, B., Bertels K. and Vassiliadis S., 2005, A survey of peer-to-peer networks, *Proceedings of the 16th Annual Workshop on Circuits, Systems and Signal Processing*.

Ratnasamy S., Francis P., Handley M., Karp R. and Shenker S., 2001, A scalable content-addressable network, *ACM SIGCOMM Conference*, 161-172pp.

Ripeanu M., Foster I. and Iamnitchi A., 2002, Mapping the Gnutella Network: Properties of large-scale peer-to-peer systems and implications for system design, *IEEE Internet Computing Journal*, 6(1).

KAYNAKLAR DİZİNİ (devam)

Ruixiong T., Yongqiang X., Qian Z., Bo L., Ben Z. Y. and Xing L., 2005, Hybrid overlay structure based on random walks, *Proceedings of the 4th International Workshop on Peer-To-Peer Systems (IPTPS'05)*.

Saltzer J., Reed D. and Clark D., 1984, End-to-end arguments in system design, *ACM Trans. Compt. Syst.*, 2, 195-206pp.

Sayıt M. F., 2010, H.264 Çok katmanlı video codec ile p2p çoklu gönderim sistemlerinde video akışlandırma, Doktora Tezi, Ege Üniversitesi Uluslararası Bilgisayar Enstitüsü (yayınlanmamış).

Sayıt M. F., Tunalı E. T. and Tekalp A. M., 2009, Bandwidth-aware multiple multicast tree formation for p2p scalable video streaming using hierarchical clusters, *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 945-948pp.

Stoica I., Morris R., Karger D., Kaashoek F. and Balakrishnan H., 2001, Chord: A scalable peer-to-peer lookup service for internet applications, *Proceedings of the ACM SIGCOMM Conference*, 149-160pp.

Tran D. A., Hua K. A. and Do T. T., 2004, A peer-to-peer architecture for media streaming, *IEEE Journal on Selected Areas in Communications*, 22, 121-133pp.

Wang, C. and Li B., 2003, *Peer-to-Peer Overlay Networks: A Survey*, Technical Report.

Zhang X., Liu J., Zhang Q. and Zhu W., 2002, gMeasure: A group-based network performance measurement service for peer-to-peer applications, *presented at the GLOBECOM*.

Zhang X. Y., Zhang Q., Zhang Z., Song G. and Zhu W., 2004, A construction of locality-aware overlay network: mOverlay and its performance, *IEEE J. Select. Areas Commun., Special Issue on Recent Advances in Service Overlay Networks*, 22, 18-28pp.

Zhao B. Y., Huang L., Stribling J., Rhea S. C., Joseph A. D. and Kubiatowicz J. D., 2004, Tapestry: A resilient global-scale overlay for service deployment, *IEEE JSAC*, 22(1).

EKLER

İngilizce - Türkçe Sözlük

Accuracy	Kümeleme hassasiyeti
Adaptive	Uyarlanabilir
Administrative	Yönetimsel
Aggressively Probing	Israrlı sorgulama
Application Layer Multicast	Uygulama seviyesinde çoklu gönderim
Application Layer Overlay Network	Uygulama seviyesinde üstüne bindirmeli ağ
Backbone	Omurga
Boot Host	Başlatma kullanıcısı
Bottleneck	Darboğaz
Broadcast	Tüme gönderim
Candidate Group	Aday grup
Client-Server	İstemci-Sunucu
Cluster Merge	Küme birleşmesi
Cluster Split	Küme bölünmesi
Clustering	Kümeleme
Co-Leader	Yardımcı lider
Complexity	Karmaşıklık
Congestion	Tıkanıklık
Connected Graph	Bağlı çizge
Connectivity	Bağlanabilirlik
Content Distribution	İçerik dağıtımı
Correctness	Kümeleme doğruluğu
Crew	Takım
Cumulative Distribution Function	Kümülatif dağılım fonksiyonu
Data Delivery Tree	Veri gönderim ağacı
Data Replication	Veri kopyalama
Decentralized	Merkezi olmayan
Delivery Time	Gönderim süresi
Deployment	Konuşlanma

Distributed Hash Tables	Dağıtık hesaplama tabloları
Distributed Storage	Dağıtık depolama
Dynamic Landmark	Dinamik işaretleme
End System Multicast	Uç sistem çoklu gönderimi
End User	Uç kullanıcı
Fault Tolerant	Hata toleranslı
File Sharing	Dosya paylaşımı
Flooding	Selbasma
Forwarding	İletim
Fragments	Küçük parçalar
Framework	Taslak
Gossipping	Dedikodu
Heterogeneity	Heterojenlik
Hop	Zıplama
Host Departure	Üye ayrılması
Hybrid	Karma
Hypertext Transfer Protocol	Hipermetin aktarma iletişim protokolü
Identical	Özdeş
Infrastructure	Alt yapı
Instant Messaging	Anlık mesajlaşma
Load Balanced	Yük dengeli
Maintenance	Sürdürülebilirlik
Membership Management	Üyelik yönetimi
Mesh	Örgü
Metadata	Üstveri
Multi Source	Çok kaynaklı
Multicast	Çoklu gönderim
Negative Acknowledgement	Olumsuz teyit
Network Mapping	Ağ haritalama
Node	Düğüm
Overlay	Üstüne bindirmeli
Packet Loss Rate	Paket kayıp oranı
Packet Replication	Paket kopyalama

Peer-to-Peer	Görevdeş
Physical Link	Fiziksel hat
Policy	Politika
Privacy	Gizlilik
Pruned	Azaltılmış
Quality of Service	Servis kalitesi
Query Processing	Sorgu işleme
Recipient	Alıcı
Rendezvous Point	Randevu noktası
Resilient Overlay Network	Dirençli üstüne bindirmeli ağ
Round Trip Time	Paket dolaşım süresi
Router	Yönlendirici
Self Organized	Kendi kendine organize olmak
Semantic Overlay Network	Anlamsal üstüne bindirmeli ağ
Single Point of Failure	Tek nokta hatası
Spanning Tree	Kapsayan ağaç
Structured	Yapısal
Super Cluster	Süper küme
Super Peer	Süper eş
Tag	Etiketlemek
Tightly	Daha sıkı
Time To Live	Yaşam süresi
Tracker	Takipçi
Triangulated Heuristic	Üçgenlere bölme yaklaşımı
Unicast	Tek yönlü gönderim
Unstructured	Yapısal olmayan
User Datagram Protocol	Kullanıcı veri bloğu iletişim protokolü
Virtual	Sanal
Wide Area	Geniş alan

ÖZGEÇMİŞ

Sercan DEMİRCİ
Adres: Uluslararası Bilgisayar Enstitüsü 35100 Bornova/İZMİR
e-mail: sercan.demirci@ege.edu.tr, sercan.ube@gmail.com

Eğitim Durumu

Yüksek Lisans	:	2008 - , Ege Üniversitesi, Uluslararası Bilgisayar Enstitüsü
Lisans	:	2004 – 2008, Doğu Akdeniz Üniversitesi Bilgisayar Mühendisliği
Lise	:	1996 – 2003, Samsun Anadolu Lisesi

Yabancı Dil

Türkçe	:	anadil
İngilizce	:	iyi derecede
Almanca	:	orta

Bilgisayar dilleri ve program deneyimi

- C/C++, C#, Java, Assembly, OpenGL(C/C++), Python, ASP.NET
- Microsoft Visual Studio 2008 Professional, Eclipse IDE, NetBeans IDE, Microsoft Windows Office

Projeler

- ASP.NET ile online konferans programı
- C# ve Microsoft SQL Server 2005 ile hasta takip programı
- C# ve Microsoft SQL Server 2005 ile otel rezervasyon programı
- Assembly ile led aydınlatma paneli
- C ile öğrenci bilgi sistemi takip programı
- Mikroişlemciler ile devre tasarımı
- C# ile SQL Server benzeri uygulama programı
- ASP.NET ile online satış programı
- NS2 ile ağ protokollerinin simülasyonu
- Sayısal veri iletişiminde kullanılan protokollerin CLOUD programı ile simülasyonu
- OpenGL ile basketbol oyunu programı

- UDP soketleri kullanılarak TCP protokolünün gerçeklenmesi
- Chang Roberts kapsayan ağaç algoritmasının telsiz duyarga ağlarında TOSSIM ile simülasyonu
- Görevdeş Ağlarda Ölçeklenebilir Gruplandırma ve Video Akışlandırma – Tübitak 1001 (2008 - ,)

Yayınlar

- Hasan Bulut, Asil Yardımcı, Sercan Demirci, Yağız Kaymak, Müge Fesci Sayıt and E. Turhan Tunalı, "*An Efficient JSD-Based Search on Interest-Based Hierarchical Clustering of Overlay Networks*", The Second International Conference on Advances in P2P Systems (AP2PS 2010).
- Onur Yılmaz, Sercan Demirci, Yağız Kaymak and Kayhan Erciyeş, "*Synchronous Distributed Spanning Tree Algorithm for Wireless Sensor Networks*", ISCSE 2010.
- Yağız Kaymak, Onur Yılmaz and Sercan Demirci, "*The Chang Roberts Distributed Spanning Tree Algorithm Simulation on TinyOS Simulator in Wireless Sensor Networks*", ICAST 2010.

Eğitim Katılımı

- Bilişim haftası kapsamında "Visual Studio 2008 ve Getirdiği Yenilikler" – DAÜ (2008)
- Robotik Eğitim Programı – DAÜ (2008)

Sertifikalar

- 2004-2005 Güz dönemi yüksek şeref öğrencisi sertifikası (3,64 / 4)
- 2004-2005 Bahar dönemi yüksek şeref öğrencisi (3,87 / 4)
- 2005-2006 Güz dönemi şeref öğrencisi (3,32 / 4)
- 2005-2006 Bahar dönemi şeref öğrencisi (3,11 / 4)
- 2006-2007 Güz dönemi şeref öğrencisi (3,44 / 4)
- 2006-2007 Bahar dönemi şeref öğrencisi (3,36 / 4)
- 2007-2008 Güz dönemi yüksek şeref öğrencisi (3,94 / 4)
- 2007-2008 Bahar dönemi yüksek şeref öğrencisi (3,71 / 4)