T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ AD-HOC AĞLAR İÇİN OĞUL ZEKÂSI TABANLI YENİ BİR YÖNLENDİRME PROTOKOLÜ

DOKTORA TEZİ

Zafer ALBAYRAK

Enstitü Anabilim Dalı	**:**	**ELEKTRONİK VE BİLGİSAYAR EĞİTİMİ**
Tez Danışmanı	**:**	**Doç. Dr. Ahmet ZENGİN**

Mart 2014

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ AD-HOC AĞLAR İÇİN OĞUL ZEKÂSI TABANLI YENİ BİR YÖNLENDİRME PROTOKOLÜ

DOKTORA TEZİ

Zafer ALBAYRAK

Enstitü Anabilim Dalı : ELEKTRONİK VE BİLGİSAYAR EĞİTİMİ

Bu tez 28/03/2014 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

Prof. Dr. **Etem KÖKLÜKAYA** **Jüri Başkanı**	**Doç. Dr.** **Cüneyt BAYILMIŞ** **Üye**	**Doç. Dr.** **Ahmet ZENGİN** **Üye**
Doç. Dr. **Resul KARA** **Üye**		**Doç. Dr.** **İhsan PEHLİVAN** **Üye**

TEŞEKKÜR

Doktora eğitimim süresince desteklerini hiçbir zaman esirgemeyen danışmanım Doç. Dr. Ahmet ZENGİN'e çok teşekkür ederim. Tez izlemelerimde bana sürekli destek olan Doç. Dr. Cüneyt BAYILMIŞ ve Doç. Dr. İhsan PEHLİVAN hocalarıma teşekkür ederim. Tez ile ilgili çalışmalarımda bana yardımlarını esirgemeyen Yrd. Doç. Dr. Bülent ÇOBANOĞLU ve Yrd. Doç. Dr. Fatih ÇELİK' e teşekkür ederim. Bunun yanında çalışmam boyunca bana desteklerini esirgemeyen aileme teşekkür ederim.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

ABC	: Karınca Koloni Tabanlı Yönlendirme Algoritması
ABR	: Oğul Zekâ Kabanlı Yönlendirme Algoritması
Acc	: Toplayıcı Arı (Accumulator)
ACLR	: Yer Farkındalıklı Karınca Kolonisi Yönlendirme
Ad-Hoc	: Hareketli Eş-Eş Kablosuz Ağlar
AI	: Yapay Zekâ (Artificial Intelligence)
AntNet	: Karınca Ağı Yönlendirme Protokolü
AODV	: Ad Hoc İsteğe Bağlı Uzaklık Vektörü Yönlendirme Protokolü
BeeAdhoc	: Bal Arıları Tabanlı Yönlendirme Protokolü
Bee-MANET	: Oğul Zekâsı Tabanlı Yönlendirme Protokolü
Bscout	: Geri Öncü Arı
BSS	: Baz İstasyonu Sistemi (Base Station System)
CM	: Birleşik Model (Coupled Model)
CBRGEN	: Ağ Trafiği Oluşturma Komutu
DECT	: Dijital Geliştirilmiş Kablosuz İletişim
DEVS	: Ayrık Olaylı Sistem Tanımı
DSDV	: Hedef Sıralı Uzaklık Vektörü
DSR	: Dinamik Kaynak Yönlendirme
DV	: Uzaklık Vektörü
EF	: Deneysel Çerçeve (Experimental Frame)
Fscout	: İleri Öncü Arı
GGSR	: Kümelenmiş Ağ Geçidi Anahtar Yönlendirme
GloMoSim	: Global Mobile Information System Simulator
GPRS	: Genel Paket Radyo Servisi (General Packet Radio Service)

GSM	: Küresel Hareketli İletişim Sistemi (Global System for Mobile)
GUI	: Grafiksel Kullanıcı Arayüzü (Graphical User Interface)
ID	: Kimlik Numarası
IEEE	: Elektrik Elektronik Mühendisleri Enstitüsü
IP	: İnternet Protokolü
ISM	: Industrial Scientific Medical
KTMY	: Konum Tabanlı Melez Yönlendirme Algoritması
LAN	: Yerel Alan Ağı (Local Area Network)
LS	: Bağlantı Durum
M&S	: Modelleme ve Benzetim
MAC	: Veribağı Katmanı
MANET	: Hareketli Gezgin Ağlar (Mobile ad hoc networks)
Mbps	: Megabits Per Second
MRP	: Çoklu Geçiş Protokolü
MS	: Mobil İstasyon (Mobile Station)
n	: Düğüm Sayısı
NSF	: Ulusal Bilim Vakfı (National Science Foundation)
Ns-2	: Network Simulator 2
Ns-3	: Network Simulator 3
NSS	: Anahtarlamalı Ağ Sistemi (Network Switching System)
OMNET++	: Objective Modular Network Test-bed in C++
OSI	: Açık Sistem Bağlantısı (Open System Interconnection)
OSPF	: Open Shortest Path First
PDNS	: Paralel/Dağıtık ağ benzeticisi
PİO	: Paket İletim Oranı (Packet Delivery Ration)
RIP	: Yönlendirme Bilgi protokolü
SI	: Oğul Zekâ (Swarm Intelligence)
SETDEST	: Topoloji Üretme Komutu
SSR	: Sinyal Sabitliği Tabanlı Uyarlamalı Yönlendirme
TCP	: İletim Kontrol Protokolü (Transmission Control Protocol)
TORA	: Geçici Sıralı Yönlendirme Algoritması

TCL	: Ağ Programlama Dili
TTL	: Yaşama Zamanı (Time to Live)
UDP	: Kullanıcı Veribloğu İletişim Kuralları
UUG	: Uçtan Uca Gecikme (End-to-End Delay)
VINT	: Sanal İnternet Testi(Virtual InterNetwork Testbed)
WIMAX	: Mikrodalga Erişimi için Küresel Çalışabilirlik
WLAN	: Kablosuz Yerel Alan Ağı
WRP	: Kablosuz Yönlendirme Protokolü
WSN	: Kablosuz Algılayıcı Ağlar (Wireless Sensor Network)
ZHLS	: Bölge Tabanlı Hiyerarşik Bağlantı Durum
ZRP	: Bölge Yönlendirme Protokolü

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Hareketli Ad-Hoc ağlar, Yönlendirme Protokolleri, Oğul Zekâsı.

İletişim teknolojilerindeki gelişmeler kablosuz cihazların daha güçlü ve daha ucuz olmalarını sağlamıştır. Böylece hızlı teknolojik gelişmeler internete bağlanan cihaz sayısının artmasına sebep olmuştur. Hareketli Ad-Hoc ağlarda araştırmacılar tarafından kabul edilen en önemli konulardan biri de yönlendirme protokolleridir. Yönlendirme protokolü geliştirme çalışması, kablosuz sistemlerde karmaşık, ölçeklenebilirlik, uyum, verimlilik ve pil ömrü gibi problemler ile ilgilenir. Kablosuz yönlendirme protokolleri bu sorunların üstesinden gelmek için geliştirilmiştir. Oğul zekâsı; termitler, arılar, karıncalar, kuşlar, balık sürüleri gibi aralarında etkileşim olan böceklerin veya diğer sosyal hayvanların topluluk halindeki davranışlarını örnek alarak, problemlere çözüm getirmeyi amaçlayan bir yapay zekâ tekniğidir.

Oğul zekâsı çözümleri dağıtık sistemlere ve ağ problemlerine uygulandıkları zaman ölçeklenebilirlik, hata toleransı, uyarlanabilirlik, modülerlik, paralellik ve otonomi gibi özelliklerinden dolayı geniş ve yüksek derecede dinamik sistemler için oldukça uygundur. Bu alanda geliştirilmiş protokollere örnek olarak ABC, AntNet, Adaptive-SDR, Beeadhoc algoritması ve Hopnet algoritması verilebilir.

Bu tez çalışmasında, Ad-Hoc ağlardaki problemlere çözüm getirmek amacıyla;

1. Ad-Hoc ağlar için oğul zekâsı tabanlı yeni bir yönlendirme protokolü geliştirilmiştir. Geliştirilen bu protokol Bee-MANET olarak adlandırılmıştır.
2. Geliştirilen protokolü modellemek ve benzetimini yapmak amacıyla büyük ölçekli ağları destekleyen, model kütüphanesi zengin ve güçlü Ns-2 ağ simülatörü kullanılmıştır.
3. Geliştirilen protokolün üstünlüklerini göstermek amacıyla farklı ölçeklerden oluşan ağlar modellendi. Modellenen ağlar farklı trafik ve topolojilerde çalıştırılarak, geliştirilen Bee-MANET protokolü literatürdeki önemli protokoller olan AODV ve BeeAdhoc yönlendirme protokolü ile karşılaştırılarak başarımı incelendi.
4. Gerçekleştirilen uygulamalarda, geliştirilen yönlendirme protokolünün belirlenen problemlere çözüm getirdiği gözlemlendi.

A NEW SWARM SWARM INTELLIGENCE BASED ROUTINGPROTOCOL FOR WIRELESS AD-HOC NETWORKS

SUMMARY

Key Words: Mobile Ad-Hoc networks, Routing Protocols, and Swarm Intelligence.

The development of communication technology has made wireless equipment's less, more powerful and less expensive. Such rapid technology improvement has contributed great growth to mobile devices connected to the Internet. One of the main fields adopted by researchers studying on Mobile Ad-Hoc Networks is to develop routing protocols in wireless systems. Routing protocol development is related to complexity, scalability, adaptability, productivity, and battery life in wireless systems. Routing protocols for wireless systems are developed in order to cope with these problems. Swarm intelligence solutions to problems when they are applied to distributed systems and network scalability, fault tolerance, adaptability, modularity, parallelism and autonomy of the properties are spacious and are well suited for highly dynamic systems. Examples of protocols developed in the art for example, ABC, Antnet, Adaptive-SDR, Beeadhoc Hopnet algorithm are given.

In this thesis, routing protocols, network throughput and packet delivery ratio in order to solve problems such as;

1. A new routing protocol for Mobile Ad Hoc Networks has been developed by inspired swarm intelligence of the honey bees in order to bring solutions to network throughput and packet delivery ration. Developed protocol called Bee-MANET.
2. Bee-MANET has been compared to BeeAdhoc and AODV protocols which are most known protocols in the network community. Using the Ns-2 network, we develop simulation models of networks with varying topologies and scales. The results were presented as graphs and evaluated.
3. Different networks modeled to demonstrate the superiority of the developed protocol and were run in different traffic and topology. Bee-MANET, AODV, and Beeadhoc algorithms are empirically compared to research large-scale behavior. The results were presented as graphs and were evaluated.
4. Bee-MANET brings the solutions to the problems such as network throughput and packet delivery ratio.

BÖLÜM 1. GİRİŞ

1.1. Giriş

Kablosuz ağlar ilk olarak ortaya çıktığı 1970'li yıllardan beri oldukça yaygın olarak kullanılmaya başlanmıştır. Kullanıcının coğrafi konumu nerede olursa olsun bilgiye erişim imkânı sağlaması en büyük özelliklerinden biridir. İnternetin gelişimiyle beraber kablosuz ağ sistemlerinin gelişimi de hızlı bir şekilde artmıştır. Son yıllarda kullanımı hızla artan mobil telefonlar ve kablosuz yerel alan ağı (Wireless Local Area Network-WLAN) özelliğine sahip taşınabilir bilgisayarlar kablosuz iletişimin önemini arttırmıştır. Günümüzde iletişimle ilgili olarak kablolu veya kablosuz olmak üzere iki tür ağ bulunmaktadır. Kablosuz ağlar yapısına göre alt yapılı ve Ad-Hoc ağlar olmak üzere iki gruba ayrılır. Alt yapılı ağlarda düğümlerin birbiri ile haberleşmesi için merkezi düğüm kullanılır. Hareketli Ad-Hoc ağlarda merkezi bir düğüm yoktur. Düğümler arası haberleşme düğümlerin birbiri üzerinden gerçekleştirilir. Uçtan uca iletimde kablosuz ağlar kablolu ağlara göre farklılıklar içermektedir.

1.2. Problem Tanımı

Günümüzde kablosuz Ad-Hoc ağ sistemleri aşağıdaki problemlere sahiptir[1]:

1. Karmaşıklık; birden fazla protokolün ve teknolojinin bir arada olması,
2. Ölçeklenebilirlik; ağdaki düğüm sayısının çok fazla olduğu durumlarda ağda dolaşan aşırı paketten dolayı bazı paketlerin genel kimlik (ID) numarası olmayabilir, bu nedenle kalabalıktan kaynaklanan tıkanma ve çarpışmalar olabilir.

3. Uyum yeteneği; Ad-Hoc ağlarının topolojisi çok sık değişir. Ad-Hoc bir ağın sürekli olarak değişen şartlara göre düğümlerin bağlantılarını tanımlaması ve yayınlaması gerekir.
4. Beka; bir veya daha fazla düğümün devre dışı kalması sistemin çalışmasını etkilememelidir.

Son zamanlarda Ad-Hoc ağlarla ilgili olarak yapılan çalışmalardan oluşturulan yönlendirme protokollerinde halen Ad-Hoc ağlarında bulunan karmaşıklık, ölçeklenebilirlik, uyum yeteneği ve beka gibi sorunların devam ettiği görülmektedir. Yukarıda bahsedilmiş olan problemler dikkate alındığında Ad-Hoc ağlarla ilgili olarak yönlendirme protokollerinin sürekli olarak geliştirileceği görülmektedir. Mevcut üretilen protokollerin her birinin tüm durumlar ve değerlendirme ölçütlerinde de tam olarak birbirlerine karşı üstünlüğünün olamaması araştırmacıları yeni yönlendirme protokolleri üzerine çalışmaya sevk etmektedir. Bu nedenle bu çalışmada, kablosuz Ad-Hoc ağların yukarıda bahsedilen eksikliklerini gidermek üzere, oğul zekâsı tabanlı yeni bir yönlendirme protokolü geliştirilmiştir.

1.3. Literatür Çalışması

Bu bölümde literatürdeki hareketli Ad-Hoc ağlarla ilgili olarak geliştirilmiş olan yönlendirme protokollerine yer verilmiştir. Geliştirilmiş olan bu yönlendirme protokolleriyle ilgili olarak geniş bir bilgi Bölüm 3'de verilmiştir.

Kablolu ağlarda kaynak ile hedef arasındaki yolun tespit edilmesi için uzaklık vektörü (Distance Vektor-DV) ve bağlantı durum (Link State–LS) yönlendirme algoritmalarını esas alan protokoller geliştirilmiştir [2]. Kablosuz ağlarda düğümlerin sürekli hareketi, sınırlı bant genişliği düğümlerin sürekli ağa bağlanması veya ağdan ayrılması ve radyo dalgalarının çevresel faktörlerden etkilenmesi gibi sebeplerden dolayı kablolu ağ yönlendirme protokollerinin hareketli Ad-Hoc ağlarda aynen kullanılması mümkün değildir. Bundan dolayı hareketli Ad-Hoc ağların kendine has karakteristik özelliklerinden dolayı yönlendirme protokolleri farklılıklar göstermektedir.

Hareketli Ad-Hoc ağlar için geliştirilen yönlendirme protokollerini tabloya-dayalı, isteğe bağlı, hibrit ve oğul zekâsı tabanlı olmak üzere dört ana sınıfa ayırabiliriz.

Tabloya-dayalı yönlendirme protokollerinde tüm ağa ait yönlendirme bilgileri sürekli güncellenir. Ağdaki topoloji değişikliği ile yönlendirme tabloları sürekli güncellenmiş olur. Bu yöntemle geliştirilmiş olan yönlendirme protokolleri: DSDV (Destination Sequenced Distance Vector, 1994), WRP (Wireless Routing Protocol, 1995), CGSR (Cluster head Gateway Switch Routing, 1997), GSR (Global State Routing, 1998), FSR (Fisheye State Routing, 1999), HSR (Hierarchical State Routing, 1999), STAR (Source Tree Adaptive Routing, 2000), WRPLite (Wireless Routing Protocol Lite, 2000), IARP (Intrazone Routing Protocol, 2002), DFR (Direction Forward Routing, 2006) olarak sıralanabilir.

İsteğe-bağlı yönlendirme protokollerinde, yönlendirme tabloları sürekli güncellenmez. Eğer bir düğüm başka bir düğüme paket gönderme isteğinde bulunursa yol bulma işlemi başlar ve böylece yönlendirme tabloları güncellenir. Bu yöntemle geliştirilmiş olan yönlendirme protokolleri: DSR (Dynamic Source Routing, 1996), ABR (Associativity Based Routing, 1996), TORA (Temporally Ordered Routing Algorithm, 1997), SSR (Signal Stability Routing, 1997), PAR (Power Aware Routing, 1998), LAR (Location Aided Routing, 1998), CBR (Cluster Based Routing, 1999), AODV (Ad hoc On-Demand Distance Vector Routing, 1999), Dynamic NIx-Vector Routing (2005), DYMO (Dynamic Manet On-demand Routing, 2006), MAODV (Multirate Ad hoc On Demand Distance Vector Routing, 2007) olarak sıralanabilir.

Hibrit yönlendirme protokolleri hem tabloya dayalı hem de isteğe bağlı yönlendirme protokollerinin özelliklerini taşımaktadır. Düğümleri coğrafi konumlarına göre değerlendirip geliştirilen konum tabanlı yönlendirme protokolleri de bu sınıfa girer. Bu yöntemle geliştirilmiş olan yönlendirme protokolleri: MFR (Most Forward With in Radius, 1984), ZHLS (Zone based Hierarchical Link State, 1999), ZRP (Zone Routing Protocol, 2002), MPR (Multi Point Relaying, 2005).

Oğul zekâsı tabanlı yönlendirme protokolleri; Termitler, arılar, karıncalar, kuşlar, balık sürüleri gibi aralarında etkileşim olan böceklerin veya diğer sosyal canlıların topluluk halindeki davranışlarını örnek alarak, problemlere çözüm getirmeyi amaçlayan bir tür yapay zekâ tekniği kullanır. Bu yöntemle geliştirilmiş olan yönlendirme protokolleri: ABC (1996), AntNet (1998), Adaptive-SDR (2002), BeeAdhoc (2006), Hopnet (2008), Fuzzy Ant Colony Based Routing Protocol (2009).

1.4. Çözüm önerisi

Hareketli Ad-Hoc ağlar için daha önce oğul zekâsı tekniği kullanılarak oluşturulmuş yönlendirme protokollerinden elde edilmiş olan simülasyon sonuçları incelendiğinde bu yolla oluşturulmuş olan yönlendirme protokollerinin geleneksel yollarla elde edilmiş olan yönlendirme protokollerine göre benzer / daha iyi sonuçlar verdiği görülmüştür [3]. Böylece, daha önce bahsedilmiş olan hareketli Ad-Hoc ağlardaki problemlerin azaltılması ve ağ çıkışının arttırılması yolunda oluşturulacak olan yeni bir yönlendirme algoritmasının oğul zekâsı tabanlı oluşturulmasına karar verilmiştir.

Bal arılarının yiyecek aramalarından ilham alınarak oluşturulan iletişim teknikleri kendi kendini organize edebilen zeki yapılı öncülerle yapıldığında daha iyi sonuç verdiği bilinmektedir [3,4]. Bu nedenle yapay zekâsı tekniğinin içerisinde bulunan oğul zekâsı, biyolojik sistemlerden elde edilip mühendislik sistemlerinde ağ iletişiminde olduğu gibi birçok güçlü özellikler göstermiştir. Buna ilave olarak zeki biyolojik canlıların özellikleri alınarak yeni tasarımlar gerçekleştirilmiştir. Gelecekteki tasarım şekli biyolojik canlıların birbiri ile basit iletişim şekli ile karmaşık ve zeki küresel davranışların kullanımının sistem performanslarının yükselmesine neden olmaktadır. İletişim ağlarının yönetimi ağ topolojilerinin ve karmaşıklıklarının hızlı değişmesinden dolayı yönetimleri gittikçe daha da zor olmaktadır. Bundan dolayı iletişim ağlarının potansiyel birçok probleminin çözümü için oğul zekâsı tabanlı yeni algoritmalar gittikçe daha da geliştirilmektedir. Bu algoritmalar birbiri ile iletişim halindeki öncüler üzerine kurulmuştur.

Oluşturacağımız yönlendirme protokolü aşağıdaki gereksinimleri karşılamalıdır:

1. Her bir öncü için farklı yazılım oluşturulmamalıdır. Yeterince basit, C++ diline uygun ve ağ katmanına kolayca uygulanabilmelidir.
2. Öncülerin işlem karmaşıklığı düşük olmalıdır. Bu durum ağ performansı için önemlidir.
3. Öncüler ağdaki düğümlere aynı anda olmayacak şekilde gönderilmelidir.
4. Ağdaki öncü kaybı kabul edilebilir oranda olmalıdır.
5. Öncülerin yapısı bir IP paketi için uygun olmalıdır. Bu durum iletişim maliyetini önemli oranda azaltacaktır.
6. Büyük ağlar için ölçeklenebilir olmalıdır.
7. Doğru yönlendirme yapılabilmesi için diğer yönlendirme tablolarından etkilenmemelidir.

1.5. Tezin Amacı

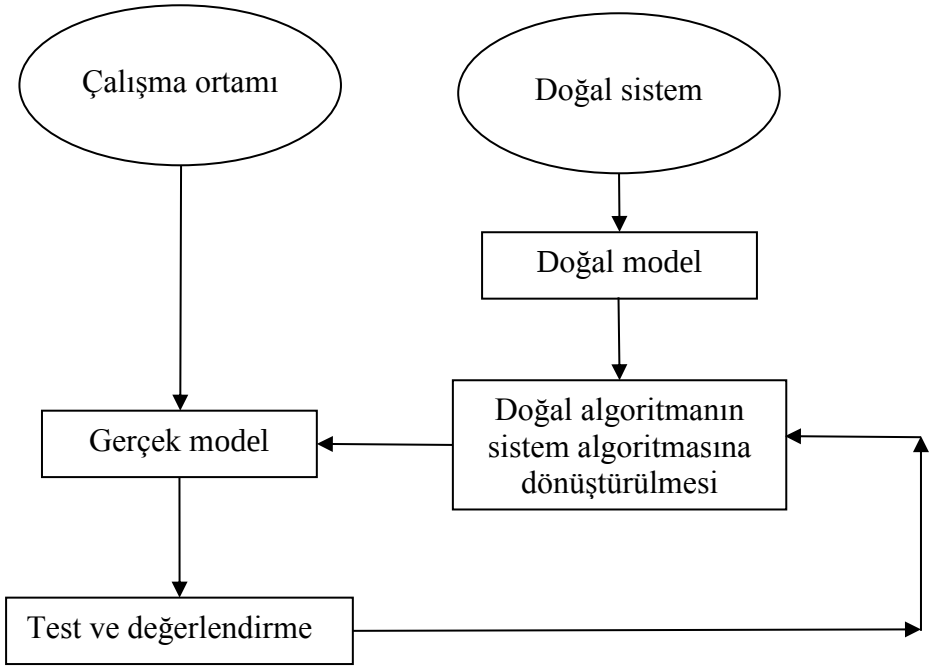
Bu tez çalışmasının amacı; kablosuz Ad-Hoc ağların sorunlarına çözüm olması amacıyla, bal arılarının yiyecek arama davranışlarından esinlenerek yeni bir oğul zekâsı tabanlı yönlendirme algoritması geliştirmek ve Ns-2 ağ simülatörü altında uygulamaktır.

1.6. Tezin Kapsamı

Hareketli Ad-Hoc ağ problemlerini karşılamak üzere geliştirlen protokolde:

1. Sosyal böceklerin doğada yiyecek arama özellikleri incelenerek bal arılarının kablosuz Ad-Hoc ağlar için oğul zekâsı tabanlı yeni bir yönlendirme protokolü geliştirmek için uygun olduğu tespit edilmiştir.
2. BeeAdhoc yönlendirme protokolünden farklı olarak, bal arılarından yiyecek arama için kovanda görevli olan öncü arıların, farklı nektar alanlarındaki bilgileri bir araya getirip, toplayıcı arı tarafından kovana getirilmesi özelliği, geliştirilen yönlendirme protokolünde kullanılmıştır.

3. Geliştirilen yönlendirme protokolünün modelleme ve simülasyonunu gerçekleştirmek için oğul zekâsı tabanlı protokol oluşturma süreci kullanılmıştır [5,6,7,8] (Şekil 1.1).
4. Modellenen protokolünün C++ programlama dili kullanılarak yazılan kodları Ns-2 ağ simülatörüne eklenmiştir.
5. Ek-B ve Ek-C'de örnek ağ topolojileri ve ağ trafikleri oluşturmak için Ns-2 ağ simülatöründeki setdest ve cbrgen yöntemleri kullanılmıştır.
6. Ek-D, Ek-E, Ek-F ve Ek-G'de deki örnek script kodlar kullanılarak ağ çıkış dosyaları oluşturulmuştur.
7. Geliştirilen protokolü test etmek amacıyla literatürde öne çıkan AODV ve oğul zekâsı tabanlı Bee-Adhoc protokolleri ile karşılaştırmalar yapılmış ve elde edilen sonuçlar sunulmuştur.



Şekil 1.1 Yapay zekâ tabanlı protokol geliştirme süreci.

1.7. Bilime Katkısı

Bu bölümde, tez çalışması sırasında elde edilen bilimsel katkılar detaylandırılmıştır. Bu çalışmada basit, dağıtık, merkezi olmayan belli hedeflere ulaşmayı amaçlayan bir ağ yönlendirme sistemi geliştirilmiştir. Bu yönlendirme sistemi sınırlı işlem ve kendi

bellek kapasitesi olan, kendi kararlarını alan, bulunduğu alandaki diğer düğümler hakkında bilgi toplayan bir sistemdir. Basit bir öncü modeli kurulurken, bir kovandaki bal arılarının nektar sahaları hakkında bilgi toplama özelliğinden esinlenilmiştir. Literatürde bulunan oğul zekâ tabanlı BeeAdhoc protokolünden farklı olarak, geliştirilen protokolde ***toplayıcı öncü arı*** kullanılmıştır. Toplayıcı öncü arı kullanılmasındaki temel hedef ağda dolaşan öncü arı sayısını azaltarak ağ kaynaklarını daha verimli kullanmaktır. Bu durum aynı zamanda iletilen veri paketi sayısının önemli sayıda artmasına neden olmuştur.

Çoklu öncü sistemi ve dinamik yönlendirme Bee-MANET modelin basit, esnek, sağlam ve ölçeklenebilir bir protokol olmasında etkili olmuştur. Deney sonuçlarından elde edilen veriler değerlendirildiğinde literatürde bulunan mevcut modellere göre benzer veya daha iyi sonuçlar alınmıştır. Deneysel çalışma kısmında değişik düğüm sayısı ve hızlarının ağ performansına ve maliyetine Ad-Hoc ağlardaki etkileri detaylandırılmaktadır.

1.8. Tez Planı

Tez aşağıdaki şekilde organize edilmiştir.

Bölüm 1'de, yapılan tezle ilgili konular kısaca belirtilip ve yapılan çalışmalar genel olarak özetlendikten sonra çalışmaların sunulduğu tezin kapsamı ile ilgili bilgiler verilmektedir.

Bölüm 2'de, kablosuz Ad-Hoc ağlarla ilgili olarak temel kavramların açıklanması ve konuyla ilgili olarak genel bir bilgi verilmesi hedeflenmiştir.

Bölüm 3'de, kablosuz Ad-Hoc ağ yönlendirme protokolleri listelenip ayrıntılı bir şekilde detaylandırılmaktadır.

Bölüm 4'de, kablosuz Ad-Hoc ağ modellerinin test edilebildiği simülasyon araçları listelenip detaylandırılmaktadır.

Bölüm 5'de, kablosuz Ad-Hoc ağlar için geliştirilmiş Bee-MANET yönlendirme protokolü ayrıntılı bir şekilde detaylandırılmaktadır.

Bölüm 6'da, Bee-MANET ve Ad-Hoc ağlar ile ilgili literatürde bulunan AODV ve BeeAdhoc yönlendirme protokollerinin başarım deneyleri ve bu deneylere ait sonuçlar grafiklerle detaylandırılmaktadır.

Bölüm 7'de, tez çalışması ile ilgili olarak elde edilen sonuçlar özetlenmekte ve ileriye yönelik bu konuda yapılabilecek çalışmalarla ile ilgili olarak öneriler verilmektedir.

BÖLÜM 2. KABLOSUZ AD-HOC AĞLAR

2.1. Giriş

Her çağda insanların en önemli gereksinimlerinden biri iletişim olmuştur [9]. Bu gereksinimi en iyi ve en kolay biçimde karşılamak amacıyla yapılan çalışmalar günümüzde oldukça ileri düzeylere ulaşmıştır. Bugün telefon, telgraf, radyo ve televizyon gibi araçların yanı sıra bilgisayar da iletişim amacıyla kullanılmaya başlanmıştır. Bilgisayarın iletişim aracı olarak kullanılması bilgisayar ağları ile olanaklı olmaktadır. 1969 yılında ilk geliştirilen bilgisayar ağıyla yalnız dört bilgisayar arasında bağlantı kurulabilirken, bugün bir bilgisayar ağı ile değişik ve birbirinden uzak yerlerde kurulu bulunan milyonlarca bilgisayar arasında iletişim sağlanabilmektedir.

Ağ, organizasyonların veya sistemlerin belli bir amaç doğrultusunda bilgilerinin birbirleri arasında paylaşımı demektir. Bilgisayar terminolojisinde ağ, bir grup bilgisayarın mantıksal şekilde birbirleri ile bağlanarak bilgilerini ve kendilerine bağlı olan çevre cihazlarını paylaşması demektir. Bir başka ifade ile bilgisayarlar arasındaki iletişim sistemine bilgisayar ağları denir. Başlangıçta bilgisayar ağları yalnızca dosya ve yazıcılarının paylaşımı için kullanılmasına rağmen bu durum günümüzde belli hedefleri olan ticari bilgi ve iş uygulamalarının paylaşımına dönüşmüştür. Bilgisayar ağları sabit (kablolu, sabit bağlantı) veya değişken olabilir.

Bilgisayar ağları ikiye ayrılır:

1. Kablolu ağlar ve
2. Kablosuz ağlar.

Düğümler arasında fiziksel bağlantı (kablo) olan ağlara kablolu, aralarında hiçbir şekilde fiziksel bağlantı olmayan ağlara da kablosuz ağlar denir[10].

2.2. Kablolu ve Kablosuz Ağlar

Günümüzde iletişimle ilgili olarak kablolu ve kablosuz olmak üzere iki tür ağ bulunmaktadır. Uçtan uca iletimde kablosuz ağlar kablolu ağlara göre farklılıklar içermektedir.

2.2.1. Kablolu ağlar

Kablolu ağlar, birbirine tel veya kablo ile bağlı olan iletişim cihazlarından meydana gelir. Genellikle bu tip ağlarda koaksiyel, burgulu çift (UTP, SMP), ve fiber optik kablolar kullanılmaktadır. Ağı genişletmek ve bağlantı gücünü arttırabilmek için anahtar, dağıtma ve tekrarlayıcı kullanılır. Genellikle bu ağlar kablosuz ağlara göre daha verimli, daha ucuz ve daha kaliteli iletişim sağlar. İletişim bir kez sağlandığında iletişimde aksama meydana gelme ihtimali çok azdır. Genellikle bağlantı hızı 100 mbps ile 1000 mbps arasındadır [9].

2.2.2. Kablosuz ağlar

Kablosuz ağlar, haberleşmek için radyo frekansı (RF) teknolojilerini kullanan kablosuz terminallerin oluşturduğu sistemlerdir [11]. Kablosuz ağlarda, ağdaki cihazlar arasındaki iletişim fiziksel kablo içermez. İletişim radyo dalgaları ile sağlanmaktadır. En büyük avantajlarından biri kablo maliyetlerinin çok düşük ve kurulumunun kolay olmasıdır. Hareketli olan kullanıcılar için avantaj sağlar. Kurulumu kolay ve hızlıdır. Ağ genişletilmek istendiğinde kabloya ihtiyaç duyulmadığından kolaydır. Bu avantajlarına rağmen bazı dezavantajları ise radyo dalgaları ile iletişim sağlandığından çevredeki radyo dalgalarından ve duvarlardan dolayı bağlantı hızı ve kalitesi kabloluya göre düşmektedir.

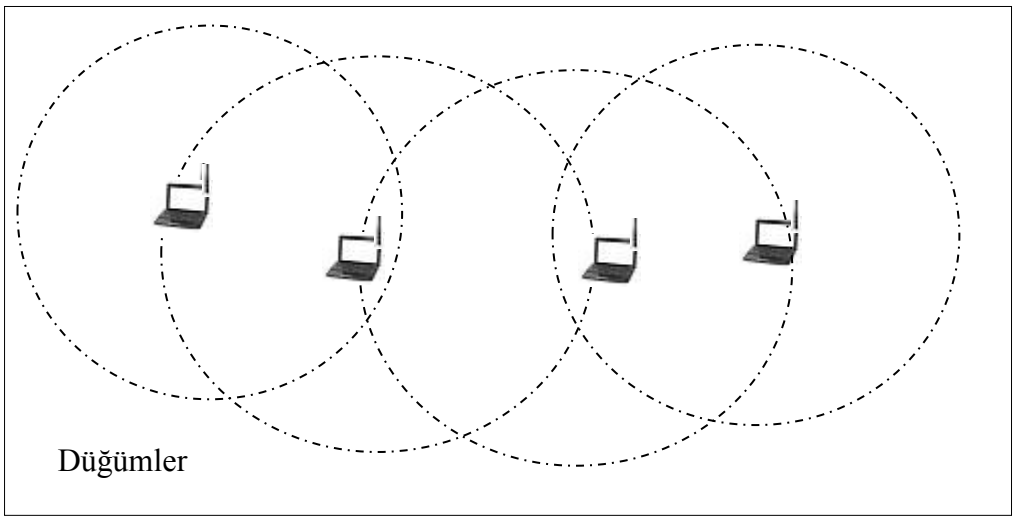
Kablosuz ağlar ikiye ayrılır:

1. Altyapılı kablosuz ağlar (Şekil 2.1)
2. Hareketli Ad-Hoc ağlar (Şekil 2.2)

Altyapılı kablosuz ağlarda ağ cihazlarının birbiriyle iletişiminde erişim noktası vardır.

Şekil 2.1 Altyapılı kablosuz ağlar.

Hareketli Ad-Hoc ağlar, kendi kendine organize olabilen ağlardır. Ağdaki bütün düğümler birbirleri ile merkezi bir nokta olmadan organize olabilir ve haberleşebilirler. Böylece ağ topolojisi sürekli bir değişim gösterir.



Şekil 2.2 Hareketli Ad-Hoc ağlar.

Kablosuz ağları kullanıldıkları yerlere göre şöyle sınıflandırabiliriz:

1. Hareketli Ad-Hoc ağlar,
2. Kablosuz algılayıcı ağlar,
3. Bluetooth,
4. HomeRF,
5. ZigBee,
6. Wİ-MAX,
7. GSM ve
8. GPRS.

2.2.2.1. Hareketli ad-hoc ağlar

Hareketli Ad-Hoc ağlar [12, 13, 14], (MANET) düğümler arasında herhangi bir merkez olmadan haberleşebilmektedir. Sürekli değişen bir ağ topolojisine sahiptir. Düğümler arasında yol bulma işlemi için yönlendirme protokolleri kullanılmaktadır. Sürekli değişen ağ topolojisinden dolayı hareketli Ad-Hoc ağlarda kullanılan yönlendirme protokolleri kablolu ve kablosuz ağlara göre farklılık göstermektedir. Topolojinin sürekli değişimi yönlendirme tablolarının güncellenmesine ve kontrol paket sayısının artmasına sebep olmaktadır. Bu protokollerdeki amaç düğümler arasındaki en kısa zamanda doğru ve daha az maliyetli yolları tespit etmektir. Burada en iyi yolu bulmaktaki izlenecek nokta yolun dinamik olarak değişen ağ topolojisini göz önüne alarak, genel ağ trafiğini arttırmadan ve kaynakları verimli kullanarak, daha fazla paketi daha az gecikmeyle iletimini minimum paket kaybı ile sağlamaktır.

Geleneksel ağlarda kullanılan uzaklık vektörü protokolü olan RIP [2] ve bağlantı durum protokolü olan OSPF [15] protokolleri hareketli Ad-Hoc ağlarda aşağıdaki sebeplerden dolayı kullanılamaz. Bunlar:

1. Kaynak düğüm ile hedef düğüm arasında tek yönlü bir bağlantı olabilir,
2. Kaynak ile düğüm arasında birden fazla yol olabilir,
3. Bant genişliği ve güç tüketimi yönlendirme güncellemelerini etkileyebilmektedir ve
4. Hızlı topoloji değişiminde yol bulma işlemi uzun sürebilmektedir.

2.2.2.2. Kablosuz algılayıcı ağlar

Kablosuz algılayıcı ağlar (KAA), çok fazla sayıda, küçük boyutlu, düşük maliyetli ve kısa mesafede kablosuz ortam üzerinden haberleşebilen algılayıcı düğümler meydana gelen bir ağdır. Bu ağda, düğümler rastgele olarak ortama bırakılabilmekte ve geliştirilen protokoller sayesinde kablosuz ortam üzerinden birbirleri ile haberleşerek kendi kendine organize olabilmektedir. Bu özellik, düğümlerin ortamdaki fiziksel büyüklük (ışık, sıcaklık, nem, basınç vb.) değişimlerini çok atlamalı yollar üzerinden merkezi ağ birimine iletmesini mümkün kılmaktadır. Kablosuz algılayıcı düğümlerin düşük maliyetli olması, normal şartlarda erişimin imkânsız olduğu bölgelere kolaylıkla yerleştirilebilmesi ve uzun süre bakım istemeden çalışabilmesi gibi özellikler kablosuz algılayıcı ağlarının çok çeşitli alanlarda kullanılabilmesini sağlamaktadır. Genel bir algılayıcı ağ; gözlem alanı, algılayıcı düğümler, çıkış düğümü ve görev yönetim düğümünden meydana gelmektedir [1].

Kablosuz algılayıcı ağlarda görev yapan bir kablosuz algılayıcı yapısal olarak algılayıcı, işlemci, alıcı/verici ve güç birimleri olmak üzere dört ana bileşenden oluşur. Bunlara ilave olarak kullanım amacına göre bir kablosuz algılayıcı, yer bulma sistemi, güç üretim birimi ve konum değiştirici bulundurabilir.

2.2.2.3. Bluetooth ağlar

Dizüstü bilgisayarlar, cep telefonları, ağ erişim noktaları ve iletişim cihazları arasında veri iletişimini sağlamak üzere oluşturulmuştur. Bluetooth teknolojisi ilk olarak Ericsson şirketi tarafından 1994 yılında 2.4 GHz bandında geliştirilmiştir. Bluetooth ses iletimine de olanak tanımaktadır. Kısa mesafede ve kişisel kullanımı için ucuz, düşük güç ve teknoloji hedeflemiştir. Yaklaşık 10 m çapında kapsama alanına sahiptir.

Bluetooth sistemi kullanıcıya birçok fayda sağlar [16]:

1. Cihazlar arasındaki kablonun görevini üstlenerek kablosuz iletişim imkânı sağlar,
2. Uygun cihazlar arasında dosya paylaşımına imkân verir,

3. Dizüstü bilgisayarlar, masaüstü bilgisayarlar, cep telefonları ve diğer iletişim cihazlarında kullanılabilir ve
4. Ofis, ev ve bankacılık gibi birçok uygulamaya imkân tanır.

2.2.2.4. HomeRF ağlar

HomeRF çalışma grubu 1999 yılında kablosuz ağ teknolojilerine çözümler üretebilmek amacıyla DECT ve IEEE 802.11 kablosuz ağ standartlarını bir araya getirdi. Kablosuz internet protokolü olarak adlandırılan bu yöntem son kullanıcılar için geliştirildi. HomeRF ISM frekans sınırı içerisinde 2.4 GHz ile maksimum 100 mW'lık arasında yayın yapmaktadır. HomeRF sistemi 10 Mbit/sn'lik veri aktarım hızına ulaşabilmektedir. Home RF özel veri ağlarında yaygın olarak kullanılmaktadır [17, 18].

2.2.2.5. ZigBee

Kişisel kablosuz ağlarda, düşük güç ile sınırlı kapasite veri iletimi sağlamak amacıyla ZigBee firması tarafından geliştirilmiş ve IEEE tarafından 802.15.4 adıyla standartlaştırılmış iletişim protokolüdür [19].

Tablo 2.1'de bu iletişim protokolünün radyo frekansları ve veri aktarım hızları gösterilmektedir.

Tablo 2.1 IEEE 802.15.4 Radyo frekansları ve veri aktarım hızları.

Band	Etki Sahası	Kanal	Veri Hızı
2.4GHz	Dünya geneli	16 kanal	250kbps
915MHz	Amerika	10 kanal	40kbps
868MHz	Avrupa	1 kanal	20kbps

ZigBee'nin diğer IEEE standartlarına göre ayırt edici özellikleri [20];

1. 10 ile 115.2Kbps arasında düşük veri hızı,
2. Standart bir batarya ile birkaç yıl süren düşük güç tüketimi,
3. Çoklu izleme ve uygulama kontrolü sağlayan ağ topolojisi,
4. Düşük maliyet, basit ve kolay kullanım ve
5. Yüksek güvenlik.

2.2.2.6. WIMAX ağlar

WIMAX (Worldwide Interoperability for Microwave Access) teknolojisi 50 kilometreye kadar 70 Mbps hıza varabilen ve 802.16e standartlarını kullanan, son kullanıcıya kablo alternatifi olarak genişband erişim sağlayabilen kablosuz erişim teknolojisidir. 802.16e standardı daha çok genişband kablosuz ve hareket halinde kullanılmaya uygun olarak geliştirilmiştir. 80216d standardı ise hareketli olmayan noktalarda daha çok kullanılmaktadır [21].

2.2.2.7. GSM ağlar

Küresel hareketli iletişim sistemi [22] (Global System for Mobile - GSM), bir cep telefonu sistemidir. Sesli iletişim için geliştirilmiştir. Farklı ülkelerde kolayca uyarlanabilir bir standart olarak yaygın kullanılan bir dijital ses ve veri hizmetidir. 2G hücresel teknoloji olarak kabul edilebilir ve dünyanın büyük bölümünde 900-1800 MHz frekans bantlarında faaliyet göstermektedir. Gerçek taşınabilirlik mobil ağ sistemlerinin tasarımının kullanımı ile mümkün olmuştur.

Bir GSM şebekesi çeşitli bileşenlerden oluşur:

1. Mobil İstasyon (Mobile Station-MS),
2. Baz İstasyonu (Base Station System-BSS) ve
3. Bir Anahtarlamalı Ağ Sistemi (Network Switching System-NSS).

2.2.2.8. GPRS ağlar

Genel Paket Radyo Servisi [23] (General Packet Radio Service – GPRS) hücresel ağlar üzerinden iletişim için kullanılan 2G teknolojisi olarak adlandırılan bir kablosuz ağdır.

Bazı özellikleri şunlardır:

Kablosuz veri iletişimi açısından birçok avantaj sunan paket anahtarlamasına sahiptir;

1. İnternet protokolü IPV4 – 6 destekler,
2. Kablosuz uygulama protokolü (WAP) destekler,
3. Veri aktarım hızı 56–114 kbit/ps ve
4. Çok çeşitli veri iletimi için kullanılabilir (cep telefonu, dizüstü bilgisayarlar veya kişisel bilgisayarlar için genişleme kartları ve noktasal satış sistemler).

BÖLÜM 3. KABLOSUZ AD-HOC YÖNLENDİRME PROTOKOLLERİ

3.1. Giriş

Kablosuz Ad-Hoc ağlar için (MANET) geliştirilmiş birçok yönlendirme protokolü bulunmaktadır. Bunlar yön bulma yöntemi ve zamanlarına göre tabloya dayalı (table driven), isteğe bağlı (on-demand), hibrit ve oğul zekâsı tabanlı olmak üzere dört gruba ayrılırlar. Tabloya dayalı yönlendirme protokollerinde her bir düğüm, ağdaki tüm düğümlere ait yönlendirme bilgilerini içeren bir ya da daha fazla tablo tutar. Ağ içerisindeki bir değişiklikte tüm düğümler tablolarını günceller. Bu durum, ağda bir düğümden diğer düğüme paket gönderilmek istendiğinde gerekli yolun bulunmasının kısa sürmesine ve ağdaki kontrol paket sayısının artmasına sebep olmaktadır. Tabloya dayalı yönlendirme protokollerinin aksine isteğe bağlı yönlendirme protokollerinde sürekli olarak düğümlere ait yönlendirme tablolarının tutulması ve takip edilmesi gereksinimi yoktur. Bunun yerine yönlendirme gerektiğinde çeşitli mekanizmalar çalıştırılarak hedefe ait yollar oluşturulur. Oluşturulan yollar paket hedefe iletilinceye kadar ya da belirli bir süre boyunca saklanır. Bir düğümden diğer düğüme paket gönderilmek istendiğinde yol bulma işlemi başlar ve kaynak düğümden hedef düğüme kontrol paketleri gönderilir. Böylece ağdaki kontrol paketi sayısı azalırken kaynak ile hedef arasındaki yol bulma süresi tabloya dayalı yönlendirme protokollerine göre daha uzun olmuş olur.

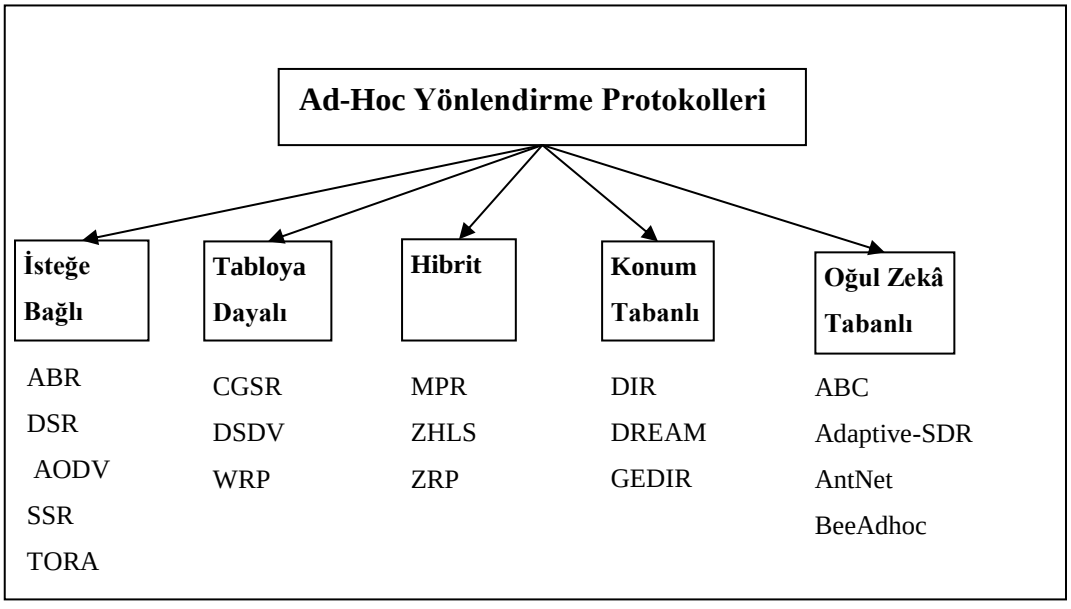
Hareketli Ad-Hoc ağ yönlendirme protokolleri paket gönderim şekline göre tekli gönderim (unicast) ve çoklu gönderim (multicast) olmak üzere ikiye ayrılır.

Tekli gönderim**,** hareketli Ad-Hoc ağlardaki birçok protokol tekli gönderim esasına göre çalışır. Böylece genel Ad-Hoc ağların yönlendirme katmanı paketi bir kaynaktan bir hedefe gönderir. Paketin iletimi gayet basit bir şekilde yönlendirme tablosundan tekli bir hedefe gönderilmiş olur. Eğer kaynakla hedef arasında birden fazla adım (hop) varsa, paket hedefe doğru bir sonraki düğüme gönderilir.

Çoklu gönderim, bir grup ağ cihazına veri göndermeye yarar. Ağdaki grup adresleri kullanılarak, birden fazla ağ cihazın tekil bir adresi dinlemesi (buradan veri beklemesi) sağlanmaktadır. Grup adresine bir paket iletildiğinde, bu grupta olan bütün ağ cihazları bu paketi alacaktır. IP protokolünün 802.3 MAC Alt Katman (802.3 MAC Sublayer) protokolünden itibaren, yani OSI katmanlı yapısının 2. seviyesinden itibaren bu tür bir destek gelmektedir. Tüme gönderim (broadcast) ise ağdaki bütün cihazlara veri iletimini sağlayan özelleşmiş bir çoklu gönderimdir. Adres alanının hepsinin 1 olması durumu, tümüne gönderimi bildirir [24] .

3.2. Ad-Hoc Yönlendirme Protokolleri

Hareketli Ad-Hoc ağlarda yönlendirme protokolleri topolojilerine göre Şekil 3.1'deki gibi sınıflandırabiliriz.



Şekil 3.1 Ad-Hoc Yönlendirme Protokollerin Sınıflandırılması.

1. İsteğe Bağlı Yönlendirme Protokolleri

    a) İlişki tabanlı yönlendirme (Associativity Based Routing-ABR)[25]
    b) Dinamik kaynak yönlendirme (Dynamic Source Routing-DSR) [26]
    c) İsteğe bağlı uzaklık vektör yönlendirme (On-Demand Distance Vector Routing-AODV) [27]
    d) Sinyal kararlılık tabanlı uyarlamalı yönlendirme (Signal Stability Based Adaptive Routing-SSR) [28]
    e) Geçici sıralı yönlendirme algoritması (Temporally Ordered Routing Algorithm-TORA) [27]

2. Tabloya Dayalı Yönlendirme Protokolleri

    a) Kümelenmiş ağ geçidi anahtar yönlendirme (Clustered Gateway Switch Routing-CGSR) [29]
    b) Hedef sıralı uzaklık vektörü (Destination Sequenced Distance Vector-DSDV) [30]
    c) Kablosuz yönlendirme protokolü (Wireless Routing Protocol -WRP) [31]

3. Hibrit Yönlendirme Protokolleri

    a) Çoklu geçiş protokolü (Multi Point Relaying-MPR)[32]
    b) Bölge tabanlı hiyerarşik bağlantı durum protokolü (Zone based Hierarchical Link State Routing Protocol-ZHLS) [33]
    c) Bölge yönlendirme protokolü (Zone Routing Protocol-ZRP) [34]

4. Konum Tabanlı Yönlendirme Protokolleri

    a) Yön yönlendirme algoritması (Directional Routing Algorithm -DIR) [35]

b) Hareketli uzaktan yönlendirme etki algoritması (Distance Routing Effect Algorithm For Mobility-DREAM) [36]
c) Coğrafi uzaklık yönlendirme (Geographic Distance Routing -GEDIR) [37]
d) En ileri yarıçap içi (Most Forward within Radius-MFR) [38]
e) KTMYP [39]

5. Oğul Zekâsı Tabanlı Yönlendirme Protokolleri

a) ABC [40]
b) Adaptive-SDR[41]
c) AntNet [42]
d) BeeAdhoc [43]
e) Hopnet [44]
f) Bulanık karınca koloni tabanlı yönlendirme protokolü (Fuzzy Ant Colony Based Routing Protocol) [45]

3.2.1. İsteğe bağlı yönlendirme protokolleri

İsteğe bağlı yönlendirme protokollerinde, tabloya dayalı yönlendirme protokollerinin aksine sürekli olarak düğümlere ait yönlendirme tablolarının tutulma gereksinimi yoktur. Bunun yerine yönlendirme gerektiğinde yön bulma mekanizmaları çalıştırılarak hedefe ait yollar oluşturulur. Oluşturulan yollar paket hedefe iletilinceye kadar ya da ihtiyaç duyulduğu sürece geçerlidir. Birçok isteğe bağlı yönlendirme protokolü bulunmaktadır. Bunların en yaygın olanları; Geçici Sıralı Yönlendirme (Temporarly Ordered Routing Algorithm, TORA), Dinamik Kaynak Yönlendirme (DSR) ve İsteğe Bağlı Uzaklık Vektör Yönlendirme (AODV) protokolleridir.

3.2.1.1. ABR yönlendirme protokolü

ABR[46] (Associativity-Based Routing) yönlendirme protokolü, yayılım (broadcast) ve düğümden düğüme(point-to-point) yönlendirmesinden meydana gelir. AODV[47]

yönlendirme protokolüne benzer biçimde sadece kaynak için istenilen yolları sağlar. Ancak, ABR alternatif yolları yönlendirme tablosunda saklamaz. Buna ilave olarak yönlendirme kararı hedef düğümde alınarak diğer yollar dikkate alınmaz. Böylece tekrarlı yol bilgilerinden kaçınılarak seçilen yol daha uzun ömürlü olmaktadır.

ABR Yönlendirme protokolü üç aşamadan meydana gelmektedir:

1. Yol keşif aşaması (Route discovery phase),
2. Yolun tekrar kurulma aşaması (Route re-construction (RRC) phase) ve
3. Yolun silinme aşaması (Route deletion phase).

Eğer kaynak düğüm yol isteğinde bulunursa yol keşif aşaması (Route discovery phase-RDP) devreye girer. Kaynak ile hedef arasında herhangi bir yol değişikliği oluştuğunda yolun tekrar kurulum aşaması (Route reconstruction-RRC) devreye girer. Eğer yolun tekrar kurulum aşaması devreye giremiyorsa yolun silinme aşaması (Route reconstruction-RRC) devreye girer.

3.2.1.2. DSR yönlendirme protokolü

Dinamik kaynak yönlendirme (Dynamic Source Routing-DSR) protokolü kaynak yönlendirmeli isteğe bağlı bir yönlendirme protokolüdür. Her düğüm, kaynak yollarını içeren bir tablo bulundurur. Düğümler hedefe bir paket göndermek istediğinde, önce belleğinde hedefe ait mevcut bir yol olup olmadığını kontrol eder. Eğer geçerli bir yol var ise bu yolu kullanarak paketi gönderir. Eğer yolun süresi dolmuş ya da yok ise kaynak ve hedefin adres bilgisi ile ağ içerisinde tek olan bir kimlik numarasına (ID) sahip yol istek paketini yayın (broadcasting) modunda ağa göndererek yön bulma mekanizmasını başlatır. Her bir ara düğüm, gelen yol istek paketindeki hedef adrese ait bir yol olup olmadığını kontrol eder. Eğer yoksa paketi komşu bir düğüme gönderir [26].

3.2.1.3. AODV yönlendirme protokolü

İsteğe bağlı uzaklık vektörü (Ad-Hoc On-Demand Distance Vector - AODV) algoritması hareketli ad hoc ağlar için tasarlanmış bir yönlendirme protokolüdür. Çoklu ve bire bir yönlendirme yapabilmektedir. Sadece kaynak düğüm ile paketin gönderileceği düğüm ve/veya düğümler arasındaki yolun oluşturulmasını sağlar. Bu işlem kaynaklar yol talep ettiği sürece devam eder. AODV yolların güncelliğini sağlamak için sıra numaralarını kullanır. AODV protokolünde kaynak, hedefe ait yolu bulabilmek için ağa bir yol istek paketi (RREQ) gönderir. Yol istek paketi, hedef hakkında en son geçerli yol bilgisine sahip düğüme ya da hedefe ulaşıncaya kadar komşu düğümler arasında dolaşır. Bir düğüm yol istek paketini komşu düğüme gönderdiği zaman, yol istek paketi ilgili kaynak düğümün kimliği, yayın kimliği, güncel sıra numarası gibi bilgileri içerir. Böylece yol istek paketinin cevabı için bir yol çizilmiş olur. Bu yol, kaynaktan veri paketleri gönderildiği müddetçe kullanılmaya devam eder. Kaynaktan veri paketi gönderilme işlemi bittikten ve zaman aşımından sonra ilgili yol yönlendirme tablosundan silinir. Eğer bağlantı sırasında bir hata olursa kaynak düğüme hata kodu gönderilir. Kaynak hata kodunu aldıktan sonra eğer tekrar paket göndermek isterse yol bulma işlemi tekrar edilir. Ağ hareketli olduğu için topolojinin değişmesiyle yollarda değişiklik meydana gelebilir. AODV' de yol aktif olduğu sürece geçerli kalır [47].

3.2.1.4. SSR yönlendirme protokolü

Sinyal sabitliği tabanlı uyarlamalı yönlendirme (Signal Stability Based Adaptive Routing-SSR) algoritması, isteğe bağlı yol bulma işlemini daha uzun konumu, daha istikrarlı ve yaşam ömrü daha fazla olan sinyali güçlü yolları tercih etmektedir. Yönlendirme protokolü güçlü ve zayıf sinyal kanallarının farklılıklarını inceler. Her bir kanal hedef ile kaynak arasındaki paket değişiminde sinyallerin ortalama sinyal gücüne göre güçlü ya da zayıf olup olmadığını inceler. Düğümün konum istikrar kriteri seçilen yolun ne kadar uzun süreli olduğunu tespit eder. Sinyal gücü ve konum istikrar kriteri ile daha güçlü sinyal kanalları tespit edilmiş olur.

Bu protokolde iki alt protokol bulunmaktadır:

1. İleri protokol (Forwarding protokol-FP) ve
2. Dinamik yönlendirme protokolü (Dynamic routing protokol-DRP).

Dinamik yönlendirme protokolü veri bağı katmanından paket alış verişindeki sinyal güç bilgilerini alır ve değerlendirir.

Bu yönlendirme protokolünde iki tür tablo bulunmaktadır:

1. Sinyal sabitliği tablosu(Signal stability table-SST) ve
2. Yönlendirme tablosu(Routing table-RT).

Sinyal sabitliği tablosunda (SST) her bir düğümün komşusu ile olan geçmiş sinyal gücü bilgileri bulunmaktadır [48].

3.2.1.5. TORA yönlendirme protokolü

Geçici sıralı yönlendirme algoritması (Temporally Ordered Routing Algorithm-TORA) yönlendirme protokolü yüksek hızda hareketli çok adımlı (multihop) kablosuz ağlar için tasarlanmıştır. Kaynak ile hedef arasında birden fazla yol bulur. Bu protokolün en büyük özelliği topolojisi değişen birbirine yakın az sayıdaki düğümler için küçük bölgesel kontrol paketleri üretmesidir. Bunu başarmak için düğümler komşu düğümlerin yönlendirme bilgilerini elde eder. Bu protokolün yolun oluşturulması, elde edilmesi ve silinmesi gibi üç temel fonksiyonu vardır.

Her bir düğüm aşağıdaki beş özelliğe sahiptir:

1. Kopan bağlantının mantıksal zamanı,
2. Yeni referans seviyesi için her bir düğümün benzersiz kimlik numarası,
3. Bir yansıma göstergesi,
4. Yayılım parametresi verme ve

5. Her bir düğümün benzersiz kimlik numarasına sahiptir.

Yukarıdaki belirtilenlerin ilk üçü referans seviyesi olarak belirtilir. Yeni referans seviyesi her bir düğümü bağlantısı koptuğu zaman tanımlanır. Yol oluşturma işlemi istek (query-QRY) ve veri paketleri ile yapılır. Kaynak içerisinde hedef düğümün olduğu kimlik numarasının(ID) olduğu yayılım paketi oluşturur [35].

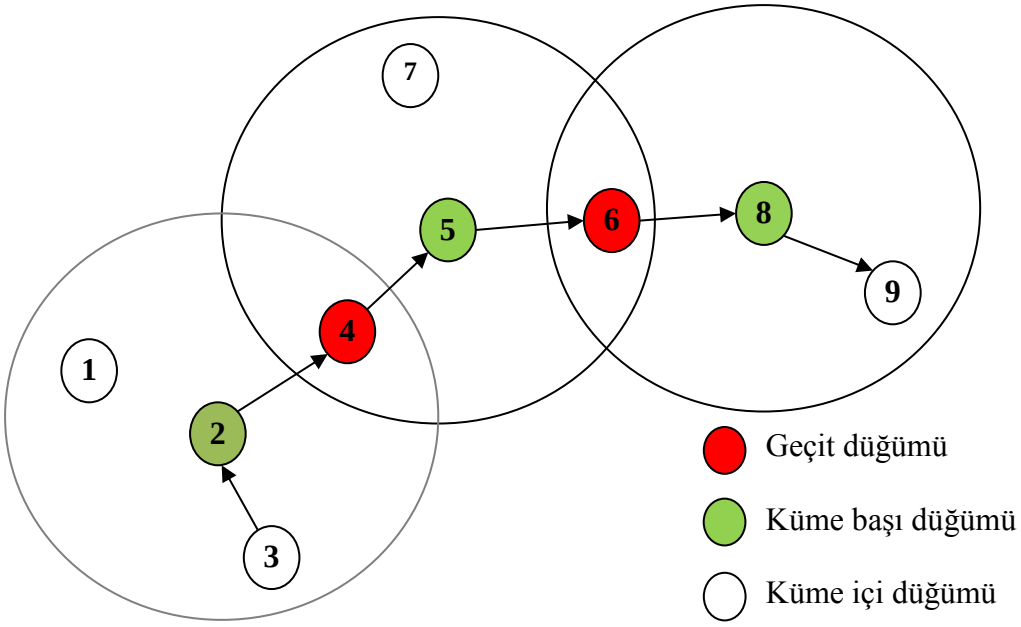
3.2.2. Tabloya dayalı yönlendirme protokolleri

Tabloya dayalı yönlendirme protokollerinde her bir düğüm, ağdaki tüm düğümlere ait yönlendirme bilgilerini içeren bir ya da daha fazla tablo tutar. Ağ içerisindeki bir değişiklikte tüm düğümler tablolarını günceller. Tabloya dayalı yönlendirme protokollerinden en bilinenleri Dinamik Hedef-Sıralı Uzaklık Vektör (Dynamic Destination-Sequenced Distance-Vector Routing Protocol, DSDV) ve Kablosuz Yönlendirme (The Wireless Routing Protocol, WRP) protokolleridir.

3.2.2.1. CGSR yönlendirme protokolü

Kümelenmiş ağ geçidi anahtar yönlendirme (Clusterhead Gateway Switch Routing Protocol-CGSR) yönlendirme protokolü temel olarak DSDV yönlendirme protokolünü kullanmaktadır. Düğümler kümelere(cluster) ayrılır ve kümedeki seçilmiş olan bir düğüm küme başı olarak(cluster head) seçilir. Bütün düğümler kümedeki düğümlerle haberleşirken küme başını seçer. İki veya daha fazla küme başının birbiri ile haberleşmesi geçit (gateway) düğümü ile gerçekleştirilir. Ağ topolojisinin hızlı değişime uğradığı ağlarda küme başı seçimi ağ performansını olumsuz olarak etkilemektedir. Bu yüzden CGSR protokolü bu durumun üstesinden gelmek için LCC(Least Cluster Change) algoritması kullanmaktadır. LLC algoritması, eğer ağ topolojisinde iki veya daha fazla küme başı aynı kümeye küme başı olarak seçildiğinde devreye girer. İlk olarak kaynak düğüm paketi küme başına gönderir. Kaynak düğüm bu paketi küme başları arası bağlantıyı sağlayan geçiş yolu düğümüne hedefe ulaşması için gönderir. Geçiş yolu düğümü bu paketi bir sonraki küme başına

gönderir. Bu durum kontrol paketinin hedef düğüme ulaşıncaya kadar devam eder. Şekil 3.2 CGSR protokolünün çalışma prensibini göstermektedir [29].



Şekil 3.2 CGSR Yönlendirme protokolü çalışma prensibi.

3.2.2.2. DSDV yönlendirme protokolü

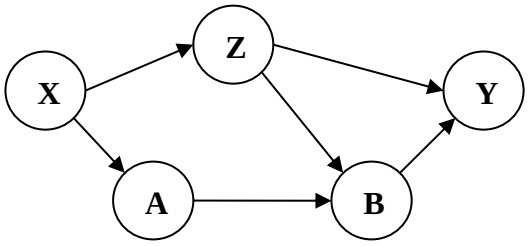
Hedef Sıralı Uzaklık Vektörü (Destination Sequenced Distance Vector - DSDV) yönlendirme algoritması Bellman-Ford[47] algoritması tabanlı hareketli ad hoc ağlar için tasarlanmış tabloya dayalı bir yönlendirme protokolüdür. Bu protokol temel yönlendirme problemlerine katkı sağlamak amacıyla 1994 yılında C. Perkins ve P. Bhagwat tarafından geliştirilmiştir [47]. Yönlendirme tablosuna yol ile ilgili her bir bilgi girildiğinde yol ile ilgili sıra numarası verilir. Bu sıra numarası ile ilgili başka bir bağlantı olsa bile yine başka bir sıra numarası verilir. Bu sayı hedef tarafından oluşturulur. Yönlendirme bilgileri düğümler arasında sık sık gönderilerek ağdaki yönlendirme tabloları güncellenir. Eğer bir düğüm yeni bir bilgi alırsa en son sıra numarasını kullanır. Eğer sıra numarası yönlendirme tablosundaki herhangi bir sıra numarası ile aynı ise daha iyi olan sıra numaralı yol kullanılır. İyi olmayan yönlendirme bilgileri kısa bir süre içinde güncellenmez. Ağda paket alış verişleri yokken yani ağ boştayken belirli periyodik aralıklarda ağdaki düğümlerin yönlendirme

tabloları güncellenir ve böylece daha fazla bant genişliği ve batarya kullanılmış olur. Ağ topolojisinin değiştiği her bir durumda yeni sıra numaraları gerekecektir. Bundan dolayı DSDV protokolü hareketliliği yüksek ağlarda çok iyi sonuç vermemektedir [47, 48].

3.2.2.3. WRP yönlendirme protokolü

Kablosuz Yönlendirme Protokolünde (Wireless Routing Protocol-WRP) her bir düğüm ağ ile ilgili olarak uzaklık tablosu, yönlendirme tablosu, bağlantı maliyet tablosu ve mesaj yeniden iletim tablosu bulundurmaktadır. Örnek olarak, x düğümünün uzaklık tablosunda hedef y düğümü için x in komşusu olan her bir z için bilgi tutulur (Şekil 3.3). Bu ayrıca hedef düğüm yolu için z düğümünün diğer komşuları içinde geçerlidir. X düğümünün yönlendirme tablosu ayrıca hedef y için bütün yolların bilgilerini tutar. Ayrıca her bir yolla ilgili olarak yönlendirme tablosuna basit, döngü veya geçersiz yol olup olmadığı ile ilgili bir etiket bilgisi girilir. Yönlendirme tablosuna bu şekilde bilgiler girilmesi yönlendirme ile ilgili sorunların çözümüne yardımcı olmaktadır.

Bağlantı maliyet tablosu düğümün her bir komşusu ile ilgili bağlantı maliyetini ve komşu düğümden gelen hata mesajlarından dolayı oluşan zaman aşım sayısını bulundurur. Mesaj iletim listesi her bir düğümün komşularından alınmamış cevap mesajlarını ve bu komşu düğüme tekrar iletilen güncelleme mesajlarını bulundurur[49].



Şekil 3.3 WRP Yönlendirme protokolü çalışma prensibi.

3.2.3. Hibrit yönlendirme protokolleri

Hibrit yönlendirme protokolleri konum tabanlı yönlendirme protokolleri ve çoklu geçiş(Multipoint Relaying-MPR) tabanlı protokoller olarak iki ana başlık altında toplanmaktadır [39]. Hibrit yönlendirme protokolleri, isteğe bağlı(on-demand) ve tabloya dayalı(table-driven) protokollerin özelliklerini bir arada barındıran protokollerdir. Tasarlanan protokoller, hem isteğe bağlı hem de tabloya dayalı protokollerin çalışma prensiplerini barındırmaktadır.

3.2.3.1. Konum tabanlı yönlendirme protokolleri

Konum tabanlı yönlendirme protokolleri tabloya dayalı ve isteğe bağlı yönlendirme protokollerin özelliklerini içerirler ve genellikle yerel düğümlerle ilgilenirler. Ağdaki düğümlerin, coğrafi konumlarının belirlenmesi (Global Positioning System- GPS) veya Galileo (Global Navigasyon Sistemi) ile gerçekleştirilir [50].

3.2.3.1.1. ZHLS yönlendirme protokolü

Bölge tabanlı hiyerarşik bağlantı durum (Zone-based Hierarchical Link State Routing Protocol- ZHLS) protokolünde ağ birbiri ile örtüşmeyen (non-overlapping) bölgelere ayrılır. Diğer hiyerarşik protokollerin aksine her bölgenin bir küme başı yoktur. ZHLS protokolü bölge seviyeli (zone-level) ve düğüm seviyeli (node-level) olmak üzere iki hiyerarşik topolojiden oluşur. Düğüm seviyesi topolojisi düğümlerin bir biri ile bölge içerisinde fiziksel olarak nasıl bağlandığını gösterir. İki bölge arasındaki bağlantı ancak en az bir bölgedeki bir düğümün diğer bölgedeki bazı düğümlere fiziksel olarak bağlantı olduğu zaman olur. Bölge seviye topolojisi bölgelerin birbiri ile nasıl bağlı olduğunu gösterir. Düğüm ve bölge olmak üzere iki tür bağlantı durum paketi(Link State Packet-LSP) vardır. Düğüm durum paketi bölge içerisindeki komşuları hakkında bilgileri bölge durum paketi de bölgeler hakkındaki bilgileri içerir. Böylece her bir düğüm bölgesindeki düğümler ile ilgili tam bir bilgiye ve yalnızca diğer bölgelerin bölge bağlantı bilgilerine sahiptir. Hedef düğümün düğüm kimliği (id) ve bölge

kimliği(zone-id) bilgisi verilerek kontrol paketi bir hedef düğümün bölgesine gider. Sonra hedef düğüm bölgesinde hedef düğüme yönlenir [33].

3.2.3.1.1. ZRP yönlendirme protokolü

Bölge yönlendirme (Zone Routing Rrotocol- ZRP protokolünde her bir a düğümün n adım (hop) sayısı uzaklığına göre bölgesi (zone) vardır.

ZRP protokolünde yönlendirme için üç farklı protokol kullanılır:

1. Bölge içi yönlendirme protokolü (Intrazone Routing Protocol) [51] ,
2. Bölgeler arası yönlendirme protokolü (Intrazone Routing Protocol) [52] ve
3. Sınır çözümleme protokolü (Bordercast Resolution Protocol) [53].

Düğümün kendi bölgesi içerisinde yönlendirme protokolü kullanılır. Eğer farklı bölgelerden bir yol isteği olduğu zaman bölgeler arası yönlendirme protokolü kullanılır. Sınır çözümleme protokolü ise bölgeler arasındaki gereksiz olan yönlendirme bilgilerini azaltmak için kullanılır.

ZRP protokolünün avantajları şöyle sıralanabilir:

1. Bölge içinde bölge içi yönlendirme protokolü kullanıldığından hedef düğüm kaynak düğümün etrafında (aynı bölge) olduğunda yol bulma gecikmesi çok kısa olmaktadır,
2. Periyodik olarak bölge içi yönlendirmeler güncellendiğinden bölge içindeki bir düğümün konumu ve diğer düğümlerle bağlantıları değiştiğinde ağdaki diğer bölge düğümleri bu durumdan etkilenmez,
3. Bölgeler arasındaki yol bulma işlemi yalnızca bölgeler arası yön bulma olduğu için ağdaki dolaşan kontrol paket sayısı azaltılmıştır,
4. Bölge içi yönlendirme, bölgeler arası yolların tespitine yardımcı olmaktadır. Bölge içi bozuk bağlantılarda kopma meydana geldiğinde bölge içi düğümler

kolayca kullanılabilir. Yönlendirme optimizasyonu düğümün bulunduğu bölge içinde yapılabilmektedir ve

5. Bölge içi yönlendirmede etkin bir biçimde kontrol paket yayılımı (broadcast) sağlanır.

Ağda oluşturulan bölgelerin yarıçapları değişken bir parametredir. Farklı bölgeler farklı yarıçap ölçüsü kullanabilir. Bölge yarıçapları doğru bir şekilde tespit edildiğinde bölge içi ve bölgeler arası yönlendirmeler yüksek performansta çalışmaktadır [51].

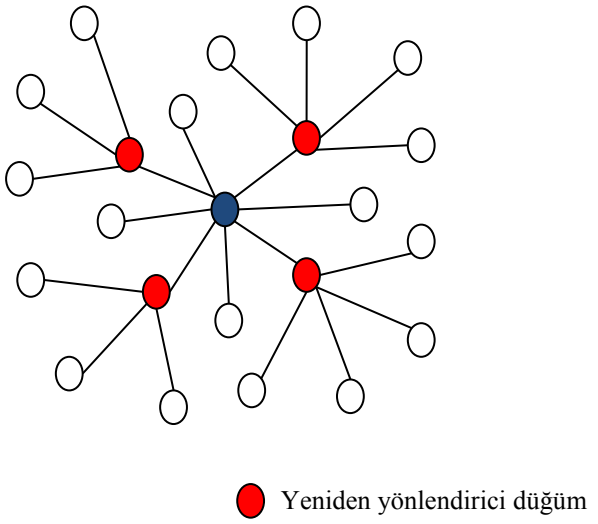
3.2.3.1.2. KTMYA yönlendirme protokolü

KTMYA (Konum Tabanlı Melez Yönlendirme Algoritması**),** hem yönlendirme ek yükünü azaltarak bant genişliğinin etkin bir şekilde kullanımını, hem de ağdaki her bir düğüm tarafından yönlendirme için yapılacak işlem sayısını ve tutulacak veri miktarını azaltarak, batarya ömrünü etkin bir biçimde kullanır. Bu protokolde hem isteğe bağlı, hem de tabloya dayalı algoritmaların çalışma esaslarından faydalanılmıştır. Ancak bu protokolde bu iki algoritma türünün dışında yeni bir yaklaşım içermektedir. KTMYA'da altyapılı ağlarda olduğu gibi merkezi bir düğüm diğer bir ifadeyle bir ana düğüm atanır ve yönlendirme bilgilerini yönetir. Düğümler hedef bir düğüme veri göndermek istediklerinde, hedef düğümün yerini ve ona gidecek yolu ana düğümden öğrenirler ve bu yol üzerinden verilerini gönderirler. Bu aşamada algoritma altyapılı ağlardan ayrılır. Çünkü altyapılı ağlarda veri de merkezi istasyon üzerinden gönderilir. Ancak bu algoritmada merkezi istasyon olarak davranan ana düğüm sadece hedefe ait yolu bulmada yardımcı olur [54].

3.2.3.2. MPR tabanlı algoritmalar

Melez yönlendirme protokollerinin bir kısmı Çoklu Geçiş (Multi Point Relaying-MPR) yöntemi ile oluşturulmuş protokollerdir. MPR algoritmaların çalışma prensibi, yönlendirme ek yükünü azaltmak için kaynak düğümden gönderilen kontrol paketinin ağdaki tüm düğümler tarafında alındıktan sonra diğer bütün düğümlere gönderilmesinden kaynaklanan ağdaki kontrol paket sayısının azaltılmasını hedefler.

MPR algoritmalarının kontrol paket yayım şeması Şekil 3.4'te görüldüğü gibi kontrol paketlerinin ağdaki bütün düğümler tarafından değil yalnızca daha önce belirlenmiş bir düğüm kümesi tarafından iletilmesini sağlar. Böylece büyük ölçekli ağlarda aynı kontrol paketinin ağdaki bütün düğümlerde tekrar tekrar iletilmesinin önüne geçilmiş olur [55].



Şekil 3.4 MPR algoritmalarının kontrol paket yayım şeması [54].

3.2.4. Oğul zekâsı tabanlı yönlendirme protokolleri

Oğul zekâsı (Swarm Intelligence) biyolojik sistemler tarafından sunulmuştur ve mühendislik sistemlerinde ve ağ iletişiminde olduğu gibi birçok güçlü özellikler göstermiştir[41]. Oğul zekâsı; termitler, arılar, karıncalar, kuşlar, balık sürüleri gibi aralarında etkileşim olan böceklerin veya diğer sosyal hayvanların topluluk halindeki davranışlarını örnek alarak, problemlere çözüm getirmeyi amaçlayan bir yapay zekâ tekniğidir [56, 57]. Doğada grup halinde yaşayan canlılar tarafından meydana getirilen oğul zekâsı, iletişim ağları gibi birçok mühendislik sistemlerinde istenen sayısız özellikler barındırır. Oğul zekâsı çözümleri, dağıtık sistemlere ve ağ problemlerine uygulandıkları zaman bir takım üstünlükler sunarlar. Bu üstünlükler;

1. Ölçeklenebilirlik: Birbirinden bağımsız varlıklar, çoğalma ve göç yoluyla sistemin boyutuna uyarlanabilen tam olarak merkezi olmayan bir çözümü

ortaya koydukları için merkezi uygulamalara göre herhangi bir ölçeklenebilirlik problemine maruz kalmamaktadır,

2. Hata Toleransı: Sistemi oluşturan bireyler / varlıklar bağımsız olarak hareket ettiklerinden dolayı diğer varlıklara bağlı olmadan çalışabilirler sistem hatalara karşı toleranslıdır. Merkezi yaklaşımlarda merkezi birim arıza yaptığında bütün sistem çökerken, bir grup varlığın ortadan kalkması / ölmesi sistemin tamamen çökmesine neden olmamakta, sadece sistemin performansında bir miktar düşüşe neden olmaktadır,
3. Uyarlanabilirlik: Sistemi oluşturan varlıkların yaşam döngüsü (doğum, ölüm, göç, vb.) ve yetenekleri, değişen sistem şartlarına uyarlanabilen bir sistem meydana getirir,
4. Hız: Sistemdeki değişikliklere yerel etkileşimler yoluyla hızlı cevap verilir,
5. Modülerlik: Bireysel ve sadece kendinden sorumlu varlıklara dayalı problem çözüm yöntemi, yüksek seviyeli, modüler ve açık yapılı sistemlerin ortaya çıkmasına neden olur ve böylece sürekliliği ve güncelliği arttırır,
6. Otonomi: Bu sistemlerde yönetim son derece dağıtık ve dinamik olduğu için herhangi bir yönetici gerekli değildir ve
7. Paralellik: Varlıkların işlemleri ve etkileşimleri doğal olarak paraleldir.

Oğul zekâsı, yukarıdaki nedenlerden dolayı özellikle geniş ve yüksek derecede dinamik sistemler için oldukça uygundur.[5]

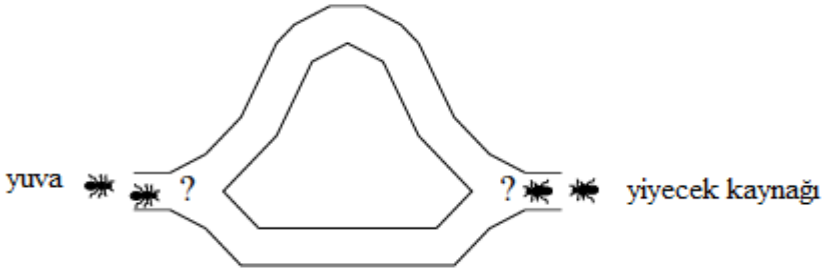
Bu alanda geliştirilmiş yönlendirmelerine protokollerine örnek olarak ABC [40], Adaptive-SDR [41], AntNet [42], BeeAdhoc [43], Hopnet [44], Fuzzy Ant Colony Based Routing Protocol [45] verilebilirler.

3.2.4.1. ABC yönlendirme protokollü

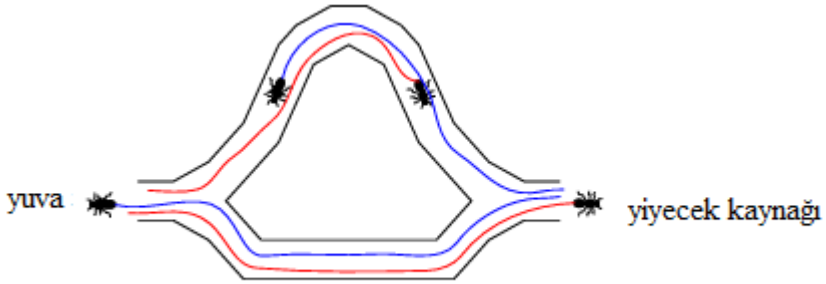
Karıncalar bireysel olarak çok sade davranışlara ve kısıtlı hafızaya sahip böceklerdir. Ancak kolektif bir şekilde çalışarak karmaşık görevleri güvenli ve tutarlı bir şekilde başarıyla yerine getirirler [58]. Bunlar:

1. 1 °C sınırları içerisinde yuva sıcaklığının ayarlanması,
2. Köprü yapımı,
3. Gıda için belirli alanlara baskın,
4. Bina ve yuvaların korunması,
5. Besin öğeleri sıralama,
6. Büyük ürünlerin taşıma işbirliği,
7. Koloni göç işlemi,
8. Yumurta ve yavru bakımı,
9. Besin kaynaklarının yuvaya en yakın yolunu bulma ve
10. Tercihen en zengin besin kaynaklarının bulunmasıdır.

ABC yönlendirme protokolünde karıncalar geçtikleri yerlere ait yol ve yiyecek ile ilgili bilgileri (pheromone) bırakarak arkadan gelen karıncaya yiyecek alanları ve yiyecek ile ilgili bilgileri bırakmış olur. Böylece ağdaki düğümlerin yönlendirme tabloları güncellenmiş olur.



Şekil 3.5 Karıncaların karar vermeleri.



Şekil 3.6 Karardan bir süre sonra.

Şekil 3.5 ve Şekil 3.6'da karıncaların yiyecek ararken nasıl karar verdiklerini göstermektedir. Şekil 3.4'de karıncalar için yuvadan yiyecek alanına doğru iki muhtemel yol bulunmaktadır. İlk olarak bir karınca yol ayrımına geldiğinde sağa ya da sola dönecektir. Rastgele sağa veya sola dönme ihtimali %50'dir. Eğer yol ayrımına iki karınca gelmişse bu iki karıncadan biri sola diğeri ise sağa dönecektir. Bir süre sonra karıncaların konumu Şekil 3.5' teki gibi olur. Yollardaki çizgiler karıncaların bırakmış olduğu feromen izleridir. Eğer karıncalardan biri uzun yolu seçtiyse ve hedefe hala ulaşamadıysa diğer karınca kısa yolu seçer. Başlangıçta yollarda feromen izi yoksa karıncalar iki yol için yolların %50 ihtimalle seçer. Eğer bir yolda daha fazla feromen izi varsa karıncanın o yolu seçme ihtimali daha yüksek olur. Çünkü daha fazla feromen izi daha kısa yol anlamına gelmektedir. Daha az seçilen yollarda feromen izi az olacağından bu izlerin zamanla ortadan kalkması daha çabuk olacaktır [40].

3.2.4.2. AntNet yönlendirme protokolü

Karınca ağı (AntNet) yönlendirme protokolü iletişim ağlarındaki problemleri çözmek amacıyla karıncaların davranışlarından ilham almıştır. Karınca koloni tabanlı algoritmalarda bir grup yapay karınca problemleri tespit etmek amacıyla ağ üzerinde hareket ederek topladığı bilgilerle de bu problemleri çözmeyi hedefler. Bu algoritmada her bir yapay karınca kaynak düğüm ile hedef düğüm arasında bir yol inşa eder. Bu yolu inşa ederken de zaman, yol ve ağın yük durumu hakkında bilgileri toplar. Daha sonra toplanan bu bilgiler ters yönde hareket eden diğer bir yapay karıncaya aktararak ziyaret edeceği düğümlerin yönlendirme tablolarını güncellemesi sağlanır.

AntNet algoritmasının çalışma prensibi aşağıdaki gibidir [42]:

1. Sabit zaman aralıklarında ağdaki her bir düğüm ağdaki diğer düğümler ile ilgili bilgi toplamak amacıyla komşu düğümlerine ileri öncü karınca(forward ant) gönderir.
2. Her bir karınca yollarını rastgele seçer.
3. Eğer düğüme gelen karınca geldiği düğüme daha önce geldi ise geriye bilgi göndererek yol ile ilgili bilgileri siler.

4. Eğer ileri öncü karınca hedefe ulaşırsa, toplamış olduğu bilgileri geri öncü (backward ant) karıncaya aktarır.
5. Geri öncü karınca almış olduğu bu bilgilerle kaynak düğüme geri döner ve yönlendirme tablolarını günceller.

3.2.4.3. Adaptive-SDR yönlendirme protokolü

Uyarlamalı oğul zekâsı tabanlı dağıtık yönlendirme (Adaptive Swarm-based Distributed Routing Adaptive-SDR) [41] protokolünde iletişim ağlarındaki problemleri çözmek amacıyla karıncaların davranışlarından ilham alınmıştır. Bu algoritma üç sürece ayrılır. Bunlar;

1. Koloni içerisindeki düğümler hakkında bilgi toplanması,
2. Özel gezgin görevlileri (karınca) kullanarak yolların bulunması ve
3. Karıncaların bulmuş olduğu yollar kullanılarak trafiğin yönlendirilmesidir.

Birinci aşama sıklıkla yerine getirilmez. Sadece algoritmanın başında ve topolojide her hangi bir değişiklik meydana geldiğinde meydana gelir. İkinci ve üçüncü aşama ise düzenli ağ işlemlerinin bir parçası olarak sürekli tekrarlanır. ABC ve AntNet yönlendirme protokollerinde ölçeklenebilirlik problemi mevcuttur. Bu problem eğer bütün düğümlerden diğer düğümlere karıncaları gönderilecek olursa ağda aşırı sayıda karınca olacağından yoğun bir trafik oluşur. Buna ilave olarak zaman aşımından dolayı karıncaların kaybolması, gelecek olan bilginin zaman aşımına uğraması problemini meydana getirir. Bu problemin çözümü ağı eşit düğüm sayılarına sahip kolonilere bölmektir. Böylece ağdaki karınca sayısı azaltılmış olur. Ağ kolonileri oluşturulduktan sonra iki tür karınca tanımlanır; bunlardan birincisi bulunduğu kolonilerin yönlendirme tablolarını oluşturur, ikincisi ise içerisinde bulunduğu bölgenin yönlendirme tablolarını günceller. Bu algoritmada işlenen toplam paket sayısı (throughput) ve gecikme (end-to-end delay) noktalarında AntNet algoritmasıyla karşılaştırıldığında daha iyi sonuçlar vermektedir [59].

3.2.4.4. BeeAdhoc yönlendirme protokolü

BeeAdhoc yönlendirme protokolü bal arılarının nektar toplama davranışlarından esinlenmiştir. Arılar, genellikle uzak mesafelere yiyecek bulmak amacıyla gitmek zorunda kalırlar. Yiyecek arama alanlarında besin kaynağı bulan arı, kolonisinin diğer üyelerine haber vermek için kovana geri döner ve bir süre sonra diğer arıların etrafında uçmaya başlarlar. Bal arıları sağırdırlar ve bu nedenle birbirleri ile sesli iletişim kuramazlar. Birbiri ile iletişimlerini değişik şekilleri yerine getirerek kurarlar. Bu şekillere sallanma dansı denir. Bu dansta besin kaynağının kovana uzaklığı, yönü, besinin kalitesi ve miktarı hakkında bilgiler mevcuttur. Suyun kısıtlı olduğu zamanlarda ise bu dans su kaynağının yerini göstermek içinde kullanılır [60].

Bu model üç tip arıya sahiptir. Bunlar *kâşif (scout), işçi(forager) ve onaylama (beeswarms)* arılarından oluşmaktadır. *Kâşif arıları,* ilk çıkış düğümünden hedef düğümüne kadarki yolları keşfeder. Bu işlem tüm komşularına belli bir zaman içinde gitmekle olur. Kâşif arı hedefine ulaştığı zaman hedefine ulaşırken takip ettiği yoldan geri döner. Kaynak düğümüne geri geldiğinde ise özel danslar vasıtası ile işçi arıları toplar. *İşçi arıları,* temel çalışan statüsündedir. Taşıma katmanından aldıkları paketi hedefe teslim ederler. İşçi arıları gecikme ve yaşam süresi olmak üzere iki türdür. Gecikme arıları ağdaki gecikme bilgilerini, yaşam süresi arıları da düğüm bataryalarının kapasite bilgilerini ziyaret ettikleri düğümlerden toplarlar. Gecikme arıları minimum gecikme ile paketleri belirtilen yollardan sevk ederken yaşam süresi arıları aynı zamanda paket güzergâhlarını belirleyerek ağın yaşam süresini arttırmaktadır. İşçi arılar PPM (point-to-point mode) ile hatta kalarak hedefe kadar hat ile ilgili bilgileri toplarlar. Hedefe ulaştığı zamanda o hedeften kaynağa kadar ağ trafiğinde kalır. Bu da kontrol paketlerinin genel giderini azaltır ve böylece batarya konusunda tasarruf edilmiş olur. TCP gibi güvenli protokoller için alınan paketleri tasdik eder. *Onaylama arıları,* eğer uygulama güvensiz UDP protokolü kullanıyorsa işçi arıların kaynağa geri dönüşünde kullanılır. UDP protokolünde paketlerin hedefe ulaşıp ulaşmadığı hakkında geri dönüş onaylaması yoktur.

Her düğüm için üç tip yönlendirme tablosu mevcuttur. Bunlar;

1. Nektar arama bölgesi için kuyruklama, yayılma gecikmelerinin tutulduğu tablo (Intra Foraging Zone -IFZ),
2. Nektar arama bölgeleri arası farklı bölgelere veri gönderildiğinde gereken yönlendirme tablosu (Inter Foraging Region-IFR) ve
3. Bölgedeki temsil edilen düğümler için bilinen hedeflerin eşlemelerinin tutulduğu yönlendirme tablosudur (Foraging Region Membership-FRM) .

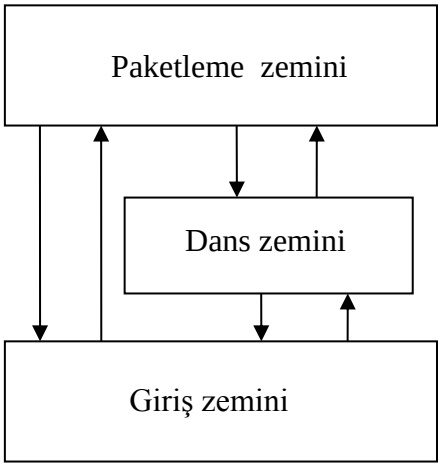
Ağdaki her bir düğüm kovanı temsil eder ve üç kısma ayrılır. Bunlar; paketleme, giriş ve dans zeminidir (Şekil 3.7). Giriş MAC katmanı ile paketleme ise taşıma katmanı ile ara yüzü oluşturmaktadır. Protokolde kullanılan önemli semboller Tablo 3.1'de gösterilmektedir.

Paketleme zemini, TCP veya UDP gibi daha üst seviye katmanı olan taşıma katmanı ile ara yüzü oluşturmaktadır. Taşıma katmanından paket geldiği zaman dans zeminine bir uygun işçi için bakılır eğer bir tane bulunursa veri paketi kapsül yapılır ve S_{sd} değişkenine yüklenir aksi takdirde veri paketi bir süre kuyrukta bekletilir. Eğer işçi arı yoksa veya belirli zamanda gelmese kâşif arı belirtilen hedef için yeni yollar bulmak amacıyla gönderilir.

Giriş zemini, MAC katmanı ile ağ katmanı arasında bir ara yüzdür. Bütün gelen ve giden paketlerle ilgilenir. Bir kâşif arı giriş zeminine geldiği zaman eğer bu arının yaşam süresi bitmedi ise veya hedefine varmamışsa bu arıyı diğer düğümlere gönderir. Kâşif arının kimliği ve kaynak düğümü ile ilgili bilgiler listeye kaydedilir. Eğer aynı kâşif arıdan bir tane daha gelirse bu arı kovanda öldürülür. Eğer düğümde aynı hedef için hem işçi arı ve kâşif arı dans zemininde var ise o zaman gerekli yol işçi arı ile kâşif arıya verilir. Yani kâşif arı işçi arıdan yol bilgisini alır. Böylece işçi arı yol bilgisini de taşımış olur. Eğer bulunulan düğüm işçi arının hedefi ise paketleme zeminine gönderilir; değilse direk MAC katman ara yüzüne bir sonraki düğüm için gönderilir.

Dans zemini, kovanın kalbidir çünkü burası yönlendirme bilgilerini sağlar. Bir işçi arı yolculuğunu bitirdikten sonra gitmiş olduğu zikzaklı yolların kalitesine göre yeni

işçileri yapmış olduğu dans ile görevlendirir. Bununla birlikte her bir işçi arının kalite ölçüsü farklıdır.



Şekil 3.7 Beeadhoc protokolünün yapısı.

Daha önce bahsedildiği gibi yaşam süresi işçileri yol kalitelerini yoldaki düğümlerin kalan batarya kapasitelerini temel alarak değerlendirir. İki senaryoya göre yaşam süresi işçileri kendilerini klonlayabilir. Birincisi yoldaki düğümün yeterince batarya kapasitesi varsa (iyi yol); ikincisi eğer yolundaki düğüm kapasitesi az olmasına rağmen çok fazla paketlerin beklediği söz konusu ise. İkinci durum iyi yoldan göndermeye daha az duyarlıdır. Diğer taraftan eğer herhangi bir paket beklemiyorsa iyi yollu işçi arı arkadaşlarının yaptıkları işi iyi yaptığından dolayı dans etmeyecektir. Bu fikir direk olarak kâşif/işçi arılarının doğal yaşam davranışlarından esinlenilmiştir. Böylece birçok işçi arının yollarının düzene konmasına yardımcı olur. Dans zemini aynı zamanda paketleme zeminine veri paketi gönderme isteğine cevap olarak bu isteğe uygun işçi gönderir. Yaşam süresi bitmiş olan işçi arılar bu fonksiyon içinde düşünülmemektedir. Eğer birçok işçi arı aynı özelliklere sahipse aralarında rastgele seçim yapılır. Buda paketlerin birçok yoldan dağıtımına yardımcı olur. Bu dağıtımın iki amacı vardır birincisi aşırı yüklenmelerdeki tıkanıklığı azaltmak; ikincisi farklı düğüm bataryalarının eşit olarak azaltılmasını sağlamaktır. Seçilen işçiden bir kopya paketleme zeminine gönderilir ve orijinal işçinin dans numarası bir azaltılarak dans zemininde saklanır. Eğer dans numarası sıfır olursa orijinal işçi paketleme zeminine gönderilir ve onun girişi giriş zemininden silinir.

Tablo 3.1 BeeAdhoc protokolünde kullanılan semboller.

Sembol	Tanımı
s	Paketin kaynak düğümü
D	Paketin hedef düğümü
i	Bulunulan düğüm
S_{sd}	*s* den *d* ye doğru yüklenmiş kaşif arı
F_{sd}	*s* den *d* ye doğru yüklenmiş işçi arı
D_{sd}	*s* den *d* ye doğru yüklenmiş veri paketi
P_{sd}	Bulunulan düğümde alınmış herhangi bir paket kaynağı *s* hedef *d*
h_{next}	Kaynaktan hedefe doğru bir sonraki adım
Ld	Hedef *d* ye doğru gidecek olan işçi arı listesi

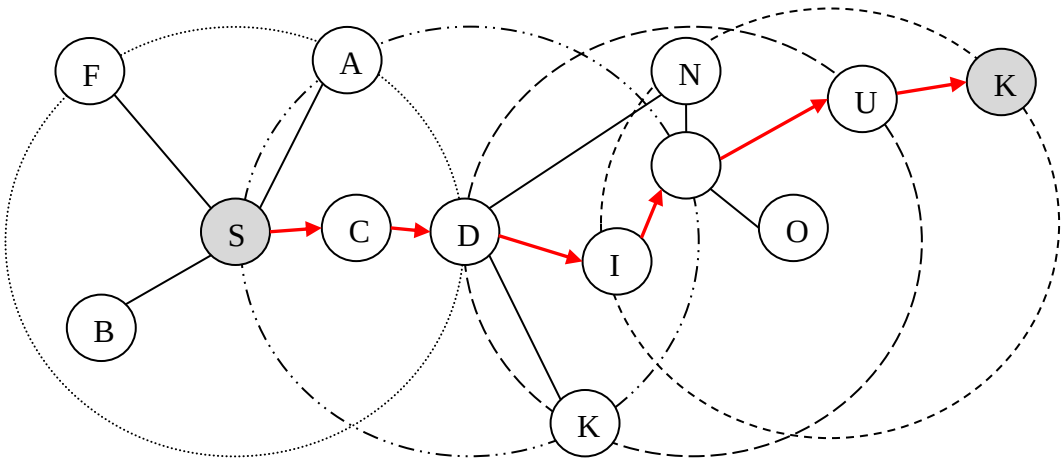
Yukarıda bahsedilen prensibe göre genç işçi arılar (en son ki yolları tespit eden) yaşlı olanlara göre daha fazla tercih edilip daha uzun ömürlü olurlar. Eğer son işçi arıda hedef düğüm için kovanı terk etti ise bu durum kaynak ile hedef düğüm arasında yol bilgisi kalmamış olur. Bir hedef için bir yol varsa eninde sonunda bir işçi kovana dönecektir ve eğer hiçbir işçi belirli bir zamanda dönmezse düğüm muhtemelen hedef için yolu kaybetmiştir. Bu da diğer algoritmalara göre daha az kontrol paketi iletimi ile sonuçlanır ve böylece daha az masraf olur [43].

3.2.4.5. Hopnet yönlendirme protokolü

Hopnet yönlendirme protokolü iki önemli noktadan oluşur. Birincisi düğümün sınırları içerisindeki yerel (local) yolun bulunması, diğeri ise sınırlar arasındaki, reaktif iletişimdir.

Bu algoritmada sadece iki düğüm vardır (Şekil 3.8):

1. Sınır (boundary) düğümler (s düğümü için yarıçap uzaklığındaki düğümler a, d, f, g) ve
2. İç düğümler (b, e, c).



Şekil 3.8 Hopnet kontrol paket yayım şeması [44].

Hopnet yönlendirme protokolü iki adet yönlendirme tablosuna sahiptir. Bunlar:

1. intraRT yönlendirme tablosu ve
2. interRT yönlendirme tablosu.

intraRT bölge içi düğümleri ile elde edilir. Düğüm bölgesi içerisindeki düğümle hızlı bir şekilde iletişim kurabilir. Buda öndeki gözcü arıların bölge içerisinde sürekli gönderilmesi ile olur. InterRT, bölge dışındaki düğümle iletişim kurulacağı zaman talep edilir. Eğer aranan düğüm bilginin gönderileceği düğümün alanında değilse interRT yönlendirme tablosu kullanılır.

Bu algoritma; teslimat oranı (delivery ratio), gecikme (end to end delay) ve ağ performansı AODV ile karşılaştırıldığında daha iyi sonuç vermektedir. Büyük ağlarda daha iyi sonuçlar alınmaktadır [44].

3.2.4.6. Bulanık karınca koloni tabanlı yönlendirme protokolü

Bulanık karınca koloni tabanlı (Fuzzy Ant Colony Based) yönlendirme protokolü Ad-Hoc ağlar için bulanık mantık (Fuzy logic) kullanılarak geliştirilmiştir. Ağ topolojisinde birbiri ile ilgili birçok faktör ele alınmıştır. Bunlar;

1. Batarya kapasitesi,
2. Trafik düzeni,
3. Bağlantı sağlamlığı ve
4. Düğümlerin hareketliliği.

Bulanık mantık için giriş değerleri şunlar seçilmiştir;

1. Sinyal gücü,
2. Batarya kapasitesi ve
3. Düğüm tampon meşguliyeti seçilmiştir.

Bu üç giriş bulanık mantıkla değerlendirildiğinde çıkış olarak bulanık maliyet için bulanık üye fonksiyonu hesaplanmıştır. Bir veya ikisinin dikkate alınması en iyi yolun seçimi için yeterli değildir. Örneğin sadece en kısa yol düşünüldüğünde ve yukarıdaki faktörler ele alınmadığında, yol üzerindeki düğümün enerjisi yeterli değilse veya çok az ize o düğümün kullanılması uygun olmaz ya da kullanılacak yol kararlı değilse, problem olabilir. Bundan dolayı bu algoritmada bağlantı yolu seçilirken yukarıdaki faktörlerin durumunu genel olarak değerlendirip en uygun yolun seçilmesi sağlanmıştır [45].

3.3. Yönlendirme Protokollerinin Başarım Değerlendirmesi

Bu bölümde Ns-2 [61] simülasyon aracı kullanılarak en yaygın bilinen ad hoc protokolleri başarım analizine tabi tutulmuştur. Sonuçlar grafikler halinde sunulmuş ve değerlendirilmiştir.

3.3.1. Simülasyon parametreleri

Ad hoc ağlar için geliştirilmiş olan ve yaygın olarak kullanılan AODV, DSDV ve BeeAdhoc protokollerinin simülasyon tabanlı analizi ve karşılaştırma çerçevesi Ns-2 ağ simülatörü altında oluşturulmuştur. Ns-2 simülatöründe aynı trafik ve ortam şartları altında tüm protokoller çalıştırılmış ve sonuçlar grafikler halinde sunulmuştur.

Böylece, simülasyon tabanlı başarım analizi yardımıyla, protokollerin ölçeklenebilirlik ve verimlilik gibi yönleri değerlendirilmiştir.

Tablo 3.2 Simülasyon model parametreleri.

Parametre	Değer	
Topografya	Düğüm Sayısı	Alan (m^2)
	10	300 x 300
	50	500 x 500
	100	700 x 700
	200	1500 x 1500
Hareket hızı (m/s)	1 ~ 20	
Paket boyutu (Bayt)	512	
Topoloji üreteci	SETDEST	
Trafik üreteci	CBRGEN	
Trafik modeli	FTP/TCP	
Simülasyon süresi (s)	50	
Bilgisayar	Core 2 Duo, 2 GB RAM	
İşletim sistemi	Fedora 6	

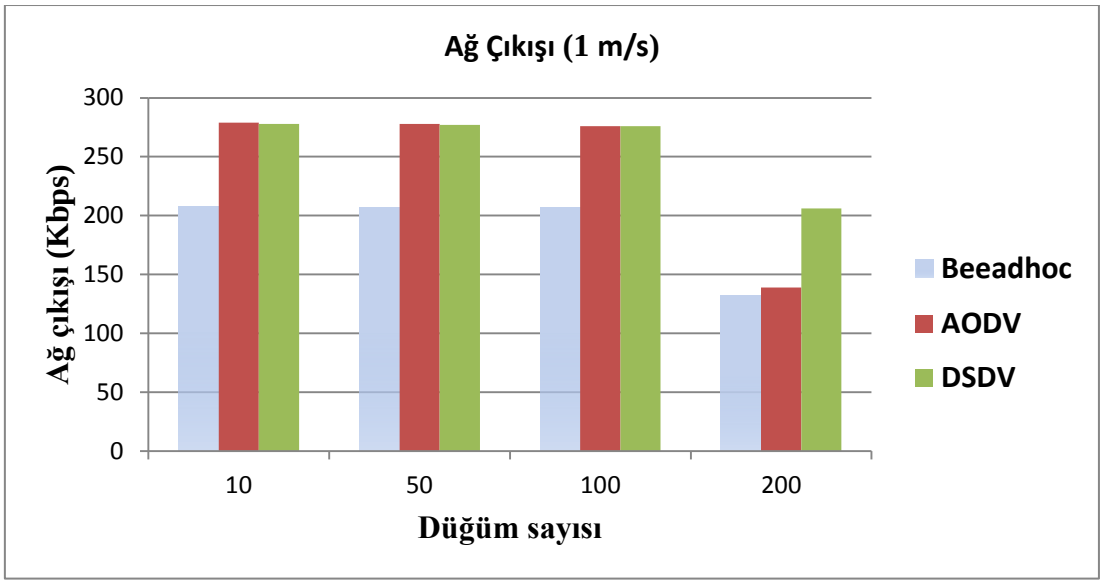
Tablo 3.2'de karşılaştırma sürecinde kullanılan simülasyon parametreleri verilmektedir. Bu çalışmada geliştirilen modeller, 1 m/s (yürüyüş hızı), 5 m/s (koşma), 10 m/s (araç üzerinde) ve 20 m/s gibi farklı hızlarda farklı düğüm sayıları için simülasyonlar çalıştırılmıştır. Düğüm sayıları ise 10, 50, 100 ve 200 olarak seçilmiştir. Burada düğüm sayılarının maksimum boyutu kullanılan bilgisayarın konfigürasyonu ile yakından ilgilidir. Kullanılan bilgisayar 200 düğüme kadar kablosuz ağların simülasyonuna izin vermektedir, bu değerden sonra bellek yetersiz hatasıyla karşılaşılmıştır. Düğüm sayısı arttıkça topografyanın da boyutu artmaktadır (bakınız Tablo 3.2). Bu senaryolar Ns-2 ağ simülatörü içerisinde bulunan setdest.exe [61] aracı yardımı ile trafik senaryosu ise yine Ns-2 ile birlikte gelen cbrgen.tcl [61] kullanılarak elde edilmiştir.

Bütün düğümlerde TCP iletim protokolü yanında FTP uygulaması kullanılmıştır. Veri paketi büyüklüğü 512 bayt olarak sabit tutulmuştur. Fiziksel ve MAC katmanlarında aynı modeller kullanılmıştır.

Algoritmalar için uygun bir değerlendirme yapabilmek maksadıyla simülasyon süresi 50 saniye gibi uzun bir süre seçilmiştir. Bu süre boyunca simülasyonun tamamlanma süresi 10 düğümlü ağ için birkaç dakika iken 200 düğümlü ağ için saatler mertebesindedir.

3.3.2. Simülasyon sonuçları

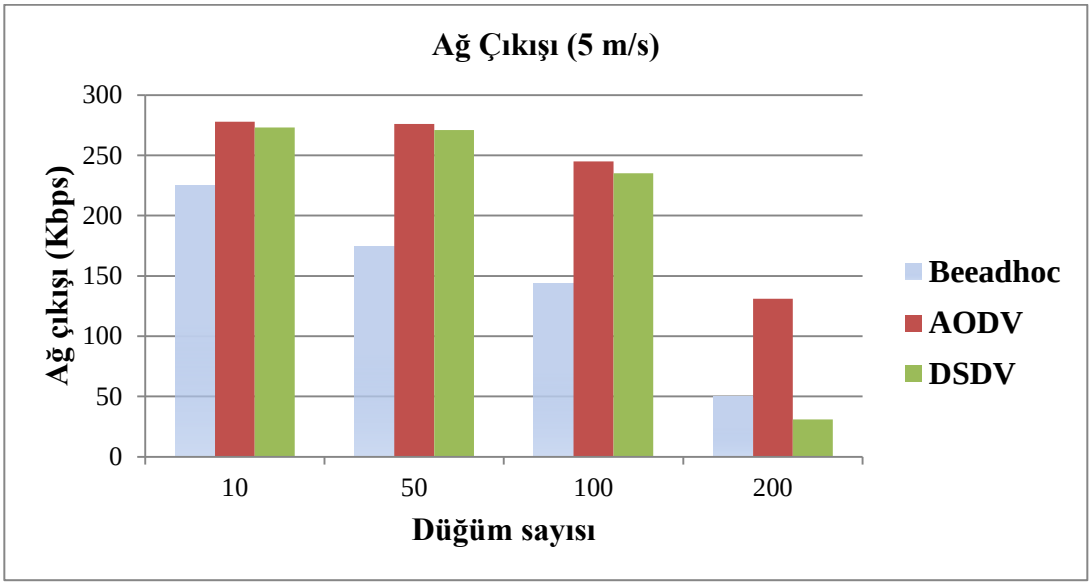
Yukarıda bahsi geçen protokollerin değişik düğüm sayısı ve hareket hızlarında ağ çıkışları (throughput) karşılaştırılmaktadır. Ağ çıkışı bir ağda herhangi bir düğümün benzetim süreci boyunca almış olduğu trafik paketleridir. Dolayısıyla protokol verimliliğini ölçmek için en uygun kriterlerden bir tanesidir. Şekil 3.9, Şekil 3.10, Şekil 3.11 ve Şekil 3.12'de dört farklı ağın değişik hızlar altındaki ağ çıkış değerleri görülmektedir.



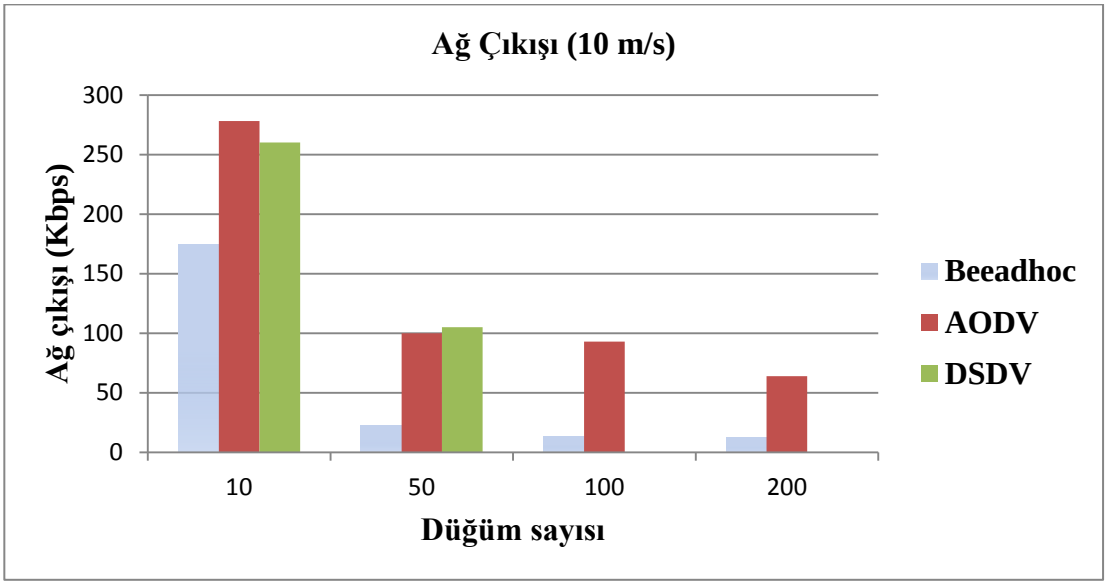
Şekil 3.9 Maksimum 1 m/s hız için ağ çıkışı.

Şekil 3.9'da, düğümlerin maksimum 1 m/s hızla hareket ettiği bir ağda BeeAdhoc, AODV ve DSDV protokollerinin çıkış değerleri verilmektedir. Tüm hız değerlerinde

AODV ve DSDV hemen hemen aynı verimlilikte çalışırken, BeeAdhoc verimliliği düşük kalmaktadır. Ayrıca hız 200 düğümlü ağda DSDV'nin daha verimli çalıştığı görülmektedir. Burada DSDV protokolünün tablo tutmamasının dezavantajı ağdaki düğüm sayısının az olmasından dolayı çok fazla hissedilmemiştir.



Şekil 3.10 Maksimum 10 m/s hız için ağ çıkışı.

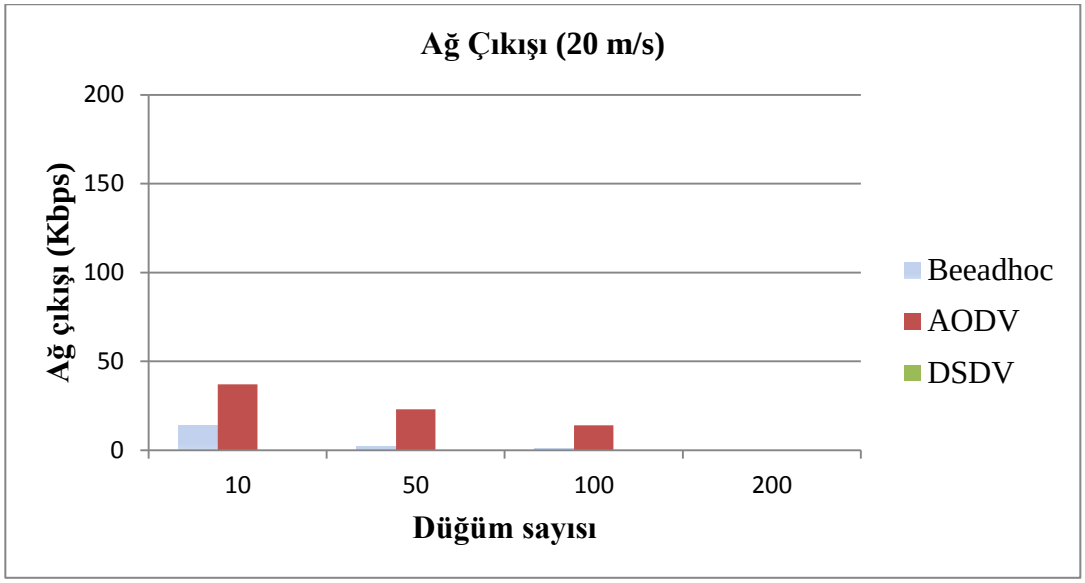


Şekil 3.11 Maksimum m/s hız için ağ çıkışı.

Şekil 3.10'da düğümlerin maksimum 5 m/s hızla hareket ettiği bir ağda BeeAdhoc, AODV ve DSDV protokollerinin çıkış değerleri verilmektedir. DSDV protokolü tablo

tutması nedeniyle düğüm sayısı fazla olan ağlardaki dezavantajı ortaya çıkmaktadır ve AODV protokolünden daha geride kalmaktadır. BeeAdhoc verimliliği 200 düğümlü ağ için daha düşük kalmaktadır.

Şekil 3.11'de düğümlerin maksimum 10 m/s hızla hareket ettiği bir ağda AODV'nin daha performanslı çalıştığı görülmektedir. 100 ve 200 düğümlü ağda DSDV yönlendirme protokolünün tablo tutması nedeniyle hiç çıkış vermemektedir.



Şekil 3.12 Maksimum 20 m/s hız için ağ çıkışı.

Şekil 3.12'de düğümlerin maksimum 20 m/s hızla hareket ettiği bir ağda ölçek arttığından dolayı AODV haricindeki protokoller neredeyse hiç çıkış vermemektedir. Bu sonuçla, AODV'nin tablo tutmaması nedeniyle daha verimli ölçeklenebildiği söylenebilir. Beeadhoc protokolü yalnızca 10 düğümlü ağda az sayıda paketi hedef düğüme iletebilmiştir.

BÖLÜM 4. NS-2 AĞ SİMÜLATÖRÜ

4.1. Giriş

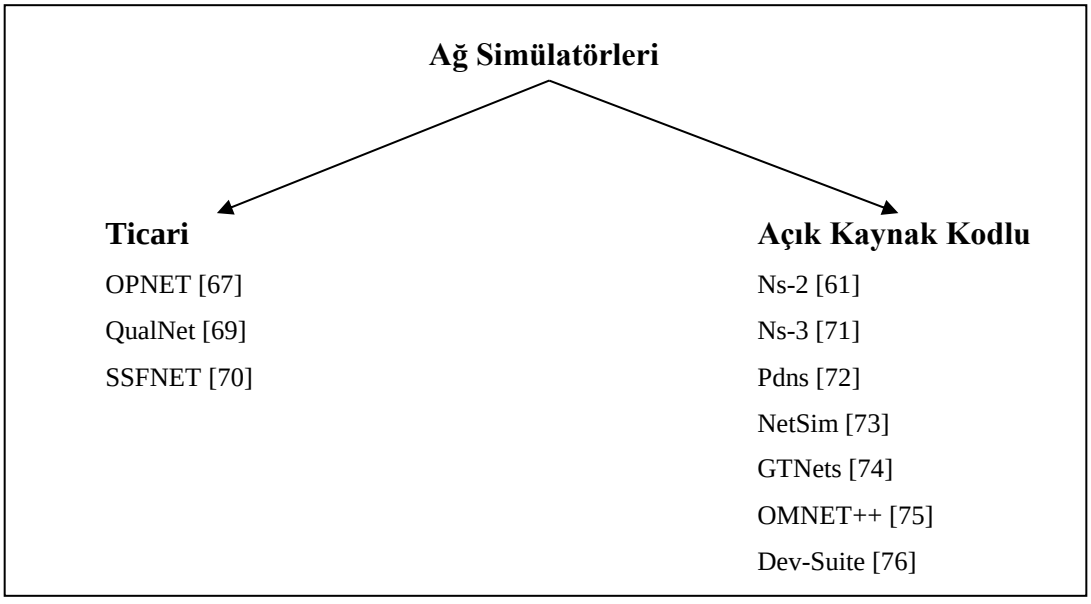
Simülasyon son yıllarda yaygınlaşan önemli bir teknolojidir. Bilgisayar ağı araştırma ve tasarımında modelleme ve simülasyon yönteminin, gerçek bir ağın modellenmesi, yeni ağ teknolojilerinin geliştirilmesi, değişik ağ şartlarında oluşturulması ve test edilmesinde önemli bir etkisi vardır [62, 63, 64]. Bilgisayar simülasyonu birçok doğal sistemin analiz ve modellenmesinde yaygın olarak fizik, kimya, biyoloji, ekonomi, inşaat, makine ve bilgisayar mühendisliği gibi alanlarda kullanılmaktadır. Simülasyon teknolojisi uygulamaları günümüzde son zamanlarda daha yaygın olarak uygulanmaktadır. Bunun sebebi ağ tasarım aşamalarının maliyeti yüksek ve zaman alıcı olmasından kaynaklanmaktadır. Tasarım süreçlerini kolaylaştırmak amacıyla araştırmacılar ve ağ cihaz üretici firmaları farklı ağ modelleme ve simülasyon araçları geliştirmişlerdir. Ağ simülasyon yazılımları günümüzde eğitim ve ticari amaçlar doğrultusunda kullanılmaktadır. Ağ işlemlerinin gerçeğe daha yakın senaryolarla analizi için ilk zamanlarda analitik yöntem ve araçlar yeterliyken günümüzde ağ donanım cihazlarının çok daha karmaşık olması, hız, protokollerin çokluğu, büyük ölçekli ağların yapısı gibi sebeplerden dolayı simülasyonu gerekli kılmaktadır [46, 47]. Örnek olarak bir ağ simülasyonunda performans analizi gerçek bir sistemin görsel analizine göre daha etkili olmaktadır. Buna ilave olarak ağ teknolojisindeki gelişmelerin hızla ilerlemesi, yeni ağ donanım cihazlarının geliştirilmesi ve farkı ağ protokollerinin İnternete katılması gibi durumlar ağ simülasyonunun önemini daha da arttırmaktadır. Bundan dolayı ağ simülasyon ortamları farklı kişi ve kuruluşların oluşturacakları yeni ağ donanım cihazları ve protokollerin test edilebilmesi için açık kaynak kodlu olmalıdır. İnternetin iletişim yapısını oluşturan TCP/IP protokolü katmanlar içeren protokoller bütünüdür. Araştırmacılar bu katmanlardan her hangi birisi komşu katmanlarla gerekli koordinasyonu sağlayarak istenilen katmanda ağ

simülasyon araçları kullanılarak geliştirme sağlamaktadırlar. Ağ simülasyon araçları araştırmacılar tarafından akademik araştırmalar ve endüstriyel gelişimler için farklı protokollerin karşılaştırılması amacıyla kullanılmaktadır. Aynı zamanda aynı protokol üzerinde farklı parametreler içinde protokoller test edilmektedir. Genellikle ağ simülatörleri çok geniş teknolojileri, protokolleri, çok karmaşık düğüm kümelerini ve düğüm bağlantılarını test edebilmektedir.

Genel olarak ifade etmek gerekirse, ağ simülatörleri gerçek dünyada kullanılan ağları modellemeye çalışır. Genel olarak bir sistem örnek alınır ve daha sonra modelin özellikleri değiştirilerek yeni sonuçlar elde edilip bu sonuçlar analiz edilir. Oluşturulan bilgisayar ortamındaki yeni modelin değiştirilmesi ve geliştirilmesi gerçek dünyadaki modele göre çok düşük maliyetli olur. Ancak ağ simülatörleri mükemmel değildir. Mükemmel bir ağın tüm özellikleri modellenemeyebilir. Ancak iyi modellenmiş bir ağ araştırmacılara gerçek dünya modeline çok yakın sonuçlar verebilir.

4.2. Ağ Simülatörlerinin Sınıflandırılması

Ağ simülatörleri ticari ve açık kaynak kodlu olmak üzere iki gruba ayrılabilir (Şekil 4.1).



Şekil 4.1 Ağ simülatörlerinin sınıflandırılması.

Ticari simülatörler, üretici firmalar tarafından kâr amacı gözetilerek oluşturulmuş olan simülatörlerdir. Bu simülatörlerin açık kaynak kodları simülatörü oluşturan firmalar tarafından genel kullanıcılara verilmemektedir. Bütün kullanıcılar simülatörü kullanabilmek için lisans ücreti ödemek zorundadırlar. Bu simülatörlere örnek olarak OPNET [67] verilebilir.

Ticari yazılımlar hem avantajlara hem de dezavantajlara sahiptir. Çalıştırdıkları personel sayesinde ticari simülatörler eksiksiz dokümanı bulunmakta ve sürekli güncellenebilmektedir. Bu avantaj olarak görünebilmekle beraber yeterli personel firmalar tarafından çalıştırılmadığında bir dezavantaja dönüşebilmektedir. Farklı versiyonlarda oluşturulmuş ticari simülatörler yeni kullanıcılar için sıkıntı oluşturabilmektedir.

4.3. Ağ Simülatörlerin karşılaştırılması

Günümüzde araştırmacılar ve ticari firmalar tarafından çalışırken iletişim ağlarının simülasyonlarını gerçekleştirmek amacıyla birçok ağ simülatörleri kullanılmaktadır. Bu simülatörlerin birinde kendine ait birçok özellikleri bulunmaktadır. Bu özellikler kullanıcıların amaçları doğrultusunda değerlendirilip her bir kullanıcı araştırma konularının ihtiyaç duyduğu özelliklere göre uygun simülatör seçimi yapılmaktadır. Bu özellikler; Simülasyon amacı, Model kütüphanesi zenginliği, Analiz, Esneklik, Doküman bolluğu, Kullanım kolaylığı, Kullanıcı arayüzü, Ölçeklenebilirlik, Performans, Ağ modeli, Lisans, Programlama dili ve çalıştığı platform olarak sıralanabilir [77]. Tablo 4.1'de yaygın olarak kullanılan bazı simülatörlerin yukarıda belirtilen özelliklere göre değerlendirilmeleri gösterilmektedir.

Ticari simülatörlerden özellikle OPNET ağ simülatörü, yukarıda belirtilen özelliklerin hemen hemen tamamında son derece güçlü olduğu görülmektedir. Eğitim ve araştırma amaçlı simülatörlerden Ns-2, analiz, kullanıcı arayüzü, kulanım zorluğu gibi olumsuzluklara rağmen güçlü kütüphanelerinin bulunması, açık kaynak kodlu olması, doküman bolluğu, kullanılan programlama dilinin yaygın olan C++ olması ve güçlü performansından dolayı birçok araştırmacı tarafından yaygın olarak kullanılmaktadır.

Tablo 4.1 Yaygın olarak kullanılan ağ simülasyon araçlarının karşılaştırılması [78].

Özellik	Ns-2	pdns	OPNET	OMNeT++	QualNet	SSFNet	DEVS-Suite
Amaç	Eğitim, Araştırma	Eğitim, Araştırma	Ticari	Eğitim, Araştırma	Ticari	Ticari, Araştırma	Eğitim, Araştırma
Model Kütüphanesi	Güçlü	Güçlü	Güçlü	Güçlü	Orta	Zayıf	Zayıf
Analiz	Orta	Orta	Çok Güçlü	Zayıf	Zayıf	Zayıf	Çok Güçlü
Esneklik	Orta	Orta	İyi	Çok İyi	İyi	İyi	Çok İyi
Doküman	Orta	Orta	Çok Güçlü	Güçlü	Zayıf	Zayıf	Orta
Kullanım Kolaylığı	Zor	Zor	Kolay	Orta	Orta	Zor	Kolay
Arayüzü	Zayıf	Zayıf	Güçlü	Güçlü	Güçlü	Güçlü	Güçlü
Ölçeklenebilir	Orta	Çok iyi	Orta	İyi	Çok iyi	Çok iyi	İyi
Performans	Güçlü	Çok güçlü	Orta	Orta	Güçlü	Çok güçlü	Çok Güçlü
Ağ Modeli	WAN	Büyük ölçek	LAN uydu	LAN, MAN, Wireless	Kablosuz uydu	Büyük ölçek	LAN, MAN, WAN
Lisans	Açık kaynak	Açık kaynak	Ticari	Açık kaynak	Ticari	Açık kaynak	Açık kaynak
Programlama dili	C++ ve OTcl	C++ ve OTcl	C++	C++	C++	Java, C++	Java
Platform	Unix, Linux	Unix	Windows	Windows	Linux, Windows	Linux, Windows	Linux, Windows

*A*çık kaynak kodlu simülatörler, ticari bir amaç gözetilmeden genellikle akademik kuruluşlar tarafından oluşturulup araştırmacıların kullanımına ücretsiz olarak sunulmaktadır. Farklı kullanıcılar veya kuruluşlar açık kaynak kodları kullanarak simülatöre katkıda bulunabilir ve aynı zamanda varsa hataları düzeltme imkânına sahiptir. Simülatör ara yüzü geliştirilecek durumlar için imkân sağlamaktadır. Böylece

geliştirilecek yeni teknolojik gelişmeler ticari simülatörlere göre daha hızlı bir şekilde simülatöre eklenebilmektedir. Bu avantajların yanında kurulumu, doküman ve sürüm problemlerinden dolayı dezavantajları da bulunmaktadır. Açık kaynak kodlu simülatörleri örnek olarak Ns-2 [61] verilebilir.

4.4. Ns-2 Ağ Simülatörü

Ns-2 [61] en popüler açık kaynak kodlu ağ simülatörlerinden biridir. Ns-2 simülatörü Ns simülatörünün ikinci versiyonudur. Kablolu simülasyonun yanı sıra yönlendirme algoritmaları, TCP ve UDP gibi kablosuz ağ protokolleri de Ns-2 içerisinde kullanılabilir. Ns-2 ağ simülatörü ayrık olay tabanlı olup University of California-Berkeley tarafından geliştirilmiştir. Temel olarak REAL [66] ağ simülatörü tabanlıdır. Ns simülatörü ilk olarak 1989 yılında geliştirilmiş ve günümüzde de halen geliştirilmeye devam etmektedir. Mevcut güncel NS projeleri DARPA [79] tarafından desteklenmektedir. Şu anda Ulusal Bilim Vakfı (NSF) gelişimine katkı sağlamaktadır [80].

Ns simülatörünün mevcut güncel versiyonu olan Ns-2 simülatörü yaygın olarak araştırmacılar tarafından kullanılarak gönüllü olarak simülatöre katkılar sunulmaktadır. Güncel değişiklikler Ns-2 simülatörünün resmi web sitesinden kullanıcılara sunulmaktadır [61, 81].

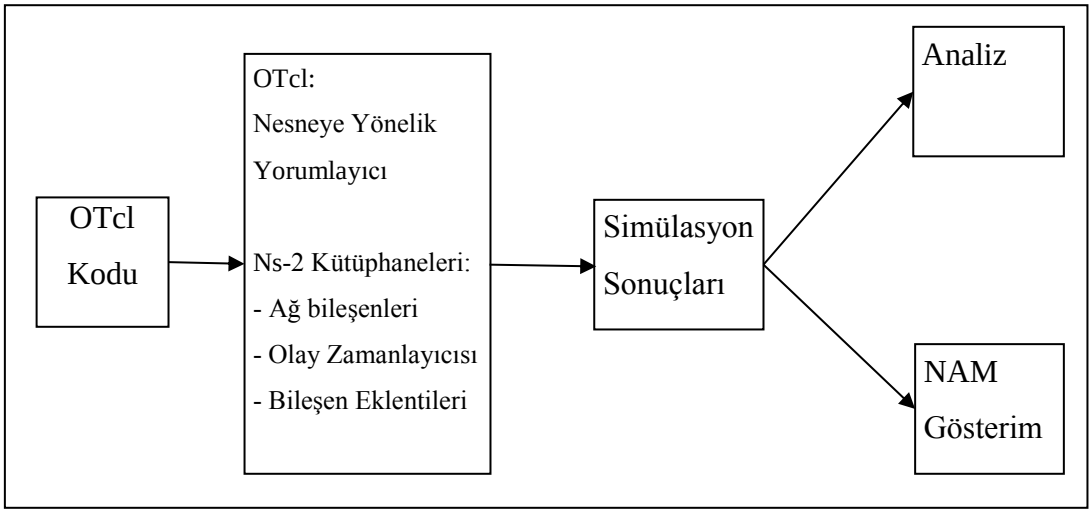
4.4.1. Ns-2 simülatörünün temel yapısı

Programlama olarak C++ nesneye yönelik bir dille geliştirilmiştir. Ns-2 içerisinde çok sayıda C++ nesneleri bulunmaktadır. Bu Tcl simülasyon komut dosyası kullanarak bir simülasyonunu kurmak için bu C++ nesneleri kullanılabilir. Buna rağmen birçok ileri düzeyde kullanıcı bu nesneleri yetersiz gördüğünde kendi nesnelerini kolayca oluşturabilmektedir. Bu nesneleri geliştirmek ve birlikte kullanabilmek için OTcl yapılandırma arayüzü kullanılmaktadır. Yapılan uygulamaların çıktıları görsel ve metin tabanlı bir dosyada incelenebilir. Ağının belirli bir davranışını analiz etmek,

kullanıcıların metin tabanlı veri ile ilgili dosyalar ayrı ayrı oluşturulan sonuçlar ayrıntılı olarak incelenebilir.

OTcl programlama dili kullanılarak düğümler, bağlantılar ve topolojiler tanımlanmıştır. Bu iki programlama dili kullanılmasının nedeni bu iki dilin kendine has özellikleridir. C++ bir tanım oluşturmak için çok etkili bir dil olmasına rağmen görsel olarak etkili bir dil değildir. Görsel editör olarak NAM (Network Animatör) kullanılmaktadır.

Görsel olarak parametrelerin değiştirilmesi çok kolay değildir. Buna ilave olarak Ns-2 veri yolu uygulamasını kontrol yolu uygulamasından ayırmaktadır. Olay işleme süresini (event processing time) ve paket işlemleri süresini azaltmak için veri yolundaki olay zamanlayıcısı ve temel ağ bileşenleri C++ dilinde yazılmıştır. OTcl programlama dili C++ programlama dilinin etkin olmadığı görsel bölümde kullanılmaktadır. Böylece bu iki dilin etkin olduğu yönler bir araya getirilerek daha etkin bir simülatör oluşturulmuştur. C++ ile protokollerin detayları oluşturulmuştur. OTcl ile simülasyon süreleri ve olay zamanları kontrol edilmektedir.



Şekil 4.2 Ns-2 simülatörünün kullanımı.

OTcl dili ağ topolojisinin kurulumu, kaynak düğüm ve hedef düğüm trafiklerinin belirlenmesi, paket gönderilme olay zamanlayıcısının(event scheduler) kurulması ve paket gönderilmesinin sonlandırılması işlemlerinde kullanılmaktadır. Bu parametreler

OTcl programlama dili ile kolayca belirlenebilir. Kullanıcı yeni bir nesne üretmek istediğinde kolayca üretebilir veya her hangi bir nesne üzerinde değişiklik yapabilir. Bu da Ns-2 simülatörünü en güçlü yapan özelliklerindendir. OTcl ile yazılmış kodun Ns-2 simülatörünün kullanıcı tarafından kullanımı Şekil 4.2'de gösterilmektedir. Ns-2 ağ simülatörünün en önemli özelliği de olay zamanlı olmasıdır.

4.4.2. Ns-2 simülatörünün kurulumu

Ns-2 ücretsiz elde edilebilen bir simülasyon aracıdır. Linux, Windows ve Mac sistemleri de dahil olmak üzere çeşitli platformlarda çalışır. Kullanıcılar genellikle Ns-2 simülatörünü Linux ortamında kullanmaktadır. Ns-2 kaynak kodları "all-in-one" ve bileşenler olarak ikiye ayrılmaktadır. All-in-one ile tüm bileşenler kullanıcının hizmetine sunulurken bileşenler ile kullanıcılar yalnızca ihtiyaç duyduğu kısımları sisteme yüklerler. Genellikle yeni kullanıcılar "all-in-one" bileşenlerini kullanmaktadır. Bu bileşenler "install" komutu ile sistemin yüklenmesini "make" komutu ile de simülatörün kullanıcıya hazır hale gelmesini sağlar [82].

Mevcut all-in-one aşağıdaki ana aşağıdaki bileşenlerden oluşur:

1. NS release 2.29,
2. Tcl/Tk release 8.4.11,
3. OTcl release 1.11 ve
4. TclCL 1.17.

İsteğe bağlı bileşenler aşağıdaki gibidir:

1. NAM 1.11,
2. Zlib 1.2.3,
3. Xgraph 12.1

Bileşen bazlı yaklaşımın fikri, yukarıda parçaları tek tek elde ederek yüklenmesidir. Bu seçenek, zaman ve bellek alanı indirme konusunda önemli miktarda tasarruf sağlamaktadır. Bu durum yeni başlayanlar için biraz sıkıntı olabilmektedir.

Lunix tabanlı işletim sistemlerinde Ns-2 yüklenmesi aşağıdaki komutlarla sırayla gerçekleştirilmektedir.

1. "ns-allinone-2.29.tar.gz" dosyasının Ns-2 web sitesinden indirilmesi. www.isi.edu/nsnam/dist
2. "ns-allinone-2.29.tar.gz" dosyasını ana dizine kopyalanıp tar dosyasından çıkarılması. Yeni bir "ns-allinone-2.29" dosyasının oluşturulur.
3. "ns-allinone-2.29" dosyasındayken "./install" komutunun çalıştırılması.
4. Terminalde "gedit~/.bash" komutu çalıştırılarak gelen dosyanın aşağıdaki gibi güncellenmelidir.

```
NS_HOME=/home/Owner/ns-allinone-2.29
PATH=$NS_HOME/tcl8.4.11/unix:$NS_HOME/tk8.4.11/
unix:$NS_HOME/bin:$PATH
LD_LIBRARY_PATH=$NS_HOME/tcl8.4.11/unix:$NS_HOME/tk8.4.11/unix:\
$NS_HOME/otcl-1.11:$NS_HOME/lib:$LD_LIBRARY_PATH
export TCL_LIBRARY=$NS_HOME/tcl8.4.11/library
```

5. "./validate" komutu ile sistemin test edilmesi.
6. "ns" yazılarak ekranda % sembolünün görünmesi.

Windows tabanlı sistemlerde Ns-2 yüklenmesi için VirtualBox platformu [83] windows işletim sisteminin bulunduğu sisteme yüklenip Lünix işletim sistemi bu platformun içine yüklenir. Daha sonra yukarıdaki Lünix işletim sistemi içinde yapılan işlemler sırayla gerçekleştirilir.

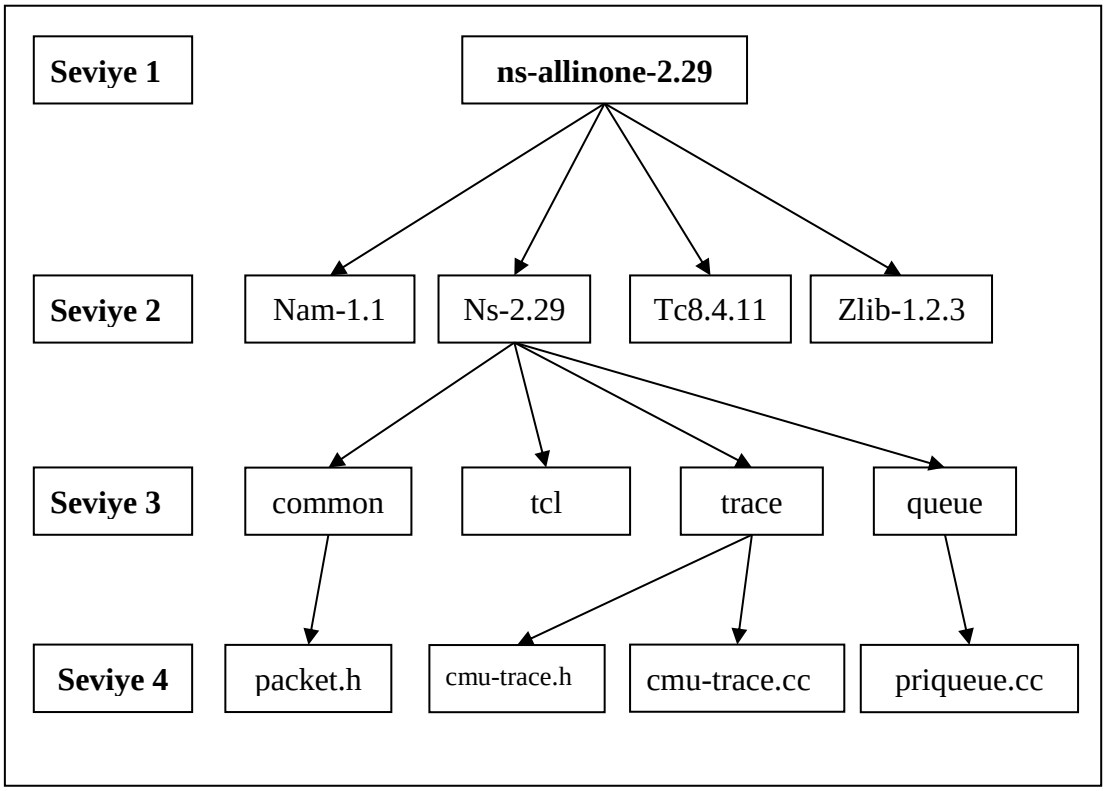
VirtualBox platformundan başka Cygwin [84] platformu da araştırmacılar tarafından sıklıkla kullanılmaktadır. Cygwin Ns-2 simülatörünü çalıştırmak için gerekli tüm paketleri yüklemez. Bunda dolayı Tablo 4.2'deki paketlerin Ns-2 simülatörüne eklenmesi gerekmektedir.

Tablo 4.2 Ns-2 simülatörünü çalıştırmak için gereken ek cygwin paketleri [82].

Kategori	Paketler
Development	gcc, gcc-objc, gcc-g++, make
Utils	patch
X11	xorg-x11-base, xorg-x11-devel

4.4.3. Ns-2 simülatörünün dizinleri

Şekil 4.3'te Ns-2 ns-allinone-2.29 versiyonunun dizin yapısın bazı sıklıkla kullanılan bazı dizinleri görünmektedir.



Şekil 4.3 Ns-2 dizin yapısı [85].

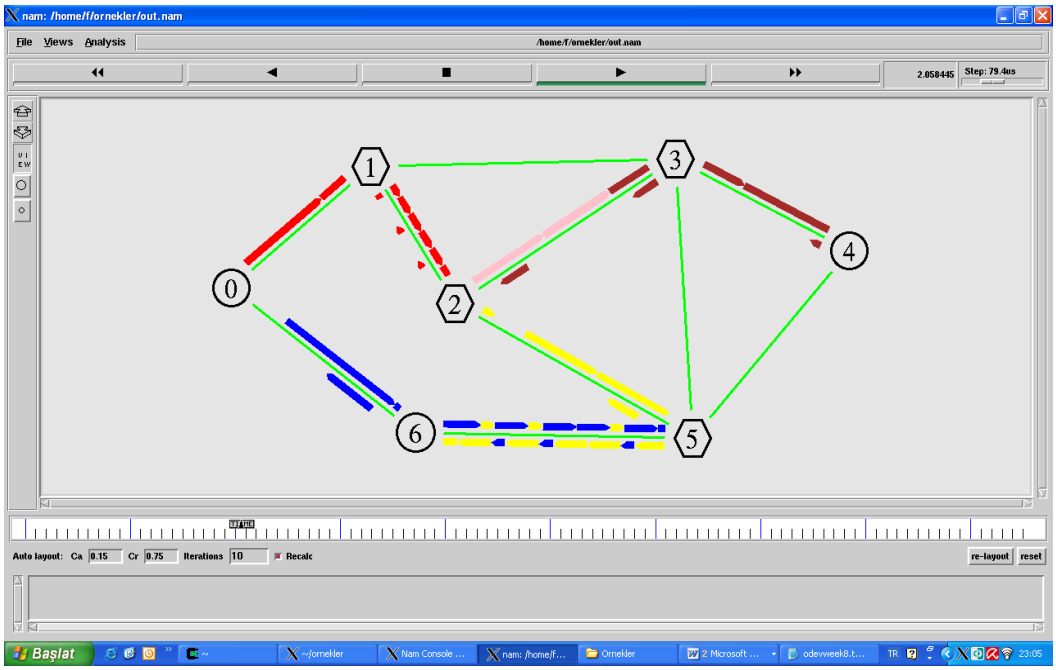
Burada ns-allinone-2.29 dizininin seviye 1 olduğunu görülmektedir. Seviye 2'de dizin tclcl-1.11 TclCL (Tcl, TclObject, TclClass) alt dizinlerini içerir. Seviye 3'de Seviye tcl, trace, queue, tcp, tools ve common gibi alt dizinler bulunmaktadır. Seviye 4'te

aşağıdaki protokol geliştirme sürecindeki güncellenmesi gereken Ns-2 dosyaları şunlardır;

1. /ns-2.29/tcl/lib/ns-lib.tcl,
2. /ns-2.29/tcl/lib/ns-packet.tcl,
3. /ns-2.29/common/packet.h,
4. /ns-2.29/trace/cmu-trace.h,
5. /ns-2.29/trace/cmu-trace.cc ve
6. /ns-2.29/queue/priqueue.cc.

4.4.4. Simülatörünün çalıştırılması

Basit olarak Ns-2 simülatörünün görselliği Şekil 4.4'de görülmektedir. Bu görsellik Ek-A'da gösterilmiş olan tcl kodları ile elde edilmiştir. Bu tcl kodu terminaldeki komut satırına "ns ilktclkodu_ns.tc" yazılarak çalıştırılmıştır.



Şekil 4.4 Simülatör ekran görünümü.

Ağda 7 adet düğüm bulunmaktadır. Her bir düğüm her bir düğüm farklı renkte oluşturulmuştur. Düğümler arasındaki bütün linkler 10MB bant genişliğinde ve TCP protokolü kullanılmıştır. Paketler 1ms sıklıkta 1 Mb boyutunda gönderişmiş ve 5 trafik tanımlanarak 10 sn sürecek bir simülasyon oluşturulmuştur. Oluşturulan trafikler sırasıyla 0-2, 2-3, 2-4, 2-6 ve 3-5, düğümleri arasındadır. Oluşturulan bu trafikler sırasıyla 0.6, 0.7, 0.8, 0.9 ve 1.0 saniyelerde başlayıp 7.5, 8.0, 8.5, 9.0, 9.5 saniyelerde sonlandırılmıştır. Simülasyon boyunca ağ ile ilgili tüm veriler "out.tr" dosyasına yazdırılmıştır.

4.4.5. Simülasyon paketlerinin incelenmesi

Paket izleme, bir simülasyon sırasında paket akışının ayrıntı bir şekilde kaydeder. Ns-2 simülasyonu sonucunda iki tür paket izleme vardır. Bunlar;

1. Metin tabanlı paket izleme ve
2. NAM paket izleme.

Metin tabanlı paket izleme sonucunda oluşturulan "out.tr" metin dosyası excel programı ile açıldığında Şekil 4.5'te olduğu gibi görünmektedir. "out.tr" dosyasındaki simgeler ve sütunların açıklamaları Tablo 4.3'de gösterilmiştir.

out.tr

	A	B	C	D	E	F	G	H	I	J	K	L
1	+	1	0	4	tcp	40	-------	0	0	6	0	0
2	-	1	0	4	tcp	40	-------	0	0	6	0	0
3	r	1,01016	0	4	tcp	40	-------	0	0	6	0	0
4	+	1,01016	4	6	tcp	40	-------	0	0	6	0	0
5	-	1,01016	4	6	tcp	40	-------	0	0	6	0	0
6	r	1,02032	4	6	tcp	40	-------	0	0	6	0	0
7	+	1,12032	6	4	ack	40	-------	0	6	0	0	1
8	-	1,12032	6	4	ack	40	-------	0	6	0	0	1
9	r	1,13048	6	4	ack	40	-------	0	6	0	0	1
10	+	1,13048	4	0	ack	40	-------	0	6	0	0	1
11	-	1,13048	4	0	ack	40	-------	0	6	0	0	1
12	r	1,14064	4	0	ack	40	-------	0	6	0	0	1
13	+	1,14064	0	4	tcp	1040	-------	0	0	6	1	2
14	-	1,14064	0	4	tcp	1040	-------	0	0	6	1	2
15	+	1,14064	0	4	tcp	1040	-------	0	0	6	2	3

Şekil 4.5 Out.tr ağ çıkış dosyası.

Tablo 4.3 Out.tr dosyasındaki simge, sütün ve açıklamaları.

Sütun	Açıklama
A	"+" Paketin kuyruğa eklendiğini
A	"-" Paketin kuyruktan silindiğini
A	"d" Paketin düştüğünü
A	"r" Paketin alındığını
B	Olayın gerçekleştiği an
C	Olayın gerçekleştiği düğümün başlangıç id'si
D	Olayın bittiği düğümün başlangıç id'si
E	Paketin tipi
F	Paketin büyüklüğü
H	Akış id'si
I	Kaynak düğümün adresi
J	Hedef düğüm adresi
K	Yönlendirme protokolünün paket sıra numarası
L	Paketin id'si

NAM(Network Animator), metin dosyasına kayıtlı simülasyon detaylarını görsel olarak kullanıcıya gösterir. NAM, tcl kodlarının içerisinde "$ns namtrace-all $file" komutu ile aktif edilir. Eğer tcl kodu içerisinde oluşturulan NAM dosyası tcl kodu içerisinde aktif edilmez ise terminalden "nam filename.nam" komutu kullanarak aktif edilebilir.

.

BÖLÜM 5. OĞUL ZEKÂSI TABANLI YENİ BİR YÖNLENDİRME ALGORİTMASI (Bee-MANET) TASARIMI ve UYGULAMASI

5.1. Giriş

Bölüm 3'te hareketli Ad-Hoc ağlar için geliştirilmiş protokoller tabloya dayalı, isteğe bağlı, hibrit, konum tabanlı ve oğul zekâ tabanlı olmak üzere beş kategoride sınıflandırılmıştı. Bee-MANET yönlendirme protokolü bal arılarının sosyal davranışı ile iletişim ağlarının ortak yönleri bir araya getirilerek geliştirilmiştir. Bee-MANET yönlendirme protokolü isteğe bağlı yönlendirme protokolü özelliği taşır. Bal arılarının doğal yiyecek arama davranışlarından esinlenildiğinden dolayı oğul zekâ tabanlı protokoller kategorisi içerisinde yer alır.

Bee-MANET, hareketli(mobile) ağlar için daha dinamik, basit, verimli, güvenilir, esnek ve ölçeklenebilir tekli yol (unicast) bir yönlendirme protokolüdür. Kaynakların daha etkin bir şekilde kullanılarak hem ağdaki iletilen kontrol paketlerini azaltmak hem de düğümlerin iletmiş olduğu veri paket miktarını (throughput) arttırmayı hedeflemektedir.

Geliştirilen Bee-MANET yönlendirme protokolü aşağıdaki özelliklere sahiptir;

1. Bee-MANET yönlendirme protokolü yönlendirme işlemini yerine getirirken yalnızca ileri (forward) ve geri (backward) öncü arılar (scout) kullanmaktadır.
2. Öncü arılar çok az bant genişliği (bandwidth) kullanmaktadır.

3. Öncü arılar yönlendirme görevini yerine getirirken başka görevler de gerçekleştirebilmelidir.
4. Bee-MANET protokolü düğümler arasındaki bağlantı kaliteleri(quality of link) için istatistiksel veri kullanmamaktadır.
5. Bir düğümün öncü arı ile ilgili işlemleri çok kısa olmaktadır.
6. Bee-MANET ağ kaynaklarını verimli kullanmaktadır.
7. Bee-MANET iletilen veri paketi oranı yüksek olmaktadır.
8. Düğümün yönlendirme tablosunun (routing table) boyutu az olmaktadır.
9. Bee-MANET büyük ölçekli topolojiler için ölçeklenebilir olmaktadır.
10. Literatürdeki oğul zekâ tabanlı ve diğer protokollere göre ağ performansı benzer/daha iyi olmaktadır.

5.2. Oğul Zekâsı Yönteminin İletişim Ağları İle Ortak Özellikleri

Bu bölümde oğul zekâ tabanlı mühendislik çalışmaları için doğada yaşayan bal arılarının yiyecek ararken göstermiş oldukları davranışlarının iletişim ağlarında yön bulma işlemi ile olan ortak yönleri incelenmiştir:

1. Ağdaki her bir düğüm bal arıları kolonisindeki bir kovana karşılıklı düşer.
2. Bal arıları kolonisindeki her bir kovana ait yiyecek arama görevini yerine getiren ***öncü arılar***(scout) iletişim ağlarındaki kontrol paketlerine karşılık gelmektedir. Her bir düğüme ait öncü arılar yiyecek ihtiyacı duyulduğunda nektar alanına yiyecek aramak için gönderilir. Gönderilen öncü arılar nektar arama alanındaki bilgileri elde ederek düğümle ilgili olarak yönlendirme bilgisi elde edilmiş olur. Böylece düğüm hedefle ilgili yolları değerlendirerek en uygun yolu seçer.
3. Öncü arılar ziyaret ettikleri düğümlerle ilgili olarak mesafe ve zaman gecikme bilgilerini elde ederler. Böylece düğümler arası mesafe ve yayılım gecikmeleri tespit edilmiş olur. Bu bilgiler öncü arıların kovana geri döndüğünde kovanda yapmış oldukları danslar ile işçi arılara aktarılır. Kovandaki işçi arı kavramı iletişim ağlarındaki veri paketlerine karşılık

gelmektedir. Veri paketleri ağdaki en az gecikmeye sahip olan en yeni yollardan seçilir.

4. Öncü arıların yalnızca belirli bir adım sayısına kadar yol bulmasına müsaade edilir. Belli bir adımdan sonra yol bulma işlemi başarısız olarak kabul edilir.
5. Öncü arılar yalnızca geçmiş oldukları yolarla ilgili bilgileri değil diğer düğümlerden gelen bilgileri de bir sonraki düğüme taşır. Bu bilgileri taşıyan öncü arılara ***toplayıcı öncü arı*** adı verilmiştir. Toplayıcı öncü arı kullanılmasındaki temel hedef ağda dolaşan öncü arı sayısını azaltarak ağ kaynaklarını daha verimli kullanmaktır. Bu durum aynı zamanda iletilen veri paketi sayısını önemli sayıda artmasını sağlamıştır.

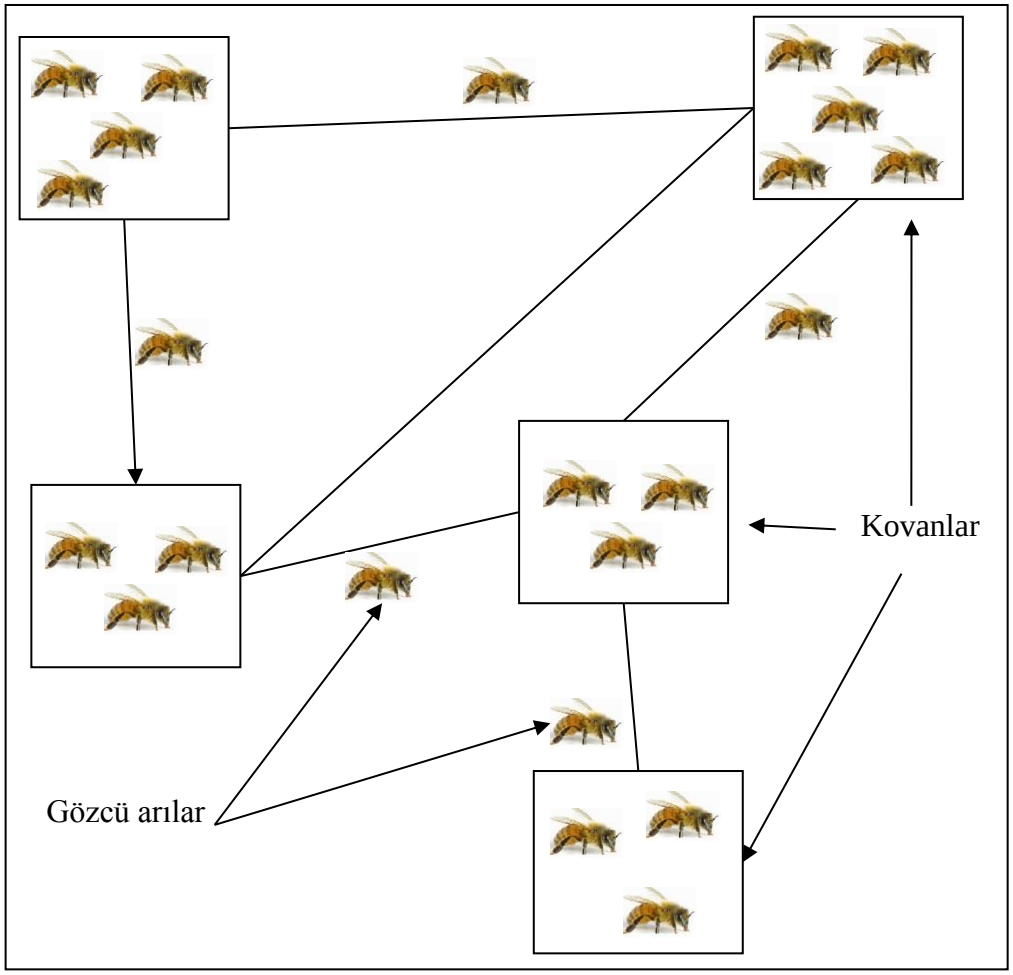
Bee-MANET algoritması, bal arılarının sosyal davranışları ile ağ sistemleri arasındaki ortak özellikler dikkate alınarak geliştirilmiştir. Bu benzerlikler Tablo 5.1'de gösterilmektedir. Oluşturulan algoritmada, ağdaki her bir düğüm bir arı kovanına sahiptir. Şekil 5.1'de görüldüğü gibi ağ bütün arılar için bir nektar arama bölgesi olarak kabul edilmiştir.

Tablo 5.1 Bal arıları ve bilgisayar ağ sistemleri arasındaki benzerlikler.

Bal arıları	Bilgisayar ağ sistemi
Kovanlar	Düğümler
Nektar	Ağ kaynakları
Nektar alanı	Ağ
Gözcü ve Toplayıcı arılar (scout)	Kontrol paketleri
İşçi arılar (foragers)	Veri paketleri
Sallanma dansları	Yönlendirme tabloları

Bal arıları nektar toplarken mümkün olduğu kadar fazla nektar toplamak isterler. Nektar kovanlar arasında belirli yol (link) boyunca toplanır. Toplanan nektar sayısı kaynak düğüm ile hedef düğüm arasındaki adım (hop) sayısıyla ters orantılıdır. Bir başka değişle bir arı daha kısa mesafelerden daha fazla nektar toplayabilir. Ağ çalışırken, gözcü arılar ağda düğümleri dolaşarak kaynak ile hedef düğüm arasında en

kısa yolları araştırırlar. Daha sonra kovandaki işçi arılar keşfedilen yolları kullanarak veri paketlerini bu yollar üzerinden hedef düğümlere iletirler. Ağdaki veri paketlerinin kaynak düğümden en kısa sürede ve verimli bir şekilde hedef düğümlere taşınması işlemi gözcü arıların işçi arıları en iyi şekilde yönlendirmesi ile olur.

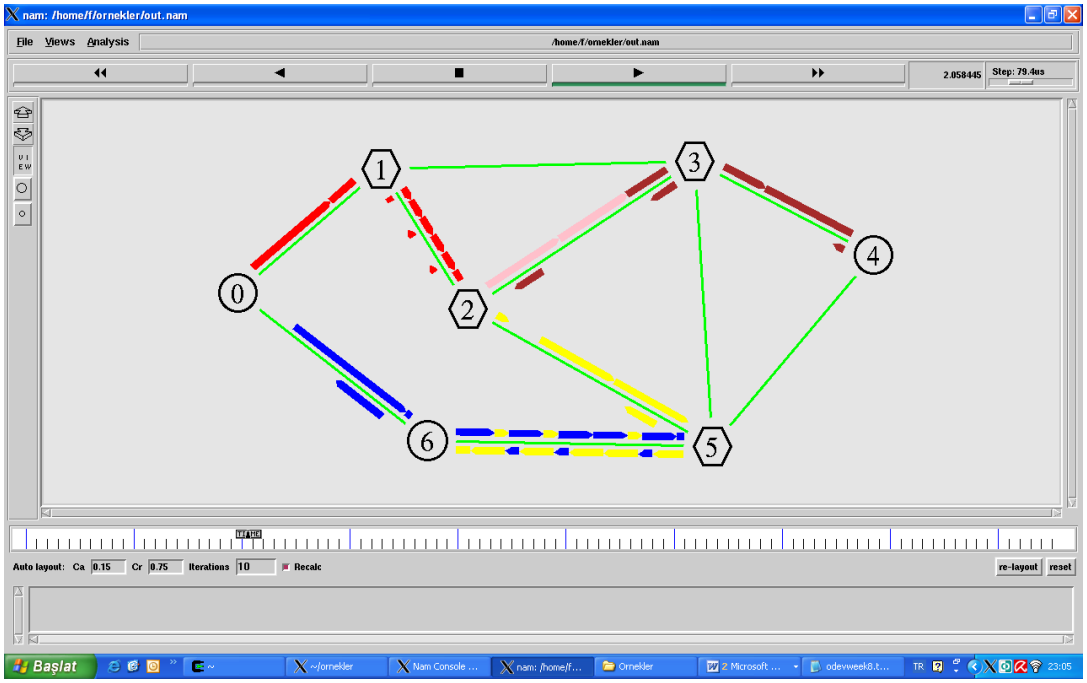


Şekil 5.1 Ağ yapısı.

Bal arılarının birbirleri ile iletişimde bir takım danslar yaparak nektar alanlarının nerelerde olduğu bilgisini birbirlerine iletirler. Dansların ağ sistemindeki karşılığını ise yönlendirme tabloları almaktadır. Veri paketleri yani işçi arılar oluşturulan bu yönlendirme tablolarını kullanarak paketleri kaynak düğümden hedef düğüme taşırlar.

Ağı oluşturan düğümler ağ kurulduğunda birbirleri ile ilgili konum ve uzaklık bilgisine sahip değildir. Bir düğümden diğer bir düğüme paket gönderilmek

istendiğinde kaynak düğümden ağa kontrol paketleri gönderilmeye (broadcast) başlanır (Şekil 5.2). Hedef düğüme ulaşan kontrol paketleri geri kaynak düğüme gelirler. Gelen bu paketlere sıra numaraları verilerek en yeni sıra numarasına sahip olan yol veri paketi gönderilmek için seçilir.



Şekil 5.2 Gözcü (scout) arılarının ağda dolaşmaları.

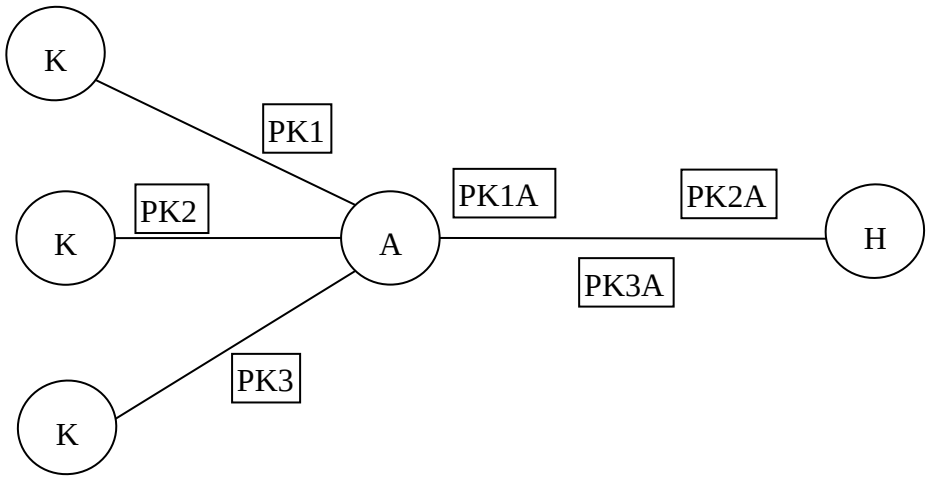
Geliştirilen algoritma bazı kurallara sahiptir. Bu kurallar aşağıdaki gibi sıralanabilir:

1. Bir paketin kaynak düğümden hedef düğüme gönderilmesi istenildiğinde, kaynak düğüm ileri öncü arı üretir ve komşu düğümler aracılığıyla ağdaki düğümler hakkında bilgi toplarlar.
2. Gözcü arı hedefe ulaştığında geçmiş olduğu düğümlerle ilgili bilgileri geldiği yoldan kaynak düğüme geri döner.
3. Ağa gönderilen gözcüler geri döndükten sonra her bir gözcünün kat ettiği yollar dikkate alınarak en yeni yol seçilir.
4. Gözcüler bir düğümü iki defa ziyaret edemez.
5. Her bir gözcünün bir yaşam süresi (Time to live-TTL) vardır. Bu süreyi aşarsa ağdan düşürülür.

5.3. Toplayıcı Arılar

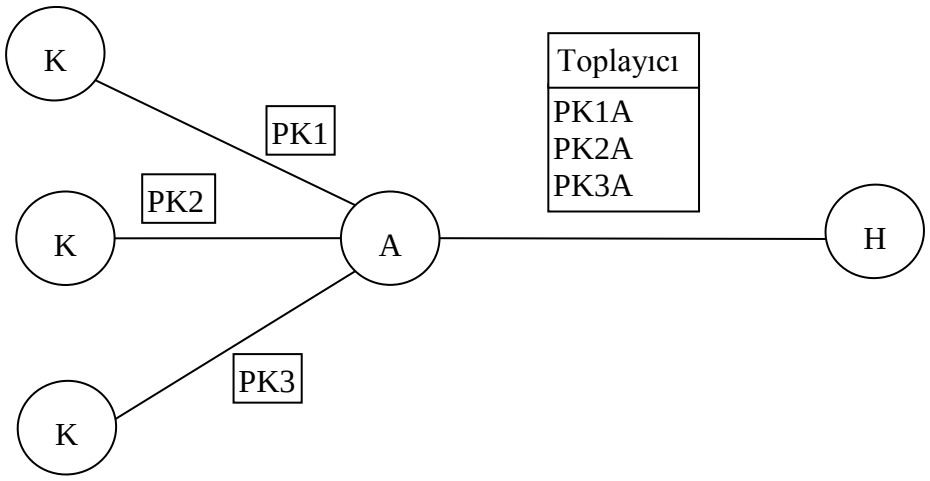
Toplayıcı mantığı ağda dolaşan kontrol paket sayısını ve paket iletim sürelerini azaltıp kaynakların etkili bir şekilde kullanılmasını hedeflemiştir. Kablosuz ad hoc ağlarda kaynak düğüm ile hedef düğüm arasında birçok yol bulunabilir. Ayrıca aynı ağ üzerinde bir düğüm birçok yol için geçiş yolu olabilir. Örnek olarak, beş düğümlü bir ağda üç kaynak düğümden (K1, K2 ve K3) bir hedef düğüme (H1) paket gönderilmek istendiğinde yönlendirme sistemi kaynaklar ile hedefler arasında yol bulma işlemine başlayacaktır.

Şekil 5.3'te toplayıcı mantığı kullanılmadan önceki kontrol paketleri görülmektedir. Burada K1, K2 ve K2 düğümlerinden H1 düğümüne veri paketi gönderilmek istendiğinde K1, K2 ve K3 düğümleri H1 düğümü için en uygun yolun bulunabilmesi için kontrol paketi gönderir. Burada A düğümüne gelen paketler A düğümünden H1 düğümüne tek tek gönderilirken A düğümünün veri iletim katmanı PK1A, PK2A ve PK3A paketlerini ayrı ayrı kullanması gerekmektedir. Böylelikle A düğümü ile H1 düğümü arasındaki bağlantı da yoğunluk olacağı, paketlerde hedefe ulaşırken gecikme meydana geleceği ve A düğümünün yoğunluğundan dolayı da bu düğümdeki paket kayıplarını artacaktır.



Şekil 5.3 Toplayıcı kullanılmadığında ağda dolaşan kontrol paketleri.

Toplayıcı mantığı kullanıldığında kullanılan ağın görünümü Şekil 5.4'te görülmektedir. Kontrol paketleri bir sonraki düğüme toplayıcı ile bir arada gönderildiğinde, A düğümünde birim zamanda veri iletim katmanına gönderileni kontrol paket sayısı azalır. Burada A düğümü K1, K2 ve K3 düğümünden kendisine gelen PK1, PK2 ve PK3 kontrol paketlerini toplayıcı listesine ekler ve 0.2 sn gibi kısa bir süre bekledikten sonra bu listeyi H1 düğümüne gönderir.



Şekil 5.4 Toplayıcı kullanıldığında ağda dolaşan kontrol paketleri.

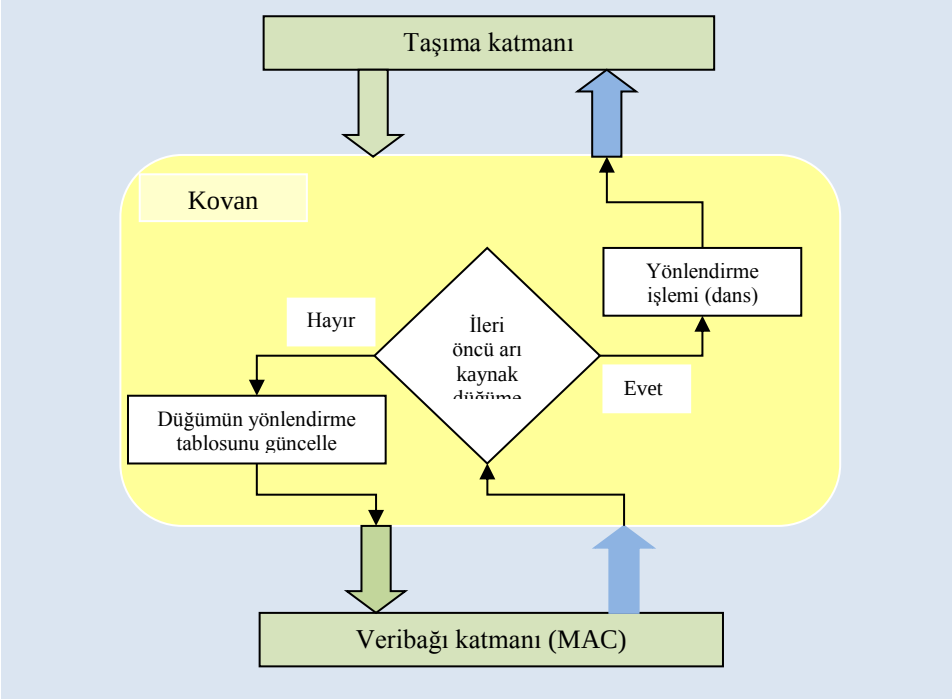
5.4. Bee-MANET Yönlendirme Protokolü

Bu bölümde, Ad-Hoc ağlar için oğul zekâsı tabanlı yeni bir yönlendirme protokolünün çalışması, kovan mimarisi, öncü arı kontrol paket yapıları ve algoritmaları, Temel Fonksiyonlar, protokolü sınıf ve nesneleri anlatılacaktır.

5.4.1. Bee-MANET yönlendirme protokolünün mimarisi

Bee-MANET, paket yönlendirmesi için kendi yönlendirme katmanına sahiptir. Düğümler olarak düşünülen kovanlar içerisinde vebibağı katmanından gelen öncü arılar (yönlendirme paketleri) yönlendirme bilgilerini danslar yardımıyla işçi arılarına (forager) aktarır (Şekil 5.5). işçi arıları yönlendirme bilgilerini kullanarak veri

paketlerini taşıma katmanına gönderir. Tablo 5.2'de kovana gelen Bee_MANET paketlerinin, kovana alınışının algoritması gösterilmiştir.



Şekil 5.5 Bee-MANET mimarisi.

Tablo 5.2 Kovanın öncü arı alımı.

```
KovanaPaketGelişi ()
{
   eğer paket bee-manet paketi ise
     {
         KovanaPaketiAl ()
     }
}
KovanaPaketiAl ()
{
    seçim (öncü arı)
     {
        durum  ileriÖncüArı:  ileriÖncüArıAlımı() ; durdur;
        durum  geriÖncüArı:  geriÖncüArıAlımı() ; durdur;
        durum  toplayıcı:  toplayıcıArıalımı() ; durdur;
        varsayılan: geçersiz Bee-Manet paketi; çıkış;
     }
}
```

5.4.2. Bee-MANET paket yapıları

Geliştirilen yönlendirme protokolünde üç tip öncü arı(*scout*) mevcuttur. Bunlar;

1. İleri öncü arı,
2. Geri öncü arı ve
3. Toplayıcı öncü arı.

Tablo 5.3, Tablo5.4 ve Tablo 5.5'te sırasıyla ileri öncü arı, geri öncü arı ve toplayıcı arı paket başlıkları ve açıklamaları görülmektedir.

Tablo 5.3 İleri öncü arı paket başlığı.

p_type	Paket tipi
hop_number	Adım sayısı
bcast_id	Yayılım ID
d_ip	Hedef IP Adresi
ss_no	Kaynak sıra numarası
ds_no	Hedef sıra numarası
s_ip	Kaynak IP adresi
delay	Yol gecikmesi

Tablo 5.4 Geri öncü arı paket başlığı.

p_type	Paket tipi
hop_number	Adım sayısı
d_ip	Hedef IP Adresi
ds_no	Hedef sıra numarası
s_ip	Kaynak IP adresi
delay	Yol gecikmesi

İleri öncü arı, ağdaki her hangi bir düğümden bir başka düğüme paket gönderilmek istendiğinde ve geçerli bir yola sahip olmadığı zaman yol kurmak üzere kaynak

düğüm, ileri öncü arı oluşturarak komşu düğümlere bu istek hakkında bir paket gönderir (broadcast). Komşu düğüm böyle bir paket aldığında (Şekil 5.6) öncelikli olarak paketin daha önce kendisine ulaşıp ulaşmadığını kontrol eder. Eğer böyle bir paket daha önce kendisine ulaştıysa bu paketi ağdan atar. Bu iletişim sırasında, ara düğümler aldıkları ileri öncü arı ile kendi yönlendirme tablolarında güncellerler.

Tablo 5.5 Toplayıcı arı paket başlığı.

p_type	Paket tipi
bcast_id	Yayılım ID
f_scout_list	Adım sayısı
delay	Yol gecikmesi

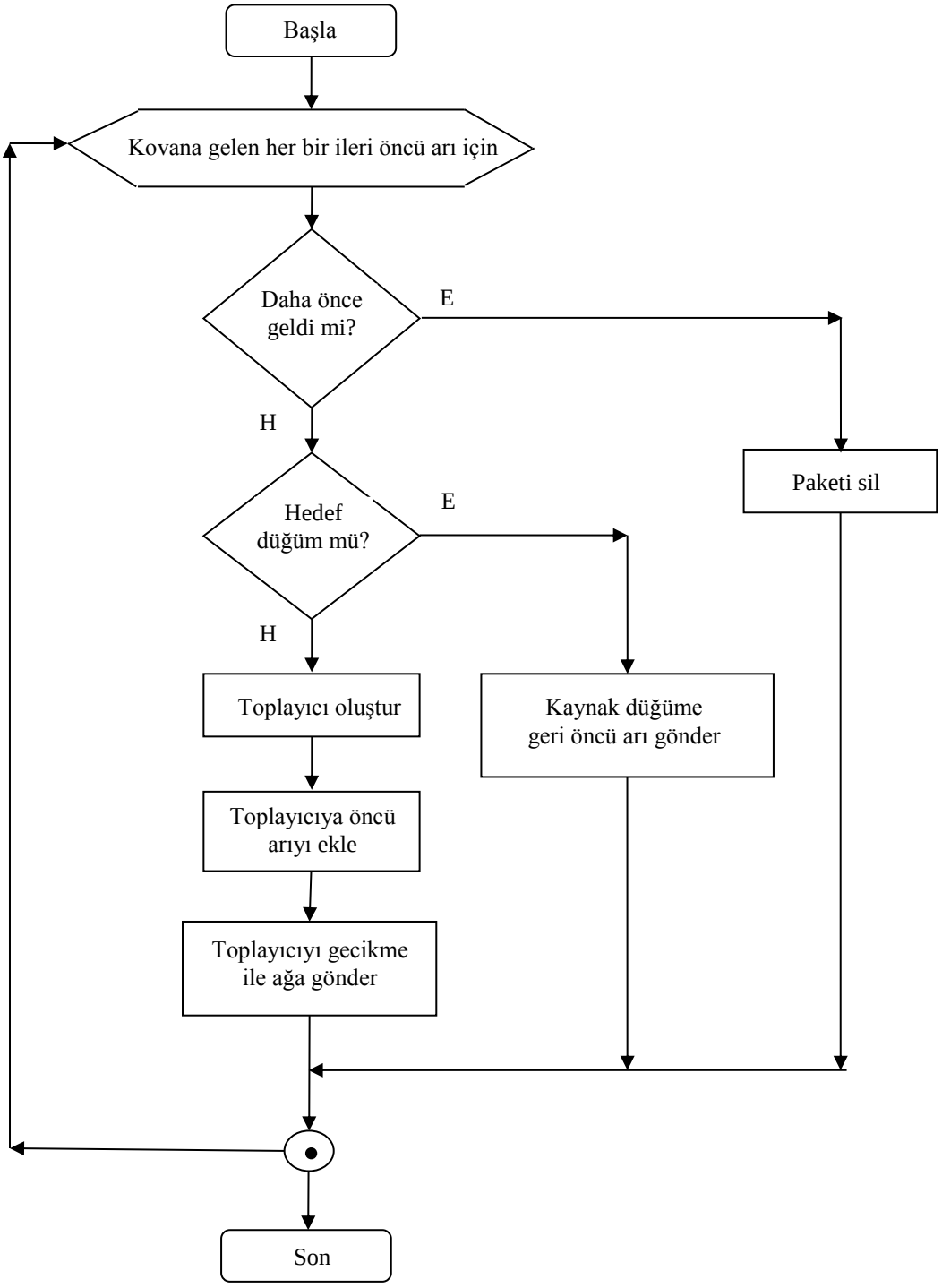
İleri öncü arı için yaşam süresi (ttl) 10 sn olarak belirlenmiştir. Eğer paketler hedef düğümlere bu süre içerisinde ulaşamamışsa paket silinerek ağdan atılır ve hedef için tekrar ileri öncü arı gönderilir. Hedef için ileri öncü arı gönderim tekrar sayısı 3 olarak belirlenmiştir.

Tablo 5.6 Kovanın geri öncü arı alımı algoritması.

```
kovanaÖncüArıAlımı ()
{
  eğer düğümdeki yönlendirme bilgisi daha yeni  || (eşit &&daha kısa yol)
    {
      yönlendirmeTablosunuGüncelle ()
        eğer hedef düğüme geldiysem
          {
            Kuyrukta bekliyen paketleri gönder
              değilse eğer bu düğüme daha önce geldiysem
                  geri öncü arıyı sil
                değilse
                    paketi hedef sıradaki bir sonraki düğüme gönder
          }
    }
}
```

Düğüme gelen bu paket eğer hedef düğüme ulaştı ise geldiği yoldan geri dönecek şekilde kaynak düğüme ***geri öncü arı*** oluşturularak gönderilir (unicast). Kaynak

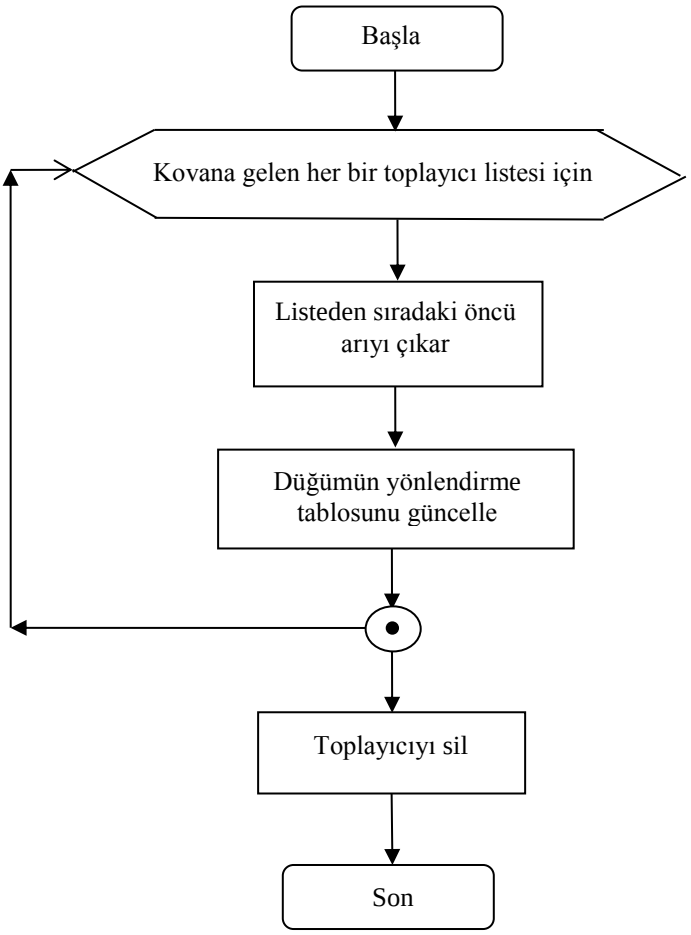
düğüme gelen geri öncü arı veri paketinin hedef düğüme gönderilmesi için yönlendirme işlemini yerine getirir (Tablo 5.6).



Şekil 5.6 Kovanın öncü arı alımı.

Eğer düğüm hedef düğüm değilse bulunulan düğümde ***toplayıcı öncü arı*** oluşturulup gelen ileri öncü arı bu toplayıcıdakine eklenir. Toplayıcı, bu düğümde kısa bir süre bekletilerek aynı düğüme başka ileri öncü arılarının gelmesi beklenir. Eğer başka ileri öncü arı gelirse aynı toplayıcıya bu öncü arı da eklenir. Belli bir süre sonra toplayıcı düğümün komşularına gönderilir.

Toplayıcı öncü arı, ağda dolaşan ileri öncü arı sayısını azaltıp kaynakların daha verimli kullanılmasını hedeflemektedir. Burada bir düğüme gelen öncü arı öncelikli olarak oluşturulan toplayıcıda kısa bir süre (0.2 ms) bekletilip başka düğümlerden farklı hedefler için öncü arıların gelmesi beklenir. Süre dolduğunda toplayıcı arı içerisindeki ileri öncü arı listesi ile ağa gönderilir.

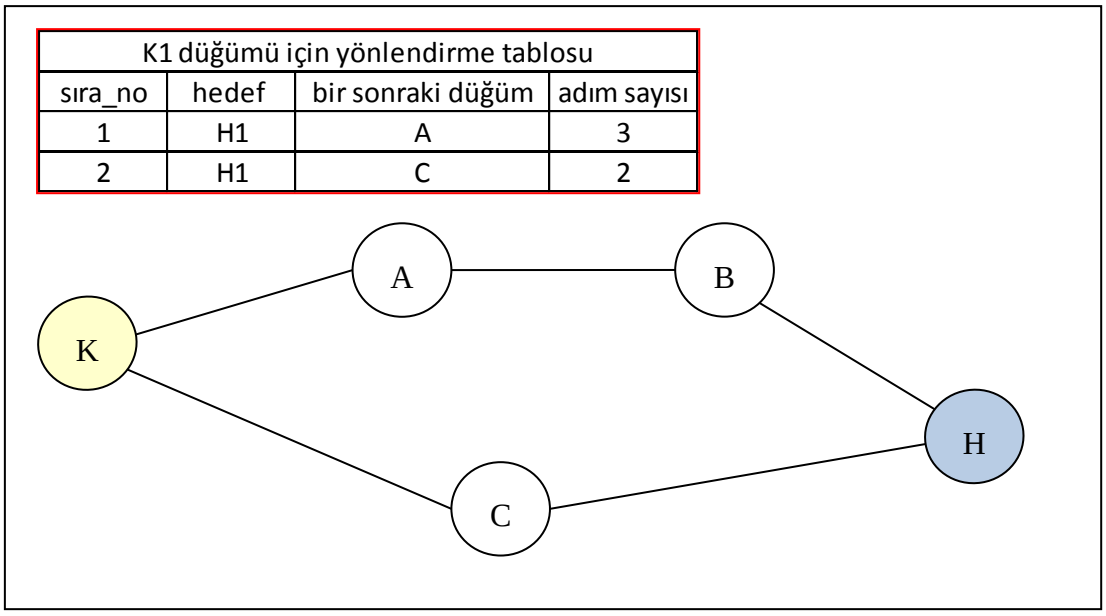


Şekil 5.7 Kovan toplayıcı arı alımı.

Kovana toplayıcı öncü arı geldiğinde öncelikli olarak toplayıcının öncü arı listesine ulaşılır. Buradaki öncü arılar teker teker çıkartılarak kovanda öncü arıların yapması gereken yönlendirme işlemlerinin yerine getirilmesi sağlanır. Toplayıcıda bulunan öncü arı listesindeki öncü arıların tamamı çıkarıldıktan sonra toplayıcı görevini yerine getirmiş olur ve kovanda yer kaplamaması için kovandan silinir (Şekil 5.7).

5.4.3. Yönlendirme tabloları

Bir ağdaki hedefe ait yol bilgilerini yönlendirme tabloları tutar. Yönlendirme tabloları veri paketleri gönderilmek istediği zaman oluşturulur. Geliştirilen modeldeki yönlendirme tabloları, hedef düğüm için kullanılan yollar tutulur. Şekil 5.8'de temsili olarak bir ağ oluşturulmuştur.

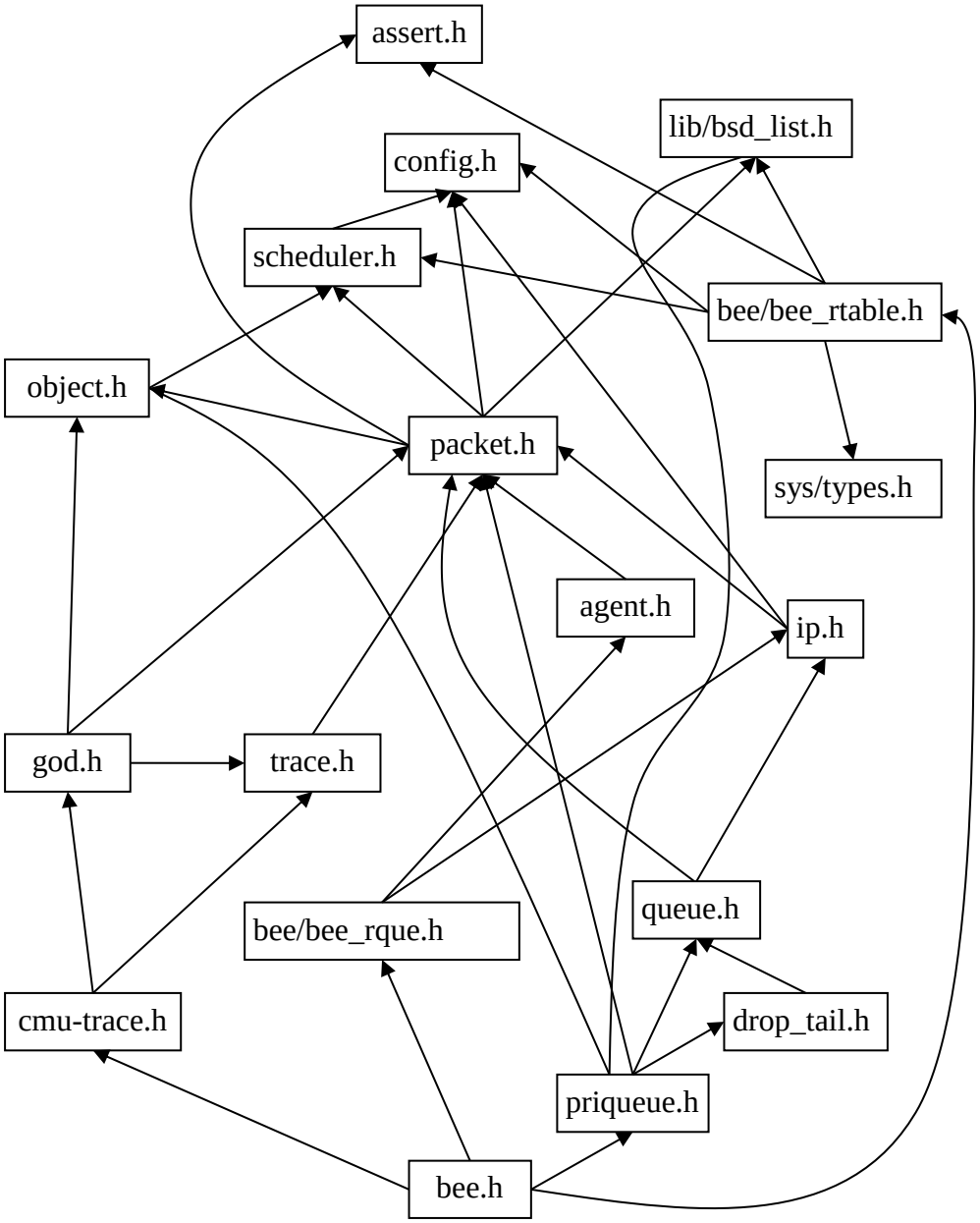


Şekil 5.8 Yönlendirme tablosu örneği.

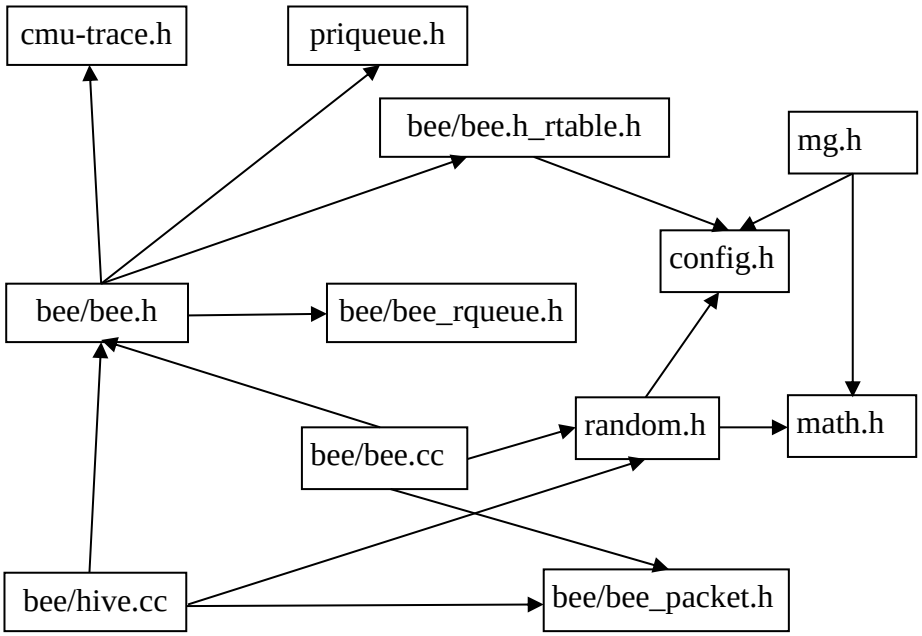
K1 ile kaynak düğüm ve H1 ile de hedef düğüm gösterilmiştir. Şekilde, K1'den H1'e gidecek olan veri paketleri için oluşturulan yönlendirme tablosu gösterilmektedir. Yönlendirme tablosunda hedef sıra no, hedef düğüm, hedefe ulaşmak için sonraki düğüm ve yolun adım sayısı tutulur. Burada Bee-MANET protokolü 2 numaralı hedef sıra numaralı yolu kullanarak veri paketlerini H1 düğümüne iletir.

5.4.4. Bee-MANET yönlendirme protokolü sınıf ve nesneleri

Şekil 5.9 ve Şekil 5.10'da Bee-MANET yönlendirme protokolünün agent ve agent.h sınıfından çıkarılmış olan dosya bağımlılıklarını gösterilmektedir.



Şekil 5.9 bee.h dosya hiyerarşisi.



Şekil 5.10 bee.cc dosya hiyerarşisi.

Geliştirilen yönlendirme protokolünün Ns-2 uygulama modülleri şunlardır:

1. bee.h, değişkenlerin, sabitlerin ve protokolün kullanmış olduğu paket alım (incomingToHive, incomingAccumulator, incomingFscout, incomingBscout) ve gönderim (forward, outgoingAccumulator, outgoingFscout, outgoingBscout) fonksiyon tanımlamaları bulunur. Sabitlerin çoğu kolaylık sağlamak için değişkenler olarak TCL hiyerarşisine uygun olarak oluşturulmuştur.
2. bee_packet.h, ileri öncü arı, geri öncü arı ve toplayıcı arı tanımlamaları ve bu arıların başlık dosyaların bulunduğu dosyadır. Ns-2 içerisindeki Bee-MANET paket başlığı olan bağlantılar kaynak yönlendirme ve IP seçenekleri olan yapılardır.
3. bee.cc, protokolün merkezi durumundadır. Bee-MANET protokolü ile ilgili olarak tüm simülasyon ayrıntılarının bulunduğu dosyadır.
4. hive.cc, düğüme gelen bütün paketlerin alınması, işlenmesi ve gönderilmesi ile ilgili olarak yapılacak olan işlemlerin bulunduğu dosyadır.
5. bee_rtable.h, yönlendirme ile ilgili olarak değişkenlerin ve fonksiyonların tanımlamaların yapılmış olduğu dosyadır.

6. bee_rtable.cc, yönlendirme işlemleri ile ilgili fonksiyonların yazılmış olduğu dosyadır.
7. bee.tcl, Bee-MANET protokolünün TCL değişkenlerini varsayılan değer olarak oluşturur.

5.4.5. Bee-MANET protokolündeki temel fonksiyonlar

Bee-MANET protokolündeki temel fonksiyonlar hive.cc dosyasında bulunmaktadır. Bu fonksiyonları iki gruba ayırabiliriz. Bunlar;

1. Paket alım fonksiyonları ve
2. Paket gönderim fonksiyonları.

5.4.5.1. Paket alımı fonksiyonları

Paket alım fonksiyonları, ağdaki her hangi bir düğüme bir paket geldiğinde çalışan fonksiyonlardır. Bunlar;

1. incomingToHive( ): Bu fonksiyon gelen Bee-MANET paketlerinin hangi paketleri olduğunu tespit eder. Eğer kovana gelen paketin türü sırasıyla fscout, bscout ve Accumulator ise sırasıyla incommingFscout, incomingBscout ve incomingAccumulator fonksiyonları çağrılacaktır.
2. incomingAccumulator( ): Eğer düğüme paket tipi "Accumulator" olan bir paket gelirse bu fonksiyon çağrılır.
3. incomingFscout( ): Eğer düğüme paket tipi "fscout" olan bir paket gelirse bu fonksiyon çağrılır.
4. incomingBscout( ): Eğer düğüme paket tipi "bscout" olan bir paket gelirse bu fonksiyon çağrılır.

5.4.5.2. Paket gönderim fonksiyonları

Paket gönderim fonksiyonları, ağdaki her hangi bir düğümden paket gönderilmek istenildiğinde çalışan fonksiyonlardır. Bunlar;

1. forward( ): Bu fonksiyon düğüme gelen paket eğer o düğümün değilse hedef yoldaki bir sonraki düğüme iletilmek için kullanılır.
2. outgoingAccumulator( ): Bu fonksiyon *"fscout"* kontrol paketlerini bir sonraki düğüme iletmek için kullanılır.
3. outgoingFscout( ): Bu fonksiyon *"fscout"* paketi göndermek için kullanılır.
4. outgoingBscout( ): Bu fonksiyon *"bscout"* göndermek için kullanılır.

BÖLÜM 6. Bee-MANET BAŞARIM DENEYLERİ VE SONUÇLARI

6.1. Giriş

Bu bölümde Bee-MANET yönlendirme protokolünün simülasyon deneyleri ve sonuçların değerlendirilmesinden bahsedilmiştir. Bee-MANET protokolü, 200 düğüm ve 20 m/s hıza kadar hareketli ad hoc ağlar altyapısına uygun bir şekilde tasarlanmıştır. Geliştirilen protokol ileri yöndeki kontrol paketlerinin düğümlerde toplanarak bir araya getirilip komşu düğümlere gönderilme yöntemi geliştirip uygulandığından ağdaki dolaşan toplam paket sayısının düşmesine sebep olmuştur. Bu durum kaynakların daha verimli kullanılması, iletim oranının artması ve iletilen veri paketlerinin sayısını arttırmıştır.

Yapılan çalışmada Bee-MANET protokolünün yanı sıra hareketli ad hoc ağlarda kullanılan AODV ve BeeAdhoc protokollerinin de değişik düğüm ve hızlarda performans analizleri test edilmiş ve karşılaştırılmıştır.

6.2. Performans Kriterleri

Geliştirilen protokol AODV ve BeeAdhoc algoritmaları ile aşağıdaki belirtilen ölçütlere göre değerlendirilip birbirleri ile karşılaştırılmıştır.

1. Uçtan uca ortalama gecikme (end to end delay): Kaynak düğümden çıkan bir veri paketinin hedef düğüme ulaşıncaya kadarki geçen süre. Uçtan uca gecikmenin iyi olarak değerlendirilebilesi için olabildiğince düşük olması gerekir.

2. Paket iletim oranı(delivery ratio): Kaynak düğümden gönderilen paketlerin hedef düğüme ulaşma oranıdır. Bu oran ağın performansı için yüksek olması gerekmektedir.
3. Kontrol paket sayısı(number of control packets): Kaynak düğüm ile hedef düğüm arasındaki yolların bulunması amacıyla ağa gönderilen paket sayısıdır. Kontrol paket sayısının az olması beklenmektedir. Kontrol paket sayısının fazla olması demek ağ kaynaklarının daha fazla kullanılması ve meşgul edilmesi demektir. Bu durum iletilen veri paketi sayısını düşürmektedir.
4. Ağ çıkışı(throughput): Hareketli ad hoc ağlardaki en önemli performans kriteridir. Ağdaki herhangi bir düğümün birim zamanda (sn) almış olduğu veri paketlerinin bit cinsinden toplamının zamana oranı ile ifade edilir. Bu oranın yüksek olması ağ performansının iyi olduğu anlamına gelmektedir.
5. Toplam enerji tüketimi (total energy consumed): Hedefe 1 kbyte veri taşımak için kontrol paketlerinin harcamış olduğu toplam enerji olarak tanımlanır. Daha az adım sayısı ile taşınan veri daha az enerji harcayacaktır. Toplam enerji tüketimi değerinin hareketli Ad-Hoc ağlar için daha az olması tercih edilmektedir.

Aşağıdaki kriterler ölçülemeyen kriterler olup ağ performansını olumlu / olumsuz etkilemektedir [39, 86].

1. Ölçeklenebilirlik(Scalability): Oluşturulan protokolün ağ performansı, ağdaki toplam düğüm sayısına ve düğümlerin hareket hızlarının artması veya azalmasından etkilenmemelidir.
2. En uygun yol bulunması: Protokol, seçilen ölçütlere göre en uygun yolu bulabilmelidir. Bu ölçüler, adım sayısı, düğümler arası gecikme, düğümler arası bant genişliği, düğüm yoğunluğu gibi sıralanabilir.
3. Bant genişliği: Kablosuz iletişim ağlarında her düğüm kısıtlı bant genişliğine sahiptir. Bu yüzden protokol mümkün olduğunca en az adım sayısı ile trafik üretmeyi hedeflemelidir. Bu da periyodik güncellemelerin azaltılması ve kontrol paket sayılarının azaltılmasıyla elde edilir.

4. Yakınsama: Ağ topolojisinde her hangi bir değişiklik meydana geldiğinde, kullanılan yönlendirme protokolü mümkün olan en kısa zamanda yeni yollar bulabilmelidir.
5. Yolun geçerliliğini yitirmesi ve düzeltilmesi: Kaynak ile hedef arasındaki kullanılan mevcut yolların bozulması durumunda bu yolların düzeltilebilmesi için gerekli olan mekanizmalar geliştirilmelidir.
6. Yük dengeleme (load balancing): Protokol ağdaki bir düğüme aşırı yüklenmemeli ve ağdaki tüm düğümlere ağ yükünü eşit bir şekilde dağıtmalıdır. Bu durum, bazı düğümlerde meydana gelmesi muhtemel paket çakışmalarının azaltılmasına yardımcı olur. Paket kaybının azalması ağ çıkışını da arttırmaktadır.
7. Döngü oluşmama (loop free): Ağdaki bazı paketler hedeflerine ulaşamayıp ağ içerisinde hareket etmeye devam ettiklerinde döngü meydana gelir. Yönlendirme protokolü, döngü oluşmalarını engelleyebilmelidir. Ağda böyle bir paket dolaştığı zaman bant genişliğini büyük oranda kullanırlar. Döngü halindeki paketler hiç bir zaman hedeflerine ulaşamamaktadır.
8. Uyuma özelliği (sleep function): Kablosuz ad hoc ağlarda düğümlerin enerjileri batarya tarafından sağlandığından bazı düğümler periyodik olarak uyku modunda (pasif moda) tutulabilir. Protokol böyle bir durumda bulunan düğümlerden ağdaki diğer düğümlerin etkilenmesini önleyebilmelidir.
9. Düğümlerin hareketlilik hızı (mobility): Protokol düğüm hareketlerinin tüm hızlarda(yürüyüş, şehir içi araç hızı ve şehirlerarası araç hızı) ağ trafiğini devam ettirebilmelidir.
10. Çoklu yol: Yönlendirme protokolü, kaynak ile hedef arasında kullanılan yoldaki bir düğümdeki çarpışmaları azaltmak için kaynak ile hedef arasında birden fazla yol tahsis edebilir. Çoklu yollar ad hoc ağlar için çok kullanışlıdır. Çünkü ad hoc ağlarda düğüm hareketleri hızlı ve topoloji değişikliği sürekli olduğu için birçok link hatası meydana gelmektedir. Eğer alternatif yollar kullanıldığında yollar gecikmeyi azaltıp paket iletim oranını yükseltir.

6.3. Simülasyon Çerçevesi

Bee-MANET, AODV ve BeeAdhoc yönlendirme protokollerinin birbiri ile karşılaştırabilmesi için simülasyon çerçevesinin ve test senaryolarının tüm protokoller için aynı olması gerekmektedir. Yapılan çalışmada simülatör çerçevesi olarak Ns-2 ağ simülatörü seçilmiştir. Fakat bu protokolleri test ederken aynı simülasyon çerçevesi kullanılarak birçok farklı sonuçlar elde edilebilir. Bundan dolayı her bir protokol için aynı simülasyon çerçevesi içinde aynı senaryo ve trafik dosyaları kullanılmalıdır. Yapılan çalışmada her bir farklı düğüm sayıları için farklı senaryolar ve bu senaryolarda farklı düğüm hızları oluşturulmuştur. Tablo 6.1'de görüldüğü üzere tüm senaryolar için simülasyon süreleri 50 saniye, trafik modeli FTP/TCP ve veri paketi boyutu 512 bayt olarak alınmıştır. Trafik üreteci olarak Ns-2 simülatörü içerisinde olan cbrgen ve topoloji üreteci olarak da setdest kullanılmıştır. Kullanılan bilgisayar olarak core i7 işlemcili dört çekirdekli ve 6 GB ram özelliği olan Fedora 6 işletim sistemi ile çalışan bir bilgisayar tercih edilmiştir.

Tablo 6.1 Simülasyon model parametreleri.

Parametre	Değer
Trafik modeli	FTP/TCP
Trafik üreteci	CBRGEN
Topoloji üreteci	SETDEST
Paket boyutu (Bayt)	512
Simülasyon süresi (s)	50
Bilgisayar	I7, 6 GB RAM
İşletim sistemi	Fedora

Diğer simülasyon parametreleri de farklı senaryolar için sabit tutulmuştur. Tablo 6.2'de görüldüğü gibi toplam 16 adet senaryo ve trafik dosyası üretilmiştir. 10 düğüm için 300 x 300 m^2, 50 düğüm için 500 x 500 m^2, 100 düğüm için 700 x 700 m^2 ve 200 düğüm için 700 x 700 m^2 simülasyon alanları kullanılmıştır. Simülasyon alanı içerisine düğümler rastgele yerleştirilip hızları minimum 0 ve maksimum hızı 1 m/s, 5

m/s, 10 m/s ve 20 m/s hızlarda olacak şekilde senaryo dosyası üretilip simülasyon sonuçları hesaplanmıştır.

Tablo 6.2 Simülasyon ağ modelleri.

Topografya				
Düğüm hareket hızı		Simülasyon alanı		Düğüm sayısı
Minimum	Maksimum	X(metre)	Y(metre)	
0	1	300	300	10
0	5	300	300	10
0	10	300	300	10
0	20	300	300	10
0	1	500	500	50
0	5	500	500	50
0	10	500	500	50
0	20	500	500	50
0	1	700	700	100
0	5	700	700	100
0	10	700	700	100
0	20	700	700	100
0	1	1500	1500	200
0	5	1500	1500	200
0	10	1500	1500	200
0	20	1500	1500	200

Trafik ve senaryo dosyaları Ek-B ve Ek-C'de görüldüğü gibi oluşturulmuştur. 10 düğüm için trafik dosyası T10, 50 düğüm için trafik dosyası T50, 100 düğüm için trafik dosyası T100 ve 200 düğüm için trafik dosyası T200 olarak adlandırılmıştır. Trafik ve senaryo dosyaları oluşturulduktan sonra tcl kodları Ek-D, Ek-E, Ek-F ve Ek-G'de verilmiştir. Örnek olarak; 10 düğümlü ağ için 2 adet trafik dosyası ve 300 x 300 m^2 alana rastgele dağıtılmış ve düğüm hızları maksimum 1 m/s olan senaryo dosyası yazıldığı gibi tcl dosyasından çağrılmaktadır.

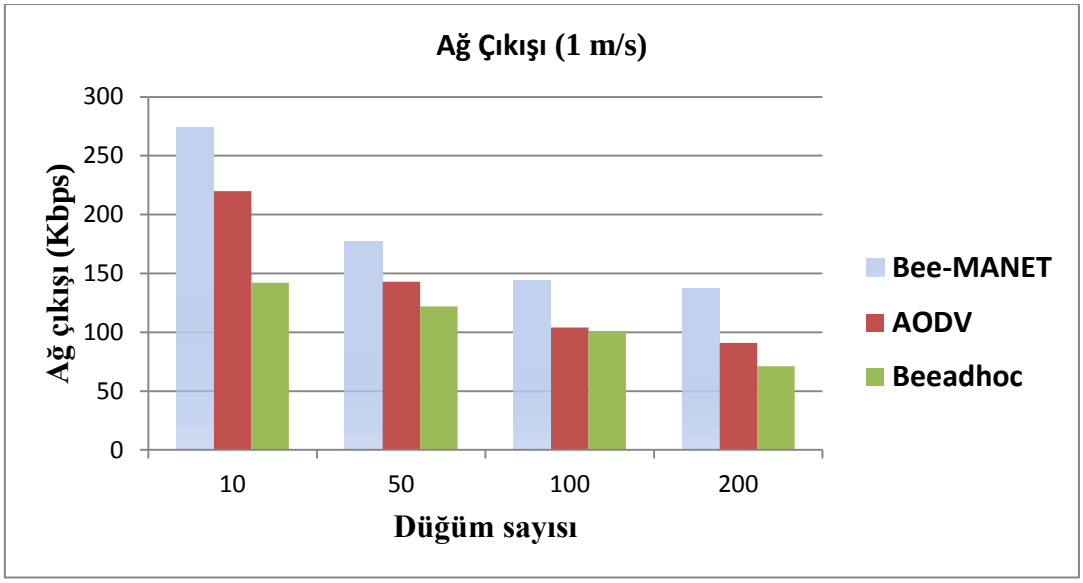
Her bir simülasyon çalışması üç defa tekrar edilmiş ve çıkan sonuçların birbirine yakın değerler olduğu görülmüştür. Çıkan üç değerin ortalamaları alınarak karşılaştırmalarda bu ortalama değerler kullanılmıştır.

6.4. Simülasyon Sonuçları

6.4.1. Ağ çıkışı

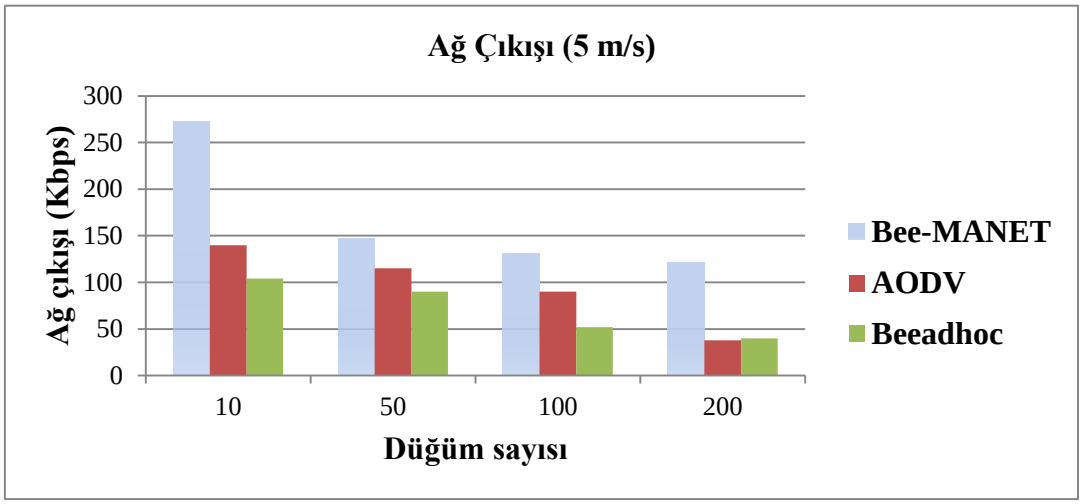
Şekil 6.1, Şekil 6.2, Şekil 6.3 ve Şekil 6.4'te dört farklı ağın değişik hızlar altındaki ağ çıkış değerleri görülmektedir.

Şekil 6.1'de, düğümlerin maksimum 1 m/s hızla hareket ettiği ağlarda BeeAdhoc, AODV ve Bee-MANET protokollerinin çıkış değerleri verilmektedir. Tüm düğüm sayılarında Bee-MANET protokolü sırasıyla AODV ve Beeadhoc protokokünden daha iyi sonuç verdiği görülmüştür. AODV ve Beeadhoc protokolleri 100 düğümlü ağda hemen hemen aynı verimlilikte çalışmışlardır. Bee-MANET protokolü, kontrol paketlerini birleştirmesi ve daha basit yapıya sahip olmasından dolayı AODV ve BeeAdhoc protokollerine göre daha fazla veri paketi iletmiştir.



Şekil 6.1 Maksimum 1 m/s hız için ağ çıkışı.

Şekil 6.2'de, düğümlerin maksimum 5 m/s hızla hareket ettiği ağlarda BeeAdhoc, AODV ve Bee-MANET protokollerinin çıkış değerleri verilmektedir. Tüm düğüm sayılarında Bee-MANET protokolü 1 m/s hızdaki ağlarda olduğu gibi sırasıyla AODV ve Beeadhoc protokokünden daha iyi sonuç verdiği görülmüştür. AODV ve Beeadhoc protokolleri 200 düğümlü ağda hemen hemen aynı verimlilikte çalışmışlardır. Bee-MANET protokolü, özellikle 10 düğümlü ağda kontrol paketlerini birleştirmesi ile AODV ve BeeAdhoc protokollerine göre daha fazla veri paketi iletmiştir.

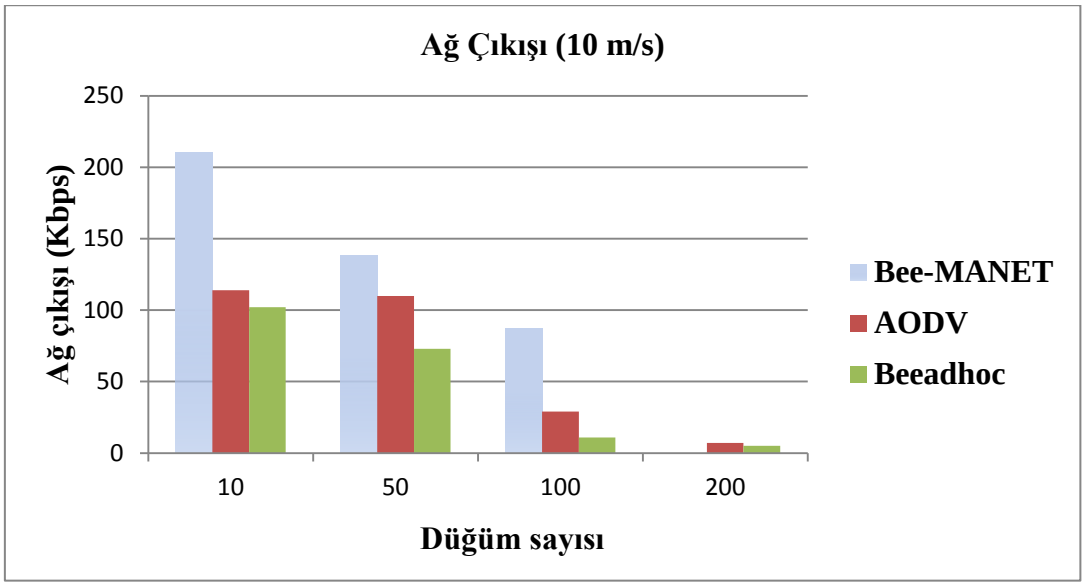


Şekil 6.2 Maksimum 5 m/s hız için ağ çıkışı.

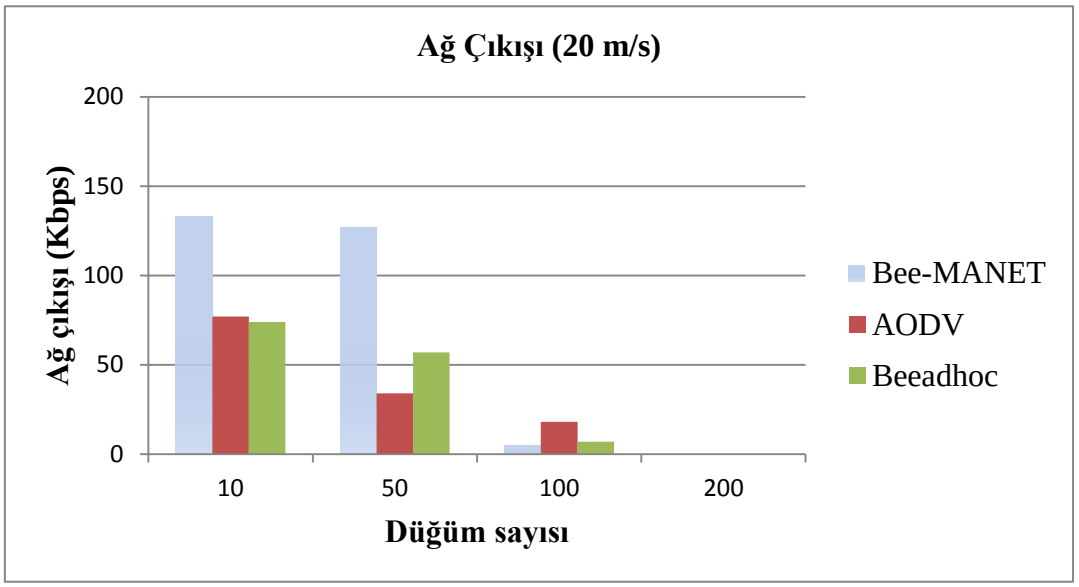
Şekil 6.3'te, düğümlerin maksimum 10 m/s hızla hareket ettiği ağlarda BeeAdhoc, AODV ve Bee-MANET protokollerinin çıkış değerleri verilmektedir. 10, 50 ve 100 düğümlü ağlarda BeeMANET protokolü AODV ve Beeadhoc AODV ve Beeadhoc protokollerinden daha iyi sonuç verdiği görülmüştür. 200 düğümlü ağda Bee-MANET protokolünün kontrol paketlerini birleştirmesinden dolaya oluşan gecikmenin etkisiyle paket iletememiştir.

Şekil 6.4'te, düğümlerin maksimum 20 m/s hızla hareket ettiği ağlarda BeeAdhoc, AODV ve Bee-MANET protokollerinin çıkış değerleri verilmektedir. 10 ve 50 düğümlü ağlarda BeeMANET protokolü AODV ve Beeadhoc AODV ve Beeadhoc protokollerinden daha iyi sonuç verdiği görülmüştür. 100 düğümlü ağda Bee-MANET protokolü, AODV ve Beeadhoc protokollerinden daha az paket ilettiği

görülmüştür. Bee-MANET protokolünün kontrol paketlerini birleştirmesi ve birleştime için beklenen sürenin 10 m/s yeye kadarki hizlarda çok verimli olduğu görülürken, 10 m/s den daha yüksek hızlarda paket iletimine olumsuz etki ettiği görülmüştür. 200 düğümlü ağda tüm protokoller ağdaki düğümlerin fazlalığı ve düğümlerin çık hızlı hareket etmelerinden dolayı oluşan hızlı topoloji değişikliği paket iletimini engellemektedir.



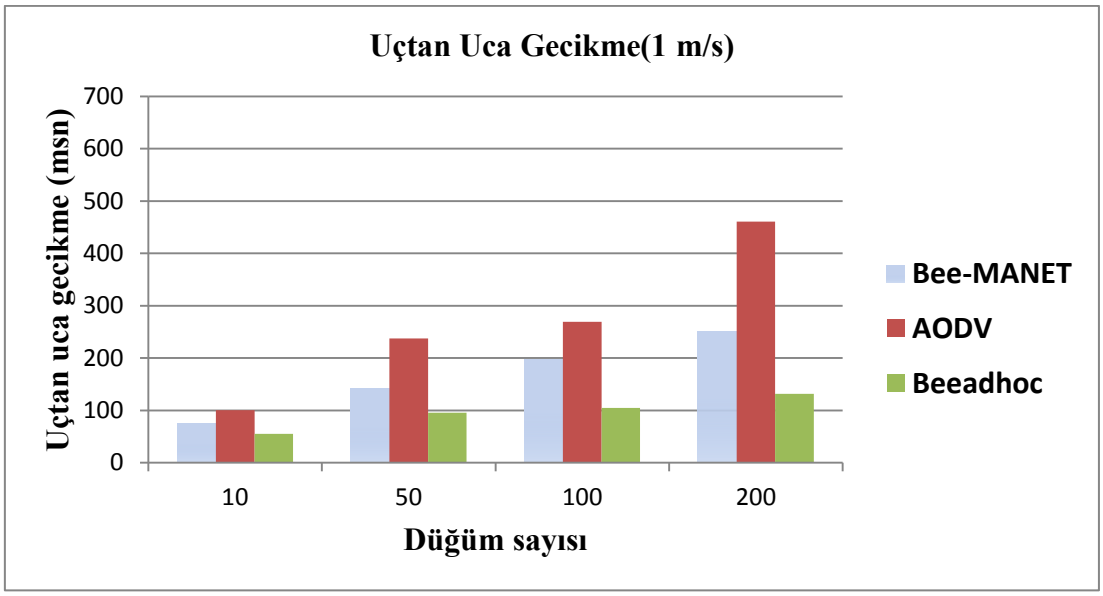
Şekil 6.3 Maksimum 10 m/s hız için ağ çıkışı.



Şekil 6.4 Maksimum 20 m/s hız için ağ çıkışı.

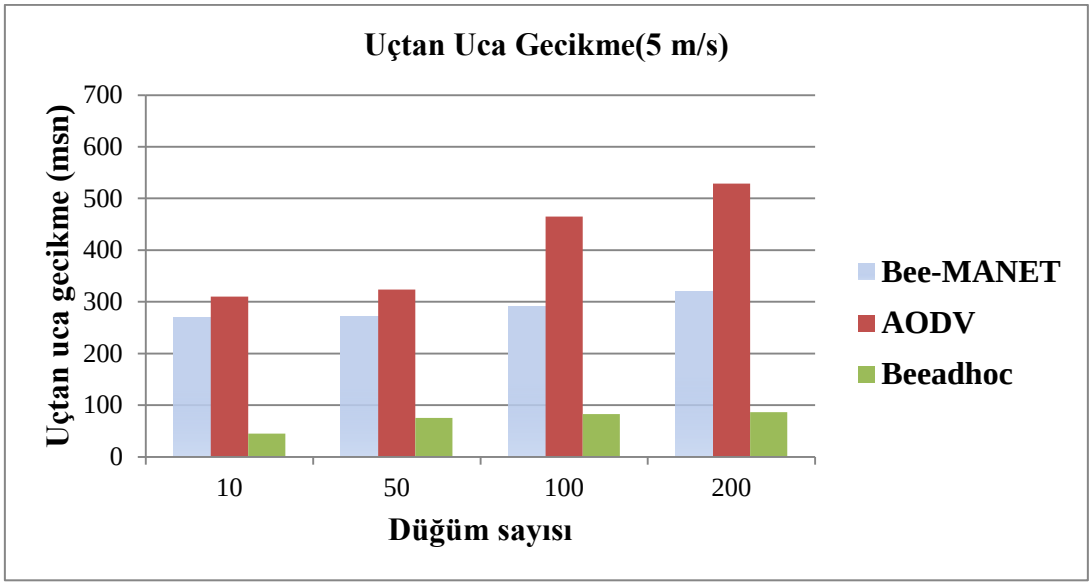
6.4.2. Uçtan uca gecikme

Şekil 6.5'te düğümlerinin maksimum 1 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim gecikmesi milisaniye olarak hesaplanmıştır. Tüm düğüm sayılarında BeeAdhoc protokolü AODV ve Bee-MANET protokollerinden daha iyi sonuç verdiği görülmektedir. Bee-MANET protokolü, 10, 50, 100 ve 200 düğümlü ağlarda BeeAdhoc protokolüne yakın, AODV protokolüne göre daha kısa sürede paket iletmiştir. Bu durum, BeeAdhoc ve Bee-MANET protokollerinin AODV protokolüne göre daha basit yapıda olmasından kaynaklanmaktadır.

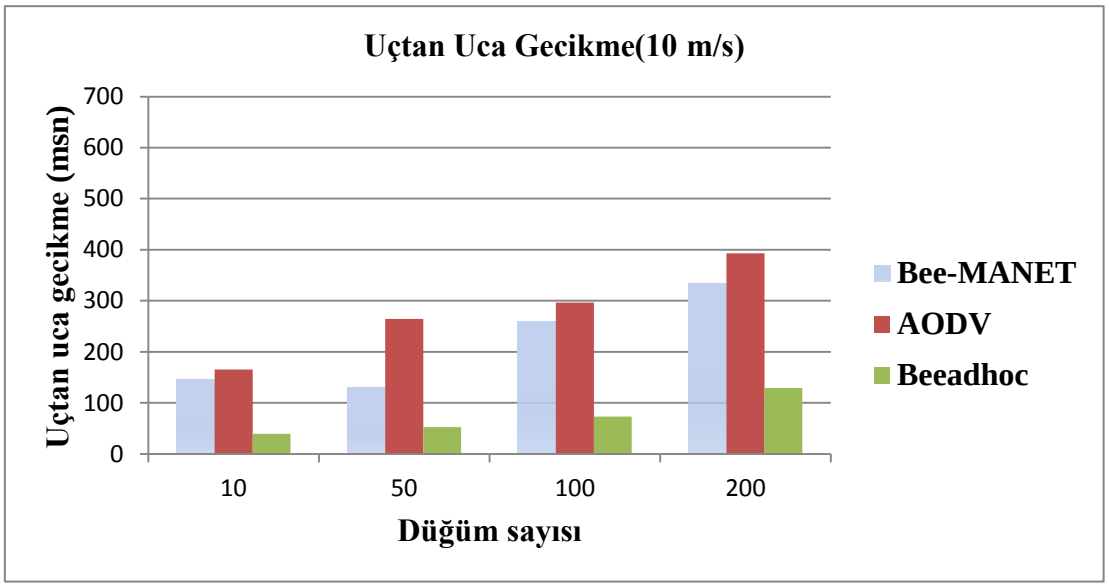


Şekil 6.5 Uçtan uca gecikme karşılaştırılması(1 m/s).

Şekil 6.6'da düğümlerinin maksimum 5 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim gecikmesi milisaniye olarak hesaplanmıştır. Tüm düğüm sayılarında BeeAdhoc protokolü düğüm hareket hızının maksimum 1 m/s olduğu ağlardan AODV ve Bee-MANET protokollerine göre daha iyi sonuç verdiği görülmektedir. Bee-MANET protokolü AODV protokolüne göre daha basit yapıda olmasından dolayı 10, 50, 100 ve 200 düğümlü ağlarda AODV protokolüne göre daha kısa sürede paket iletmiştir. Düğümlerin daha hızlı hareket etmesi ağ topolojilerinin daha hızlı değişmesine ve böylelikle paketlerin uçtan uca iletilmesi süresi artmıştır.



Şekil 6.6 Uçtan uca gecikme karşılaştırılması(5 m/s).

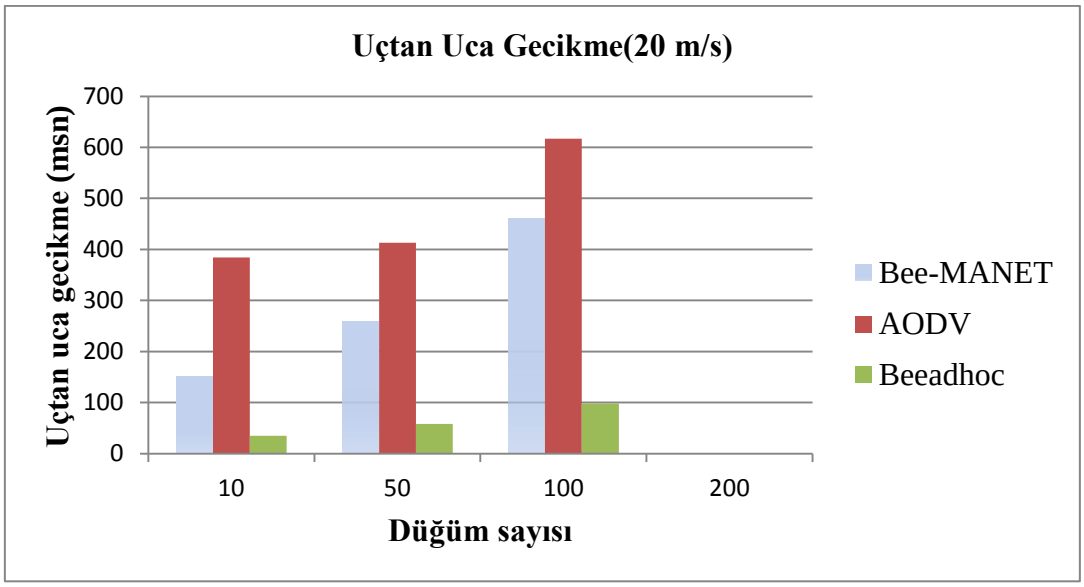


Şekil 6.7 Uçtan uca gecikme karşılaştırılması(10 m/s).

Şekil 6.7'de düğümlerinin maksimum 10 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim gecikmesi milisaniye olarak hesaplanmıştır. Tüm düğüm sayılarında BeeAdhoc protokolü düğüm hareket hızının maksimum 1 m/s ve 5 m/s ağlarda olduğu gibi AODV ve Bee-MANET protokollerine göre daha iyi sonuç verdiği görülmektedir. Bee-MANET protokolü, 10, 50, 100 ve 200 düğümlü ağlarda AODV protokolüne göre daha kısa sürede paket iletmiştir. Düğümlerin daha

hızlı hareket etmesi ağ topolojilerinin daha hızlı değişmesine ve böylelikle paketlerin uçtan uca iletilmesi süresi artmıştır.

Şekil 6.8'de düğümlerinin maksimum 20 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim gecikmesi milisaniye olarak hesaplanmıştır. 10, 50 ve 100 düğüm sayılarında her üç protokolde paket iletebilmelerine rağmen 200 düğümlü hiçbir protokol ağ çıkışı vermediğinden uçtan uca gecikme bu hızda ölçülememiştir. 20 m/s maksimum hızda ve 200 düğümlü ağda, düğümlerin çok hızlı hareket etmesi ve ağın topolojisinin çok daha hızlı değişmesi paket iletilememesine neden olmuştur.

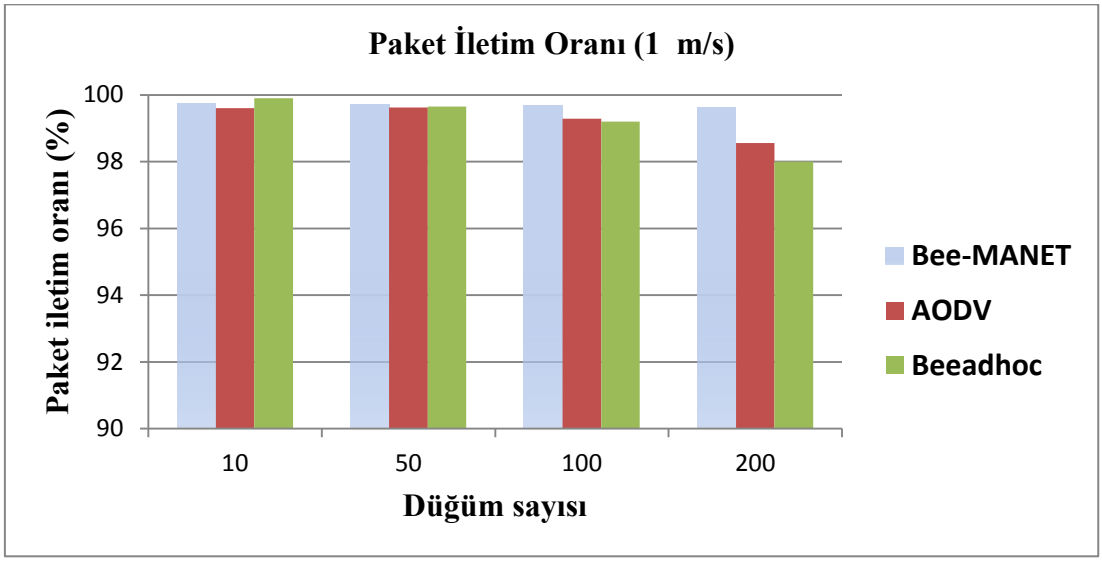


Şekil 6.8 Uçtan uca gecikme karşılaştırılması(20 m/s).

6.4.3. Paket iletim oranı

Şekil 6.9'da düğümlerinin maksimum 1 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim oranı yüzde olarak hesaplanmıştır. 10 ve 50 düğüm sayılarında BeeAdhoc protokolü sırasıyla Bee-MANET ve AODV protokollerinden daha iyi sonuç verdiği görülmesine rağmen, 100 ve 200 düğüm sayılarında Bee-MANET protokolü AODV ve BeeAdhoc protokollerine göre daha iyi sonuç vermiştir. Her üç protokol için, ağdaki düğümlerin hızının maksimum 1 m/s gibi çok az olduğu durumlarda oluşturulan yolların geçerlilik süresi uzun olması, daha

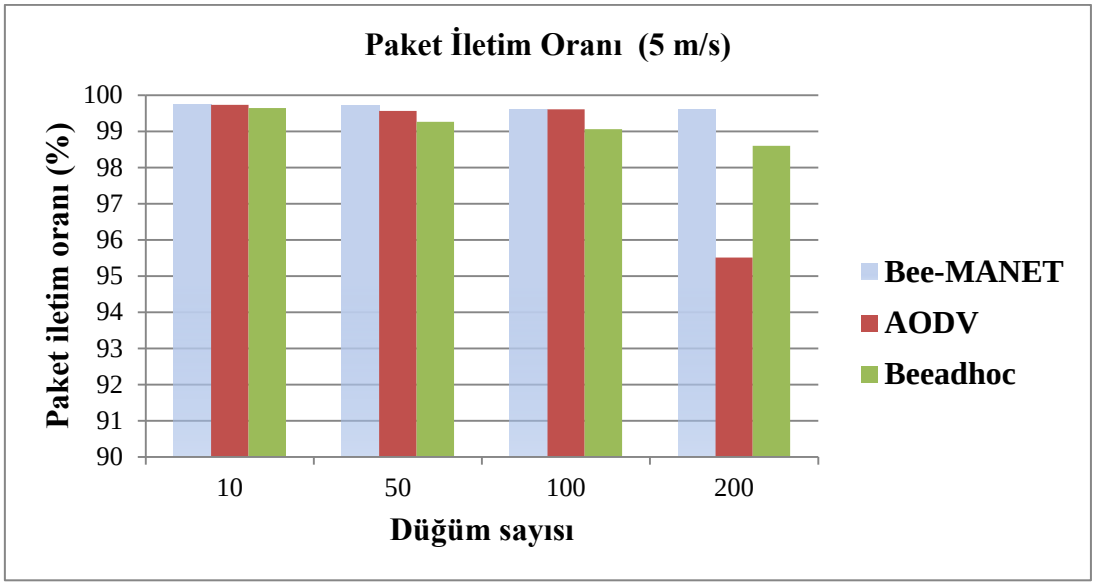
az paket kaybına, dolayısıyla daha yüksek paket iletim oranı gerçekleşmesine neden olmuştur.



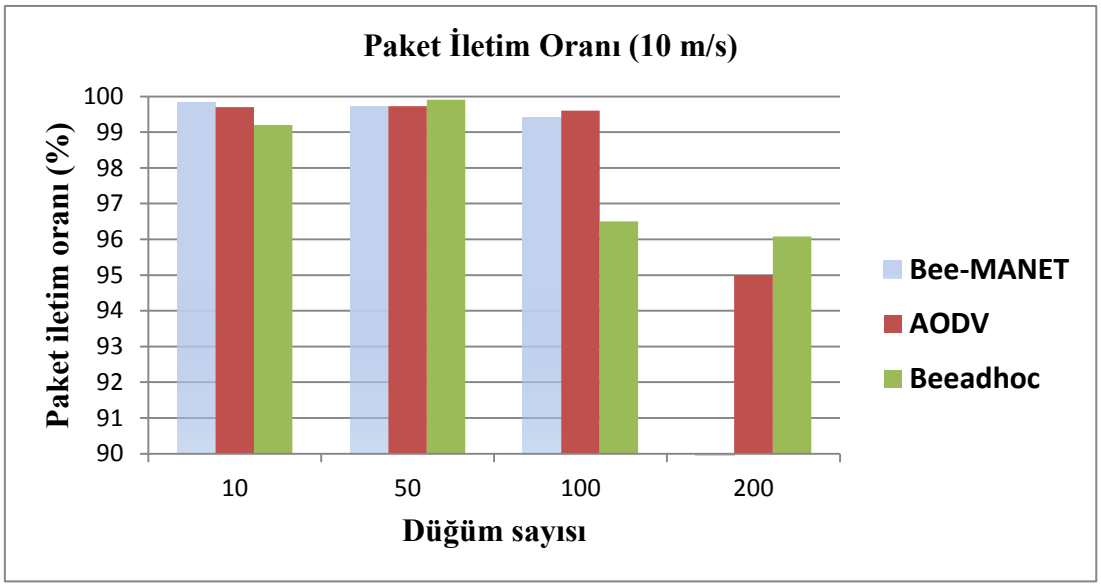
Şekil 6.9 Paket iletim oranı karşılaştırılması (1 m/s).

Şekil 6.10'da düğümlerinin maksimum 5 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim oranı yüzde olarak hesaplanmıştır. 10, 50, 100 ve 200 düğüm sayılarında Bee-MANET protokolü BeeAdhoc ve AODV protokollerinden daha iyi sonuç verirken 200 düğümlü ağda hızda BeeAdhoc protokolü AODV protokolünden daha iyi sonuç vermiştir. Bee-MANET ve BeeAdhoc protokollerinin daha basit yapıda ve daha az kontrol paket göndermesi ağ trafiğinin daha az olmasına ve böylelikle daha az paket çarpışmasının meydana gelmesine sebep olmaktadır. Bu durum, paket iletim oranlarının AODV protokolüne göre daha fazla olmasına sebep olmaktadır.

Şekil 6.11'de düğümlerinin maksimum 10 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim oranı yüzde olarak hesaplanmıştır. 10, 50, 100 ve 200 düğüm sayılarının hepsinde BeeAdhoc ve AODV protokolleri paket iletebilmişlerdir. Bee-MANET protokolü 10, 50 ve 100 düğümlü ağlarda paket iletebilmesine rağmen 200 düğümlü ağda paket iletemediğinden uçtan uca gecikme hesaplanamamıştır. Bunun sebebi, düğüm sayısının fazla olduğu ağlarda muhtemel adım sayısının da fazla olmasına paketin müsaade edilen sürede hedefe ulaşmamasıdır.



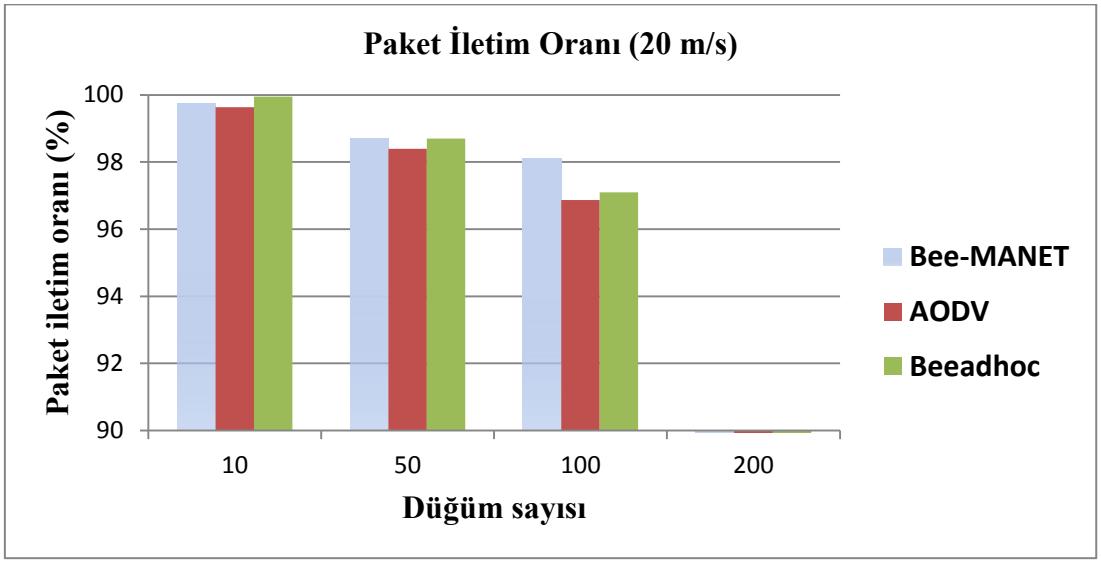
Şekil 6.10 Paket iletim oranı karşılaştırılması (5 m/s).



Şekil 6.11 Paket iletim oranı karşılaştırılması (10 m/s).

Şekil 6.12'de düğümlerinin maksimum 20 m/s hızla hareket ettiği 10, 50, 100 ve 200 düğümlü ağlarda uçtan uca paket iletim oranı yüzde olarak hesaplanmıştır. 200 düğümle ağda AODV, Bee-MANET ve Beeadhoc protokolleri paket iletemediğinden uçtan uca gecikme hesaplanamamıştır. 10 düğümlü ağda Beeadhoc protokolü Bee-MANET ve AODV protokollerine göre daha iyi sonuç vermiştir. 50 ve 100 düğümlü ağlarda ise Bee-MANET protokolü Beeadhoc ve AODV protokollerine göre daha iyi sonuç vermiştir. 200 düğümlü ağda tüm protokollerin paket iletememesinin sebebi,

düğüm sayılarının fazla olduğu ağlarda muhtemel adım sayısının da fazla olması ve böylece paketlerin müsaade edilen sürede hedefe ulaşamamasıdır.



Şekil 6.12 Paket iletim oranı karşılaştırılması (20 m/s).

6.5. Sonuçların Değerlendirilmesi

Tez çalışmasında, yapay zeka tekniklerinden biri olan oğul zekası kullanılarak oluşturulan bal arısı algoritmasından verimli sonuçlar alınmıştır. Sonuçlar incelendiğinde aşağıdaki çıkarımlar yapılabilir;

1. Büyük ölçekli ağlarda Ns-2 simülasyon ortamı verimli sonuçlar alınabilecek bir ortamdır.
2. Ns-2 ortamında setdest ve cbrgen.tcl kullanılarak ortamı kolayca senaryo ve trafik topolojileri üretilebilmekte; deneyler ve sonuçlar tüm simülasyon parametreleri için kısa zamanda gerçekleştirilebilmektedir.
3. Ns-2 simülasyon ortamı kullanılarak Ad-Hoc ağlara uygulanan Bee-MANET yönlendirme protokolü veri toplama problemine daha iyi bir çözüm getirmektedir.
4. Bee-MANET protokolü, daha basit yapıda olmasından dolayı AODV ve BeeAdhoc protokollerine göre daha fazla veri paketi iletmiştir.

5. Bee-MANET protokolü, AODV protokolüne göre daha az ağ trafiğine sahip olmasından dolayı uctan uca gecikmeyi azaltmıştır.
6. Bee-MANET protokolü, daha basit yapıda ve ağ trafiğini az olmasından dolayı AODV ve BeeAdhoc protokollerine göre daha yüksek paket iletim oranına sahiptir.
7. Bee-MANET protokolü, oluşturulan kontrol paketlerini birleştirmesi ile bir miktar gecikme oluşturmasından dolayı uctan uca gecikmede BeeAdhoc algoritmasının gerisinde kalmıştır.

Sonuç olarak; Ns-2 ortamı esnek ve sade yapısıyla, yeni bir yönlendirme protokolünün oluşturulması kolaylığı ve Ns-2 ortamına kolayca eklenmesi Ns-2 simülatörünün üstünlüklerindendir. Geliştirilen Bee-MANET protokolü veri toplama ve paket iletim oranı problemlerine uygun çözümler getirmektedir. Geliştirilen düğüme gelen kontrol paketlerini birleştirip komşu düğümlere birleştirilmiş paketleri tek bir pakette gönderilmesi veri iletim problemleri çözümüne katkı sağlamıştır.

BÖLÜM 7. TARTIŞMA VE ÖNERİLER

Kablosuz Ad-Hoc ağlar için geliştirilen oğul zekâsı tabanlı yönlendirme protokolü Bee-MANET Ad-Hoc ağlarda veri iletimi ve paket iletim oranı problemlerine çözüm getirmektedir.

Bu çalışma, geliştirilen ortamın değişik özelliklerini göstermek üzere iki aşamadan oluşmuştur:

1. Bal arılarının yiyecek aramalarından esinlenerek geliştirilen BeeAdhoc protokolünün Ns-2 ağ simülatöründe modellenmesi,
2. Geliştirilen protokolünün büyük ölçekli ve yüksek hareket hızındaki ağlarda davranışlarının incelenmesi.

Bee-MANET protokolünün bir düğüme gelen kontrol paketlerini birleştirmesi, düğüm trafiğini ve ağ trafiğindeki kontrol paketi sayısını azaltarak daha fazla veri paketi iletimini sağlamaktadır. Bee-MANET protokolünün verimli çalışmasının nedeni Ad-Hoc ağların sosyal canlıların davranışları ile olan ortak özellikleridir. Bal arılarının danslar ile nektar alanları hakkında toplamış oldukları bilgileri kolonideki diğer üyelere aktarabilme özellikleri, bilgisayar ağlarındaki kaynak düğüm ile hedef düğüm arasındaki yolların bulunması ile benzerlik göstermektedir. Böylece, sosyal canlıların yiyecek aramadaki davranışlarından esinlenilmiş olmaktadır.

Kablosuz Ad-Hoc ağlar için geliştirilen Bee-MANET protokolünün Ns-2 ortamında uygulanması oğul zekâsı tabanlı geliştirilen yöntem ve tekniklerin ayrık olaylı modelleme ve benzetim ortamlarında gerçekleştirilmesi daha kolay, hızlı ve esnek bir şekilde olduğunu göstermektedir.

Geliştirilen yönlendirme protokolü nihai bir hedef değil literatürde bir basamaktır. Hedefe giden yolda daha detaylı modeller oluşturan ve daha verimli çalışan modelleme ve benzetim ortamları oluşturmak mümkündür. Yapılan çalışma açık kaynak kodlu olduğundan her türlü bilimsel katkıya açık ve geliştirilebilir bir özelliğe sahiptir.

Geliştirilen model, çok yüksek hızlarda ve büyük ağlarda AODV protokolünün gerisinde kalmaktadır. Daha iyi bir optimizasyon ve kümeleme yöntemleri kullanılarak daha iyi sonuçlar alınabilir.

Geliştirilen modelde kullanılan toplayıcı öncü arı yöntemi kablosuz sensör ağlar için kullanıldığında ağ performansını arttırabilir.

Geliştirilen Bee-MANET protokolünün performansı yalnızca Ns-2 ağ simülatöründe uygulanmıştır. Bee-MANET protokolü, yaygın olarak kullanılan ve doğruluğu tartışılmayan simülasyon ortamlarında (OPNET, OMNET, DEVS, vb.) bulunan diğer protokollerle karşılaştırılabilir.

KAYNAKLAR

[1] AKYILDIZ, I. F., SU, W., SANKARASUBRAMANIAM, Y., CAYIRCI, E., Wireless Sensor Networks: A Survey, Elsevier Computer Networks, 38 393–422, USA, 2002.

[2] STEENSTRUP, M. E. (Ed.). Routing in Communications Network. Prentice-Hall, 1995.

[3] ALBAYRAK, Z., ZENGIN, A.,ÇELIK, F., Swarm Intelligence Based Routing Protocols for Mobile ad hoc Networks(MANETs), International Science and Technology Conference, 2010.

[4] FAROOQ, M., From the wisdom of the hive to intelligent routing in telecommunication networks, Doctoral dissertation, Dortmund University of Technology, 2006.

[5] ZENGİN, A. "Dağıtık Simülasyon Sistemleri İçin Yeni Bir Yönlendirme Algoritması ve Uygulaması, Fen Bilimleri Enstitüsü, Sakarya Üniversitesi, 2004.

[6] BERNARD P. ZEIGLER, HERBERT P., and TAG K. Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems. Academic Press, second edition, 2000.

[7] RAHMAN, M.A., PAKŠTAS, A., WANG, F. Z., Network Modelling and Simulation Tools, Simulation Modelling Practice and Theory, International Journal of the Federation of European Simulation Societies – EUROSIM, 17, pp. 1011-1031,USA, 2009.

[8] VANGHELUWE H., Multi-Formalism Modeling and Simulation, proefschrift, Universiteit Gent Faculteit Wetenschappen, 2001.

[9] ANDREW, S., Computer Networks 3rd edition, Prentice-Hall, 1996.

[10] HANDE, A., POLK, T., WALKER, W., BHATIA, D., Self-Powered Wireless Sensor Networks for Remote Patient Monitoring in Hospitals, Moleculer Diversity Preservation International, 10.3390/s6091102, Switzerland, 22 September 2006.

[11] BAYILMIŞ, C., IEE 802.11 KLAN Kullanarak Can Segmentleri Genişleten Arabağlaşım Birimi Tasarımı, Kocaeli Üniversitesi, 2006.

[12] CORSON, M., Internet Engineering Task Force (IETF) Mobile Ad Hoc Networks (MANET) Working Group Charter, Chaired by Joseph Macker and http://www.ietf.org/html.charters/manet-charter.html.

[13] JUBIN, J., TORNOW, J.D., The DARPA packet radio network protocols, Proceedings of the IEEE 75 (1), (1987.

[14] PERKINS, C.E., Ad Hoc Networking, Addison Wesley, 2001.

[15] TANENBAUM , A. S., Computer Networks, Prentice-Hall Inc., 1996.

[16] LEE, D. J., & LEE, W. C. (2000). Ricocheting bluetooth. In Microwave and Millimeter Wave Technology, 2000, 2nd International Conference on. ICMMT 2000 (pp. 432-435). IEEE.

[17] BÜBER, Ş., Kablosuz Ağlar ve Bilgisayar Ağları, http://www.mutasyon.net, Erişim Tarihi: 11.03.20011.

[18] ALAGÖZ, F., Mobil Ağlar ve Veri Erişim Stratejileri, Fen Bilimleri Enstitüsü, Kahramanmaraş Sütçü İmam Üniversitesi, 2005.

[19] KARASULU, B.; TOKER, L.; KORUKOĞLU, S., ZigBee - IEEE 802.15.4 Standardı Temelli Kablosuz Algılayıcı Ağları, XIV. Türkiye'de İnternet Konferansı - Inet-tr'09, sayfa : 131-134 İstanbul, 2009.

[20] CHANDRA, P., DOBKIN, D. M., BENSKY, A., OLEXA, R., LIDE, D. A., DOWLA, F., RF & Wireless Networking, Elsevier, s-510, USA, 2008.

[21] FAZEL, K., KAISER,S., Multi-Carrier and Spread Spectrum Systems: From OFDM and MC-CDMA to LTE and WiMAX, 2nd Edition, John Wiley & Sons, 2008.

[22] HEINE, G., HORRER, M. . GSM networks: protocols, terminology, and implementation. Artech House, Inc., 1999.

[23] HARRINGTON, S., Computer graphics. McGraw-Hill, 1987.

[24] TANENBAUM K., Computer Networks, ISBN 0-13-394248-1, page 280, 431-432. Prentice Hal, 2002,

[25] CHAIKEONG T, Associativity-Based Routing for Ad-Hoc Mobile Networks, Wireless Personal Communications 4: 103–139, Kluwer Academic Publishers. Printed in the Netherlands, 1997

[26] ABOLHASAN M., WYSOCKI T., DUTKIEWICZ E., A review of routing protocols for mobile ad hoc Networks, AD HOC NETWORKS, 2004.

[27] EHSAN H., UZMI Z., Performance Comparison Of Ad Hoc Wireless Network Routing Protocols, 0-7803-8680-IEEE, INMIC, 2004.

[28] RABAEY J.M., AMER M.J., DA SILVA J.L., PATEL D., ROUNDY, S., Pico Radio Supports Ad Hoc Ultra-Low Power Wireless Networking, IEEE Computer Society 2000 Published Conference Proceedings, Volume 33, Issue 7, July 2000.

[29] CHIANG C.C., WU H.K., LIU W., GERLA M., Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel, IEEE Singapore International Conference on Networks, SICON'97, pp. 197-211, Singapore, 16.-17. April 1997.

[30] PANTHONG S., JANTARANG S., 3G Mobile Wireless Routing Optimization By Genetic Algorithm, CCECE 2003 - CCGEI 2003, Montreal, May 2003.

[31] MURTHY S., ACEVES G.L., An Efficient Routing Protocol For Wireless Networks, Mobile Networks and Applications, Volume 1, Issue 2 (October 1996),1997.

[32] RAJU J., ACEVES J.J., A comparison of on-demand and table driven routing for ad hoc wireless Networks, Communications 2000. ICC 2000, IEEE International Conference on Volume 3, June 2000.

[33] JOA M., LU T., A Peer-to-Peer zone-based two-level link state routing for mobile Ad Hoc Networks IEEE Journal on Selected Areas in Communications, Special Issue on Ad- Hoc Networks, Aug. 1999.

[34] DAI F., DAI Q., WU J., Power efficient routing trees for ad hoc wireless Networks using directional antenna, Ad Hoc Networks, Volume 3, Issue 5, September 2005.

[35] JUNG J., PARK T., KIM C., A forwarding scheme for reliable and energy-efficient data delivery in cluster-based sensor Networks, IEEE Communications Letters, Volume 9, Issue 2, Feb. 2005.

[36] BASAGNI, S., CHIAMTAC, I., VIOLET, R., Woodward, B.A., A Distance Routing Effect Algorithm for Mobility (DREAM), In Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking (pp. 76-84), 1998.

[37] PHAM P., PERREAU S., Multi-path routing protocol with load balancing policy in mobile ad hoc network, IFIP Int'l Conference on Mobile and Wireless communications Networks (MWCN 2002), September 2002.

[38] LEE Y.J., Routing And Efficient Evaluation Techniques For Multi-Hop Mobile Wireless Networks, School of Electrical and Computer Engineering Georgia Institute of Technology December 2005.

[39] KARA R., Gezgin Tabanlı Ağlar İçin Yeni Bir Konum Tabanlı Melez Yönlendirme Algoritması , Fen Bilimleri Enstitüsü, Sakarya Üniversitesi, 2009.

[40] SCHOONDERWOERD R., HOLLAND O., BRUTEN J., ROTHKRANTZ L., Load balancing in telecommunications networks. Adaptive Behavior, 5(2), 1997draft-ietf-manet-zone-brp-02.txt, July, 2002.

[41] KASSABALIDIS I., EL-SHARKAWI M.A, MARKS R.J., ARABSHAHI P., GRAY A.A. , Swarm Intelligence for Routing in Communication Networks, University of Washington, 2001.

[42] CARO G.DI., DORIGO M., Mobile agents for adaptive routing, InSystem Sciences., Proceedings of the Thirty-First Hawaii International Conference on (Vol. 7, pp. 74-83). IEEE, 1998.

[43] FAROOQ M., From the Wisdom of the Hive to Intelligent Routing in Telecommunication Network, PhD Dissertation, University of Dortmund, 2006.

[44] JIANPING W., ESEOSA O., Parimala Thulasiraman,Ruppa K. Thulasiram, Hopnet: A hybrid ant colony optimization routing algorithm for mobile ad hoc networks, University of Manitoba, Canada.

[45] GOSWAMI M.M., DHARASKAR R.V., Fuzzy Ant Colony Based Routing Protocol For Mobile Ad Hoc Network, 2009 International Conference on Computer Engineering and Technology.

[46] CORSON M., EPHREMIDES A., A distributed routing algorithm for mobile radio networks, ACM Wireless Networks Journal, 1995.

[47] PERKINS C., ROYER E., Ad-Hoc on-demand distances vector routing. In Proceedings of Second IEEE Workshop on Mobile Computing Systems and Applications, 1999.

[48] DUBE R., Signal Stability based adaptive routing for Ad Hoc Mobile networks, IEEE Pers. Comm., Feb. 1997, pp. 36-45. http://www.cs.umd.edu/projects/mcml/papers/pcm97.ps.

[49] MURTHY, S., GARCIA A. , Mobile Networks and Applications, Kluwer Academic Publishers, 2003.

[50] YANG, L., HONG, M., Delay Based Load Aware Routing Protocol for Ad Hoc Networks, 4th IEEE Consumer communications and Networking, Conference, Jan. 2007.

[51] ZYGMUNT J.H., PEARLMAN M.R., SAMAR P., The Zone Routing Protocol (ZRP) for Ad Hoc Networks, draft-ietf-manet-zone-zrp-04.txt, July, 2002.

[52] ZYGMUNT J.H., PEARLMAN M.R., SAMAR P., The Interzone Routing Protocol (IERP) for Ad Hoc Networks, draft-ietf-manet-zone-ierp-02.txt, July, 2002.

[53] ZYGMUNT J.H., PEARLMAN M.R., SAMAR P., The Border cast Resolution Protocol (BRP) for Ad Hoc Networks, draft-ietf-manet-zone-brp-02.txt, July, 2002.

[54] KARA, R., ÖZÇELIK, İ. Pozisyon tabanlı ad hoc yönlendirme algoritmalarında bulanık mantık yolu ile yol seçimi. 5. Uluslararası İleri Teknolojiler Sempozyumu, 2009.

[55] RAJU, J., ACEVES, J.J., A comparison of on-demand and table driven routing for ad hoc wireless Networks, Communications 2000. ICC 2000, IEEE International Conference on Volume 3, 2000.

[56] BONABEAU, E., DORIGO, M., THÉRAULAZ, G., Swarm intelligence: from natural to artificial systems, Oxford University Press, 1999.

[57] LIPPERTS, S., KRELLER, B. , "Mobile agents in telecommunications networks a simulative approach to load balancing", Proc. 5th Intl. Conf. Information Systems, Analysis and Synthesis, ISAS'99, 1999.

[58] HOLLDOBLER, B., WILSON, E.O., Journey to the Ants, Belknap Press, Harvard University Press.

[59] BECKERS R., DENEUBOURG J.L., & Goss, S. (1992). Trails and U-turns in the selection of a Path by the Ant Lasius Niger. In J. Theor. Biol. 159,397-415.

[60] FRISCH K.V., The Dance Language and Orientation of Bees. Harvard University Press, Cambridge, 1967.

[61] Ns-2 Network Simulator http://www.isi.edu/nsnam/ns/ , Erişim Tarihi: 23.05.2012.

[62] BAJAJ, S., BRESLAU, L., ESTRIN, D., et al., Improving Simulation for Network Research, USC Computer Science Dept. Technical Report, 1999.

[63] RICHARD, M., Fujimoto, Kalyan S. Perumalla & George F. Riley, Network Simulation, Morgan &Claypool, Erişim Tarihi: 12.01.2012.

[64] Breslau, L., Advances in Network Simulation, http://www.isi.edu/~johnh/PAPERS/Breslau00a.pdf, Erişim Tarihi: 17.06.2013.

[65] RAHMAN, M. A., PAKSTAS, A., WANG, F. Z., Network Modeling and Simulation Tools, Simulation Modeling Practice and Theory, vol. 17, no. 6, pp. 1011–1031, 2009.

[66] FUJIMOTO, R.M., PERUMALLA, K.S., RILEY G.F., Network Simulation, Morgan &Claypool, 2007.

[67] Opnet Modeller, http://www.opnet.com/products/modeler/home.html, Erişim Tarihi: 20.06.2013.

[69] QualNet Network Simulator , http://web.scalable-networks.com/content/qualnet, Tarihi: 20.06.2013.

[70] SSFNet Network Simulator, http://www.ssfnet.org/internetPage.html, Erişim Tarihi: 20.06.2013.

[71] NS3 Network Simulator, http://www.nsnam.org, Erişim Tarihi: 20.06.2013.

[72] Pdns Network Simulator, http://www.cc.gatech.edu/computing/compass/pdns/, Erişim Tarihi: 20.06.2013.

[73] NetSim Network Simulator, http://www.boson.com/netsim-cisco-network-simulator, Erişim Tarihi: 20.06.2013.

[74] The Georgia Tech Network Simulator (GTNetS) , http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/, Erişim Tarihi: 20.06.2013.

[75] OMNeT++ Network simulator, www.omnetpp.org, Erişim Tarihi: 20.06.2013.

[76] KIM S., SARJOUGHIAN, H., ELAMVAZHUTHI, V., DEVS-Suite: A Simulator Supporting Visual Experimentation Design and Behavior Monitoring, in Proceedings of the Spring Simulation Conference, San Diego, CA, pp. 29–36, 2009.

[77] SUNGUNG K., HESSAM S., VIGNESH E., DEVS-Suite: A Simulator Supporting Visual Experimentation Design and Behavior Monitoring, InProceedings of the 2009 Spring Simulation Multiconference on ZZZ (p. 161). Society for Computer Simulation International. Arizona State University, 2009.

[78] ÇOBANOĞLU B., ZENGİN A., EKİZ H., TUNCEL S., Comparative Study of Network Simulation Tools, International Science and Technology conference, p.70-75, Famagusta, October , 2010.

[79] Defense Advanced Research Projects Agency, www.darpa.mil/, Erişim Tarihi: 13.01.2014

[80] National Science Foundation, http://www.nsf.gov/, Erişim Tarihi: 13.01.2014

[81] The Network Simulator Wiki–Contributed Code, http://nsnam.isi.edu/nsnam/index.php/, 11.02.2011.

[82] ISSARIYAKUL, T., HOSSAIN, E. (2011). Introduction to network simulator NS2. Springer, 2011.

[83] VirtualBox Platform, https://www.virtualbox.org/wiki/Downloads, Erişim Tarihi: 01.12.2010.

[84] Cygwin Platform, www.cygwin.com, Erişim Tarihi: 09.11.2011.

[85] CHUNG, J., CLAYPOOL, M., Ns by example. http://nile.wpi.edu/NS/, Erişim Tarihi: 09.10.2013.

[86] CORSON, S., MACKER, J., Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, IETF, http://www.ietf.org/rfc/rfc2501.txt, 1999.

EKLER

Ek-A. “ilktclkodu_ns.tc” simülasyon kodu

```
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red
$ns color 3 pink
$ns color 4 yellow
$ns color 5 brown

#Open the Trace files
set file1 [open out.tr w]
set winfile [open WinFile w]
$ns trace-all $file1

#Open the NAM trace file
set file2 [open out.nam w]
$ns namtrace-all $file2

#Define a 'finish' procedure
proc finish {} {
        global ns file1 file2
        $ns flush-trace
        close $file1
        close $file2
```

```
        exec nam out.nam &
        exit 0
}

#7 adet düğümün oluşturulması
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]

# Düğümlerin geometrik şekli
$n6 shape circle
$n1 shape hexagon
$n2 shape hexagon
$n3 shape hexagon
$n5 shape hexagon
$n4 shape circle

$n0 color black
$n1 color black
$n2 color black
$n3 color black
$n4 color black
$n5 color black
$n6 color black

#düğümler arası baglantılar
$ns duplex-link $n0 $n1  1Mb 1ms DropTail
$ns duplex-link $n1 $n2  1Mb 4ms DropTail
$ns duplex-link $n2 $n3  1Mb 2ms DropTail
```

```
$ns duplex-link $n3 $n4  1Mb 1ms DropTail
$ns duplex-link $n1 $n3  1Mb 3ms DropTail
$ns duplex-link $n2 $n5  1Mb 2ms DropTail
$ns duplex-link $n0 $n6  1Mb 1ms DropTail
$ns duplex-link $n6 $n5  1Mb 5ms DropTail
$ns duplex-link $n3 $n5  1Mb 4ms DropTail
$ns duplex-link $n4 $n5  1Mb 4ms DropTail

#düğüm yönlerinin verilmesi (for NAM)
$ns duplex-link-op $n0 $n1 orient right-up
$ns duplex-link-op $n5 $n3 orient up
$ns duplex-link-op $n6 $n5 orient right
$ns duplex-link-op $n0 $n6 orient right-down
$ns duplex-link-op $n2 $n5 orient right-down
$ns duplex-link-op $n1 $n3 orient right
$ns duplex-link-op $n3 $n4 orient right
$ns duplex-link-op $n2 $n3 orient right-up
$ns duplex-link-op $n1 $n2 orient right-down

#Set Queue Size of link (n3-n4) to 10
$ns queue-limit $n3 $n4 20

#Kuyruğu görüntüleme
$ns duplex-link-op $n3 $n5 queuePos 0.3"
$ns duplex-link-op $n2 $n1 queuePos 0.3"
$ns duplex-link-op $n1 $n3 queuePos 0.3"

# düğümler arasındaki linklerin renklendirilmesi
$ns duplex-link-op $n0 $n1 color "green"
$ns duplex-link-op $n6 $n5 color "green"
$ns duplex-link-op $n1 $n2 color "green"
$ns duplex-link-op $n3 $n4 color "green"
$ns duplex-link-op $n2 $n3 color "green"
```

```
$ns duplex-link-op $n1 $n3 color "green"
$ns duplex-link-op $n0 $n6 color "green"
$ns duplex-link-op $n3 $n5 color "green"
$ns duplex-link-op $n4 $n5 color "green"
$ns duplex-link-op $n2 $n5 color "green"

#TCP bağlantılarının ayarlanması
#n3-n5
set tcp0 [new Agent/TCP/Newreno]
$ns attach-agent $n3 $tcp0
set sink0 [new Agent/TCPSink/DelAck]
$ns attach-agent $n5 $sink0
$ns connect $tcp0 $sink0
$tcp0 set fid_ 1
$tcp0 set window_ 808
$tcp0 set packetSize_ 55

# FTP nin  TCP baglantısı ile ayarlanması
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ftp0 set type_ FTP

#TCP bağlantılarının ayarlanması
# n0-n2
set tcp1 [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp1
set sink1 [new Agent/TCPSink/DelAck]
$ns attach-agent $n2 $sink1
$ns connect $tcp1 $sink1
$tcp1 set fid_ 2
$tcp1 set window_ 800
$tcp1 set packetSize_ 55
```

```
# FTP nin  TCP baglantısı ile ayarlanması
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ftp1 set type_ FTP

#TCP bağlantılarının ayarlanması
#n2-n3
set tcp2 [new Agent/TCP/Newreno]
$ns attach-agent $n2 $tcp2
set sink2 [new Agent/TCPSink/DelAck]
$ns attach-agent $n3 $sink2
$ns connect $tcp2 $sink2
$tcp2 set fid_ 3
$tcp2 set window_ 800
$tcp2 set packetSize_ 55

#FTP nin  TCP baglantısı ile ayarlanması
set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
$ftp2 set type_ FTP

#TCP bağlantılarının ayarlanması
#n2-n6
set tcp3 [new Agent/TCP/Newreno]
$ns attach-agent $n2 $tcp3
set sink3 [new Agent/TCPSink/DelAck]
$ns attach-agent $n6 $sink3
$ns connect $tcp3 $sink3
$tcp3 set fid_ 4
$tcp3 set window_ 800
$tcp3 set packetSize_ 55

# FTP nin  TCP baglantısı ile ayarlanması
```

```
set ftp3 [new Application/FTP]
$ftp3 attach-agent $tcp3
$ftp3 set type_ FTP

#TCP bağlantılarının ayarlanması
#n2-n4
set tcp4 [new Agent/TCP/Newreno]
$ns attach-agent $n2 $tcp4
set sink4 [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink4
$ns connect $tcp4 $sink4
$tcp4 set fid_ 5
$tcp4 set window_ 800
$tcp4 set packetSize_ 55

# FTP nin  TCP baglantısı ile ayarlanması
set ftp4 [new Application/FTP]
$ftp4 attach-agent $tcp4
$ftp4 set type_ FTP

$ns at 3.0 "$n4 add-mark m3 blue box"

$ns at 0.6 "$ftp0 start"
$ns at 7.5 "$ftp0 stop"
$ns at 0.7 "$ftp1 start"
$ns at 8.0 "$ftp1 stop"
$ns at 0.8 "$ftp2 start"
$ns at 8.5 "$ftp2 stop"
$ns at 0.9 "$ftp3 start"
$ns at 9.0 "$ftp3 stop"
$ns at 1.0 "$ftp4 start"
$ns at 9.5 "$ftp4 stop"
```

```
$ns at 10.0 "finish"
$ns run
```

Ek-B. Trafik dosyalarının oluşturulması kodları

```
ns cbrgen.tcl -type tcp -nn10 -seed 1.0 -mc 2 -rate 4.0 > T10
ns cbrgen.tcl -type tcp -nn50 -seed 1.0 -mc 2 -rate 4.0 > T50
ns cbrgen.tcl -type tcp -nn50 -seed 1.0 -mc 2 -rate 4.0 > T100
ns cbrgen.tcl -type tcp -nn50 -seed 1.0 -mc 2 -rate 4.0 > T200
```

10 düğümlü ağlar için trafik dosyası(T10).

```
set tcp_(0) [$ns_ create-connection  TCP $node_(0) TCPSink $node_(1) 0]
$tcp_(0) set window_ 32
$tcp_(0) set packetSize_ 512
set ftp_(0) [$tcp_(0) attach-source FTP]
$ns_ at 1 "$ftp_(0) start"
set tcp_(1) [$ns_ create-connection  TCP $node_(4) TCPSink $node_(5) 0]
$tcp_(1) set window_ 32
$tcp_(1) set packetSize_ 512
set ftp_(1) [$tcp_(1) attach-source FTP]
$ns_ at 1 "$ftp_(1) start"
#Total sources/connections: 2/2
```

50 düğümlü ağlar için trafik dosyası(T50).

```
set tcp_(0) [$ns_ create-connection  TCP $node_(0) TCPSink $node_(1) 0]
$tcp_(0) set window_ 32
$tcp_(0) set packetSize_ 512
set ftp_(0) [$tcp_(0) attach-source FTP]
$ns_ at 1 "$ftp_(0) start"
set tcp_(1) [$ns_ create-connection  TCP $node_(4) TCPSink $node_(5) 0]
$tcp_(1) set window_ 32
$tcp_(1) set packetSize_ 512
```

```
set ftp_(1) [$tcp_(1) attach-source FTP]
$ns_ at 1 "$ftp_(1) start"
#Total sources/connections: 2/2
```

100 düğümlü ağlar için trafik dosyası(T100).

```
set tcp_(0) [$ns_ create-connection  TCP $node_(90) TCPSink $node_(2) 0]
$tcp_(0) set window_ 32
$tcp_(0) set packetSize_ 512
set ftp_(0) [$tcp_(0) attach-source FTP]
$ns_ at 1 "$ftp_(0) start"

set tcp_(1) [$ns_ create-connection  TCP $node_(80) TCPSink $node_(5) 0]
$tcp_(1) set window_ 32
$tcp_(1) set packetSize_ 512
set ftp_(1) [$tcp_(1) attach-source FTP]
$ns_ at 1 "$ftp_(1) start"

set tcp_(2) [$ns_ create-connection  TCP $node_(40) TCPSink $node_(1) 0]
$tcp_(2) set window_ 32
$tcp_(2) set packetSize_ 512
set ftp_(2) [$tcp_(2) attach-source FTP]
$ns_ at 1 "$ftp_(2) start"

set tcp_(3) [$ns_ create-connection  TCP $node_(60) TCPSink $node_(10) 0]
$tcp_(3) set window_ 32
$tcp_(3) set packetSize_ 512
set ftp_(3) [$tcp_(3) attach-source FTP]
$ns_ at 1 "$ftp_(3) start"
```

200 düğümlü ağlar için trafik dosyası(T200).

```
set tcp_(0) [$ns_ create-connection  TCP $node_(190) TCPSink $node_(90) 0]
$tcp_(0) set window_ 32
```

```
$tcp_(0) set packetSize_ 512
set ftp_(0) [$tcp_(0) attach-source FTP]
$ns_ at 1 "$ftp_(0) start"

set tcp_(1) [$ns_ create-connection  TCP $node_(150) TCPSink $node_(50) 0]
$tcp_(1) set window_ 32
$tcp_(1) set packetSize_ 512
set ftp_(1) [$tcp_(1) attach-source FTP]
$ns_ at 1 "$ftp_(1) start"

set tcp_(2) [$ns_ create-connection  TCP $node_(100) TCPSink $node_(90) 0]
$tcp_(2) set window_ 32
$tcp_(2) set packetSize_ 512
set ftp_(2) [$tcp_(2) attach-source FTP]
$ns_ at 1 "$ftp_(2) start"

set tcp_(3) [$ns_ create-connection  TCP $node_(90) TCPSink $node_(50) 0]
$tcp_(3) set window_ 32
$tcp_(3) set packetSize_ 512
set ftp_(3) [$tcp_(3) attach-source FTP]
$ns_ at 1 "$ftp_(3) start"
```

Ek C. Senaryo dosyalarının oluşturulması kodları

```
./setdest -n 10 -p 2.0 -s 1 -t 50 -x 300 -y 300 > S10M1

./setdest -n 10 -p 2.0 -s 5 -t 50 -x 300 -y 300 > S10M5

./setdest -n 10 -p 2.0 -s 10 -t 50 -x 300 -y 300 > S10M10

./setdest -n 10 -p 2.0 -s 20 -t 50 -x 300 -y 300 > S10M20

./setdest -n 50 -p 2.0 -s 1 -t 50 -x 500 -y 500 > S50M1

./setdest -n 50 -p 2.0 -s 5 -t 50 -x 500 -y 500 > S50M5

./setdest -n 50 -p 2.0 -s 10 -t 50 -x 500 -y 500 > S50M10
```

./setdest -n 50 -p 2.0 -s 20 -t 50 -x 500 -y 500 > S50M20

./setdest -n 100 -p 2.0 -s 1 -t 50 -x 700 -y 700 > S100M1

./setdest -n 100 -p 2.0 -s 5 -t 50 -x 700 -y 700 > S100M5

./setdest -n 100 -p 2.0 -s 10 -t 50 -x 700 -y 700 > S100M10

./setdest -n 100 -p 2.0 -s 20 -t 50 -x 700 -y 700 > S100M20

./setdest -n 200 -p 2.0 -s 1 -t 50 -x 1500 -y 1500 > S200M1

./setdest -n 200 -p 2.0 -s 5 -t 50 -x 1500 -y 1500 > S200M5

./setdest -n 200 -p 2.0 -s 10 -t 50 -x 1500 -y 1500 > S200M10

./setdest -n 200 -p 2.0 -s 20 -t 50 -x 1500 -y 1500 > S200M20

10 düğümlü ağ için senaryo dosyası (S10M1).

nodes: 10, pause: 0.00, max speed: 1.00, max x: 300.00, max y: 300.00
$node_(0) set X_ 180.737714677454
$node_(0) set Y_ 74.041823321185
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 14.119539744422
$node_(1) set Y_ 169.960464432784
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 13.156213712993
$node_(2) set Y_ 215.137915045183
$node_(2) set Z_ 0.000000000000
$node_(3) set X_ 280.588141843289
$node_(3) set Y_ 136.606290148131
$node_(3) set Z_ 0.000000000000
$node_(4) set X_ 81.646704879244
$node_(4) set Y_ 227.057879843758
$node_(4) set Z_ 0.000000000000
$node_(5) set X_ 121.703070149347
$node_(5) set Y_ 176.484159219243

$node_(5) set Z_ 0.000000000000
$node_(6) set X_ 25.757886762181
$node_(6) set Y_ 37.372329848224
$node_(6) set Z_ 0.000000000000
$node_(7) set X_ 23.123412348719
$node_(7) set Y_ 46.325846211001
$node_(7) set Z_ 0.000000000000
$node_(8) set X_ 274.907261920327
$node_(8) set Y_ 291.321450866494
$node_(8) set Z_ 0.000000000000
$node_(9) set X_ 282.025140980545
$node_(9) set Y_ 103.935697259993
$node_(9) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(0) setdest 181.228313081779 154.416544646640 0.412748182391"
$ns_ at 0.000000000000 "$node_(1) setdest 138.565180548829 297.868866160681 0.691256292812"
$ns_ at 0.000000000000 "$node_(2) setdest 196.187616048154 90.626458625845 0.670529546093"
$ns_ at 0.000000000000 "$node_(3) setdest 282.313164840717 269.508481597939 0.152411538805"
$ns_ at 0.000000000000 "$node_(4) setdest 283.636438938877 9.630346624905 0.060482725869"
$ns_ at 0.000000000000 "$node_(5) setdest 37.999612233583 277.196517925913 0.344138031262"
$ns_ at 0.000000000000 "$node_(6) setdest 295.437210763558 107.123220987066 0.524961354721"
$ns_ at 0.000000000000 "$node_(7) setdest 222.603144031357 168.358957864964 0.578717836499"
$ns_ at 0.000000000000 "$node_(8) setdest 132.387370950676 112.877596583809 0.489267886097"
$ns_ at 0.000000000000 "$node_(9) setdest 224.985614557985 50.892039338486 0.389977261451"

$god_ set-dist 0 1 1
$god_ set-dist 0 2 1
$god_ set-dist 0 3 1
$god_ set-dist 0 4 1
$god_ set-dist 0 5 1
$god_ set-dist 0 6 1
$god_ set-dist 0 7 1
$god_ set-dist 0 8 1
$god_ set-dist 0 9 1
$god_ set-dist 1 2 1
$god_ set-dist 1 3 2
$god_ set-dist 1 4 1
$god_ set-dist 1 5 1
$god_ set-dist 1 6 1
$god_ set-dist 1 7 1
$god_ set-dist 1 8 2
$god_ set-dist 1 9 2
$god_ set-dist 2 3 2
$god_ set-dist 2 4 1
$god_ set-dist 2 5 1
$god_ set-dist 2 6 1
$god_ set-dist 2 7 1
$god_ set-dist 2 8 2
$god_ set-dist 2 9 2
$god_ set-dist 3 4 1
$god_ set-dist 3 5 1
$god_ set-dist 3 6 2
$god_ set-dist 3 7 2
$god_ set-dist 3 8 1
$god_ set-dist 3 9 1
$god_ set-dist 4 5 1
$god_ set-dist 4 6 1
$god_ set-dist 4 7 1

```
$god_ set-dist 4 8 1
$god_ set-dist 4 9 1
$god_ set-dist 5 6 1
$god_ set-dist 5 7 1
$god_ set-dist 5 8 1
$god_ set-dist 5 9 1
$god_ set-dist 6 7 1
$god_ set-dist 6 8 2
$god_ set-dist 6 9 2
$god_ set-dist 7 8 2
$god_ set-dist 7 9 2
$god_ set-dist 8 9 1
$ns_ at 17.037324597869 "$god_ set-dist 6 9 1"
$ns_ at 17.335099310243 "$god_ set-dist 7 9 1"
$ns_ at 27.445965621208 "$god_ set-dist 2 8 1"
$ns_ at 35.212100499787 "$god_ set-dist 1 8 1"
$ns_ at 42.963418762472 "$god_ set-dist 2 3 1"
$ns_ at 44.207028851710 "$god_ set-dist 1 3 1"
$ns_ at 44.472278329552 "$god_ set-dist 3 7 1"
$ns_ at 50.311836008340 "$god_ set-dist 2 9 1"
$ns_ at 51.028992737098 "$god_ set-dist 3 6 1"
$ns_ at 55.400447730001 "$god_ set-dist 1 9 1"
$ns_ at 96.576110136800 "$god_ set-dist 7 8 1"
# Destination Unreachables: 0
# Route Changes: 11
# Link Changes: 11
# Node | Route Changes | Link Changes
#    0 |             0 |            0
#    1 |             3 |            3
#    2 |             3 |            3
#    3 |             4 |            4
#    4 |             0 |            0
#    5 |             0 |            0
```

```
#    6 |         2 |         2
#    7 |         3 |         3
#    8 |         3 |         3
#    9 |         4 |         4
```

Ek-D. 10 düğüm ve düğüm hareketi 1 m/s olan tcl kodu

```
set val(chan)          Channel/WirelessChannel    ;# kanal tipi
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy            ;# network interface type
set val(mac)           Mac/802_11                 ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)            LL                         ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                         ;# max packet in ifq
set val(nn)            10                         ;# number of mobilenodes
set val(rp)            Bee-MANET                  ;# routing protocol
set val(cp)            "T10"
set val(sc)            "S10M1"

# Main Program
set ns_ [new Simulator]
set tracefd     [open 10Bee-MANET.tr w]
set winfile [open WinFile w]
$ns_ trace-all $tracefd
set namtrace   [open 10Bee-MANET.nam w]
$ns_ namtrace-all-wireless $namtrace 300 300
```

```
set topo [new Topography]
$topo load_flatgrid 300 300
set god_ [create-god $val(nn)]
$ns_ node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
                 -ifqLen $val(ifqlen) \
                 -antType $val(ant) \
                 -propType $val(prop) \
                 -phyType $val(netif) \
                 -channelType $val(chan) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
                 -macTrace OFF \
                 -movementTrace ON
for {set i 0} {$i < $val(nn) } {incr i} {
        set node_($i) [$ns_ node]
}
puts "Loading connection pattern..."
source $val(cp)
puts "Loading scenario file..."
source $val(sc)
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
```

```
    $ns_ at 50.0 "$node_($i) reset";
}
$ns_ at 50.0 "stop"
$ns_ at 50.01 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
        exec nam 10Bee-MANET.nam &
       exit 0
}
puts "Simülasyon başlıyor..."
$ns_ run
```

Ek-E. 50 düğüm ve düğüm hareketi 5 m/s olan tcl kodu

```
set val(chan)          Channel/WirelessChannel    ;# kanal tipi
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy            ;# network interface type
set val(mac)           Mac/802_11                 ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)            LL                         ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                         ;# max packet in ifq
set val(nn)            50                         ;# number of mobilenodes
set val(rp)            Bee-MANET                  ;# routing protocol
set val(cp)            "T50"
```

```
set val(sc)             "S50"


# Main Program
set ns_ [new Simulator]
set tracefd     [open 50Bee-MANET.tr w]
set winfile [open WinFile w]
$ns_ trace-all $tracefd
set namtrace   [open 50Bee-MANET.nam w]
$ns_ namtrace-all-wireless $namtrace 500 500
set topo [new Topography]
$topo load_flatgrid 500 500
set god_ [create-god $val(nn)]

$ns_ node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
                 -ifqLen $val(ifqlen) \
                 -antType $val(ant) \
                 -propType $val(prop) \
                 -phyType $val(netif) \
                 -channelType $val(chan) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
                 -macTrace OFF \
```

```
                    -movementTrace ON
for {set i 0} {$i < $val(nn) } {incr i} {
            set node_($i) [$ns_ node]
}
puts "Loading connection pattern..."
source $val(cp)
puts "Loading scenario file..."
source $val(sc)
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
   $ns_ at 50.0 "$node_($i) reset";
}
$ns_ at 50.0 "stop"
$ns_ at 50.01 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
   global ns_ tracefd
   $ns_ flush-trace
   close $tracefd
       exec nam 50Bee-MANET.nam &
      exit 0
}
puts "Simülasyon başlıyor..."
$ns_ run
```

Ek-F. 100 düğüm ve düğüm hareketi 10 m/s olan tcl kodu

```
set val(chan)          Channel/WirelessChannel    ;# kanal tipi
```

```
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy            ;# network interface type
set val(mac)           Mac/802_11                 ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)            LL                         ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                         ;# max packet in ifq
set val(nn)            100                        ;# number of mobilenodes
set val(rp)            Bee-MANET                  ;# routing protocol
set val(cp)            "T100"
set val(sc)            "S100m5"

# Main Program
set ns_ [new Simulator]
set tracefd     [open 100Bee-MANET.tr w]
set winfile [open WinFile w]
$ns_ trace-all $tracefd
set namtrace   [open 100Bee-MANET.nam w]
$ns_ namtrace-all-wireless $namtrace 700 700
set topo [new Topography]
$topo load_flatgrid 700 700
set god_ [create-god $val(nn)]
$ns_ node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
```

```
                -ifqLen $val(ifqlen) \
                -antType $val(ant) \
                -propType $val(prop) \
                -phyType $val(netif) \
                -channelType $val(chan) \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON \
                -macTrace OFF \
                -movementTrace ON

for {set i 0} {$i < $val(nn) } {incr i} {
        set node_($i) [$ns_ node]
}
puts "Loading connection pattern..."
source $val(cp)
puts "Loading scenario file..."
source $val(sc)
for {set i 0} {$i < $val(nn) } {incr i} {
   $ns_ at 50.0 "$node_($i) reset";
}
$ns_ at 50.0 "stop"
$ns_ at 50.01 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
   global ns_ tracefd
   $ns_ flush-trace
```

```
    close $tracefd
        exec nam 100Bee-MANET.nam &
       exit 0
}
puts "Simülasyon başlıyor..."
$ns_ run
```

Ek-G. 200 düğüm ve düğüm hareketi 20 m/s olan tcl kodu

```
set val(chan)          Channel/WirelessChannel    ;# kanal tipi
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy            ;# network interface type
set val(mac)           Mac/802_11                 ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)            LL                         ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                         ;# max packet in ifq
set val(nn)            200                        ;# number of mobilenodes
set val(rp)            Bee-MANET                  ;# routing protocol
set val(cp)            "T200"
set val(sc)            "S200"

# Main Program
set ns_ [new Simulator]
set tracefd     [open 200Bee-MANET.tr w]
set winfile [open WinFile w]
```

```
$ns_ trace-all $tracefd
set namtrace   [open 200Bee-MANET.nam w]
$ns_ namtrace-all-wireless $namtrace 1500 1500
set topo [new Topography]
$topo load_flatgrid 1500 1500
set god_ [create-god $val(nn)]
$ns_ node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
                 -ifqLen $val(ifqlen) \
                 -antType $val(ant) \
                 -propType $val(prop) \
                 -phyType $val(netif) \
                 -channelType $val(chan) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
                 -macTrace OFF \
                 -movementTrace ON

for {set i 0} {$i < $val(nn) } {incr i} {
        set node_($i) [$ns_ node]
}
puts "Loading connection pattern..."
source $val(cp)
```

```
puts "Loading scenario file..."
source $val(sc)
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at 50.0 "$node_($i) reset";
}
$ns_ at 50.0 "stop"
$ns_ at 50.01 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
        exec nam 200Bee-MANET.nam &
       exit 0
}
puts "Simülasyon başlıyor..."
$ns_ run
```

ÖZGEÇMİŞ

Zafer ALBAYRAK, 09.12.1975 yılında Sakarya' da doğdu. İlk, orta ve lise eğitimini Sakarya'da tamamladı. 1993 yılında Sakarya Fatih Teknik Lisesi, Elektrik Bölümünden mezun oldu. 1995 yılında başladığı Marmara Üniversitesi Elektrik Eğitimi bölümünü 1999 yılında bitirdi. 2001 yılında L.M.U. (London Metropolitan University) Üniversitesi, MSc (Computer Science) yüksek lisans programına başlayıp 2003 yılında mezun oldu. 2004 – 2006 yılları arasında Sakarya Üniversitesi Hendek Meslek Yüksekokulunda misafir öğretim elemanı olarak dersler verdi. 2006 yılından itibaren Gaziosmanpaşa Üniversitesi Niksar Teknik Bilimler Meslek Yüksekokulu Bilgisayar Teknolojileri Bölümünde Öğretim Görevlisi olarak görev yapmaktadır. Evli ve 3 çocuk babasıdır.