
GAZİ ÜNİVERSİTESİ
Gazi
1926

ARAÇTAN ARACA TASARSIZ AĞ UYGULAMASI

İbrahim KÖK

YÜKSEK LİSANS TEZİ
BİLGİSAYAR BİLİMLERİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

OCAK 2015

İbrahim KÖK tarafından hazırlanan "ARAÇTAN ARACA TASARSIZ AĞ UYGULAMASI" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ / OY ÇOKLUĞU ile Gazi Üniversitesi Bilgisayar Bilimleri Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. M. Ali AKCAYOL

Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/onaylamıyorum ..……

Başkan : Doç. Dr. Erdoğan DOĞDU

Bilgisayar Mühendisliği Anabilim Dalı, TOBB Ekonomi ve Teknoloji Üniversitesi ……...

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/onaylamıyorum

Üye : Yrd. Doç. Dr. Hacer KARACAN

Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum/onaylamıyorum ……...

Tez Savunma Tarihi: 29/01/2015

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

…….……

Doç. Dr. Nurettin TOPALOĞLU

Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

İbrahim KÖK

29.01.2015

ARAÇTAN ARACA TASARSIZ AĞ UYGULAMASI

(Yüksek Lisans Tezi)

İbrahim KÖK

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Ocak 2015

ÖZET

Son yıllarda yaşanan teknolojik gelişmelerin neticesinde araçsal tasarsız ağlar konusunda yapılan çalışmalar ve uygulamalar hızla artmıştır. Bu ağların uygulama alanlarındaki farklılıklara bağlı olarak servis kalitesi ihtiyaçları da farklılaşmaktadır. Bu tez çalışmasında araçsal tasarsız ağ bileşenleri, mimarisi ve uygulamaları detaylı olarak incelenmiştir. Araçtan araca uygulamalar için farklı öncelikte paket üreten dört trafik kaynağı oluşturulmuştur. Bu kaynaklar kullanılarak araçsal tasarsız ağlar için yeni bir öncelik kuyruğu yapısı (VanetPriQ) geliştirilmiştir. VanetPriQ performansının değerlendirilmesi için benzetim ortamında kuyruk yapısı, paket boyutları, kuyruk uzunluğu ve araç hızı gibi parametreler değiştirilerek farklı senaryolar oluşturulmuştur. Bu senaryolar Ns-2 simülasyon aracı kullanılarak test edilmiştir. Simülasyon sonuçları öncelik kuyruğu yapısının yüksek öncelikli paketlerin hedefe ulaşma ihtimalini artırdığını ve uçtan uca gecikme süresini azalttığını göstermiştir.

Bilim Kodu : 902.1.063

Anahtar Kelimeler : Araçsal tasarsız ağlar, paket önceliklendirme, kuyruk yapısı, servis kalitesi

Sayfa Adedi : 83

Danışman : Prof. Dr. M. Ali AKCAYOL

VEHICLE TO VEHICLE AD HOC NETWORK APPLICATION

(M. Sc. Thesis)

İbrahim KÖK

GAZİ UNIVERSITY

INFORMATICS INSTITUTE

January 2015

ABSTRACT

Studies and applications in vehicular ad hoc networks have been increased related to recent technological advances. Quality of service requirements, depending on differences in the application area of these networks, are also differentiates. In this thesis, vehicular ad hoc network components, architecture and applications are observed in detail. Four traffic sources which generates packet with different priority have been created for vehicle to vehicle applications. A novel priority queue structure (VanetPriQ) has been developed for vehicular ad hoc networks by using these traffic sources. In simulation environment, different scenarios have been created for evaluating the performance of VanetPriQ by changing parameters such as queue structure, packet size, queue lenght and vehicle speed. These scenarios has been tested by using Ns-2 simulation tool. Simulation results showed that priority queue structure has been increased possibility of reaching to destination of high priority packests and decreased end to end delay time.

Science Code : 902.1.063

Key Words : Vehicular Ad Hoc network, packet prioritization, queue structure, quality of service

Page Number : 83

Supervisor : Prof. Dr. M. Ali AKCAYOL

TEŞEKKÜR

Yüksek lisans eğitimimin ve bu tez çalışmasının her aşamasında bilgi, öneri, yardım ve katkılarıyla beni yönlendiren tez danışmanım Prof. Dr. M. Ali AKCAYOL'a teşekkürü bir borç bilirim.

Ayrıca manevi destekleriyle beni hiçbir zaman yalnız bırakmayan aileme sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
Byte	Bayt
MHz	Megahertz
s	Saniye
ms	Milisaniye
m/s	Metre/saniye

Kısaltmalar	**Açıklamalar**
AODV	On-Demand Distance Vector Routing (İsteğe Bağlı Mesafe Vektörü Yönlendirme)
VANET	Vehicular Ad Hoc Network (Araçsal Tasarsız Ağlar)
AU	Application Unit (Uygulama Ünitesi)
CBR	Constant Bit Rate (Sabit Bit Oranı)
CBQ	Class Based Queueing (Sınıf Tabanlı Kuyruklama)
DMP	Dynamic Multilevel Priority (Dinamik Düzeyli Öncelik)
DRR	Dynamic Round Robin (Dinamik Sıralı)
DSR	Dynamic Source Routing (Dinamik Kaynak Yönlendirme)
DSRC	Dedicated Short Range Communications **(**Tahsisli Kısa Mesafeli Haberleşme)
FIFO	First In First Out (İlk Gelen İlk Çıkar)
FTP	File Transfer Protocol (Dosya Transfer Protokolü)
IEEE	The Institute of Electrical and Electronic Engineers
NAM	Network Animator
OBU	On Board Unit (Araç Üstü Ünite)

Kısaltmalar	Açıklamalar
OTCL	Object Oriented Tool Command Language (Nesne Yönelimli Aracı Komut Dili)
RED	Random Early Detection (Rastgele Erken Algılama)
RCP	Resource Command Processor (Kaynak Komut İşlemcisi)
RSU	Road Side Unit (Yol Kenarı Ünitesi)
TORA	Temporally Ordered Routing Algorithm (Geçici Sıralı Yönlendirme Algoritması)
TCL	Tool Command Language (Araç Komut Dili)
TCP	Transmission Control Protocol (İletim Kontrol Protokolü)
UDP	User Datagram Protocol (Kullanıcı Datagram Protokolü)
VanetPriQ	Vehicular Ad Hoc Network Priority Queue (Araçsal Tasarsız Ağ Önceliklendirilmiş Kuyruk)
V2I	Vehicle To Infrastructure (Araçtan Altyapıya)
V2V	Vehicle To Vehicle (Araçtan Araca)
WAVE	Wireless Access in Vehicular Environments (Araç Ortamlarına Kablosuz Erişim)

1. GİRİŞ

Donanım, yazılım ve iletişim teknolojilerindeki son gelişmeler farklı ortamlarda uygulanabilen farklı ağ türlerinin tasarımını ve uygulamasını zorunlu hale getirmiştir. Günümüzde, tüm uygulamalar kablosuz iletişim yapmaya başlamış; kablosuz teknolojiler hareketlilik, erişilebilirlik ve esneklik sayesinde her türlü bilgi aktarımında kullanılır hale gelmiştir. Herkes farklı aygıt türleri ve iletişim teknolojileri kullanarak hareketli ortamda dahi bilgiye erişebilmektedir.

Her yıl trafikteki araç sayısının hızla artıyor olması ve bununla birlikte dünyada her gün milyonlarca insanın trafik kazalarında yaralanması veya hayatını kaybetmesi giderek önemli bir problem haline gelmiştir. Bu gelişmeler araçsal tasarsız ağ (Vehicular Ad Hoc Network-VANET) olarak isimlendirilen yeni bir ağ türünün doğmasına yol açmıştır ve araçsal tasarsız ağların uygulamarına yönelik çalışmalar artmıştır. Bu ağ teknolojisi ile yapılan ilk çalışmalar ulaşım sistemlerinde güvenliğe yönelik olsa da iş, eğlence, sürüş yardımı ve kamu hizmetleri gibi alanlarda da kullanılabilir hale gelmiştir [1-6].

Uygulama alanları genişleyen VANET ağlarda oluşan trafik tipi büyük farklılıklar göstermekte ve her bir trafik tipi, bant genişliği, gecikme, gecikme sürelerindeki değişim ve bulunabilirlik faktörleri açısından kendisine özgü gereksinimlere sahip olmaktadır [7].

Literatürde VANET ağların uygulama alanlarını ve etkinliğini artırmak için çok sayıda çalışma bulunmaktadır. Kumar ve Varshini yapmış oldukları çalışmada, kablosuz ağlar için dinamik çok seviyeli kuyruk sıralama şeması geliştirmişlerdir. Bu şemada üç ayrı kuyruk kullanılmıştır. En yüksek önceliğe sahip gerçek zamanlı veri paketleri birinci kuyruğa, gerçek zamanlı olmayan veri paketleri ise diğer iki kuyruğa yerleştirilmiştir. Önerilen dinamik çok seviyeli önceliklendirme (Dynamic Multilevel Priority-DMP) şeması dinamik sıralı (Dynamic Round Robin-DRR) şemasıyla karşılaştırılmıştır. Birinci seviye kuyruktaki verilerde uçtan uca gecikme ve bekleme süreleri DRR'ye göre daha iyi performans göstermiştir. Ancak ortalama görev bekleme süresinde DRR'nin daha iyi performansa sahip olduğu belirtilmiştir [8].

Xu, Wang ve Toh yaptıkları çalışmada, mobil tasarsız ağlarda sıralama algoritmalarının karşılaştırmalı analizini yapmışlar. Bu çalışmada, tek kuyruk modeli, öncelikli kuyruk modeli, dinamik öncelikli kuyruk modeli ve adaptif kuyruk modelleri kullanılmıştır. Uçtan uca ve bir düğümden diğer düğüme performans analizleri yaparak servis kalitesi parametrelerinin etkisi incelenmiştir. Ortam kanalındaki paket trafiği az olduğu durumda tüm algoritmalar benzer performans göstermiştir. Ancak, trafiğin yoğun olduğu durumda öncelik kuyruğu, adaptif öncelik kuyruğu ve dinamik öncelik kuyruğu daha iyi performans ortaya koymuştur [9].

Chun ve Baker yaptıkları çalışmada, mobil tasarsız ağlarda kuyruk dinamiklerini ve farklı sıralama algoritmalarının ağ performansına etkisini DSR ve GPRS yönlendirme protokollerini kullanarak analiz etmişlerdir. Kontrol paketlerini önceliklendirerek yönlendirme protokollerinin buna etkisini ölçmüşlerdir. DSR ile önceliklendirilmiş kontrol paketleri ortalama gecikmeyi azaltmış ve nadiren iletişim hızını etkilemiştir. GPSR ile ortalama gecikmenin arttığı gözlemlenmiştir. Ayrıca buffer boyutunun performansa etkisi de araştırılmış ve büyük buffer hacminin ortalama gecikmeyi artırdığı görülmüştür [10].

Metin ve Deliç yaptıkları çalışmada, ağ cihazlarında kullanılacak kuyruklama yöntemlerinin servis kalitesine etkisini incelemişlerdir. Bu amaçla rastgele erken algılama (Random Early Detection-RED), sınıf tabanlı kuyruk(Class Based Queueing-CBQ), ilk gelen ilk çıkar (First In First Out-FIFO), ağırlıklandırılmış adil kuyruklama (Weighted Fair Queueing-WFQ) kuyruk yönetimi algoritmaları ile benzetimler yapmışlardır. Sonuç olarak, ağ üzerinde trafik yoğunluğu %90'ın altında ise tüm kuyruklama yöntemlerinin aynı sonucu verdiği, onun dışında CBQ veya WFQ kullanımının verimliliği artırdığı belirtilmiştir. Ayrıca, ağ üzerindeki paket boyutunu düşürmenin ortalama gecikme değerini düşürdüğü ifade edilmiştir [11].

Qian, Lu ve Moayeri yaptıkları çalışmada, araçsal tasarsız ağlarda güvenli bir ortam erişim (Medium Access Control-MAC) protokolü tasarlamayı amaçlamışlardır. Farklı uygulamalar için dört farklı öncelik sınıfı oluşturulmuştur. Mesaj paketleri öncelik sınıflarına göre tahsisli kısa mesafeli haberleşme (Dedicated Short Range Communication-DSRC) kanalları üzerinden iletilmiştir. Simülasyon tasarımı için bu öncelik sınıflarına göre dört farklı kuyruk kullanılmıştır. Araçsal ağ uygulamalarında servis kalitesinin güvenlikle ilgili gereksinimlerini karşılayan bir protokol tasarımı yapılmıştır [12].

Mi, Liu, Xu ve Li yaptıkları çalışmada, araçsal tasarsız ağlarda gerçek zamanlı mesajlar için öncelik kuyruğu algoritması önermişlerdir. Önerilen algoritma FIFO kuyruk şemasıyla karşılaştırılarak ortalama uçtan uca gecikme ve paket kayıp oranları analiz edilmiştir. Öncelikli kuyruk yapısının hem paket kayıp oranını hem de ortalama uçtan uca gecikme süresini azalttığı belirtilmiştir [13].

Mu'azu ve arkadaşları tarafından yapılan çalışmada, araçsal tasarsız ağlar için gerçek zamanlı uygulama mesajları ile öncelikli data akış şeması önermişlerdir. Çalışmada, uygulama türlerine göre trafik kazası için yüksek, çarpışma uyarısı için normal ve diğer uygulamalar için düşük öncelik kullanılmıştır. Paket trafiği tek kuyruk üzerinde önceliklerine göre tutulmuş ve yüksek öncelikli paket iletimi DSRC'nin kritik ve güvenlik için kullanılan 172. kanalından, diğerleri ise servis kanallarından iletilmişlerdir. AODV yönlendirme protokolü ile gerçekleştirilen uygulama sonucunda hem paket dağıtımı hem de ortalama iletim hızında önceliklendirilmiş trafik akışının daha iyi performans gösterdiği belirtilmiştir [14].

Araçsal ağ uygulamalarının hızla gelişmesi, ağ trafik tiplerindeki çeşitlilik ve kaynaklardaki kısıtlılık sebebiyle servis kalitesi ve ağ yönetimi konularını daha önemli hale getirmiştir. Uygulama alanlarındaki farklılıklara göre ihtiyaç duyulan servis kalitesi (Quality of Service-QoS) de farklılaşmaktadır.

Bu çalışmada, kapsamında araçsal tasarsız ağ uygulamaları incelenmiş ve araçtan araca olan uygulama paketlerinin önceliklendirilmesi ve önceliklendirilen paketlerin hedefe iletiminde öncelik durumunu baz alan öncelikli kuyruk yapısı geliştirilmiştir.

Araçtan araca olan tüm uygulamalar önceliğine göre dört gruba ayrılmıştır. Her öncelik grubuna göre ayrı paket tanımlamaları ve paket trafiği oluşturulmuştur. Paketler geliştirilen kuyruk yapısında önceliğine göre kuyrukta sıralanmıştır. Geliştirilen kuyruk yapısı Drop tail kuyruk yapısı ile karşılaştırılarak deneysel sonuçlar verilmiştir.

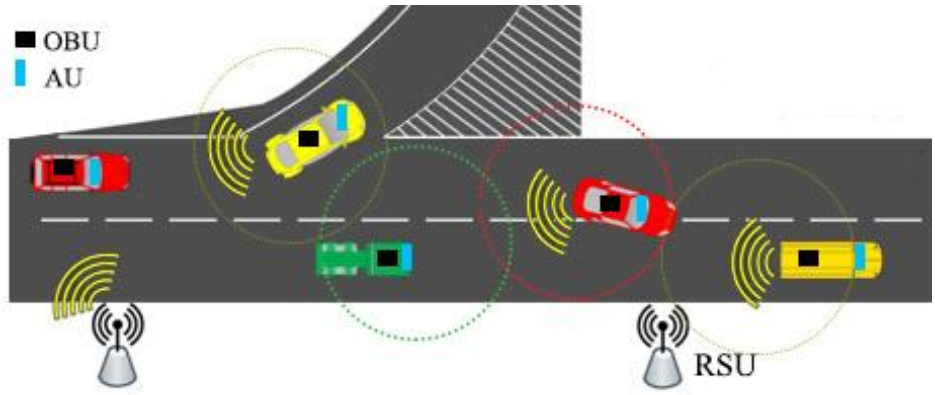
Bu tez altı bölümden oluşmaktadır. Birinci bölümde giriş ve literatürdeki çalışmalar sunulmuştur. İkinci bölümde araçsal tasarsız ağ mimarisi ve bileşenleri, karakteristikleri, iletişim teknolojileri, gereksinimleri, zorlukları, uygulamaları ve simülatörleri hakkında bilgi verilmiştir. Üçüncü bölümde ise uygulama için kullanılacak sistem ve benzetim ortamları ile yapının uygulanma aşamaları ve simülasyon senaryolarına yer verilmiştir. dördüncü bölümde ise senaryolara göre uygulama çıktıları ve analizleri yer almaktadır. Son olarak beşinci bölümde sonuçlar sunulmuştur.

2. ARAÇSAL TASARSIZ AĞLAR

Mobil tasarsız ağlar sabit altyapı gerektirmeyen ortam koşullarına hızlı ve ön hazırlıksız olarak kendini adapte ederek uç birimler arasında iletimi sağlayan kablosuz ağ türüdür. Araçsal tasarsız ağlar mobil tasarsız ağların bir uygulaması olarak sınıflandırmaktadır [15]. Araçsal tasarsız ağlar, mobil araçlar arasında ve araç ile yol kenarı birimi arasında oluşturulan ağlardır [2,16]. Araçsal tasarsız ağ mimarisi, zorlukları, karakteristikleri ve uygulama alanları bakımından mobil tasarsız ağlardan ve diğer ağ türlerinden farklılaşmaktadır.

2.1 Araçsal Tasarsız Ağ Bileşenleri ve Mimarisi

Araçsal tasarsız ağlar için sistem bileşenleri, yol kenarı ünitesi (Road Side Unit-RSU), araç üstü ünitesi (On Board Unit-OBU) ve uygulama ünitesi (Application Unit-AU) olmak üzere Şekil 2.1'de görüldüğü gibi üç çeşittir [15].



Şekil 2.1. Araçsal tasarsız ağ bileşenleri

2.1.1. Yol kenarı ünitesi

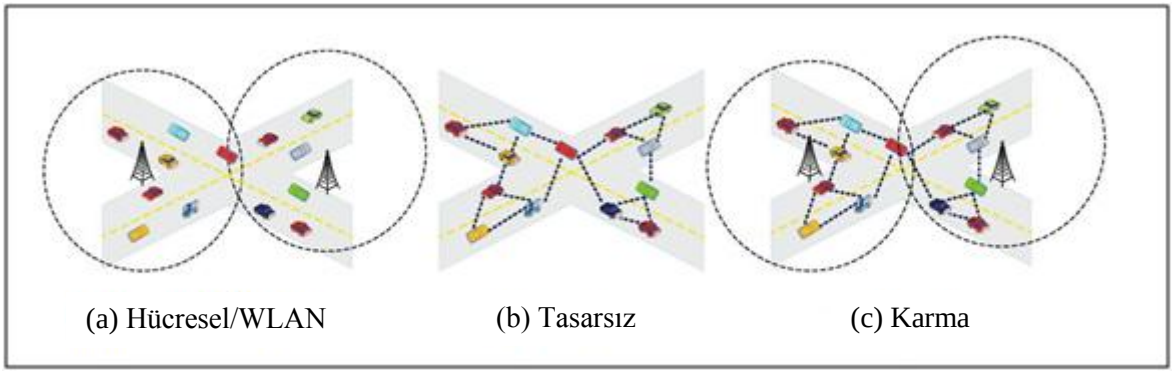
Yol kenarı ünitesi, yol kenarına, park ve kavşak yerlerine yerleştirilmiş sinyal alma ve gönderme birimidir. Temel fonksiyonu ise bilginin yeniden dağıtımını sağlayarak tasarsız ağ iletişim sahasını genişletmek, düşük köprü, kaza uyarı ya da çalışma alanı gibi güvenlik uygulamalarını işletmek, araçlar ile iletişim sağlamak, bilgi sağlayıcı olarak hareket etmek ve araçlara İnternet erişimini sağlamaktır.

2.1.2. Araç üstü ünitesi

Araç üstü üniteleri, yol kenarındaki RSU veya diğer araçlardaki OBU'larla veri iletişimi için kullanılan araç üzerine yerleştirilmiş birimlerdir. Üzerinde kaynak komut işlemcisi (Resource Command Processor-RCP) bulunur. Bağlantı için IEEE 802.11p ve güvensiz uygulamalar için IEEE 802.11a/b/g/n gibi iletişim teknolojilerini kullanır. Temel işlevleri, kablosuz radyo erişimi, tasarsız ve coğrafi yönlendirme, ağ yoğunluk kontrolü, güvenli mesaj transferi ve veri güvenliğidir.

2.1.3. Uygulama ünitesi

Uygulama ünitesi araç içinde OBU'nun sağladığı iletişim kapasitesini kullanarak İnternet ve benzeri uygulamaları kullanmaya yardımcı araç içi birimdir [15]. Araçsal tasarsız ağlar kablosuz tasarsız ağlarla aynı yapıya sahiptir. Araçsal tasarsız ağ mimarisi Şekil 2.2'de görüldüğü gibi hücresel/WLAN, tasarsız (Ad Hoc) ve karma (Hybrid) olmak üzere üç kategoriye ayrılmaktadır [4,17,18].



Şekil 2.2. Araçsal tasarsız ağ mimarisi [17]

Şekil 2.2.a'da görülen hücresel/WLAN iletişim, araçtan sabit altyapılı sistemlere olan (vehicle to insfructure-V2I) haberleşmeyi veya tersi yönde yapılan haberleşmeyi (Insfructure To Vehicle-I2V) içermektedir. Bu haberleşme hücresel telefon haberleşmesi ve WLAN olarak ikiye ayrılabilir. Hücresel telefonlar altyapı sayesinde WLAN gibi teknolojilerle de haberleşebilir. Araçlar arasında gerçekleştirilecek haberleşme için kullanılacak sinyal bu amaçla da kullanılabilir. Böylelikle bir araç çok sayıda modülü birlikte bulundurmak zorunda kalmamaktadır [19].

Şekil 2.2.b'de gösterilen iletişim, araçtan araca (vehicle to vehicle-V2V) olan doğrudan haberleşmeyi kapsamaktadır. Herhangi bir altyapı ihtiyacı duyulmadan kurulabilmektedir. Bir altyapı ya da erişim noktası olmadığı için ortama erişimin ve yönlendirmenin yeniden çözümlenmesi ve koordine edilmesi gerekmektedir. Buradaki haberleşme bilgi kaynağı ve varış düğümü pozisyonu açısından değerlendirilerek ikiye ayrılabilir. Birincisinde haber kaynağının ve varış düğümünün birbirleriyle doğrudan konuşabilir bir mesafe içerisinde doğrudan haberleşmesi yapılırken, ikincisinde bu şekilde haberleşmek mümkün değildir. Bu durumda aradaki diğer araçlar yönlendirici olarak kullanılarak iki düğümün haberleştirilmesi sağlanmaktadır.

Şekil 2.2.c'de ise araçtan araca ve araçtan altyapıya (vehicle to x-V2X) olan haberleşmenin birlikte yapıldığı iletişimi kapsamaktadır. Özellikle seyrek trafik yoğunluğu nedeniyle araçtan araca haberleşmenin yapılamadığı durumlarda V2I ve V2V haberleşmenin birlikte ve işbirliği içinde kullanımı bağlantı sorunlarını giderir [19].

2.2. Araçsal Tasarsız Ağların Karakteristik Özellikleri

Araçsal tasarsız ağlar mobil tasarsız ağlarla karşılaştırıldığında çok sayıda kendine özgü farklı karakteristik özelliklere sahiptir. Bu karakterler şu şekilde sıralanmıştır.

2.2.1. Yüksek hareketlilik ve hızlı değişen topoloji

Araçlar özellikle otoyollarda çok hızlı hareket etmekte, bağlantılar hızlı bir şekilde kurulup kesilmektedir. Böylece her bir aracın diğerleriyle iletişim süresi birkaç saniye sürmektedir [20,21]. Araçlar arasındaki bir bağlantının yaşam süresini, kapsama alanı ve araçların yönü belirlemektedir. Kapsama alanının artırılması bağlantı süresini artırmakta ancak farklı yönlere giden araçlar arasındaki bağlantı süresi daha az, aynı yöne giden araçların bağlantı süresi ise daha fazla olmaktadır [15].

2.2.2. Araç üstü algılayıcılarla etkileşim ve coğrafi pozisyon kullanılabilirliği

Araçlar üzerinde iletişim bağlantısı sağlama ve yönlendirmeyi şekillendirmek için araçlarda algılayıcı bulunduğu varsayılmaktadır [17]. Araçlara elektronik harita bilgisi bulunan ve yönlendirme amaçlarına göre konum bilgisi sağlayan GPS konum belirleme sistemleri bağlanabilmektedir.

2.2.3. Hareketlilik ve topoloji tahmini

Araçsal tasarsız ağlar yol işaretleri, trafik ışıkları ve diğer araçların hareketlerine göre şekillenen yol topolojisi tarafından kısıtlanır ve bu yönüyle düğümlerin rastgele hareket ettiği diğer mobil tasarsız ağlardan farklılaşır [15]. Otoyol, yol ve caddelerin önceden inşa edilmesi araçsal düğümlerin bu güzergahlara bağlı kalmasını sağlar. Böylece aracın hızı ve harita verildiğinde aracın ileride bulanacağı konum tahmin edilebilir [2,17,21].

2.2.4. Sınırsız pil gücü ve depolama

Araçlarda bulunan şarj edilebilir aküler sayesinde araçsal tasarsız ağlar için bir güç kısıtlaması bulunmamaktadır [2,21]. Uzun ömürlü bu aküler sayesinde araç üzerindeki üniteye sürekli olarak güç sağlanmaktadır [15].

2.2.5. Değişken ağ yoğunluğu

Araçsal tasarsız ağ yoğunluğu trafik yoğunluğuna bağlı olarak değişir [15]. Ağ yoğunluğunun fazla oluşu yayın fırtınası (broadcast storm) problemini ortaya çıkarırken, seyrekliği ise ağa bağlı kalmayı olumsuz yönde etkilemektedir. Düşük yoğunlukta acil mesajların gerçek zamanda alıcıya ulaştırılması oldukça zordur. Bu durumda mesaj, gecikme toleranslı olarak bekletilip, bağlantı oluşturulduğunda iletilebilir. Az yoğunluktan dolayı aynı mesajın aynı araç tarafından birçok kez yayınlanması gerekebilir. Fakat çok yoğun ortamda sadece belirli araçlar tarafından yayınlanmalıdır. Aksi takdirde iletişim kanalına erişimde yüklenme olacağından ortama erişim problemleri doğacaktır. Yoğunluk sadece bölgeye göre değil zamana göre de değişir.

Gün içerisinde şehir içi ve otoyollarda trafik daha yoğun ve acil mesaj iletimi için uygun iken, gece saatlerinde trafik bu yollarda daha az olmakta ve yönlendirmedeki sorunlar artmaktadır [22].

2.2.6. Yüksek hesaplama yeteneği

Araçsal tasarsız ağlardaki her aracın işlemci, geniş hafıza kapasitesi, gelişmiş anten teknolojisi ve global konum sistemi (Global Positioning System-GPS) gibi hesaplama ve algılama kaynaklarına sahip olması araçların hesaplama yeteneğini artırmaktadır [15].

2.3. Araçsal Tasarsız Ağlarda Kablosuz Erişim Teknolojileri

Günümüzde araçların birbiriyle haberleşmesi için gerekli iletişim arabirimini sağlayacak çok sayıda farklı kablosuz erişim teknolojileri mevcuttur. Bu iletişim teknolojileri güvenli ve güvensiz uygulamaları kullanarak yol güvenliğini ve trafik verimliliğini artırmayı, sürücü ve yolcu konforunu sağlamayı amaçlamaktadır. Bu teknolojilerin bazıları düğümler arası iletişimi koordine etmek için altyapı sistemleri üzerinde çalışmakta iken bazıları ise Ad Hoc olarak çalışmaktadır [15].

Hücresel sistemler (2/2.5/2.75/3G)

1990'lı yılların başında Avrupa'da geliştirilen ikinci nesil (Second Generation-2G) hücresel teknoloji Global Mobil Sistem (Global System for Mobile-GSM) en yaygın kullanılan kablosuz iletişim teknolojisi olmuştur. İlk kurulan GSM sistemleri çok düşük hızlı (9,6 kbps) ses ve mesaj iletimini desteklemekteydi. Daha yüksek veri iletim hızları gerektiren görüntü, video ve İnternet gibi servislerin desteklenmesi ise 1990'lı yılların sonunda Avrupa Haberleşme Standartları Enstitüsü (European Telecommunications Standards Institute-ETSI) tarafından geliştirilen ve 2,5 nesil(2.5G) sistem olarak isimlendirilen Genel Paket Radyo Servisleri (General Packet Radio Services-GPRS) teknolojisiyle mümkün olmuştur. GPRS ile veri iletimi 170 kbps kadar ulaşmıştır [15,23]. Daha sonra GSM'in veri miktarı saniyede 384 kbps kadar geliştirilmiş ve 2.75G olarak bilinen EDGE (Enhanced Data Rate for GSM Evolution) teknolojisi geliştirilmiştir [6].

Kullanıcılar tarafından talep edilen servislerin daha yüksek veri iletim hızı gerektirmesi ve kablolu İnternet bağlantı hızıyla yarışabilmesi için 2 Mbps hızına kadar çıkabilen üçüncü nesil (Third Generation-3G) teknoloji ve evrensel mobil haberleşme sistemi (Universal Mobile Telecommunication Systems -UMTS) geliştirilmiştir [23].

WLAN/ Wi-Fi

Kablosuz yerel alan ağı (Wireless Local Area Network-WLAN) ya da kablosuz bağlantı (Wireless Fidelity-Wi-Fi) araç-araç arası ve araç-altyapı arasında kablosuz erişim sağlayabilmektedir. Bu ağlarda kablosuz bağlantı sağlayabilmek için IEEE 802.11'in IEEE 802.11b, IEEE 802.11a ve IEEE 802.11g standartları kullanılmaktadır [15].

WiMAX

WiMAX kablosuz metropolitan alan ağlarının kapsama alanını artırmak için geliştirilmiş bir teknolojidir [14]. Noktadan noktaya kablosuz geniş bant erişimi için geliştirilmiş IEEE 802.16e standartlarına dayanmaktadır. IEEE 802.16e İnternet protokolü üzerinde video, multimedya ve ses özellikleri gerektiren uygulamalar için uygun servis kalitesi, güvenilir iletişim, geniş bir kapsama alanı ve yüksek veri oranı sağlar [15].

Dedicated Short Range Communications (DSRC) / Wireless Access in Vehicular Environments (WAVE)

DSRC/WAVE yol güvenlik mesajı ve kontrolü için son derece kısa gecikme gereksinimini karşılayabilen kablosuz teknolojidir. Araçsal ağlar, 5.9 Ghz bandında, 75 MHz'lik band genişliğine sahip, yüksek veri transfer hızı (6-27 Mbps) sunan DSRC (Dedicated Short Range Communications) standardını kullanır. Bu standarda göre yol üzerinde hareket halindeki araçlar mevki, zaman, yön, hız, trafik durumu bilgilerini içeren rutin trafik mesajını yaymak zorundadır [24].

2.4. Araçsal Tasarsız Ağlarda Ortaya Çıkan Sorunlar

2.4.1. Sinyal zayıflaması

Haberleşen iki araç arasında engel olarak duran nesneler araçsal tasarsız ağların verimliliğini etkileyen zorluklardan biridir. Bu engeller, yol boyunca bulunan diğer araçlar ve binalar olabilir.

2.4.2. Bant genişliği sınırlamaları

Araçsal tasarsız ağlardaki kilit noktalardan biri de içerik işlemleri ve bant genişliği yönetiminden sorumlu olan ve düğümler arası iletişimi kontrol eden merkezi bir koordinatörün olmamasıdır. Bu ağ uygulamaları için özellikle yüksek yoğunluklu ortamlarda, bant genişliği frekansının 10-20 MHz olması sebebiyle kanal sıkışıklığı yaşanma ihtimali yüksektir. Bu durumda, araçlar mesaj göndermek istediğinde bir süre beklemek zorunda kalacaktır Bu sebepten bant genişliğini etkili bir şekilde kullanmak gerekmektedir [15].

2.4.3. Bağlanılabilirlik

Ağdaki sık kopmalara yol açan yüksek hareketlilik ve hızlı değişen topoloji sebebiyle bağlantılar arası iletişimi devam ettirmek için gereken zaman mümkün olduğunca uzun olmalıdır. Bu durum iletim gücü artırılarak çözülebilir, ancak bunun sonucunda mesajlarda bozulmalar meydana gelebilir.

2.4.4. Küçük çaplı etki

Kısa mesafeli ağ çapı etkisi, düğümler arası haberleşmede zayıf bağlantıya yol açabilir. Böylece bir düğüm için ağın genel topolojisini devam ettirmek zorlaşmaktadır.

2.4.5. Güvenlik ve gizlilik

Araçsal tasarsız ağlarda güvenlik ve gizliliğin makul çerçevede tutulması ana zorluklardan biridir. Bilginin kaynağından güvenli bir şekilde alınması alıcı için önemlidir. Gönderilen verilerin zararlı kişiler tarafından amacı dışında değiştirilmesi, trafik akışında değişmeye ve ulaşım sisteminde kaosa neden olabilir. Diğer önemli durum ise mesajın iletildiği süre boyunca doğruluğunun ve bütünlüğünün sağlanma gerekir. Aksi bir durum, kazalara neden olabilir ve araçsal ağların amacını boşa çıkarır. Her aracın bir MAC adresi olması durumunda, bu aracı ve dolayısıyla sürücüsünü izlemek mümkün olacaktır. IEEE 802.11p standartı ile dinamik olarak MAC adresi ataması mümkündür. Bu yöntem, araçların amacı dışında yetkisiz kişiler tarafından izlenmesini engelleyici bir olabilir [6].

Araçsal ağlarda araçlar birbirleriyle iletişim sağlamak istediklerinde aynı kapsama alanında bulunan bir ya da birden çok araç bulunabilir. Bu nedenle öncelikle bu iletişimin yapılacağı araçlar arasında bir tür kimlik denetimi yapılabilir ve iki tarafında birbirlerinin kimliklerini onaylaması gerekebilir. Bu sayede ağa yetkili birimlerin erişimi sağlanır ve kötü niyetli birimlerin ağa erişimleri de engellenir [25].

Burada kullanılabilirlik (availability), gizlilik (confidentiality) ve güvenirlik (authenticity) alanlarında çeşitli saldırılar olabilmektedir [1].

Kullanılabilirliğe yönelik saldırılar

Bu saldırılar ortam erişim protokolünün açıklarından faydalanarak düğümlerin iletişimlerinin kesilmesine ya da anormal durumların oluşmasına sebep olarak hizmet engellemeyi amaçlarlar. Bu saldırılara örnek olarak hizmet engelleme saldırıları (Denial of Service-DoS), onaysız yayın değişikliği (Broadcast Tampering), kötü amaçlı yazılım (Malware), çoklu istenmeyen e-posta gönderimi (spam mail), kara delik saldırıları (black hole attack) verilebilir.

Güvenirliliğe yönelik saldırılar

Bu saldırı türlerinde saldırgan, mesajları değiştirerek iletinin bütünlüğünü bozmaktadır. Saldırgan, orijinal paketleri alıp yanlış alıcılara yönlendirirebileceği gibi, başka birimlerin kimliğini kullanarak yanlış paketler de gönderebilmektedir. Bu saldırılara örnek olarak maskeleme (masquerading), yeniden gönderme saldırıları (replay attack), konum bilgisi sızdırma (global positioning system spoofing), tünel açma (tunneling), konum yanıltma (position faking), onaysız mesaj değiştirme (message tampering), mesaj gizleme, uydurma, değiştirme (message suppression/ fabrication/ alteration), anahtar veya sertifika değiştirme (key and/or certificate replication) ve sybil verilebilir.

Gizliliğe yönelik saldırılar

Araçsal ağ düğümleri arasında mesaj değişim gizliliği, gizli dinleme yoluyla mesajların yasa dışı toplanması ve broadcast mesajları yoluyla konum bilgilerinin toplanması gibi tekniklerle saldırıya açık haldedir. Gizli dinleme durumunda saldırganlar yoldaki kullanıcıların haberleri olmaksızın onların bilgilerini toplayıp kullanabilmektedir.

2.4.6. Yönlendirme protokolü

Araçsal tasarsız ağlarda diğer sorunlardan birisi de en az paket kaybı ile en kısa sürede paket dağıtımını yapacak protokolün tasarımıdır. Etkili bir yönlendirme protokolü tasarımı çeşitli açılardan önemlidir. Öncelikle paket dağıtım yüzdelerinden faydalanarak sistem güvenliğinin artırılması, şehir merkezlerindeki binaların sebep olduğu parazit miktarının azaltılması, çakışmaların önlenmesi için ölçeklenebilirliğin hesaba katılması ve özellikle acil durumlarda mümkün olan en kısa sürede paket dağıtımının yapılması gerekmektedir [15].

2.5. Araçsal Tasarsız Ağ Uygulamaları

Araçsal tasarsız ağ uygulamalarını güvenlik ve güvenlik dışı uygulamalar olarak iki ana kategoriye ayırmak mümkündür. Güvenlik uygulamaların amacı ulaşım altyapısıyla ilgili bütün güvenliği artırmayı amaçlamaktadır. Güvenlik dışı uygulamaların amacı ise sürücü ve yolcu konforunu artırmaktır. Bu alandaki uygulamalar müşterilere araçlarla ilgili basit işlemler veya ödeme sistemleri ile ilgili servisler sağlarlar. Ayrıca eğlenceye yönelik (film, müzik vb.) servisler, basit araç bakımları, paralı yol (köprü, otoyol vb.) veya park ödemelerinin e-ödeme olarak yapılmasını sağlarlar.

2.5.1. Güvenlik uygulamaları

Güvenlik uygulamaları**,** insan yaşamının korunması amacıyla kazaların önlenmesi ve yol güvenliğinin geliştirilmesi için araç-araç arası ya da araç-yol kenarı haberleşme altyapısı arasındaki kablosuz haberleşmeyi kullanmaktadırlar [15,26].

Kavşak çarpışması önleme

Bu sistem, altyapıdaki algılayıcılar vasıtasıyla kavşağa yaklaşan araçlardan bilgilerin toplanması, işlenmesi ve analiz edilmesiyle kazaların önlenmesini sağlamaktadır. Bu kategori altında trafik ışık ihlali uyarısı, kör nokta uyarısı, sola dönüş yardımı, fren lambası hareket yardımı, kavşak çarpışma uyarısı, yaya geçidi uyarısı gibi birçok uygulama bulunmaktadır [27].

Toplum güvenliği

Toplum güvenliği uygulamaları daha çok kaza meydana geldiğinde sürücülere yardımı ve onlara yardıma gelecek ilk yardım hizmetlerinin daha hızlı ulaşmasını amaçlamaktadır. Yaklaşan acil durum aracı uyarısı, acil durum aracı sinyal algılama, kaza sonrası uyarı bu kategorideki uygulamalardır [27].

İşaret genişletmeleri

Bu uygulamanın amacı sürüş sırasında dikkatsiz sürücülerin yol kenarına yerleştirilmiş işaretçilerle uyarılmasını sağlamaktır. Dönüş hızı uyarısı, araç içi tabela bildirimi, altyapı tabanlı yol durum uyarısı, çalışma alanı uyarısı, yanlış yol uyarıları bu kategoride yer almaktadır [15].

Araç izleme ve bakımı

Bu uygulamaların amacı sürücülere araçla ilgili bildirim mesajları göndererek bakım ve tamir mesaj bildirimleri sağlamaktır. Bunlar anlık bakım uyarısı ve güvenlik uyarı bildirimi gibi uygulamalardır [27].

Gelişmiş sürüş

Bu kategori, sürüşü kolaylaştırmaya yönelik uygulamaları içerir. Bu uygulamalar sürücü asistanı gibi çalışmaktadır. Otoyol yardımı, sisteme entegre hız sabitleyici, duruma göre far parlaklığı azaltma, otomatik vites değişimi, sollama yardımı ve bilgi ekranından uyarı işaretlerinin gösterimi gibi uygulamalar bu kategoride yer almaktadır [27].

Diğer araçlardan bilgi aktarımı

Kaza öncesi algılama, acil elektronik fren lambası, otoyol birleşme yardımı, görüş artırma, zincirleme kaza uyarısı, araç temelli yol durum uyarıları bu kategoride yer almaktadır [15].

Yukarıda verilen güvenlik uygulamaları örneklerinin haberleşme şekilleri, bu haberleşmeler arasında kabul edilebilir gecikme süreleri ve uygulamanın öncelik sıraları Çizelge 2.1'de sunulmuştur. Araçtan araca olan uygulamaların önceliği haberleşme türü alanında (1),(2),(3),(4) şeklinde verilmiştir.

Çizelge 2.1. Araçtan araca güvenlik uygulamaları

Güvenlik uygulamaları	**Uygulama örnekleri**	**Haberleşme türü ve önceliği**	**Gecikme (s)**
Kavşak çarpışma önleme (Intersection Collision Avoidance)	Trafik ışık ihlali uyarısı	V2I ve I2V	0,1
	Sola dönüş yardımı		0,1
	Kavşak çarpışma uyarısı		0,1
	Fren lambası hareket yardımı		0,1
	Yaya geçidi uyarısı		0,1
Kamu güvenliği (Public Safety)	Yaklaşan acil araç	V2V (1)	1
	Acil araç sinyal önalımı	V2I	1
	SOS servisi	V2I ve I2V	1
	Kaza sonrası uyarı	V2I ve I2V	0,5
İşaret uyarıları (Sign Extension)	Araç içi tabela bildirimi	I2V	1
	Dönüş hız uyarısı	I2V	1
	Yanlış yön uyarısı	V2V (2)	0,1
	Düşük köprü ve alçak park uyarısı	I2V	1
	Çalışma alanı uyarısı	I2V	1
Araç tanılama ve bakımı (Vehicle Diognastic and Maintenance)	Güvenlik uyarı bildirimi Anlık bakım uyarısı	V2I ve I2V	5
Diğer araçlardan bilgi aktarımı (Information transfer from other vehicles)	İşbirliğine dayalı çarpışma ön uyarısı	V2V (2)	0,1
	Şerit değiştirme uyarısı	V2V (2)	0,1
	Otoyol birleşme yardımı	V2V (2)	0,1
	Görüş artırma	V2V (2)	0,1
	Zincirleme kaza uyarısı	V2V (1)	0,1
	İşbirliğine dayalı seyir kontrolü	V2V ve I2V	0,1
	Acil elektronik fren lambaları	V2V (2)	0,1
	Yol durum uyarısı	I2V	1
	Araç temelli yol durum uyarıları	V2V (2)	0,5
	Araçtan araca yol özellik bildirimi	V2V (2)	0,5
	Kaza öncesi algılama	V2V (2)	0,02
	Otoyol/raylı sistem çarpışma uyarısı	V2V ve I2V	1

2.5.2. Güvenlik dışı uygulamalar

Bu uygulamalar sürücü ve seyahat edenlerin rahatlığını ve konforunu amaçlamaktadır. Uygulamalar müşterilere araçlarla ilgili basit işlemler veya ödeme sistemleri ile ilgili servisler sağlarlar. Bu uygulamalar sürücü ve yolculara hava ve trafik bilgisi sunarken aynı zamanda yakındaki petrol, restoran, otel gibi yerler hakkında da bilgiler verirler [18].

Trafik yönetimi

Bu gruptaki uygulamalar trafik akışını yönetmeye yöneliktir. Yol kenarı biriminden periyodik yapılan bildirimlerle araçların yol giriş çıkışları, eğim ve rampa hakkında uyarılması amaçlanır [27].

Otomatik ücret ödeme

Bu alandaki uygulamalar sürücülere araçlarla ilgili basit işlemler veya ödeme sistemleri ile ilgili servisler sağlarlar. Ayrıca, sürücü ve yolculara zaman bakımından fayda sağlamakla birlikte trafik sıkışıklığı gibi sorunları da ortadan kaldırmaktadır. Yakıt ücreti ödeme, park yeri ödeme, geçiş ücreti ödeme gibi rahatlık ve konforla ilgili uygulamalar bu grupta yer almaktadır [15].

Diğer araçlardan bilgi aktarımı

Bu gruptaki uygulamalar sürücülere rahat ve güvenli sürüş imkanı sunmakla birlikte yolculara İnternet erişimi ve anlık mesajlaşma gibi fırsatlar sunmaktadır [26].

Yukarıda verilen güvenlik dışı uygulamaların haberleşme şekilleri, bu haberleşmeler arasında kabul edilebilir gecikme süreleri ve uygulamaların öncelik sıraları Çizelge 2.2'de sunulmuştur.

Çizelge 2.2. Araçtan araca güvenlik dışı uygulamalar

Güvenlik dışı uygulamalar	**Uygulama örnekleri**	**Haberleşme türü ve önceliği**	**Gecikme** (s)
Trafik yönetimi (Traffic management)	Akış rampa ölçümü Akıllı trafik akış kontrolü	V2I	1
Otomatik ücret ödeme (Toll Collection)	Serbest geçiş ödeme	V2I ve I2V (3)	0,05
Diğer araçlardan bilgi aktarımı (Information transfer from other vehicles)	İşbirliğine dayalı parlaklık azaltma	V2V (4)	1
	Anlık mesajlaşma	V2V (4)	1
	Uygun far ayarı	I2V	1
	Uygun motor gücü yönetimi	I2V	1
	Rota seyir ve yönlendirme genişletmesi	I2V	1
	İlgi çekici alan/konu bildirimi	I2V	1
	Harita indirme ve güncelleme	V2V (4) V2I	1
	GPS konum doğrulama	I2V	1

2.6. Araçsal Tasarsız Ağ Yönlendirme Protokolleri

Kablosuz ortamda temel yönlendirme algoritmalarının kullanımı geniş alana taşma, düz adresleme, yaygın dağınık bilgi, boş komşu, fazla güç tüketimi, parazit ve yük dengeleme gibi sorunlar meydana getirmektedir [28]. Bu nedenle araştırıcılar kablosuz ağlar için geniş çaplı yönlendirme protokolleri geliştirmeyi amaçlamışlardır. Bu protokoller, yükü kontrol ederek paket kayıplarını azaltırken çıktı sayısını maksimize etmektedir. Bu protokoller araçsal tasarsız ağ haberleşme mimarisi açısından araçtan araca yönlendirme protokolleri ve araçtan altyapıya yönlendirme protokolleri olarak iki kategoride sınıflandırılmaktadır [18].

2.6.1. Araçtan araca yönlendirme protokolleri

Araçtan araca yönlendirme protokolleri, topoloji tabanlı yönlendirme protokolleri, konum tabanlı yönlendirme protokolleri, broadcast tabanlı yönlendirme protokolleri, multicast tabanlı yönlendirme protokolleri, geocast tabanlı yönlendirme protokolleri ve küme tabanlı yönlendirme protokolleri olarak altı gruba ayrılmaktadır [29,18].

Topoloji tabanlı yönlendirme protokolleri

Topoloji tabanlı yönlendirme protokolleri kaynaktan hedefe veri paketlerini göndermek için ağ içindeki bağlantı bilgilerini kullanırlar [30]. Bu protokoller tabloya dayalı, isteğe bağlı ve hibrid protokoller olmak üzere üç ana kategoriye ayrılmaktadır [18,31].

Tabloya dayalı yönlendirme protokolleri

Bu protokoller ağdaki tüm kaynak-hedef çiftleri arasındaki yolları bulurlar ve periyodik yol güncellemeleriyle en yeni yol bilgilerini oluştururlar. Güncelleme mesajları ağ topolojisinde hiç değişiklik olmasa bile gönderilir. Bu kategorideki protokoller uzaklık vektörü (distance vector) ve bağlantı durumu (link state) algoritmaları değiştirilerek geliştirilmiştir. Protokoller yönlendirme bilgilerini yönlendirme tablolarında saklamaktadırlar. Periyodik güncellemeler sayesinde bu protokoller çok yavaş sonuca ulaşabilirler ve çok miktarda yönlendirme ek yükü (overhead) oluştururlar [31]. Yaygın kullanılan tabloya dayalı yönlendirme protokolleri, hedef sıralı uzaklık vektörü yönlendirme protokolü (destination sequenced distance vector-DSDV) ve kablosuz yönlendirme protokolü (wireless routing protocol-WRP) olarak sayılabilir.

İsteğe bağlı yönlendirme protokolleri

Bu protokoller düğümler arası kullanılacak yol bilgilerini sürekli oluşturmazlar. Yollar sadece gerektiğinde yani düğümlerden herhangi birisi paket göndermek istediğinde oluşturulur. Bu yüzden bu kategorideki protokollere isteğe bağlı (on-demand) yönlendirme protokolleri denir. Literatürde yer alan isteğe bağlı yönlendirme protokollerinden bazıları, isteğe bağlı uzaklık vektörü yönlendirme protokolü (Ad Hoc on-demand distance vector routing-AODV), dinamik kaynak yönlendirme protokolü (dynamic source routing-DSR), geçici sıralı yönlendirme algoritması (temporally ordered routing algorithm-TORA) olarak sıralanabilir.

Hibrid protokoller ise yönlendirmenin daha etkili ve ölçeklenebilir olması için tabloya dayalı protokol ile isteğe bağlı protokollerin birleşimden oluşmaktadır. Bu protokollere örnek olarak alan yönlendirme protokolü (zone routing protocol-ZRP) ve hibrid tasarsız yönlendirme protokolü (hybrid ad hoc routing protocol-HARP) verilebilir [18].

Konum tabanlı yönlendirme protokolleri

Konum tabanlı yönlendirme protokolleri genellikle yerleştirilmiş düğümlerle ilgilenirler. Yerelleştirme, düğümlerin coğrafi konumlarının belirlenmesini sağlayan GPS ile gerçekleştirilir [31]. Paket yönlendirilirken GPS ile elde edilen yerel pozisyon bilgisi göz önüne alınacağından düğümler komşularına ilişkin pozisyon bilgisi tutmak zorundadırlar. Pozisyon bilgisiyle yönlendirme yapıldığında, ara düğüm kendisine ulaşan bir paketi yola koyup koymayacağına rahatlıkla karar verebilir. Burada önemli olan konum bilgisinin doğru elde edilmesi ve mesajın gideceği konumun doğru saptanmasıdır [19]. Bu protokollere örnek olarak gecikme toleransı olmayan ağ (non-delay tolerant network-non-DTN) yönlendirme protokolleri, gecikme toleranslı ağ (delay tolerant network-DTN) yönlendirme protokolleri ve hibrid yönlendirme protokolleri verilebilir [18].

Broadcast tabanlı yönlendirme protokolleri

Broadcast tabanlı yönlendirme protokollerinde her düğümün mesajı bir diğerine ilettiği basit taşma metodu uygulanır. Bu süreç mesajın tüm hedeflere ulaşmasını sağlar. Ağdaki düğüm sayısının fazlalığı broadcast mesaj sayısını artırarak çarpışmaya ve yüksek bant genişliği kullanımına neden olur. Bu nedenle düğüm sayısının az olduğu ağlar için uygundur. Bu protokollere örnek olarak kentsel çok sıçramalı yayın (urban multi-hop broadcast-UMB), dağıtılmış araçsal yayın (distributed vehicular broadcast-DV-CAST) ve BROADCOMM olarak verilebilir [18].

Multicast tabanlı yönlendirme protokolleri

Multicast tabanlı yönlendirme, bir kaynaktan belli alandaki birçok hedefe yapılan yayın türüdür. Geleneksel multicast protokolleri sabit kablolu ağlar için tasarlanmıştır, ancak araçsal tasarsız ağlar bu ağlardan oldukça farklıdır. Bu sebepten Vanet multicast protokolleri yüksek düğüm hareketliliği, yüksek hareket hızı ve hızlı değişen topoloji gibi karakteristik özelliklere uyum sağlamak zorundadır. Mobil tasarsız ağlar için kullanılan multicast protokollerinin birçoğu araçsal tasarısız ağlar içinde kullanılabilir. Bu protokollere örnek olarak multicast tasarsız isteğe bağlı mesafe vektörü (multicast ad hoc on-demand distance vector-MAODV), adaptif talep odaklı multicast yönlendirme (adaptive demand-driven multicast routing-ADMR) verilebilir [18].

Geocast tabanlı yönlendirme protokolleri

Geocast tabanlı yönlendirme belli bölgedeki düğümlere yapılan broadcast servisi olarak adlandırılmaktadır. Bu protokollerin temel amacı kaynak düğümden uygun coğrafi alandaki tüm düğümlere paket iletimi yapmaktır. Uygun alan (zone of relevance-ZOR) dışındaki düğümler hızlı hareket gerektirmediğinden onlara iletim yapılmamaktadır [13n]. Bu protokollere örnek olarak araçlar arası geocast (inter-vehicle geocast-IVG), önbellekten geocast yönlendirme (cached geocast routing-CGR), sürekli geocast yönlendirme (abiding geocast routing-AGR), sağlam araç yönlendirme (robust vehicular routing-ROVER) ve mobicast gibi protokoller verilebilir [18].

Küme tabanlı yönlendirme protokolleri

Küme tabanlı yönlendirme protokollerinde ağ yapısı ölçeklenebilirlik açısından kümelenerek oluşturulur. Küme içindeki bir düğüm küme başı (Cluster-Head) olarak koordinasyon ve diğer gruplarla haberleşme işlevlerini üstlenir. Yol, konum bilgisi de kullanılarak kurulmaya çalışılır. Küme başı grup üyeleriyle ilgili adres ve konum bilgilerini içeren tablolar oluşturur. Bir düğüm mesaj göndermek isterse varış adresini bu tablolara bakarak elde eder. Eğer varış adresi bu tablolarda yoksa ilgili düğümün konumuna ilişkin bir istek yayınlanır. Diğer küme başları kendi tablolarını kontrol ederler, ilgili düğüm kendi gruplarındaysa bu mesaja cevap verirler [29]. Bu protokollere örnek olarak küme tabanlı yönlendirme (cluster-based routing-CBR), küme tabanlı yönlü yönlendirme protokolü (cluster-based directional routing protocol-CBDRP), taşmaya dayalı küme tabanlı konum yönlendirme algoritması (location routing algorithm with cluster-based flooding-LORA-CBF) olarak sıralanabilir [18].

2.6.2. Araçtan altyapıya yönlendirme protokolleri

Araçsal tasarsız ağlarda seyrek ve yoğun trafik durumlarına göre yönlendirme protokollerinin performansı belirgin bir şekilde değişiklik göstermektedir. Trafik ile ilgili etkenler nedeniyle araçsal ağlar ağ bölümlemelerini istenilen düzeyde yapamazlar [18]. Buna çözüm olarak farklı araçsal uygulamalarda istenmeyen erteleme gecikmesini azaltmak ve araçsal iletişimi daha güvenli kılmak için yol boyunca erişim noktaları yerleştirilmesi önerilmiştir [32]. Bu yapı için statik altyapı tabanlı yönlendirme protokolleri ve mobil altyapı temelli yönlendirme protokolleri kullanılmaktadır.

Statik altyapı tabanlı yönlendirme protokolleri

Bu protokoller iletim aralığında bulunan araçlara paket iletiminin sağlanabilmesi için yol boyunca ve kavşaklarda bulunan yol kenarı ünitelerini kullanırlar. Yol kenarı ünite sayısı ve dağıtımı uygulanacak olan protokol bağlıdır. Statik düğüm destekli adaptif yönlendirme protokolü (static node-assisted adaptive routing protocol-SADV), yol kenarı destekli yönlendirme (roadside-aided routing-RAR), köşe tabanlı akıllı aç gözlü yönlendirme (vertex-based predictive greedy routing-VPGR), hareket vektörü yönlendirme algoritması (motion vector routing algorithm-MOVE) protokolleri bu kategorideki en yaygın kullanılan protokollerdir [33].

Mobil altyapı temelli yönlendirme protokolleri

Bu protokollerin sabit altyapı tabanlı yönlendirme protokollerinin donanımı, bakımı, kurulumu gibi maliyet gereksinimleri yoktur. Ek olarak kuruldukları alanda bağlanabilirlik sağlama gibi kısıtları mevcut değildir. Bu protokoller yol kenarı ünitesi yerleştirilmiş yerlerdeki ağ geçitlerinden faydalanırlar. Mobil ağ geçidi yönlendirme protokolü (mobile gateway routing protocol-MGRP), tahmine dayalı yönlendirme (prediction-based routing-PBR) protokolleri bu kategoride yer almaktadır [19].

2.7. Araçsal Tasarsız Ağ Simülatörleri

Araçsal tasarsız ağlar için simülasyonun iki yönü bulunmaktadır. Bunlardan birincisi trafik simülasyonu diğeri ise ağ simülasyonudur. Hareketlilik simülasyonları kentsel hareketlilik izlerinin oluşturulmasına yardım ederken, ağ simülasyonları, düğümler arası topolojiler oluşturup bunların değerlendirilmesine ve çeşitli koşullarda uygulanmasına imkan sağlamaktadır. Ancak bu ikisi arasında doğrudan bir bağlantı bulunmamaktadır. Araçsal tasarsız ağlar için bu simülatörlerin birlikte kullanılması gerekmektedir [34,35].

2.7.1. Hareketlilik simülatörleri

Simulation of Urban MObility (SUMO)

Geniş yol ağlarının işlenmesi için tasarlanmış açık kaynak kodlu yol trafik simülasyon paketidir. Temel olarak farklı araç tipleri, araç hareketleri, tek araç yönlendirme, çok şeritli yolda şerit değiştirme, kavşak bazlı yol hakkı kuralları ve yol hakkı hiyerarşisi, açık grafik kullanıcı arayüzü kütüphanesi ve dinamik yönlendirme gibi özelliklere sahiptir. SUMO, 10.000 caddeye kadar geniş ortamlarda çalışabilmekte ve farklı ağ formatlarını da içine alabilmektedir. En büyük eksikliği ise kendinden oluşturulmuş izlerin ağ simülatörleri tarafından doğrudan kullanılamamasıdır [36].

MObility model generator for VEhicular networks (MOVE)

Araçsal ağlar için SUMO üzerine oluşturulmuş Java programlama dili temelli mobil hareketlilik simülatörüdür [15]. Temelde trafik seviyesi üzerine odaklanmış ve çok iyi görselleştirme özellikleri barındırmaktadır. MOVE harita editörü ve araç hareketlilik editörünün birleşiminden oluşur. Harita editörü ağ senaryolarına göre topolojik haritalar oluştururken, hareketlilik editörü otomatik olarak hareket şekilleri oluşturur, bu şekiller kullanıcı tarafından da girilebilmektedir. Ayrıca Tiger veritabanı veya Google Earth'den hareketlilik izleri oluşturarak simülasyonu kolaylaştırır [34].

STreet RAndom Waypoint (STRAW)

Ayrık ağ simülatörü olan JIST/SWAN için geliştirilmiş araç hareketlilik modelidir. Amerika'daki şehirlerin araç hareket modellerini kullanarak doğru simülasyon sonuçları sağlamaktadır. Araçsal ağlar için uygun detay seviyesine sahip bir hareketlilik modeli ağ simülasyonunun doğruluğu için kritik öneme sahiptir. Bunu sağlayabilmek için STRAW harita verileri üzerindeki caddeler için düğüm hareketlerini kısıtlar ve hareketliliği araç sıkışıklığına göre sınırlandırır [36].

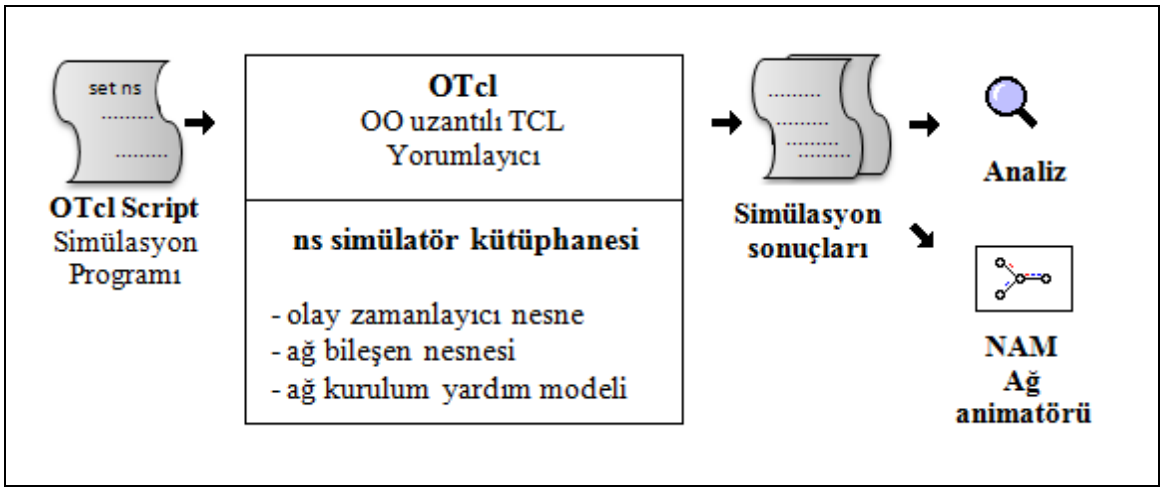
VanetMobiSim

Kullanıcı hareketliliğinin modellenmesi için esnek bir yapıya sahiptir. Farklı ağ simülatörleri için doğrudan hareketlilik modelleme oluşturur. Bu yazılım özellikle araçsal iletişim için yazılmıştır ve araçsal tasarsız ağ simülatörü için gerekli özelliklerin tümünü sağlamaktadır [37].

2.7.2. Ağ simulatörleri

Network Simulator 2 ve 3 (NS 2 ve NS 3)

ns, ağ simülasyonu oluşturmak ve gerçekleştirmek için ilk olarak 1989 yılında geliştirilmeye başlanmıştır. Akademik araştırmalar için büyük öneme sahip açık kaynak kodlu bir ayrık olay ağ simülatörü olarak tanımlanmaktadır. ns kullanımı, 1995 yılında ABD Savunma Bakanlığı İleri Araştırma Projeleri Ajansının (Defense Advanced Research Projects Agency-DARPA) sponsorluğunda ivme kazanmıştır ve günümüzde de simülatörün geliştirilmesi geniş bir kullanıcı grubu tarafından sürdürülmektedir. ns ile kablolu ya da kablosuz ağlarda istenilen miktarda düğümler ve bu düğümler arası linkler tanımlanabilmekte, yönlendirme algoritmaları ile çoklu yayın protokolleri kullanılabilir ve Ad-Hoc network, Wi-Fi, WiMAX gibi bir takım popüler kablosuz ağ uygulamalarının modellemeleri ve simülasyonu gerçekleştirilebilmektedir. ns simülatörü, ağ araştırma ve eğitimini destekleyerek ücretsiz açık kaynak kodu ile karşılaştırmalı bir benzetim ortamı sunmaktadır [38].



Şekil 2.3. ns-2 simülatörü genel bileşenleri

ns-2, C++ tabanlı bir simülatördür ve TCL dilinin nesneye yönelik bir versiyonu olan OTCL ile bir ağ simülasyonu gerçekleştirmek mümkündür. ns-3 simülatörü ise araştırma geliştirme ve akademik faaliyetlerde kullanılmak üzere özellikle internet tabanlı sistemler için geliştirilen ayrık olay ağ simülatörü olarak tanımlanmaktadır. ns-3, C++ ve Python dilleri kullanılarak yazılmaktadır. ns-3'te kod yapıları Doxygen isimli bir yazılım dokümantasyon program vasıtasıyla uygulanmaktadır. ns-2 den ns-3'e geçişte en önemli değişiklik script dilinin seçimidir. ns-3 simülatörü ns-2'nin genişletilmiş hali değildir. Her iki simülatörde C++ dilinde yazılmış fakat ns-3, ns-2'nin tüm özelliklerini bünyesinde barındırmamaktadır. ns-3'ü ön plana çıkaran ve önemini artıran IPv6 adresleme, daha fazla internet protokolü kullanım imkanı, düğümler üzerinde çoklu arayüzleri hatasız işleme gibi yeni ve etkili yeteneklere sahip olmasıdır [38].

OMNeT++

Açık kaynak ayrık olay ağ simülatörüdür. Bilgisayar ağları simülasyonu ve ağ kuyruk simülasyonları için kullanılabilmektedir. OMNeT++ belirli bir simülasyon modülü geliştirmek için gerekli olan yapıyı sağlamaktadır, fakat bu modüller OMNeT++'tan bağımsız geliştirilirler ve kendi döngülerini takip ederler.

JIST/SWAN

Cornell üniversitesi tarafından Java tabanlı olarak geliştirilen ayrık olay simülasyon aracıdır. JIST herhangi bir ağ simülatörü için kullanılabileceği gibi SWAN için de kullanılmaktadır. SWAN ise 10.000'den fazla düğümün benzetimini gerçekleştirebilmektedir [37].

Global Mobile Information System Simulator (GloMoSim)

Global mobil bilgi sistem monitörü (GlomoSim) heterojen haberleşmeye sahip binlerce düğümlü ağları simüle edebilmektedir. Hem kablolu hem de kablosuz ağ tasarım süreçlerinde kullanılabilmektedir. GloMoSim, C temelli paralel simülasyon dili olan PARSEC'te yazılmıştır. Genişletilip birleştirebilir şekilde tasarlanmıştır. Bu sayede yeni protokoller ve modüller kendi dili kullanılarak eklenebilmektedir [39].

2.7.3. Vanet simulatörleri

Araçsal tasarsız ağ simülatörleri hem ağ hem de hareketlilik simülatörlerini birlikte barındırmaktadırlar. Bu yüzden hibrid simülatörler olarak da bilinmektedirler. Bu simulatörler Şekil 2.4'te gösterilmiştir.

Veins

Bu hibrid simülatör iki ayrı simülatör olan yol trafik simülasyonu SUMO ile ağ simülatörü OMNeT++'dan oluşmuştur. Araçlar arası haberleşme için bu iki simülatör bir TCP soket ile birbirine bağlı olup paralel olarak çalışmaktadır [40,41].

GrooveNET

Gerçek araçlarla simüle edilmiş araçlar arasında haberleşme sağlayabilen araç-araç arası hibrid ağ simülatörüdür. GrooveNet kablosuz ağ ara yüzü ve GPS bulunan gerçek araçlarla cadde harita topolojisi üzerinde binlerce sanal aracı içeren simülasyonda araçsal ağ protokollerinin uygulanarak değerlendirilmesine imkan sağlamaktadır. Bu simülatör, tasarım yaklaşımı, araçsal ağ protokollerinin stres testi, doğruluk ve hızlı gelişimine olanak

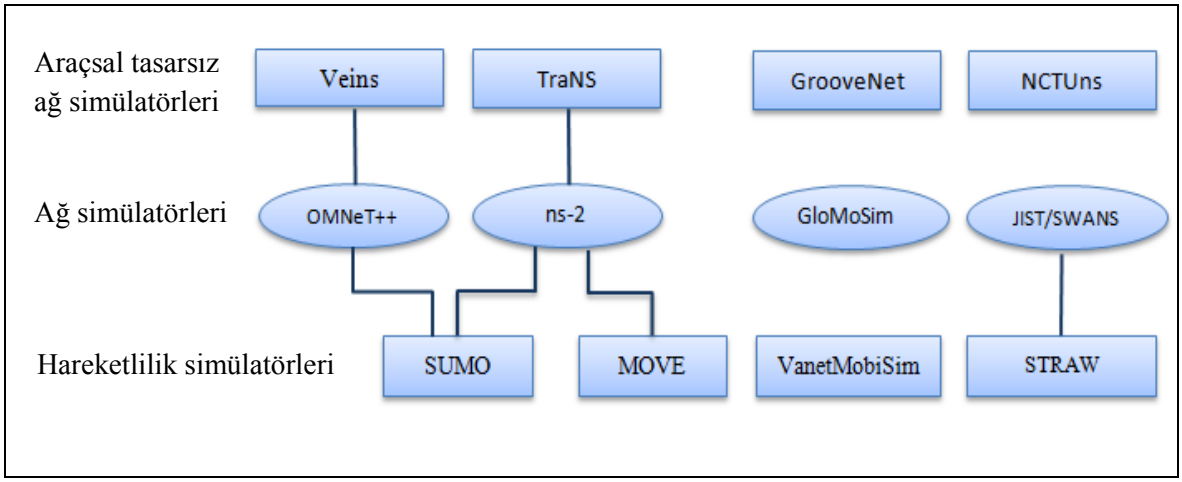
sağlamış ayrıca yollarda çok atlamalı haberleşme test prototiplerinin gelişimine ortam hazırlamıştır [42].

National Chiao Tung University Network Simulator (NCTUns)

Yol ağları üzerinde araç hareketlerini ve kablosuz iletişim protokollerini simüle etme özelliğine sahip ağ ve trafik simülatörüdür. NCTUns aslına uygun sonuçlar üretmek için Linux TCP/IP protokol yığınlarını kullanmaktadır. Bu simülatör IEEE 802.11b kablosuz LAN, IEEE 802.11e QoS kablosuz LAN, IEEE 802.16d WiMAX kablosuz ağlar, uydu ağları, araç araç arası ve araç altyapı arası akıllı taşıma sistemleri için kablosuz araç ağları, çok arayüzlü mobil düğümlü heterojen kablosuz ağlar, IEEE 802.16e mobil WiMAX ağlar, IEEE 802.11p/1609 WAVE gibi kablosuz araçsal ağlarını desteklemektedir [43].

Traffic and Network Simulation Environment(TraNS)

Hareketlilik üretici ve ağ simülatörünü birlikte barındıran ve gerçekçi araçsal tasarsız ağ uygulaması oluşturmaya yarayan simülasyon ortamıdır. TraNS araç hareketleri ve hareketlilik modelleri arasında dönüt sağlamaktadır. Java ve C++ da yazılmış olup Linux ve Windows ortamında çalışabilmektedir. Ağ simülatörü olarak ns-2, trafik simülatörü olarak SUMO kullanmaktadır. Güncel olarak TIGER (Topologically Integrated Geographic Encoding and Referencing) dosyalarını kullanmakla birlikte Google Earth'den görselleştirmeye izin verir ve 3000 araçlık geniş ölçekli ağ benzetimi yapabilmektedir [36].



Şekil 2.4. Araçsal tasarsız ağ simülatörleri

3. ARAÇTAN ARACA TASARSIZ AĞ UYGULAMASI

3.1. Kullanılan Sistem ve Simülasyon Ortamları

Geliştirilecek uygulama için literatür incelendiğinde yaygın olarak Linux tabanlı sistemlerin kullanıldığı görülmüştür [43]. Bu sebeple çalışmada işletim sistemi olarak Linux dağıtımlarından Ubuntu 12.04 LTS versiyonu kullanılmıştır. Ağ uygulamalarına yönelik yazılım geliştirmede ve eğitsel amaçlı kullanımda yaygın olarak ns-2 kullanıldığından ağ uygulamalarının benzetimi ns-2.35 versiyonu üzerinde yapılmıştır [1],[44]. Hareketliklik senaryoları için SUMO ve MOVE simülatörleri kullanılmıştır. Simülasyon senaryolarının çıktıları ise Trace Graph programı ile değerlendirilmiştir.

3.2. Uygulama Tanımı ve Adımları

Tez kapsamında geliştirilen uygulama araçtan araca haberleşme mimarisini içermektedir. Çizelge 2.1 ve 2.2'de verilen araçsal tasarsız ağ uygulamaları arasından araçtan araca olan uygulamalar seçilerek bunlar önem sırasına göre sıralanmıştır. Bu sıralama sonrasında uygulamalar için dört farklı öncelik grubu belirlenmiştir. Bu gruplara özgü olarak sırasıyla LcmApp, WmApp, TcApp, MmApp trafik üreticileri, önceliklendirilmiş PT_LCM, PT_WM, PT_TC, PT_MM paket tipleri ve bunlara yönelik iz dosyaları oluşturulmuştur. Uygulamalar için oluşturulan bileşenler Çizelge 3.1'de verilmiştir.

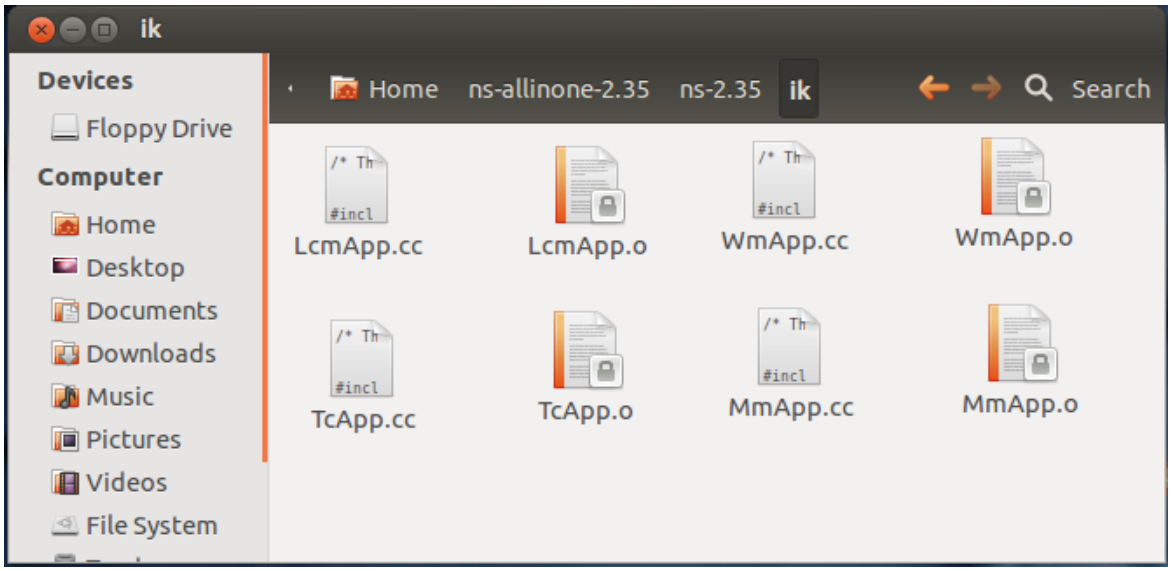
Çizelge 3.1. Mesaj paketleri tipleri ve öncelikleri

Mesaj tipleri	Trafik üreticisi	Paket tipleri	Paket önceliği
Hayati ve kritik mesajlar (Life Critical Message)	LcmApp	PT_LCM	1
Uyarı mesajları (Warning Message)	WmApp	PT_WM	2
Otomatik ödeme mesajları (Toll Collection)	TcApp	PT_TC	3
Multimedia mesajları (Multimedia Message)	MmApp	PT_MM	4

Oluşturulan bu yapılar kullanılarak yeni geliştirilen öncelik kuyruğu yapısının perfomansı test edilmiştir. Daha sonra simülasyon senaryoları oluşturularak benzetimler yapılmış, sonuçlar drop tail kuyruk yapısı ile karşılaştırmalı olarak analiz edilmiştir.

3.2.1. Trafik Oluşturulması

Belirlenen uygulamalara yönelik trafik oluşturulması için ns-2'de yeni trafik üretici dosyaları oluşturmak gerekmektedir. Bu sebeple öncelik sırasına göre birinci önceliğe sahip paket trafiği üretimi için LcmApp.cc, ikinci önceliğe sahip paket trafiği üretimi için WmApp.cc, üçüncü önceliğe sahip paket trafiği üretimi için TcApp.cc ve dördüncü önceliğe sahip paket trafiği üretimi için MmApp.cc dosyaları oluşturulmuştur.



Şekil 3.1. Trafik üretici dosyaları

Life Critical Message Application (LcmApp)

Bu trafik üretici kritik mesaj paket trafiği oluşturmaktadır. Birinci gruptaki uygulamaların paket trafikleri bu üretici ile sağlanmıştır. Üretici kodlaması için ns-2.35'te tanımlı trafgen.h dosyası mevcuttur ve diğer paket trafiklerinin oluşturulmasında da bu dosya içindeki TrafficGenerator sınıfı kullanılmıştır. LcmApp.cc dosyası içinde LCM_Traffic adında yeni bir sınıf türetilerek değişken tanımlamaları ve metodlar oluşturulmuştur. Ayrıca bu trafik üreticinin *.tcl dosyasına eklenerek kullanılmasını sağlayacak LCMTrafficClass sınıfı oluşturulmuştur.

Daha sonra üreticinin başlangıç durumu için yapıcı fonksiyonu `LCM_Traffic()`, trafiğin başlatılmasını sağlayan `LCM_Traffic::start()` ve `start()` fonksiyonundan tetiklenen `LCM_Traffic::init()` ve bir sonraki paket için mesafe aralığının hesaplanmasını sağlayan `LCM_Traffic::next_interval()` fonksiyonları kullanılmıştır. LcmApp.cc dosyasının ana hatları Şekil 3.2'de gösterilmiştir.

```
#include <stdlib.h>
#include "random.h"
#include "trafgen.h" // TrafficGenetor sınıfını header dosyası
#include "ranvar.h"
     // Kritik mesaj trafiği için LCM_Traffic sınıfı
class LCM_Traffic : public TrafficGenerator {
  public:
      trafik kaynağı sınıfına özgü public metod tanımları
  protected:
      trafik kaynağı sınıfına özgü protected değişken ve
     metod tanımları
       };
static class LCMTrafficClass : public TclClass {
 public:
      // Tcl dosyalarından LcmApp trafiğinin oluşturulması
      LCMTrafficClass():TclClass("Application/Traffic/LcmApp") {}
      TclObject* create(int, const char*const*) {
            return (new LCM_Traffic());
      }
} class_lcm_traffic;
     // Lcm_traffic sınıfı yapıcı fonksiyonu
LCM_Traffic::LCM_Traffic() : seqno_(0)
{    Yapıcı fonksiyonun oluşturulması
           Değişkenlere değer bind edilmesi  }

void LCM_Traffic::init()
{    Belli aralıklarla hayati mesaj paket trafiğinin
     başlatılması   }

void LCM_Traffic::start()
{    Paketler aralıklarının hesaplanması
     init()   fonksiyonun   çağrılarak   trafiğin   tetiklenmesi
}
double LCM_Traffic::next_interval(int& size)
{    Gönderilecek paket aralıklarının yeniden hesaplanması  }
```

Şekil 3.2. Kritik mesaj üreticisi kod blokları

Warning Message Application (WmApp)

Bu trafik üretici uyarı mesajları paket trafiği oluşturmaktadır. İkinci gruptaki uygulamaların paket trafikleri bu üretici ile sağlanmıştır. LcmApp trafiği için gerçekleştirilen kod adımları WmApp.cc dosyasının oluşturulması için de uygulanmıştır.

```
…
#include "trafgen.h" // TrafficGenetor sınıfını header dosyası
      // Uyarı mesajı trafiği için Wm_Traffic sınıfı
class WM_Traffic : public TrafficGenerator {
 public:
      trafik kaynağı sınıfına özgü public metod tanımları
  protected:
      trafik kaynağı sınıfına özgü protected değişken ve
     metod tanımları
};
     // Tcl dosyalarından WmApp trafiğinin oluşturulması
static class WMTrafficClass : public TclClass {
 public:
      WMTrafficClass() : TclClass("Application/Traffic/WmApp") {}
      TclObject* create(int, const char*const*) {
            return (new WM_Traffic());
      }
} class_wm_traffic;
WM_Traffic::WM_Traffic() : seqno_(0)
{     Yapıcı fonksiyonun oluşturulması
      Değişkenlere değer bind edilmesi
}
void WM_Traffic::init()
{     Belli aralıklarla uyarı mesajı paket trafiğinin
      başlatılması }

void WM_Traffic::start()
{     Paketler aralıklarının hesaplanması
      init() fonksiyonun çağrılarak trafiğin tetiklenmesi }

double WM_Traffic::next_interval(int& size)
{     Gönderilecek paket aralıklarının yeniden tanımlaması }
```

Şekil 3.3. Uyarı mesaj üreticisi kod blokları

Bu üreticiye özgü `WM_Traffic`, `WMTrafficClass` sınıfları ile `WM_Traffic()`, `WM_Traffic::init()` ve `WM_Traffic::next_interval()` fonksiyonları tanımlanmıştır.WmApp.cc dosyasının kodlamaları ana hatlarıyla Şekil 3.3'te gösterilmiştir.

Toll Collection Application (TcApp)

Bu trafik üretici otomatik ücret ödeme işlemleri için paket trafiği oluşturmaktadır. Üçüncü gruptaki uygulamaların paket trafikleri bu üretici ile sağlanmıştır. Daha önce açıklanan trafik üretici işlem adımları TcApp.cc dosyasının oluşturulması için de uygulanmıştır.

```
…
#include "trafgen.h"  // TrafficGenetor sınıfını header dosyası
    // Otomatik ücret ödeme mesaj trafiği için Tc_Traffic sınıfı
class TC_Traffic : public TrafficGenerator {
public:
     trafik kaynağı sınıfına özgü public metod tanımları
protected:
     trafik kaynağı sınıfına özgü protected değişken ve metod
tanımları
};
   // Tcl dosyalarından TcApp trafiğinin oluşturulması
static class TCTrafficClass : public TclClass {
 public:
     TCTrafficClass() : TclClass("Application/Traffic/TcApp") {}
     TclObject* create(int, const char*const*) {
          return (new TC_Traffic());
     }
} class_tc_traffic;

TC_Traffic::TC_Traffic() : seqno_(0)
{    Yapıcı fonksiyonun oluşturulması
     Değişkenlere değer bind edilmesi }

void TC_Traffic::init()
{    Belli aralıklarla uyarı mesajı paket trafiğinin
     başlatılması }
void TC_Traffic::start()
{    Paketler aralıklarının hesaplanması
     init() fonksiyonun çağrılarak trafiğin tetiklenmesi   }
double TC_Traffic::next_interval(int& size)
{    Gönderilecek paket aralıklarının yeniden tanımlaması }
```

Şekil 3.4. Otomatik ödeme mesajı üreticisi kod blokları

Bu trafik üreticisine özgü `TC_Traffic`, `TCTrafficClass` sınıfları ile `TC_Traffic()`, `TC_Traffic::init()` ve `TC_Traffic::next_interval()` fonksiyonları tanımlanmıştır. TcApp.cc dosyasının kodlamaları ana hatlarıyla Şekil 3.4'te verilmiştir.

Multimedia Application (MmApp)

Bu trafik üretici multimedia (ses, görüntü, internet) mesajları paket trafiği oluşturmaktadır. Dördüncü gruptaki uygulamaların paket trafikleri bu üretici ile gerçekleştirilmiştir.

```
…
#include "trafgen.h"  // TrafficGenetor sınıfını header dosyası
// Multimedia mesaj trafiği için MM_Traffic sınıfı
class MM_Traffic : public TrafficGenerator {
  public:
  trafik kaynağı sınıfına özgü public metod tanımları
  protected:
  trafik kaynağı sınıfına özgü protected değişken ve
  metod tanımları
};
static class MMTrafficClass : public TclClass {
 public:
      // Tcl dosyalarından MmApp trafiğinin oluşturulması
      MMTrafficClass()   :   TclClass("Application/Traffic/LcmApp")
{}
      TclObject* create(int, const char*const*) {
            return (new MM _Traffic());
      }
} class_lcm_traffic;

MM_Traffic:: MM_Traffic() : seqno_(0)
{   Yapıcı fonksiyonun oluşturulması
    Değişkenlere değer bind edilmesi  }
void MM_Traffic::init()
{   Belli aralıklarla multimedia mesaj paket trafiğinin
    başlatılması      }
void MM_Traffic::start()
{   Paketler aralıklarının hesaplanması
    init() fonksiyonun çağrılarak trafiğin tetiklenmesi  }
double MM_Traffic::next_interval(int& size)
{   Gönderilecek paket aralıklarının yeniden hesaplanması  }
```

Şekil 3.5. Multimedia mesaj üreticisi kod blokları

Bu trafik üreticisine özgü `MM_Traffic`, `MMTrafficClass` sınıfları ile `MM_Traffic()`, `MM_Traffic::init()` ve `MM_Traffic::next_interval()` fonksiyonları tanımlanmıştır. MmApp.cc dosyasının kodlamaları ve açıklamaları Şekil 3.5'te gösterilmiştir.

3.2.2. Paket tiplerinin oluşturulması

ns-2.35 içerisinde çeşitli uygulamalarda kullanmak üzere tanımlı yönlendirme ve data paketleri gibi standart paket tipleri mevcuttur. Bu paketler sistemde simülasyon yapmak için kullanılmaktadır. Ancak farklı uygulamaların işe koşulması gerektiği durumlarda uygulamaya özgü paket tanımlamaları yapılması gerekmektedir. Hedeflenen uygulama için araçsal tasarsız ağlarda kullanılan uygulamalar önem ve öncelik sırasına göre dört farklı gruba ayrılmıştır. Bu öncelik gruplarına göre paket trafik üreticilerin kullanacağı paketler oluşturulmuştur. Geliştirilen trafik üreticilerine özgü paket tipleri ve paket bilgilerini oluşturmak için ns-2.35 içerisindeki (˜/ns 2.35/common/packet.h) ilgili dosya Şekil 3.6'daki gibi değiştirilmiştir.

```
~/ns-2.35/common/packet.h

typedef unsigned int packet_t;// Standart  paket tipi tanımları

static const packet_t PT_TCP = 0; // Standart  paket tipi
static const packet_t PT_UDP = 1;
static const packet_t PT_CBR = 2;
…    // diğer paket tipleri
     // Yeni paket tiplerinin tanımlanması
static const packet_t PT_LCM =73;
     // Life Critical Message -Hayati mesaj paket tipi
static const packet_t PT_WM =74;
     // Warning Message - Uyarı Mesajları paket tipi
static const packet_t PT_TC =75;
     // Toll Collection - Otomatik Ödemeleri paket tipi
static const packet_t PT_MM =76;
     // Multi Media Message - Multimedia mesaj paket tipi
…
static packet_t PT_NTYPE = 78; // This MUST be the LAST one
…
```

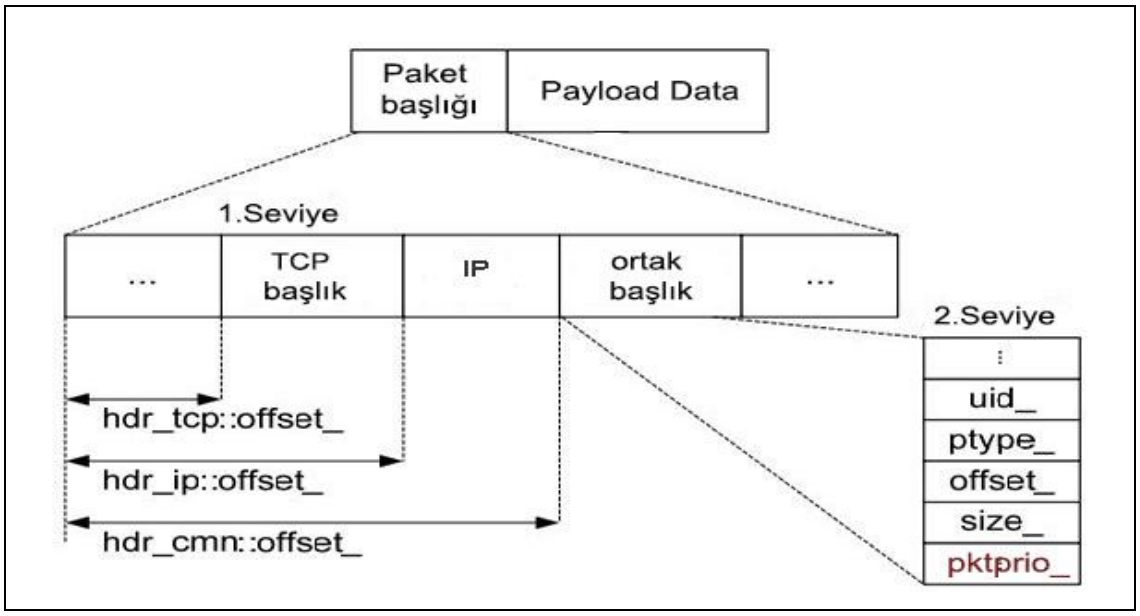
Şekil 3.6. Yeni paket tanımlamaları

```
~/ns 2.35/common/packet.h

class p_info {
public:
…
static void initName()
      {      …
             // Paket etiket bilgileri
             name_[PT_LCM]="Lcm";
             name_[PT_WM]="Wm";
             name_[PT_TC]="Tc";
             name_[PT_MM]="Mm";
             name_[PT_NTYPE]= "undefined";
      }
#define DATA_PACKET(type) ( (type) == PT_TCP || \
             …  //Diğer Paket tipleri
                // Paket tip tanımlamaları
                       (type) == PT_LCM || \
                       (type) == PT_WM || \
                       (type) == PT_TC || \
                       (type) == PT_MM \
                              )
}
```

Şekil 3.7. Paket tip ve etiket bilgileri

Şekil 3.7'de tipleri ve bilgileri tanımlanan paketlerin öncelikli kuyruk yapısı içinde kullanılması için her paketin önceliğini gösterir bir alanına sahip olması gerekmektedir. Bu nedenle ns-2'deki paket başlık yapısı, paket önceliğini gösterecek `pktprio_` alanı eklenerek genişletilmiştir. Bu alan standart paket başlığı içinde bulunan ortak başlık(common header) alanına tanımlanmıştır. Standart paket başlığı ve genişletilen kısım Şekil 3.8'de gösterilmiştir.



Şekil 3.8. Paket başlık yapısı

Şekilde gösterilen öncelik alanı tanımlaması ˜/ns-2.35/common/ yolundaki packet.h dosyası içindeki ortak başlık yapısına eklenmiştir. Bu alan tanımlaması Şekil 3.9'da görüldüğü gibi tanımlanmıştır.

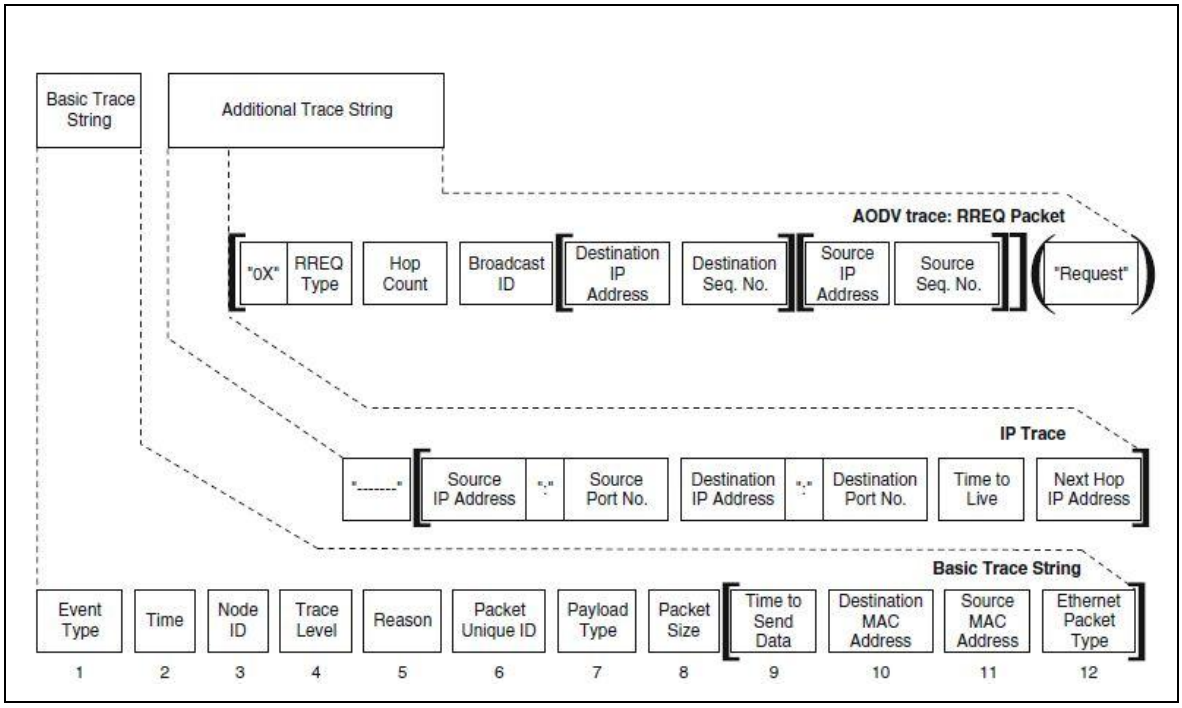
```
      …                          // packet.h dosyasındaki diğer kod
                                    satırları
struct hdr_cmn {
      enum dir_t { DOWN= -1, NONE= 0, UP= 1 };
      packet_t ptype_;      // paket tipi
      int    size_;         // paket boyutu
      int    uid_;          // eşsiz id
      int    error_;        // hata bayrağı
      int     errbitcnt_;
      int     fecsize_;
      double ts_;           // timestamp: zaman damgası
      int     iface_;       // receiving interface (label)
      dir_t direction_;     // yön: 0=none, 1=up, -1=down

      int     pktprio_;     // paket öncelik alanı
       …
               }
```

Şekil 3.9. Paket öncelik alanının açılması

3.2.3. İz dosyalarının oluşturulması

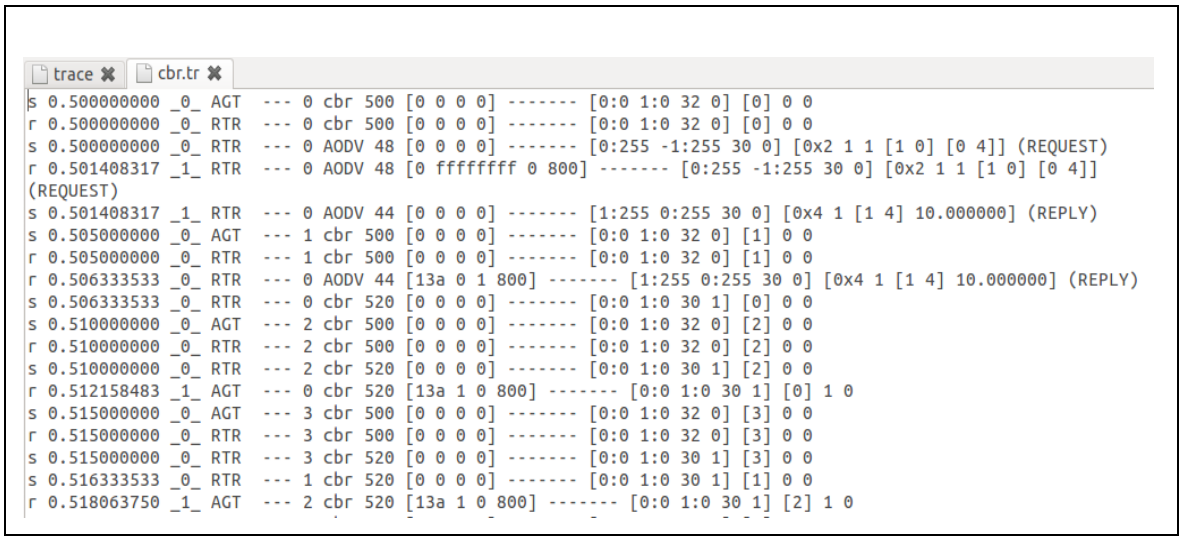
ns-2'de uygulama tiplerine göre oluşturulan standart iz dosyaları mevcuttur. Bu iz dosyası simülasyona yönelik çıktıları yani izleri yansıtmaktadır. Kablosuz ağ uygulamalarında temel iz (Basic Trace) ve ek iz (Additional Trace) dizilim alanı olmak üzere Şekil 3.10'da görüldüğü gibi iki bölümden oluşmaktadır.



Şekil 3.10. Kablosuz ağ paketi iz dosyası alanları [41]

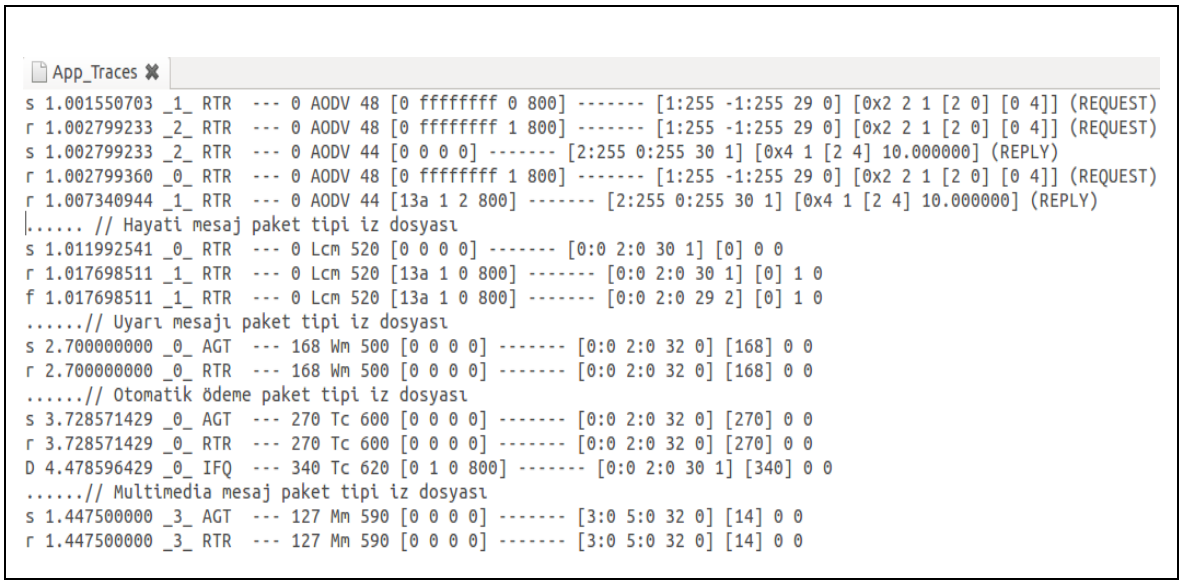
Şekil 3.10'da gösterilen alanların genel olarak anlamları şu şekildedir.
Event Type: Olayın tipini
(s: Gönderilen paket, r: Alınan paket, d: Düşen paket f: İletilen paket)
Time: Olayın zamanı,
Node ID: Düğüm numarası,
Packet Unique ID: Paketin eşsiz numarası,
Payload Type: Paketin Payload tipi,
Packet Size: Paketin boyutu
IP Trace: Kaynak düğüm ve hedef düğüm izleri
AODV trace: Protokol paketlerine ilişkin izleri
RREQ/RREP: Rota istek paketi (Route Request)/Rota cevap paketi(Route Reply)

Standart CBR (Constant Bit Rate) data paket trafiği kullanılarak oluşturulan kablosuz bir ağ uygulama dosyası cbr.tcl çalıştırıldığında üretilen cbr.tr uzantılı iz dosyasının görünümü Şekil 3.11'de verilmiştir.



Şekil 3.11. CBR iz dosyası

Uygulamada geliştirilen paket üreticilerine özgü paket tanımlamaları yapıldığından yine bu üreticilere özgü bir iz dosyası oluşturulması gereği doğmuştur. Bu sebeple her uygulamaya yönelik bir iz dosyası cmu-trace dosyasında oluşturulmuştur. Oluşturulan iz dosyalarının testi için yapılan benzetim sonucunda istenen izlerin oluştuğu gözlemlenmiştir. Alınan iz dosyası görünümü Şekil 3.12'de verilmiştir.



Şekil 3.12. Yeni iz dosyası

Paket türüne göre oluşturulan iz dosyası metotlarının ilgili dosyaya eklenmesi Şekil 3.13'te gösterilmiştir. Ayrıca gösterilen paket format metodu kodlarının bir örneği (format_lcm) Ek-2'de verilmiştir.

```
~/ns-2.35/trace/ cmu-trace.cc
            … // diğer kod satırları
void CMUTrace::format(Packet* p, const char *why)
{
            Ortak paket başlık yapısına erişim değişkenleri ve
            işaretçisi tanımlama
            … // diğer kod satırları
        switch( paket tipi ) {
             // paket tipine göre iz dosyasının seçilmesi
            case PT_AODV:
            aodv packet formatı metodunun çağrılması
                 break;
            …
            // ik uygulama iz dosyaları-application trace formats
            case PT_LCM:
            format_lcm() Lcm paket formatı metodunun çağrılması
                 break;
            case PT_WM:
            format_wm() Wm paket formatı metodunun çağrılması
                 break;
            case PT_TC:
            format_tc() Tc paket formatı metodunun çağrılması
                 break;
            case PT_MM:
            format_mm() Mm paket formatı metodunun çağrılması
                 break;
```

Şekil 3.13. Yeni iz dosyası metotları

3.2.4. Öncelikli kuyruğun oluşturulması

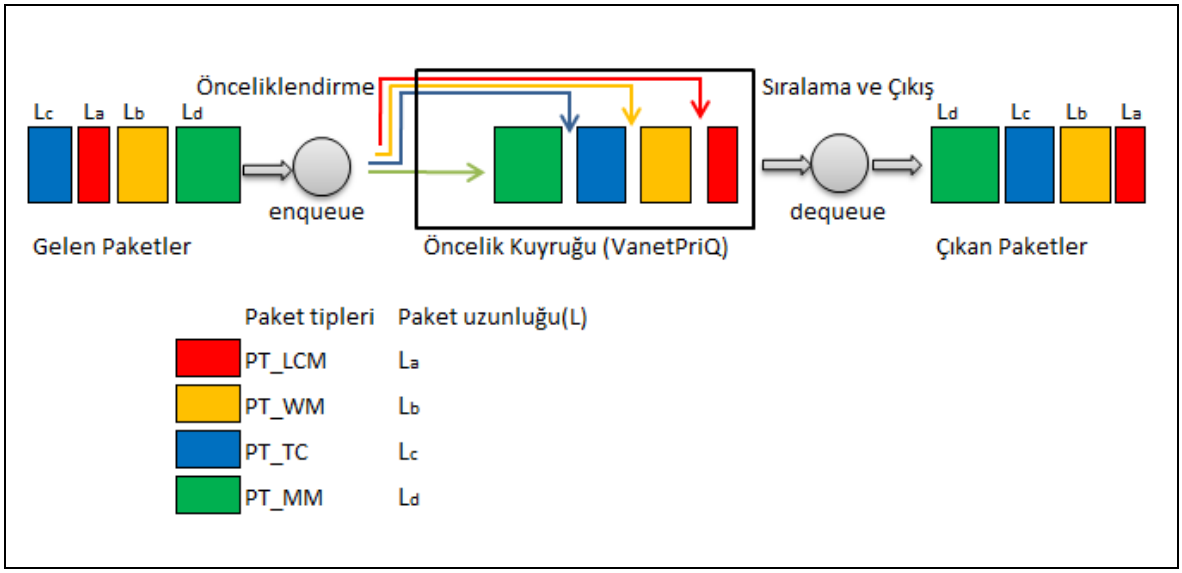
ns-2'de TCP ve UDP gibi ağ iletişim protokolleri ile mevcut trafik kaynakları FTP (File Transfer Protokol) ve CBR kullanılarak Drop tail, RED gibi yönlendirici kuyruk yönetimi yapıları ile ilgili uygulamalar yapılabilmektedir. Ancak tez kapsamında geliştirilen kuyruk yapısı Drop tail kuyruk yapısı ile karşılaştırılacağından bu kısımda Drop tail kuyruk yapısı hakkında bilgi verilmiş ve geliştirilen araçsal tasarsız ağ öncelikli kuyruk yapısı anlatılmıştır.

Drop tail

Bu kuyruk yapısı geleneksel ağ kuyruk yapısı olan FIFO yapısına göre çalışmaktadır. Ağ cihazına gelen paketler önem ve öncelik sırasına alınmadan tek bir kuyruğa konulur ve kuyruğa daha önce gelen paket daha önce çıkar. Burada kuyruk dolduktan sonra yeni paket kabul edilmez ve tüm paketler atılır.

Öncelikli kuyruk yapısı (VanetPriQ)

Hedeflenen öncelikli kuyruk yapısı ağ üzerindeki trafik akışını istikrarlı bir şekilde düzenleyerek ağın servis kalitesini artırmayı amaçlamaktadır. Bu çalışmada yeni paket üreticileri tarafından belli önceliklere sahip olarak üretilen paket tiplerini kullanarak servis kalitesini artıracak bir kuyruk yapısı geliştirilmiştir. Geliştirilen kuyruk yapısında tek bir kuyruk kullanılmıştır. Gelen paketler öncelik sırasına göre bu kuyruğa eklenmekte ve öncelik sırasına göre kuyruktan çıkmaktadırlar. Kuyrukta, kritik mesaj paketleri (PT_LCM) birinci önceliğe, uyarı mesaj paketleri (PT_WM) ikinci önceliğe, otomatik ödeme mesaj paketleri (PT_TC) üçüncü önceliğe ve multimedia mesaj paketleri (PT_MM) dördüncü önceliğe sahiptir ve kuyruğa da bu şekilde sıralanmaktadır. Geliştirilen kuyruk yapısı Şekil 3.14'de gösterilmiştir.



Şekil 3.14. VanetPriQ kuyruk yapısı

3.2.5. Simülasyon senaryolarının oluşturulması

ns-2 simülatöründe, kullanıcıların yazmış olduğu senaryoları yorumlamak için OTCL programlama dili kullanılır. OTCL dili aslında TCL (Tool Command Language) dilinin bir nesne uzantısı sayılmaktadır. TCL, California'da Berkeley Üniversitesinde John Ousterhout tarafından geliştirilen güçlü ve yorumlanabilir bir dinamik programlama dilidir. TCL yüksek ölçüde genişletilebilir bir programlama dilidir ve geniş bir kullanım alanına sahiptir. TCL dili tamamen C++ programlama dili ile uyumludur.

Tez kapsamında geliştirilen trafik üreticilerin çalışması, iz dosyalarının görülmesi ve tasarlanan kuyruk yapısının test edilip sonuçlarının alınması için çeşitli OTCL senaryo dosyaları oluşturulmuştur. Bu dosyalarda gerçekleştirilen işlem adımları şu şekildedir.

Parametrelerin kurulumu

Bu kısımda benzetimi yapılacak senaryonun kanal tipi, yayın modeli, mac tipi, bağlantı katmanı yönlendirme protokolü, kuyruk yapısı ve uzunluğu gibi parametreler kurularak açıklamaları Şekil 3.15'te gösterilmiştir.

```
#===================================
#     Simülasyon Parametrelerinin Kurulumu
#===================================
set val(chan)   Channel/WirelessChannel  ;# kanal tipi
set val(prop)   Propagation/TwoRayGround ;# sinyal yayın modeli
set val(netif)  Phy/WirelessPhy          ;# ağ ara yüz tipi
set val(mac)    Mac/802_11               ;# MAC tipi
set val(ifq)    Queue/DropTail/VanetPriQ ;# Arayüz kuyruk tipi
set val(ll)     LL                       ;# bağlantı katmanı
set val(ant)    Antenna/OmniAntenna      ;# anten modeli
set val(ifqlen) 300                      ;# kuyruk uzunluğu
set val(nn)     6                        ;# düğüm sayısı
set val(rp)     AODV                     ;# Yönlendirme
Protokolü
set val(x)      1104                     ;# X koordinatı
set val(y)      697                      ;# Y koordinatı
set val(stop)   10.0                     ;# simülasyon zamanı
```

Şekil 3.15. Simülasyon parametrelerinin kurulumu

Gösteri ve iz dosyalarının oluşturulması

Şekil 3.16'da gösteri nesnesinin ve iz dosyasının oluşturulması ve isimlendirilmesi yapılmıştır.

```
#İz dosyasının oluşturulması
set tracefile [open vpq_test.tr w]
$ns trace-all $tracefile

#.nam (gösteri nesnesi) dosyasının oluşturulması
set namfile [open vpq_test.nam w]
$ns namtrace-all $namfile
```

Şekil 3.16. Gösteri nesnesi ve iz dosyaları

Mobil düğüm parametrelerinin oluşturulması

Başlangıçta kurulumu yapılan parametreler mobil düğümler üzerine yüklenerek ilgili parametrelerle düğümlerin yapısı Şekil 3.17'de verildiği gibi oluşturulmuştur.

```
#===================================
# Mobil düğüm parametrelerinin ayarlanması
#===================================
$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) \
                -channel       $chan \
                -topoInstance  $topo \
                -agentTrace    ON \
                -routerTrace   ON \
                -macTrace      OFF \
                -movementTrace ON
```

Şekil 3.17. Mobil düğüm parametreleri

Düğüm tanımlamaları

Gösteri nesnesi üzerinde oluşturulan düğümlerin x, y ve z koordinatlarındaki başlangıç pozisyonları Şekil 3.18'de verildiği gibi atanmıştır.

```
#==================================
# Düğüm koordinat atamaları
#==================================
set n0 [$ns node]
$n0 set X_ 205
$n0 set Y_ 530
$n0 set Z_ 0.0n
$ns initial_node_pos $n0 20

# ... Diger düğümler
# n1,n2,n3,n4,n5
```

Şekil 3.18. Düğüm tanımlamaları

Agent tanımlamaları

Paket trafiği üretecek kaynak düğümlere ait UDP protokol yüklenicisi tanımlanmış ve ilgili olduğu düğüme Şekil. 3.19'da verildiği gibi bağlanmıştır.

```
#==================================
# Agent Tanımlamaları
#==================================
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1

set udp2 [new Agent/UDP]
$ns attach-agent $n2 $udp2

set udp3 [new Agent/UDP]
$ns attach-agent $n3 $udp3
```

Şekil 3.19. Agent tanımlamaları

Paket üretici tanımlamaları

Oluşturulan TCL dosyasının bu adımında kullanılacak paket üreticileri ve üretecekleri paketlere ilişkin parametreler ayarlanmaktadır. Paketlerin boyutu, iki paket arası gönderim aralığı ve bağlı olduğu agent'lar bu kısımda ayarlanmaktadır. Değiştirilmeyen parametreler için default olarak (ns-default.tcl) tanımlanmış değerler geçerli olacaktır. Yapılandırma ayarları Şekil 3.20'de verilmiştir.

```
#================================================
# Uygulama katmanı paket trafikleri
#================================================
set lcmApp [new Application/Traffic/LcmApp]
$lcmApp set packetSize_ 500
$lcmApp set interval_ 0.005
$lcmApp attach-agent $udp0

set wmApp [new Application/Traffic/WmApp]
$wmApp set packetSize_ 500
$wmApp set interval_ 0.005
$wmApp attach-agent $udp1

set tcApp [new Application/Traffic/TcApp]
$tcApp set packetSize_ 500
$tcApp set interval_ 0.005
$tcApp attach-agent $udp2

set mmApp [new Application/Traffic/MmApp]
$mmApp set packetSize_ 500
$mmApp set interval_ 0.005
$mmApp attach-agent $udp3
```

Şekil 3.20. Paket üreticilerin yapılandırılması

Ayrıca trafik üreticisinin sahip olduğu diğer değişkenlere parametreler *set* komutu kullanılarak atanabilmektedir.

Hedef düğüm ve kaynak düğüm bağlantılarının kurularak paket trafiklerinin başlatılması

Hedef düğüme bağlı null agent oluşturulmuş ve kaynaklar hedefe bağlanmıştır. Son işlem olarak trafiklerin belirlenen zamanlarda çalışması Şekil 3.21'de verildiği gibi sağlanmıştır.

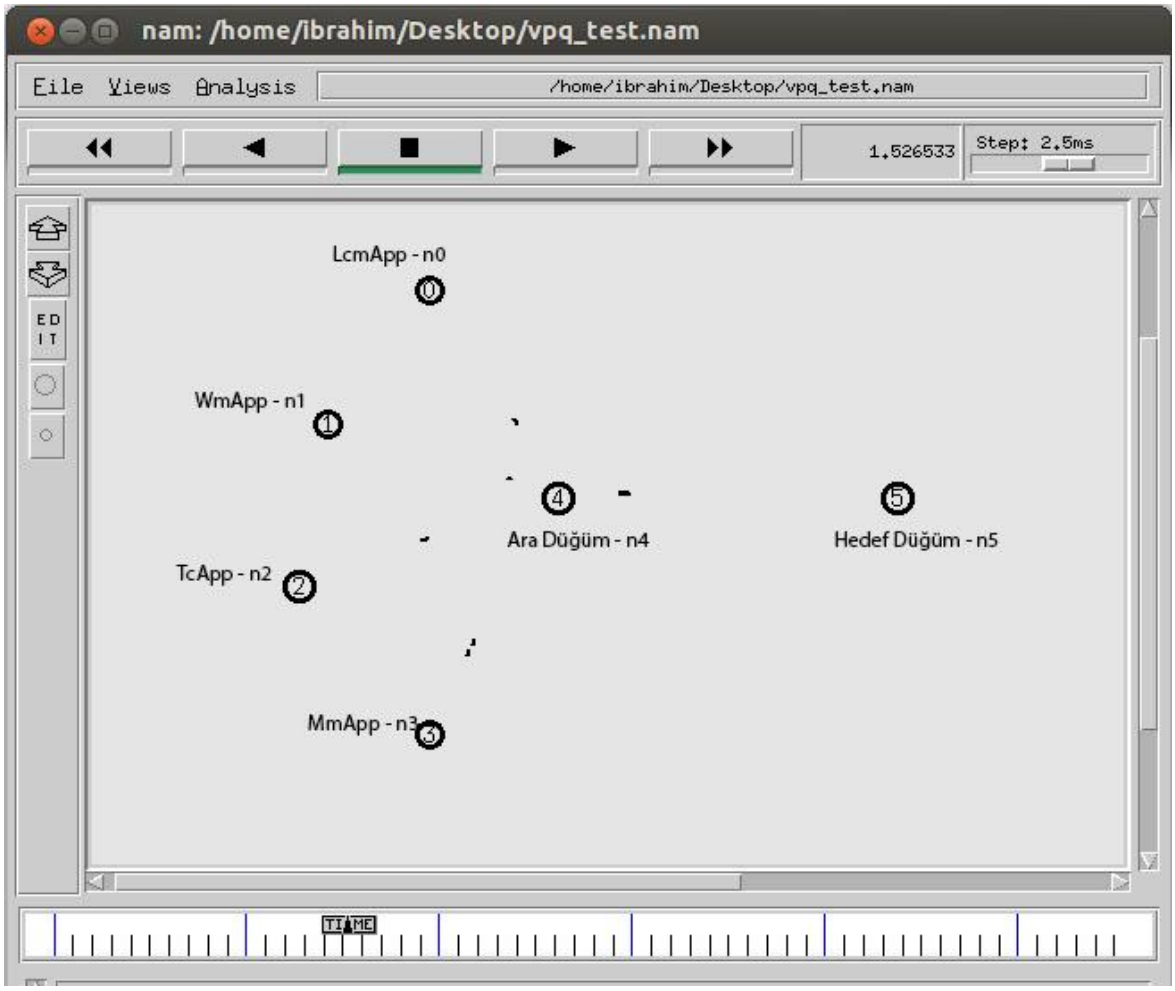
```
#Hedef Düğüm e tanımlı Null agent
set null [new Agent/Null]
$ns attach-agent $n5 $null

#Kaynaktan hedefe bağlantı
$ns connect $udp0 $null
$ns connect $udp1 $null
$ns connect $udp2 $null
$ns connect $udp3 $null

#Trafiklerin başlatılması ve durdurulması
$ns at 1.1 "$lcmApp start"
$ns at 3.0 "$lcmApp stop"
$ns at 1.0 "$wmApp start"
$ns at 3.0 "$wmApp stop"
$ns at 1.2 "$tcApp start"
$ns at 3.0 "$tcApp stop"
$ns at 1.3 "$mmApp start"
$ns at 3.0 "$mmApp stop"
```

Şekil 3.21. Trafik üreticilerin başlatılması ve durdurulması

Benzetimi yapılacak senaryoda dördü kaynak, biri ara ve biri de hedef olmak üzere 6 adet düğüm kullanılmıştır. Bir numaralı düğüme (n0) LcmApp trafik üretici, iki numaralı düğüme (n1) WmApp trafik üreticisi, üç numaralı düğüme TcApp trafik üreticisi ve dört numaralı düğüme (n3) ise MmApp trafik üreticileri bağlanmıştır. Kaynak düğümlerin kapsama alanı içerisine dört nolu (n4) ara düğümü yerleştirilmiştir. Kaynaklardan üretilen tüm paketlerin ara düğüm üzerinden hedefe gönderilmesi sağlanmıştır. Bu senaryo için düğüm konumları Şekil 3.22'de verilmiştir.



Şekil 3.22. Senaryo için düğüm konumları

3.2.6. Simülasyon senaryolarının uygulanması

Gerçek hayatta araçlar üzerinde 45KB (46080 byte), 145KB (158720 byte), 155KB (158720 byte) gibi kuyruk uzunluk değerleri kullanılmaktadır. Bu değerler arasındaki kuyruk uzunluğu seçimi kullanılan servise, ağdaki paket miktarına ve paket uzunluğuna göre ayarlanabilmektedir [46]. Ayrıca araçlar arası haberleşmede kullanılan paketin maximum uzunluğu 1400 byte olarak belirtilmiştir [47]. Yapılan çalışmalarda ise yaygın olarak 500 byte kullanıldığı ve bu değerin önerilen servise göre birkaç 100 byte daha artabileceği ifade edilmiştir [48].

Geliştirilen kuyruk yapısının test edilmesi için oluşturulan simülasyon senaryoları üzerinde kuyruk türü, kuyruk uzunluğu ve paket uzunluğu gibi parametreler gerçek hayattaki değerlere göre ayarlanmıştır. Senaryo 1, Çizelge 3.2'deki parametre değerleriyle sabit düğümler üzerinde gerçekleştirilmiştir.

Çizelge 3.2. Senaryo 1 simülasyon parametreleri

Senaryo	Kuyruk türü	Kuyruk uzunluğu (byte)	Paket uzunluğu (byte)	Simülasyon süresi (s)
1	(a) Drop Tail - VanetPriQ	150000	Eşit (500)	10
	(b) Drop Tail - VanetPriQ	150000	Farklı (500,600,700,800)	10
	(c) Drop Tail - VanetPriQ	75000	Eşit (500)	10

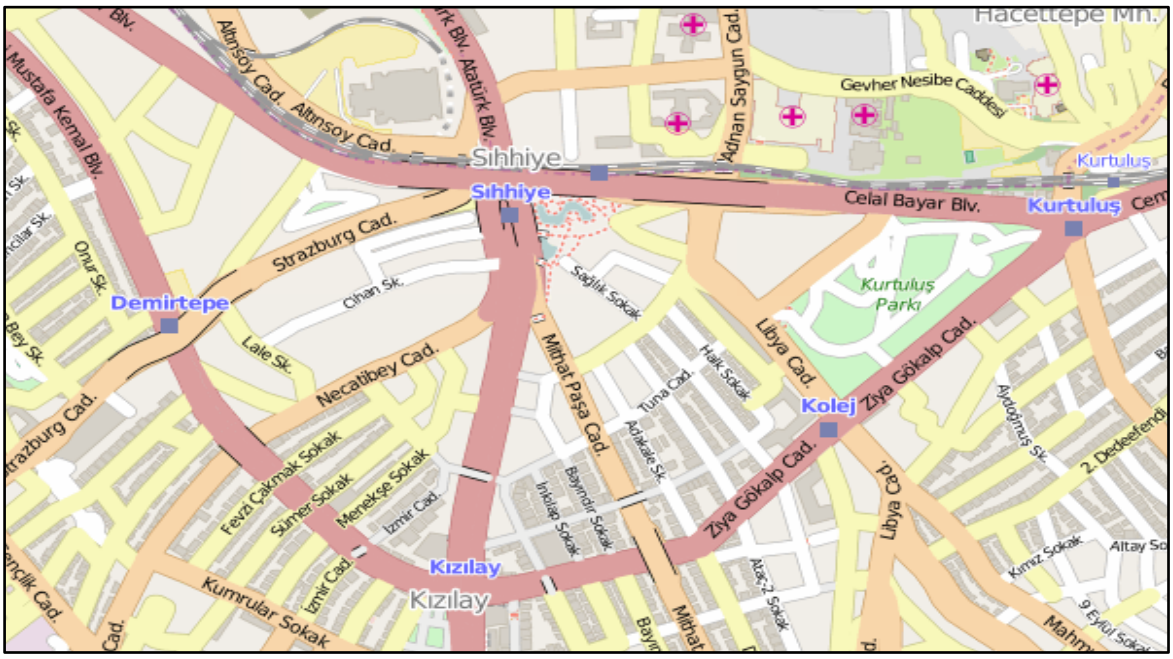
Bu senaryoda kuyruk türü, kuyruk uzunluğu ve paket uzunluğu parametreleri değiştirilerek geliştirilen kuyruk yapısının performansı ölçülmüştür. Daha sonra geliştirilen kuyruk yapısı kullanılarak hareketli düğümler üzerinde Çizelge 3.3'te gösterilen senaryo 2 oluşturulmuştur.

Çizelge 3.3. Senaryo 2 simülasyon parametreleri

Senaryo	Kuyruk türü	Kuyruk/paket uzunluğu (byte)	Hız (m/s)	Simülasyon süresi (s)
2	(a) VanetPriQ	75000/500	10	10
	(b) VanetPriQ	75000/500	30	10

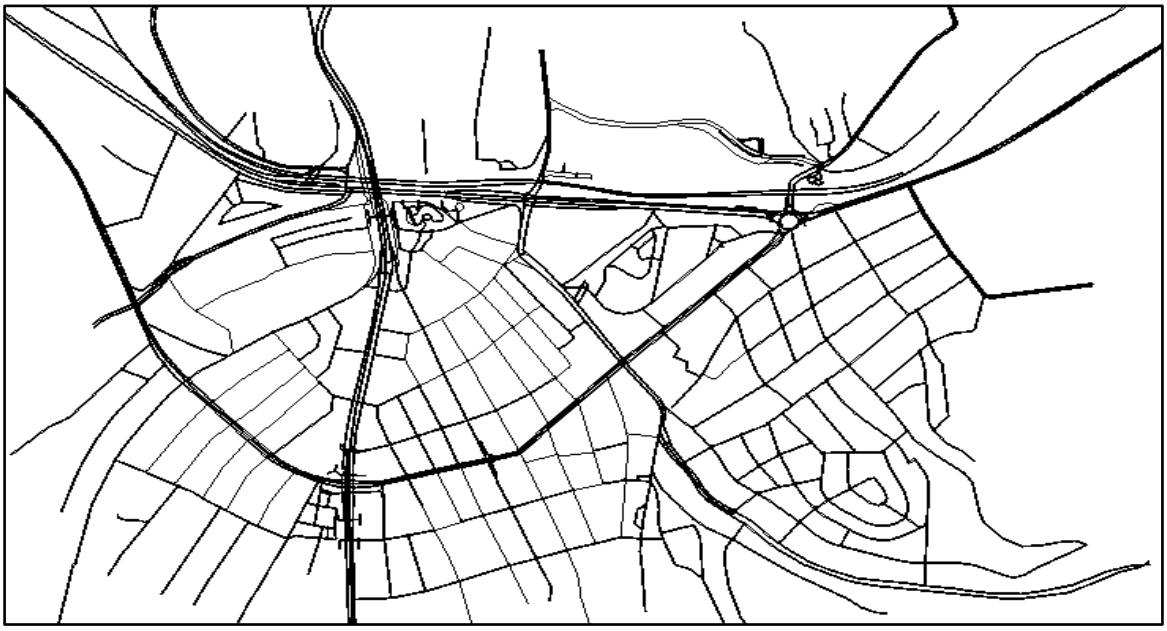
Senaryo 2 Ankara haritası üzerinde düşük ve yüksek hızlı düğümlerle(araçlarla) test edilmiştir. Bu senaryoda VanetPriQ yapısında 75000 byte'lık kuyruk uzunluğu ve 500 byte'lık paket uzunluğu kullanılmıştır. Araçlar Ankara haritası üzerinde düşük(10m/s) ve yüksek(30/ms) hızlarda hareket ettirilmiştir. Hareketli senaryo için kullanılacak harita, iz dosyası, araçların güzargah ve yol bilgileri ile bunların ns-2'ye aktarılması işlemleri aşağıda açıklanmıştır.

Hareketli düğüm senaryosu için kullanılan Şekil 3.23'de verilen Ankara haritası açık lisanslı OpenStreetMap [49] sitesinden indirilmiştir.



Şekil 3.23 Ankara haritası (tez.osm)

İndirilen *.osm uzantılı harita dosyası araçların hareket edebileceği Şekil 3.24'te verilen tez.net.xml uzantılı iz dosyasına dönüştürülmüştür.



Şekil 3.24 Ankara haritası iz dosyası (tez.net.xml)

Haritanın dosyasının iz dosyasına dönüştürmesi işlemi `netconvert --osm-files tez.osm -o tez.net.xml` komut satırı ile gerçekleştirilmiştir. Oluşturulan iz dosyası üzerinde araç ve bu araçlara ait güzergah bilgilerinin tanımlanabilmesi için Şekil 3.25'te verilen tez.rou.xml dosyası oluşturulmuştur. Bu dosya içinde araç adı, başlangıç zamanı, başlama hızı, maximum hızı ve güzergah bilgileri yer almaktadır. Senaryo 2'deki araçlara ilişkin parametreler bu dosya üzerinde ayarlanmıştır.

```
<routes>
    <vehicle id="cankaya_kurtulus_0" depart="0.00"
     departSpeed="10" maxSpeed="10">>
    <route edges="10652796#1 10652796#2 231667500#0 68695128#0
    68695128#1 68695128#2 68695128#3 68695128#4 68695128#5
    68695128#6 68695128#7 68695128#8 68695128#9 68695128#10
    68695128#11 8695128#12 68695128#13"/>
    </vehicle>
     … // diğer araç ve yol tanımlamaları
</routes>
```

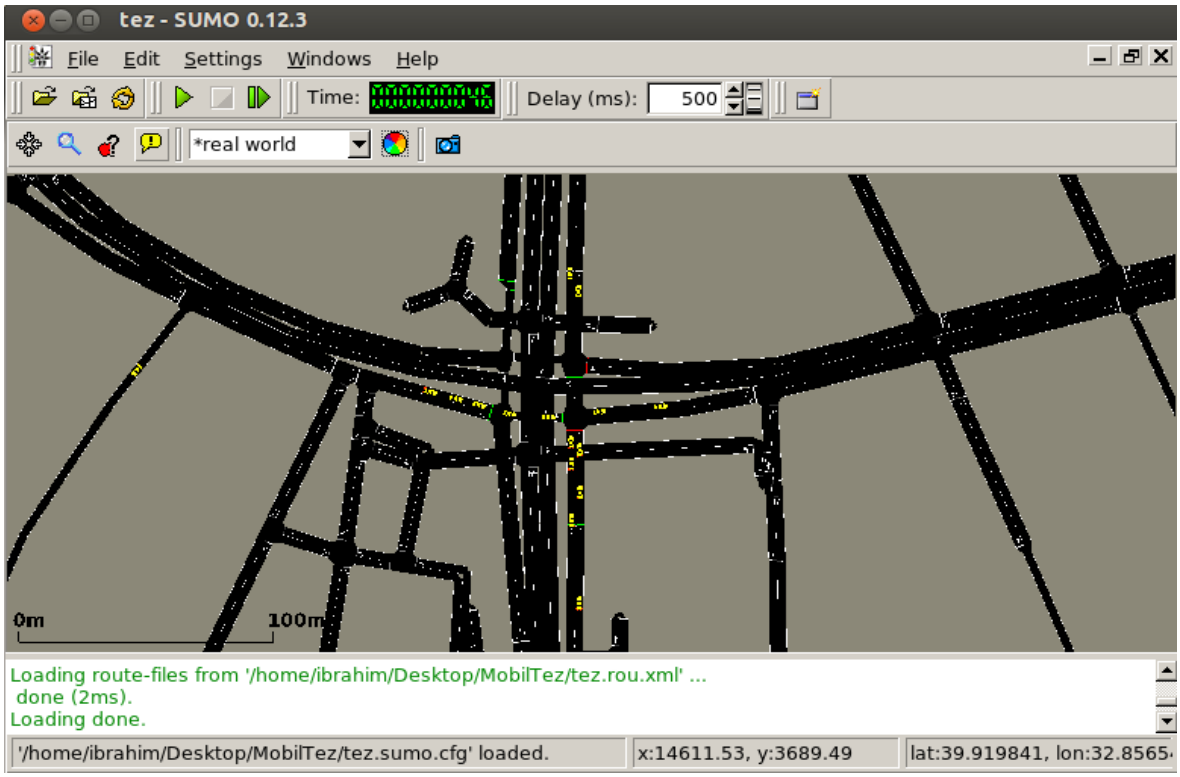
Şekil 3.25 Araç ve yön bilgileri (tez.rou.xml)

tez.rou.xml dosyası içine Kurtuluş'tan Maltepe istikametine 13, Kurtuluş'tan Sıhhiye istikametine 5, Maltepe'den Çankaya istikametine 5, Maltepe'den Kurtuluş istikametine 7, Çankaya'dan Sıhhiye istikametine 10, Kumrular Sokaktan Kurtuluş istikametine 5 ve Çankaya'dan Kurtuluş istikametine 5 araç olmak üzere toplam 50 adet araç tanımlaması yapılmıştır. Araçlar rastgele olacak şekilde belirlenen istikametlerde ve hızlarda hareket ettirilmiştir. Bu tanımlamalar sonrası araçların hareketleri ve çalışması SUMO simülatöründe görselleştirilmiştir. Görselleştirme işlemi için harita iz dosyası (tez.net.xml) ve araç yol bilgileri dosyası (tez.rou.xml) kullanılarak Şekil 3.26'te verilen SUMO konfigürasyon dosyası tez.sumo.cfg oluşturulmuştur.

```
<configuration>
 <input>
 <net-file value="/home/ibrahim/Desktop/MobilTez/tez.net.xml"/>
 <route-files
  value="/home/ibrahim/Desktop/MobilTez/tez.rou.xml"/>
 </input>
 <output>
 <netstate-dump
  value="/home/ibrahim/Desktop/MobilTez/tez.sumo.tr"/>
 </output>
 <time>
 <begin value="0"/>
 <end value="300"/>
 </time>
</configuration>
```

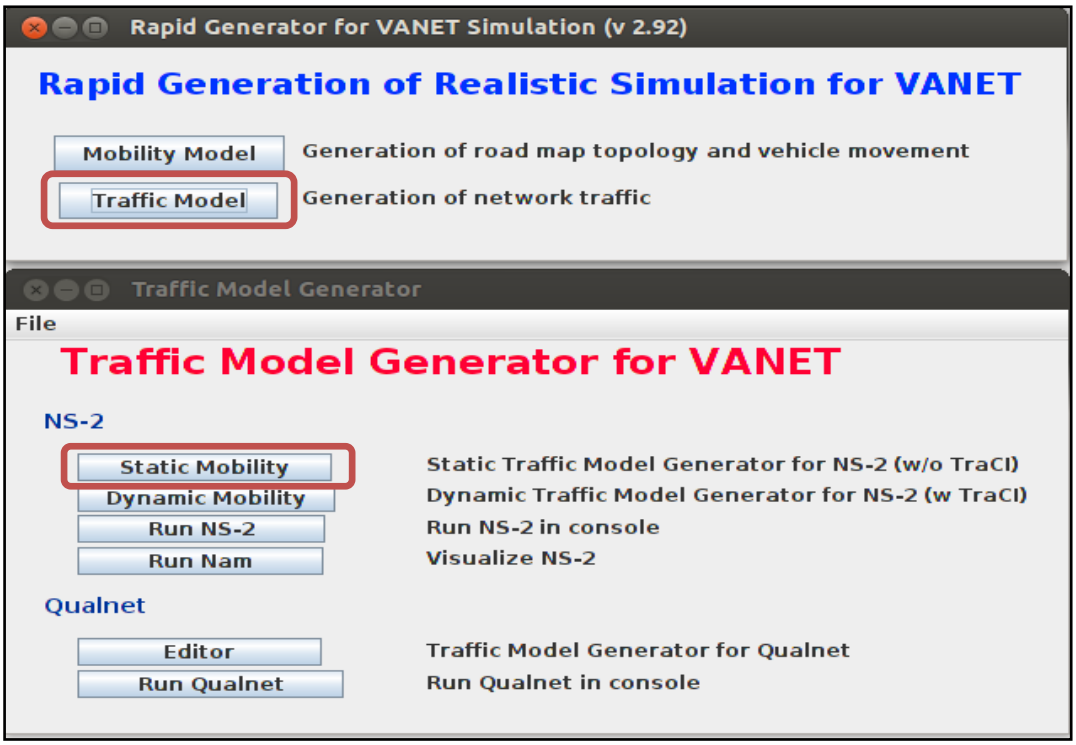
Şekil 3.26. SUMO konfigürasyon dosyası (tez.sumo.cfg)

Konfigürasyon dosyası SUMO simülatörü üzerinde çalıştırılarak araçların hareket tanımlamaları Şekil 3.27'da görüldüğü gibi test edilmiştir. Test sonrası tez.sumo.tr uzantılı iz dosyası çıktı olarak alınmıştır.



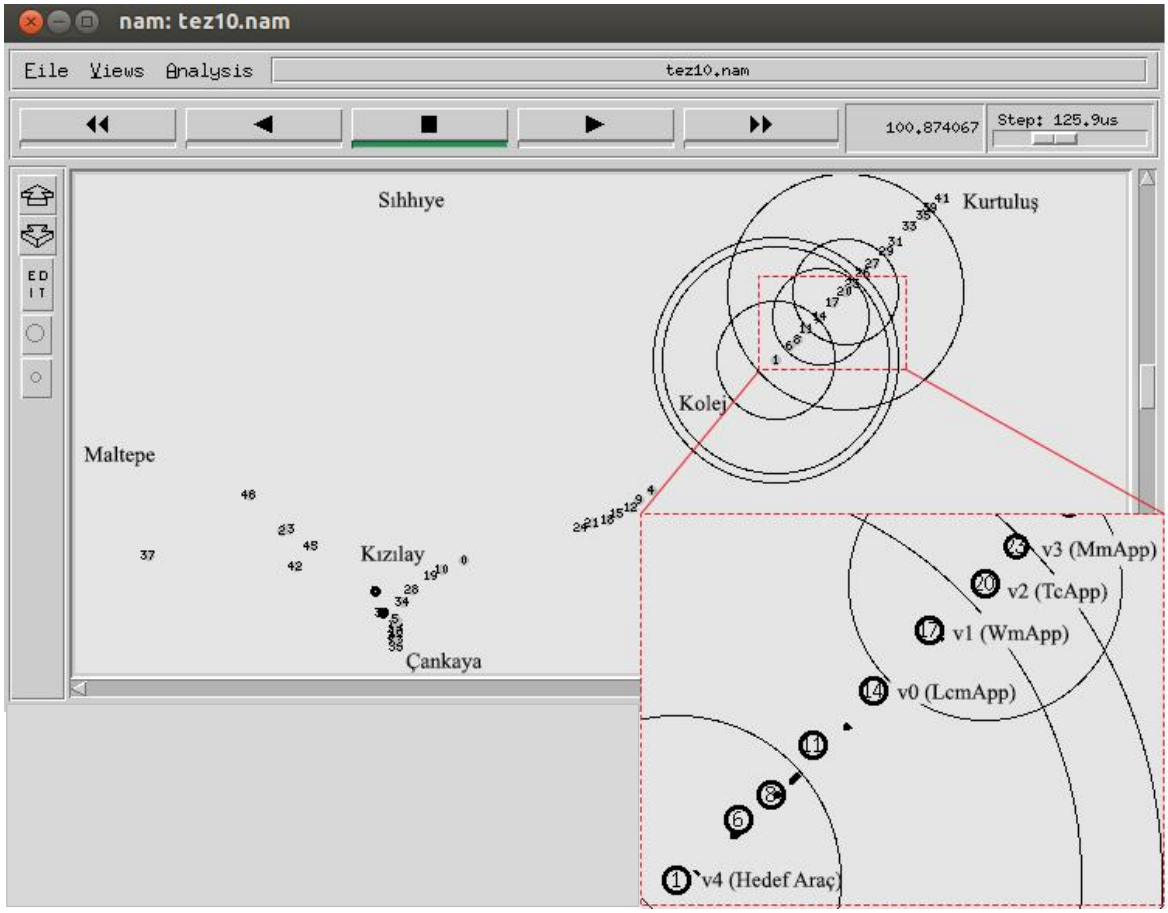
Şekil 3.27. SUMO gösteri ekranı

Araçların ns-2 ortamında da belirtilen parametre değerlerine göre hareketlendirilmesi için MOVE hareketlilik simülatörü kullanılmıştır. Şekil 3.28'de verilen ilk ekranda Traffic Model, ikinci ekranda Static Mobility butonları tıklanmıştır. Gelen ekranda yukarıda oluşturulan tez.sumo.tr dosyası ve harita dosyası (tez.net.xml) import edilerek ns-2'ye aktarımları yapılmıştır.



Şekil 3.28. MOVE simülatör ekranı

Senaryo 2 için Kurtuluş'tan Maltepe ve Sıhhıye istikametine giden araçlar kullanılmıştır. ns-2 üzerinde hareket ettirilen araçlara tez kamsamında geliştirilen trafik kaynakları Şekil 2.29'da gösterildiği gibi bağlanmıştır. Kaynak araçlardan (v0,v1,v2,v3) öncelikli paket trafikleri başlatılmış ve paketler kaynak hedef arasındaki 3 araç üzerinden hedef araca (v4) gönderilmiştir.



Şekil 3.29. Senaryo 2 araç konumları ve trafik kaynakları

Araçların Şekil 2.29'de kırmızı olarak çizilen kesitte senaryo 2 a ve b durumlarına göre benzetimleri yapılmıştır. Analizlerde özellikle hız durumlarına göre kaynak araçlarca üretilen paket sayıları, hedef araç tarafından alınan paket sayıları ve oranları, atılan paket sayıları ve paket türüne göre ortalama gecikme süreleri incelenmiştir. Elde edilen sonuçlar 4. bölümde detaylı olarak sunulmuştur.

4. UYGULAMA ÇIKTILARI VE ANALİZİ

4.1. Senaryo 1 Çıktısı ve Analizi

Senaryo 1'de paket uzunluğu, kuyruk türü ve kuyruk uzunluğu parametreleri değiştirilerek 3 farklı durum (a),(b),(c) oluşturulmuş ve bunların performansa etkisi incelenmiştir. Senaryo 1 a'da Drop tail ve VanetPriQ kuyruk yapısında kuyruk uzunluğu 150000 byte, tüm paket uzunlukları 500 byte olarak ayarlanmıştır. 10 saniyelik simülasyon sonucunda düğüm ve paket bazında elde edilen sonuçlar kuyruk türüne göre sırasıyla Çizelge 4.1 ve Çizelge 4.2'de verilmiştir.

Çizelge 4.1. Senaryo 1 (a) Drop tail çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	500	500	500	-
Kaynak düğüm	node 0	node 1	node 2	node 3	-
Üretilen paket sayısı	233	233	233	233	932
Atılan paket sayısı	34	0	39	13	86
İletilen paket sayısı	196	223	191	218	838
İletilen paket oranı (%)	84,7	100	81,5	93,5	89,9

Hedef düğüm node 5	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	196	233	190	218	837
Alınan paket oranı (%)	84,7	100	81	93	89,8
Paket bazlı ort. gecikme (s)	4,1	3,9	4,0	4,4	4,09

Kayıp paket sayısı	3	0	4	2	9
Kayıp paket oranı (%)	1,2	0	1,7	0,8	3,8

Senaryo 1 a'daki Drop tail kuyruk yapısında tüm kaynak düğümler (node 0, node 1, node 2, node 3) eşit sayıda (233) paket üretmiş ve toplam 932 (%100) paket üretimi yapılmıştır. Bu paketler ara düğüm (node 4) üzerinden hedef düğüme (node 5) iletilmiştir. Hedef düğüm paketlerin 837 tanesini (%89,8) almıştır. Toplamda üretilen paketin 86 tanesi (%36,9) atılmış, 9 tanesi (%3,8) ise kaybolmuştur. Hedef düğüm tarafından alınan paketler düğüm bazında incelendiğinde sırasıyla en fazla uyarı mesaj (WM-%100), multimedia mesaj (MM-%93), kritik mesaj (LCM-%84) ve otomatik ödeme mesaj (TC-%81) paketlerinin alındığı görülmüştür.

Çizelge 4.2. Senaryo 1 (a) VanetPriQ çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	500	500	500	-
Kaynak düğüm	node 0	node 1	node 2	node 3	-
Üretilen paket sayısı	233	233	233	233	932
Atılan paket sayısı	0	0	0	0	0
İletilen paket sayısı	230	233	228	233	924
İletilen paket oranı (%)	98,7	100	97,8	100	99,1

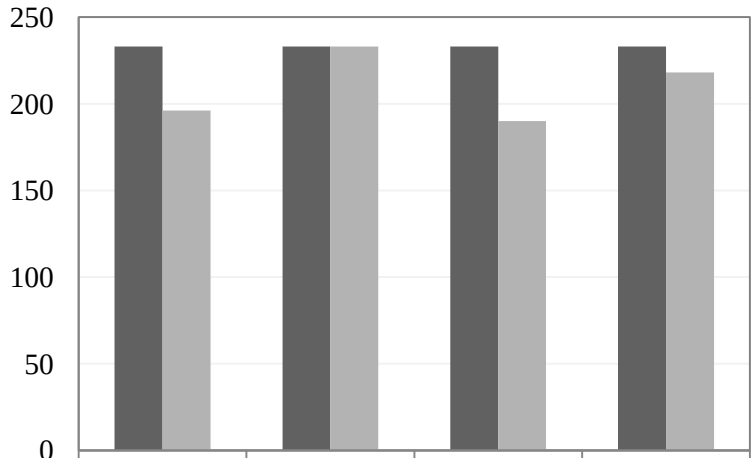
Hedef düğüm node 5	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	229	211	179	140	759
Alınan paket oranı (%)	98	90	76	60	81,4
Paket bazlı ort. gecikme (s)	3,9	4,2	4,6	4,0	4,1

Kayıp paket sayısı	4	22	54	93	173
Kayıp paket oranı (%)	1,7	9,4	23	40	18,5

Senaryo 1 a'daki VanetPriQ kuyruk yapısında tüm kaynak düğümler (node 0, node 1, node 2, node 3) eşit sayıda paket üreterek toplam 932 (%100) paket üretimi yapılmıştır. Bu paketler ara düğüm (node 4) üzerinden hedef düğüme (node 5) iletilmiştir. Hedef düğüm paketlerin 759 tanesini (%81,4) almıştır. Üretilen paketlerin 173 tanesi (%18,5) kaybolmuş ancak hiç paket atılmamıştır. Hedef düğüm tarafından alınan paketler incelendiğinde sırasıyla en fazla kritik mesaj (LCM-%98), uyarı mesaj (WM-%90), otomatik ödeme mesaj (TC-%76), multimedia mesaj (MM-%60) paketlerinin alındığı görülmüştür.

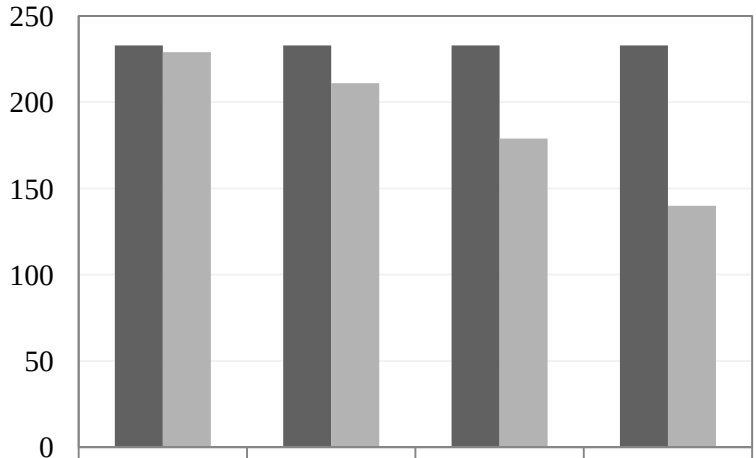
Drop tail kuyruk yapısında paketler herhangi bir öncelik sonralık ilişkisi içinde olmadığından paket türleri geldiği sırada kuyruktan ayrılmıştır. Hedefin aldığı paket sayıları WM-233>MM-218>LCM-196>TC-190 ve oranları 100>92>84,7>81 şeklinde oluşmuş olup önceliğine göre sıralı bir ilişki göstermemiştir. VanetPriQ kuyruk yapısında ise paketlere öncelik ataması yapıldığından kuyruk içerisinde paketler önceliğine göre sıralanmış ve o sırada kuyruktan gönderilmiştir. Bu nedenle VanetPriQ'da hedefin aldığı paket miktarları (LCM-229>WM-211>TC-179>MM-140) ve yüzde olarak oranları büyükten küçüğe (98>90>76>60) sıralı bir ilişki oluşturmuştur. Hedefin aldığı LCM paketleri oranı VanetPriQ ile %13,3 oranında artma göstermiştir. Diğer paketlerde ise önceliklerine göre azalma görülmüştür.

Senaryo 1 a'da kullanılan kuyruk yapılarında üretilen ve hedef tarafından alınan paket sayıları grafiği Şekil 4.7 ve Şekil 4.8'de verilmiştir.



	LCM	WM	TC	MM
Üretilen paket sayısı	233	233	233	233
Alınan paket sayısı	196	233	190	218

Şekil 4.1. Senaryo 1 (a) Drop tail üretilen ve alınan paket sayıları



	LCM	WM	TC	MM
Üretilen paket sayısı	233	233	233	233
Alınan paket sayısı	229	211	179	140

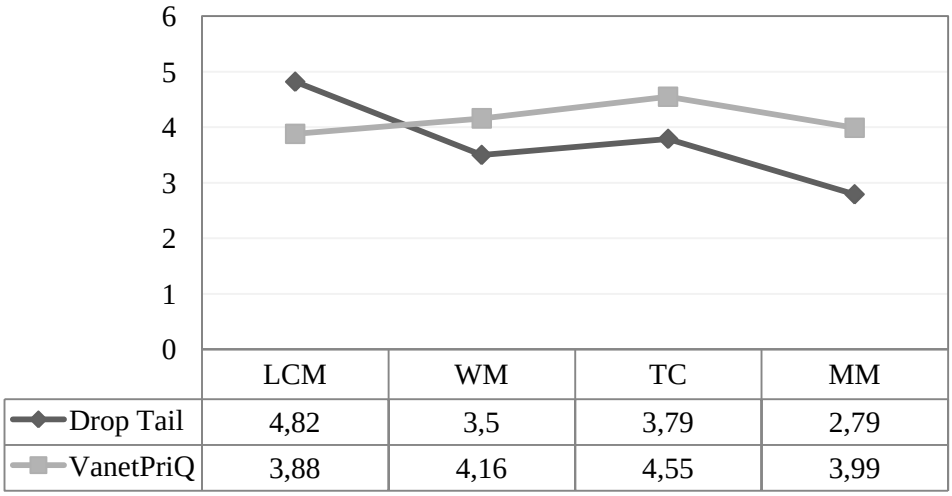
Şekil 4.2. Senaryo 1 (a) VanetPriQ üretilen ve alınan paket sayıları

Senaryo 1 a'da paket bazlı gecikme oranlarının belirlenebilmesi için VanetPriQ'daki hedefin aldığı paket sayıları dikkate alınmıştır. Bu paket sayılarının Drop tail'de ne kadar sürede alındığı hesaplanmıştır. Belirtilen sayılardaki paketlerin her iki kuyruk yapısında ortalama hedef tarafından alınma süreleri bulunmuştur. Aradaki süre farkına göre oransal olarak gecikmeler hesaplanmıştır. Hesaplamaya göre oluşan gecikme değerleri Çizelge 4.3'te verilmiştir.

Çizelge 4.3. Senaryo 1 (a) paket bazlı ortalama gecikme değerleri

Paket türü	Paket sayısı	Drop tail (s)	VanetPriQ(s)	Fark (s)	Oran (%)	Durum
LCM	229	4,82	3,88	-0,94	20	Azalma
WM	211	3,5	4,16	0,66	15	Artma
TC	179	3,79	4,55	0,76	20	Artma
MM	140	2,79	3,99	1,2	43	Artma

Sonuçlara göre VanetPriQ öncelikli kuyruk yapısı LCM paketlerinin ortalama gecikmesini %20 oranında azaltmıştır. Ancak WM paketlerini ortalama gecikmesini %15, TC paketlerini %20, MM paketlerini ise %43 oranında artırmıştır. Yukarıdaki çizelge değerlerine göre oluşan paket gecikme değişimleri Şekil 4.3'te yer almaktadır.



Şekil 4.3. Senaryo 1 (a) paket bazlı gecikme değişimleri

Senaryo 1 b'de Drop tail ve VanetPriQ kuyruk yapılarında kuyruk uzunluğu 150000 byte, paket uzunlukları ise LCM paketleri için 500 byte, WM paketleri için 600 byte, TC paketleri için 700 byte ve MM paketleri için 800 byte olarak ayarlanmıştır. Her iki kuyruk yapısında paket uzunluğu etkisi incelenmiştir. 10 saniyelik simülasyon sonucunda düğüm ve paket bazında elde edilen sonuçlar kuyruk türlerine göre sırasıyla Çizelge 4.4 ve Çizelge 4.5'te verilmiştir.

Çizelge 4.4. Senaryo 1 (b) Drop tail çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	600	700	800	-
Kaynak düğüm	node 0	node 1	node 2	node 3	-
Üretilen paket sayısı	252	211	180	158	801
Atılan paket sayısı	46	26	17	0	89
İletilen paket sayısı	204	184	162	158	708
İletilen paket oranı (%)	80,9	87,2	0,9	100	88,3

Hedef düğüm node 5	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	204	183	162	158	707
Alınan paket oranı (%)	80	86	90	100	88,2
Paket bazlı ort. gecikme (s)	4,44	4,48	4,14	3,17	4,10

Kayıp paket sayısı	2	2	1	0	5
Kayıp paket oranı (%)	0,7	0,9	0,5	0	0,6

Senaryo 1 b'deki Drop tail kuyruk yapısında üretilecek paket uzunlukları farklı olması nedeniyle kaynak düğümlerden farklı sayıda paket üretimi yapılmıştır. Kaynaklar bu uzunlukları dikkate alarak toplamda 801 (%100) paket üretimi yapmıştır. Bu paketler ara düğüm (node 4) üzerinden hedef düğüme (node 5) iletilmiştir. Hedef düğüm paketlerin 707 tanesini (%88,2) almıştır. Üretilen toplam paketin 89 tanesi (%11,1) atılmış, 5 tanesi (%0,6) ise kaybolmuştur. Hedef düğümün aldığı paket oranları incelendiğinde sırasıyla en fazla MM paketleri (%100), TC paketleri (%90), WM paketleri (%86), LCM paketleri (%80) alınmıştır.

Çizelge 4.5. Senaryo 1 (b) VanetPriQ çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	600	700	800	-
Kaynak düğüm	node 0	node 1	node 2	node 3	-
Üretilen paket sayısı	252	211	180	158	801
Atılan paket sayısı	1	0	0	0	1
İletilen paket sayısı	248	211	175	158	792
İletilen paket oranı (%)	98,4	100	97,2	100	98,8

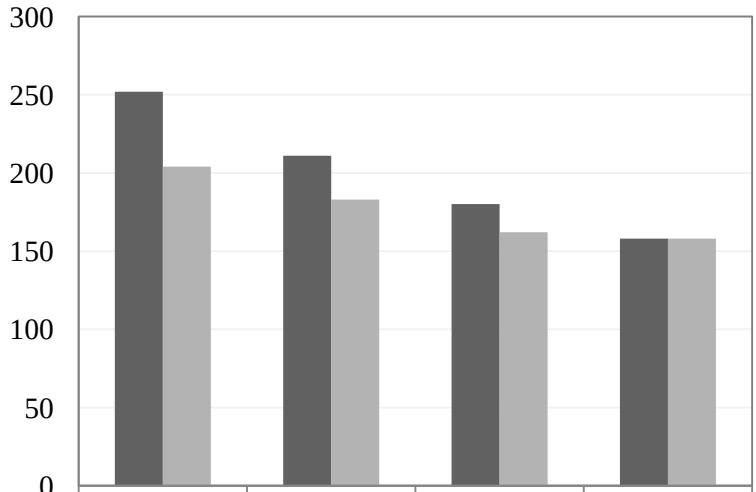
Hedef düğüm node 5	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	247	198	134	74	653
Alınan paket oranı (%)	98	93	74	46	81,5
Paket bazlı ort. gecikme (s)	4,62	4,04	3,46	3,93	4,13

Kayıp paket sayısı	4	13	46	84	147
Kayıp paket oranı (%)	1,5	6,1	25,5	53	18,3

Senaryo 1 b'deki VanetPriQ kuyruk yapısında toplam 801 (%100) paket üretimi yapılmıştır. Bu paketler ara düğüm (node 4) üzerinden hedef düğüme (node 5) iletilmiştir. Hedef düğüm paketlerin 653 tanesini (%81,5) almıştır. Üretilen paketlerin 147 tanesi (%18,3) kaybolmuş, 1 tanesi ise atılmıştır. Hedef düğümün aldığı paket oranları incelendiğinde sırasıyla en fazla LCM paketleri (%98), WM paketleri (%93), TC paketleri (%74), MM paketleri (%46) alınmıştır. VanetPriQ farklı paket uzunluklarında Drop tail'e göre LCM paketlerini %18, WM paketlerini ise %7 oranında artırarak hedefe ulaştırmıştır. TC paketlerini %16 ve MM paketlerini ise % 64 oranında azaltarak hedefe iletmiştir.

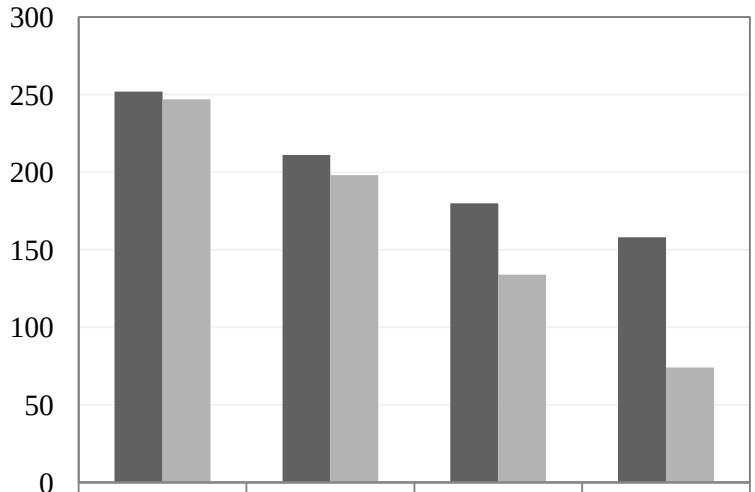
Senaryo 1 b'de farklı paket uzunlukları nedeniyle toplam üretilen paket sayısını düşmüştür. Ancak oransal olarak Senaryo 1 a ve b'de hedef tarafından alınan paket oranları benzer sonuçlar vermiştir. Paket uzunlukları nedeniyle paket sayısı azalsa da hedefe iletilen toplam paket oranları birbirine yakın değerler (Drop tail a-%88,2 / b-%89,8 ve VanetPriQ a-%81,4 / b- %81,5) çıkmıştır.

Senaryo 1 b'de kullanılan kuyruk yapılarında üretilen ve hedef tarafından alınan paket sayıları grafiği Şekil 4.4 ve Şekil 4.5'te verilmiştir.



	LCM	WM	TC	MM
Üretilen paket sayısı	252	211	180	158
Alınan paket sayısı	204	183	162	158

Şekil 4.4. Senaryo 1 (b) Drop tail üretilen ve alınan paket sayıları



	LCM	WM	TC	MM
Üretilen paket sayısı	252	211	180	158
Alınan paket sayısı	247	198	134	74

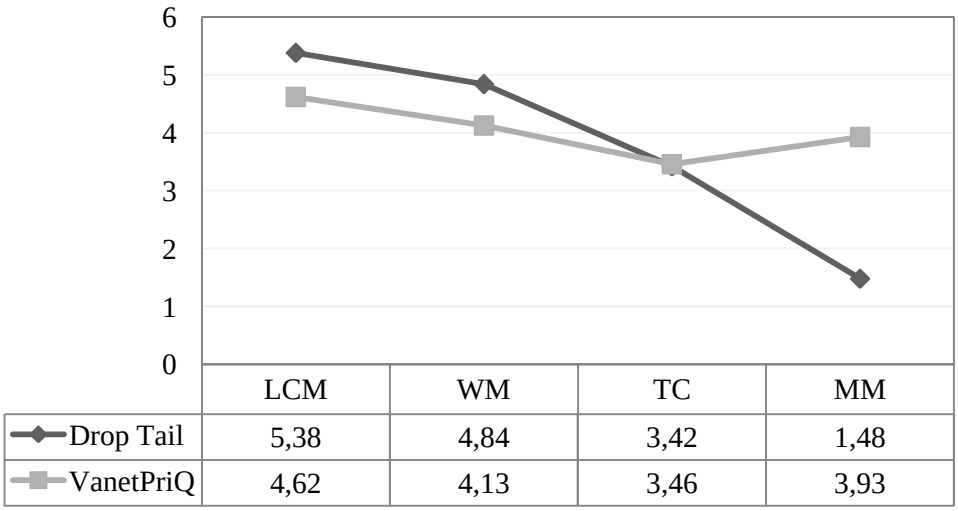
Şekil 4.5. Senaryo 1 (b) VanetPriQ üretilen ve alınan paket sayıları

Senaryo b'deki duruma göre oluşan farklı uzunluklardaki paketlerin önceliklerine göre ortalama gecikme değerleri Çizelge 4.6'da verilmiştir.

Çizelge 4.6. Senaryo 1 (b) paket bazlı ortalama gecikme değerleri

Paket türü	Paket sayısı	Drop tail (s)	VanetPriQ(s)	Fark (s)	Oran (%)	Durum
LCM	247	5,38	4,62	-0,76	14	Azalma
WM	198	4,84	4,13	-0,71	14	Azalma
TC	134	3,42	3,46	0,04	1	Artma
MM	74	1,48	3,93	2,45	165	Artma

VanetPriQ öncelikli kuyruk yapısı LCM ve WM paketlerinin ortalama gecikmesini %14 oranında azaltmıştır. Ancak TC paketlerinin ortalama gecikmesini %1, MM paketlerinin ise %165 oranında artırmıştır. Yukarıdaki çizelge değerlerine göre paketlerin gecikme değişimleri ise Şekil 4.6'da verilmiştir.



Şekil 4.6. Senaryo 1 (b) paket bazlı gecikme değişimleri

Senaryo 1 c'de Drop tail ve VanetPriQ kuyruk yapılarında kuyruk uzunluğu 75000 byte, paket uzunlukları 500 byte olarak ayarlanmıştır. Her iki kuyrukta kuyruk uzunluğu parametresinin etkisi incelenmiştir. 10 saniyelik simülasyon sonucunda düğüm ve paket bazında elde edilen sonuçlar kuyruk türlerine göre sırasıyla Çizelge 4.7 ve Çizelge 4.8'de verilmiştir.

Çizelge 4.7. Senaryo 1 (c) Drop tail kuyruk yapısı çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	500	500	500	-
Kaynak düğüm	node 0	node 1	node 2	node 3	-
Üretilen paket sayısı	233	233	233	233	932
Atılan paket sayısı	73	36	68	54	231
İletilen paket sayısı	158	197	164	178	697
İletilen paket oranı (%)	67,8	84,5	70,4	76,4	74,8

Hedef düğüm node 5	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	158	197	164	177	696
Alınan paket oranı (%)	67,8	84,5	70,3	75,9	74,6
Paket bazlı ort. gecikme (s)	3,31	3,29	3,48	3,60	3,42

Kayıp paket sayısı	2	0	1	2	5
Kayıp paket oranı (%)	0,8	0	0,4	0,8	5,3

Senaryo 1 c Drop tail kuyruk yapısında kaynak düğümler eşit sayıda (233) paket üretmiş ve toplamda 932 (%100) paket üretimi yapılmıştır. Üretilen bu paketlerin 696 tanesi (%74,6) hedef düğüm tarafından alınmıştır. Toplamda üretilen paketin 231 tanesi (%24,7) atılmış, 5 tanesi (%5,3) ise kaybolmuştur. Hedef düğüm tarafından alınan paket türleri incelendiğinde en fazla WM paketlerinin (%84,5), ikinci olarak MM paketlerinin (%75,9), üçüncü olarak TC paketlerinin (%70,3) ve son olarak LCM paketlerinin (%67,8) alındığı görülmüştür.

Çizelge 4.8. Senaryo 1 (c) VanetPriQ kuyruk yapısı çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	500	500	500	-
Kaynak düğüm	node 0	node 1	node 2	node 3	-
Üretilen paket sayısı	233	233	233	233	932
Atılan paket sayısı	16	41	42	26	125
İletilen paket sayısı	195	179	173	193	740
İletilen paket oranı (%)	83,7	76,8	74,2	82,8	79,4

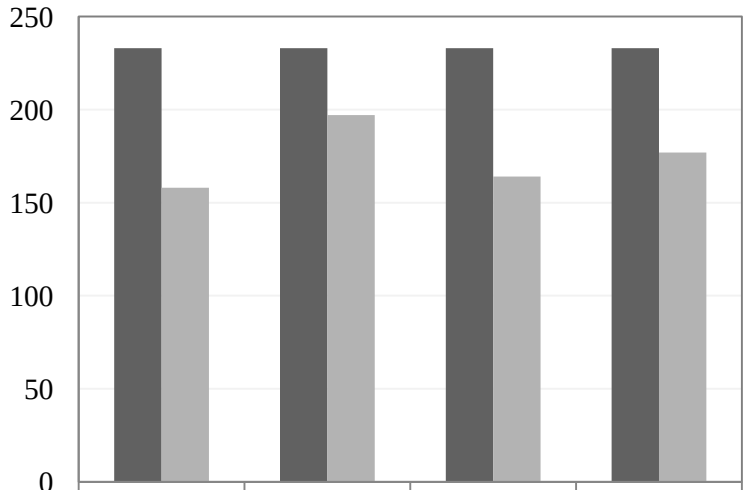
Hedef düğüm node 5	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	194	162	126	110	592
Alınan paket oranı (%)	83	69	54	47	63,5
Paket bazlı ort. gecikme (s)	3,16	3,25	3,44	3,11	3,23

Kayıp paket sayısı	23	30	65	97	215
Kayıp paket oranı (%)	9,8	12,8	27,8	41,6	23

VanetPriQ kuyruk yapısında tüm kaynak düğümler (node 0, node 1, node 2, node 3) eşit sayıda (233) paket üreterek toplam 932 (%100) paket üretimi yapılmıştır. Hedef düğüm üretilen paketlerin 592 tanesini (%63,5) almıştır. Üretilen paketlerin 215 tanesi (%23) kaybolmuş, 125 tanesi (%13,4) atılmıştır. Hedef düğümün aldığı paket türleri incelendiğinde sırasıyla en fazla LCM paketleri (%83), WM paketleri (%69), TC paketleri (%54), MM paketleri (%47) alınmıştır.

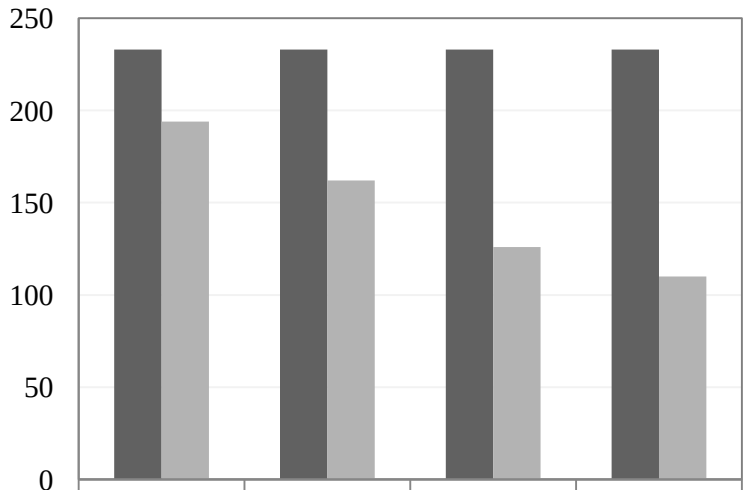
Senaryo 1 c'de kuyruk uzunluğunun azaltılması sonucu atılan paket miktarında senaryo 1 a'ya göre artış görülmüştür. VanetPriQ için Senaryo 1 a'da üretilen paket trafiği için yeterli kuyruk uzunluğu (150000 byte) tanımlandığından atılan paketin olmadığı tespit edilmiştir. Senaryo c'de ise kuyruk uzunluğu (75000 byte) trafik yoğunluğunda dolmuş ve paketlerin atılmasına neden olmuştur. Bu nedenle üretilen toplam paketin hedef tarafından alınma oranı da (%81,4'ten %63,5'e) düşmüştür.

Senaryo 1 c'de kullanılan kuyruk yapılarında üretilen ve hedef tarafından alınan paket sayıları grafiği Şekil 4.7 ve Şekil 4.8'de verilmiştir.



	LCM	WM	TC	MM
Üretilen paket sayısı	233	233	233	233
Alınan paket sayısı	158	197	164	177

Şekil 4.7. Senaryo 1 (c) Drop tail üretilen ve alınan paket sayıları



	LCM	WM	TC	MM
Üretilen paket sayısı	233	233	233	233
Alınan paket sayısı	194	162	126	110

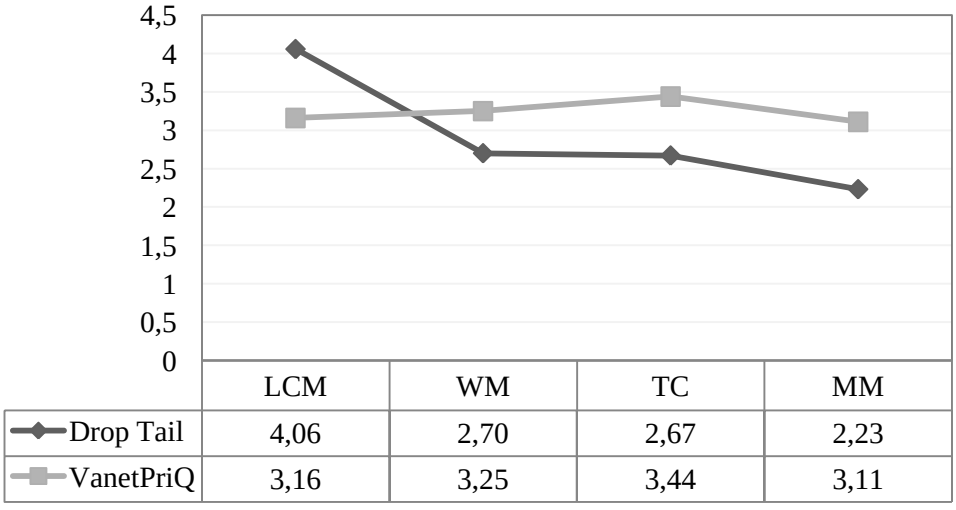
Şekil 4.8. Senaryo 1 (c) VanetPriQ üretilen ve alınan paket sayıları

Senaryo 1 c'de VanetPriQ kuyruk yapısının paket bazında ortalama gecikmeye etkisini incelendiğinde VanetPriQ öncelikli kuyruk yapısı LCM paketlerini %22 oranında daha kısa sürede hedefe iletmiştir. Ancak WM paketlerini %20, TC paketlerini %29, MM paketlerini ise %39 oranında daha geç hedefe ulaştırmıştır. Paketlerin önceliklerine göre ortalama gecikme değerleri grafiği Çizelge 4.9'da verilmiştir.

Çizelge 4.9. Senaryo 1 (c) paket bazlı ortalama gecikme değerleri

Paket türü	Paket sayısı	Drop tail (s)	VanetPriQ (s)	Fark (s)	Oran (%)	Durum
LCM	194	4,06	3,16	-0,9	22	Azalma
WM	162	2,70	3,25	0,55	20	Artma
TC	126	2,67	3,44	0,77	29	Artma
MM	110	2,23	3,11	0,88	39	Artma

Çizelge 4.9'daki değerlere göre oluşan paket bazlı gecikme değişimleri grafiği ise Şekil 4.9'da verilmiştir.



Şekil 4.9. Senaryo 1 (c) paket bazlı gecikme değişimleri

4.2. Senaryo 2 Çıktısı ve Analizi

Senaryo 2'de hareketli düğümlerin(araçların) kuyruk uzunluğu parametresi 75000 byte, paket uzunluğu 500 byte olarak ayarlanmıştır. Araçlar öncelikle düşük(10m/s) hızda hareket ettirilerek düğüm ve paket bazında Çizelge 4.10'daki sonuçlar alınmıştır.

Çizelge 4.10. Senaryo 2 (a) 10m/s hızda simülasyon çıktıları

Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	500	500	500	-
Kaynak araçlar	v0	v1	v2	v3	-
Üretilen paket sayısı	225	225	225	225	900
Atılan paket sayısı	0	0	2	161	163
İletilen paket sayısı	225	225	223	64	737
İletilen paket oranı (%)	100,0	100,0	99,1	28,4	81,9

Hedef araç v4	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	225	225	223	64	737
Alınan paket oranı (%)	100,0	100,0	99,1	28,4	81,9
Paket bazlı ort. gecikme (s)	0,97	1,20	2,00	1,09	1,29

Kayıp paket sayısı	0	0	0	0	0
Kayıp paket oranı (%)	0	0	0	0	0

Şekil 3.28'de gösterilen kaynak araçlardan(v0-LCM,v1-WM,v2-TC,v3-MM) toplamda 900 (%100) paket üretilmiştir. Hedef araç(v4) üretilen paketlerin 737 tanesini (%81,9) almıştır. Üretilen paketlerin 163 tanesi (%18) atılmıştır. Atılan paketlerin çoğunluğu ise önceliği düşük olan MM paketinden olduğu görülmüştür. Hedef araç 10m/s hızda LCM ve WM paketlerinin tamamını(%100) almıştır. TC paketlerinin 223 tanesini(%99,1), MM paketlerinin ise 64 tanesini(28,4) almıştır. Düşük hızda yüksek öncelikli paketlerin yüksek oranlarda paketi hedefe ilettiği görülmüştür. Paket atılmaları ise büyük oranda(%72,4) düşük öncelikli paketlerde yaşanmıştır. Paket bazlı gecikmeler incelediğinde önceliği yüksek olan paketlerin daha kısa sürede hedefe iletidiği sonucu çıkmıştır.

İkinci olarak araçlar yüksek hızda (30m/s) hareket ettirilerek düğüm ve paket bazında Çizelge 4.11'deki sonuçlar alınmıştır.

Çizelge 4.11. Senaryo 2 (b) 30m/s hızda simülasyon çıktıları

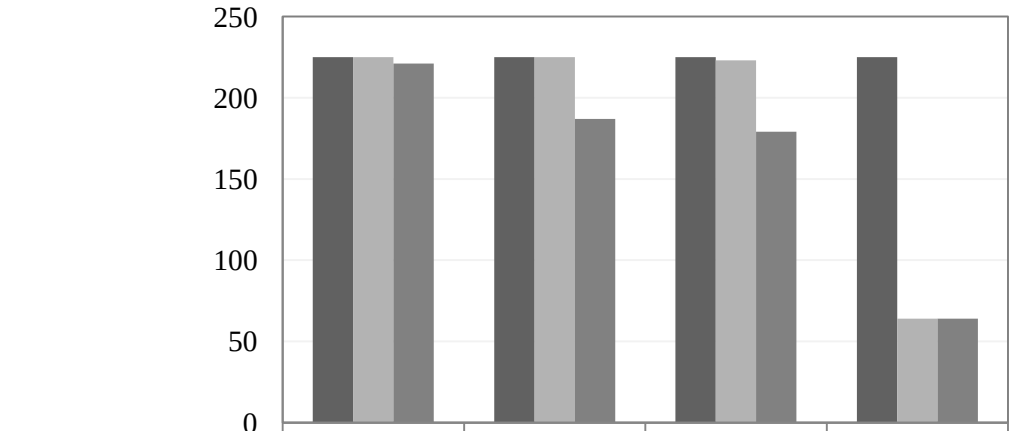
Paket türü	LCM	WM	TC	MM	Toplam
Paket uzunluğu	500	500	500	500	-
Kaynak araçlar	v0	v1	v2	v3	-
Üretilen paket sayısı	225	225	225	225	900
Atılan paket sayısı	4	38	45	161	248
İletilen paket sayısı	221	187	180	64	652
İletilen paket oranı (%)	98,2	83,1	80,0	28,4	72,4

Hedef araç v4	LCM	WM	TC	MM	Toplam
Alınan paket sayısı	221	187	179	64	651
Alınan paket oranı (%)	98,2	83,1	79,6	28,4	72,3
Paket bazlı ort. gecikme (s)	1,95	3,00	7,16	6,77	4,16

Kayıp paket sayısı	0	0	1	0	0
Kayıp paket oranı (%)	0	0	0	0	0

Senaryo 2 b'de kaynak araçlar (v0-LCM,v1-WM,v2-TC,v3-MM) 30m/s hızlarda toplamda 900 (%100) paket üretmiştir. Üretilen paketlerin 652 tanesi(%72,4) hedefe iletilirken 248 tanesi ise (%27,5) atılmıştır. Paket atılma sayıları öncelik azaldıkça artma göstermiştir. Hedef araç üretilen paketlerin 651 tanesini (%72,3) almıştır. Alınan paket sayıları ve oranları en fazla yüksek öncelikli paketlerde görülmüştür. Hedef araç LCM paketlerinin 221 tanesini (%98,2), WM paketlerinin 187 tanesini (%83,1), TC paketlerinin 179 tanesini (%79,6) ve MM paketlerinin 64 tanesini(28,4) almıştır. Hız parametresi artırıldığında toplamda hedef tarafından alınan paket miktarı (%9,6) azalmıştır. Atılan paket miktarı ise artma (%9,5) göstermiştir.

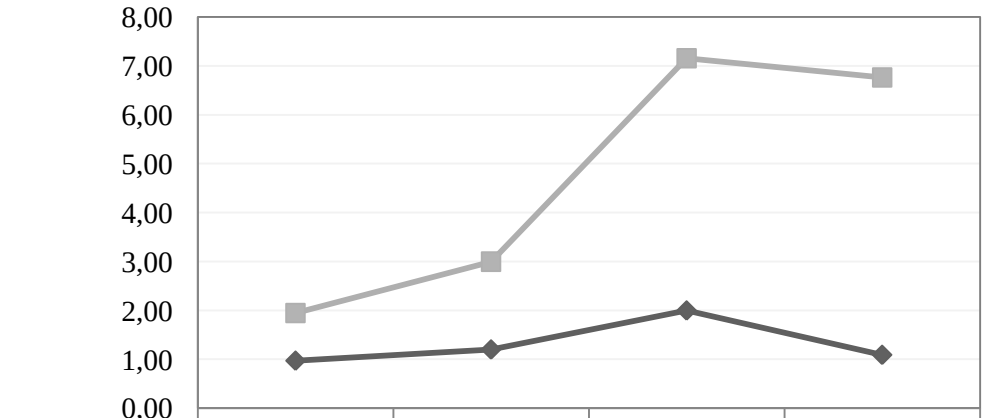
Hız değişimine göre kaynak araçlarca üretilen ve hedef araç tarafından alınan paket sayıları grafiği Şekil 4.10'da verilmiştir.



	v0	v1	v2	v3
Üretilen paket sayısı	225	225	225	225
Alınan paket sayısı (Hız 10m/s)	225	225	223	64
Alınan paket sayısı (Hız 30m/s)	221	187	179	64

Şekil 4.10. Senaryo 2 hız değişimine göre üretilen ve alınan paket sayıları

Düşük hızda (10m/s) paketler önceliklerine göre daha etkili iletildiğinden gecikme süreleri de önceliklerine göre şekillenmiştir. Hız parametresine göre paketlerin ortalama gecikme sürelerindeki değişim grafiği Şekil 4.11'de verilmiştir.



	LCM	WM	TC	MM
Hız 10m/s	0,97	1,20	2,00	1,09
Hız 30m/s	1,95	3,00	7,16	6,77

Şekil 4.11. Senaryo 2 hız değişimine göre paket bazlı ortalama gecikme süreleri

6. SONUÇ VE TARTIŞMA

Bu çalışmada araçsal tasarsız ağların bileşenleri, mimarisi, karakteristikleri, kablosuz erişim teknolojileri, zorlukları ve gereksinimleri, uygulama alanları, yönlendirme protokolleri, hareketlilik modelleri ve simülasyon ortamları detaylı olarak incelenmiştir. Araçsal tasarsız ağ türünün günümüzün ulaştırma alanında yaşanan sorunların çözümüne yönelik büyük katkılar sunacağı öngörülerek bu alandaki uygulama trafiklerini yönetecek öncelikli kuyruk yapısı önerilmiştir. Çalışmada önerilen kuyruk yapısı çeşitli senaryolarla test edilmiştir. Senaryolar farklı parametrelere göre değerlendirilmiştir.

Paket türlerine göre hedefe ulaşan paket oranları senaryo 1'de Drop tail ortalama olarak LCM paketlerini %78, WM paketlerini %90, TC paketlerini %80, MM paketlerini % 89 oranında hedefe ulaştırırken, VanetPriQ LCM paketlerini %98, WM paketlerini %84, TC paketlerini %69, MM paketlerini %51 oranında hedefe ulaştırmıştır. Sonuç olarak VanetPriQ yüksek öncelikli paketlerin hedefe ulaşma oranını artırmayı başarmıştır.

Paket türlerine göre gecikme etkisi için senaryo 1 a ve c incelendiğinde geliştirilen öncelik kuyruğu yapısı VanetPriQ birinci önceliğe sahip LCM paketlerini ortalama %21 oranında daha kısa sürede hedefe iletmiştir. WM, TC, MM paketleri ise sırasıyla %18, %24, %41 oranında daha geç iletilmiştir. Ancak gerçek hayattaki araçsal tasarsız ağ uygulamalarına göre paket boyutları farklılaşacağından senaryo 1 b'deki durum daha gerçekçi olacaktır. Bu nedenle senaryo 1 b için VanetPriQ yapısının paket gecikmelerine bakıldığında 1. önceliğe sahip LCM ve 2. önceliğe sahip WM paketlerinin yaklaşık %14 oranında daha kısa sürede hedefe iletildiği, TC ve MM paketlerinin ise hedefe daha geç iletildiği görülmüştür.

Kuyruk uzunluğu etkisi için senaryo 1 a (150000 byte) ve c (75000 byte) durumları değerlendirildiğinde kuyruk uzunluğu ağdaki trafik için yetersiz olduğunda atılan paket sayısı oranının VanetPriQ'da %13, Drop tail'de %15 arttığı, hedefe ulaşan paket oranın ise VanetPriQ'da %17,9, Drop tailde %13,6 oranında azaldığı tespit edilmiştir. Bu bilgilerden hareketle araç üstü üniteler için belirlenecek kuyruk uzunluğu, ağ yoğunluğuna, uygulama mesajları paket boyutuna ve kullanılacak servislere göre dikkatli bir şekilde belirlenmelidir.

Hız değişimlerinin araçsal ağ performansına etkisi düşük (10m/s) ve yüksek (30m/s) hızlı araçlarla test edilmiştir. Elde edilen sonuçlara göre araçsal ağ performansı hız değişiminden genel olarak alınan paket sayısı yönünden %9,6, atılan paket sayısı yönünden %9,5 oranında etkilenmiştir. Hız artışı bu performansı olumsuz yönde etkilerken, azalışı ise olumlu yönde etkilemektedir.

Elde edilen sonuçlar itibariyle geliştirilen VanetPriQ kuyruk yapısı paketlerin öncelik ve önem sıralarını dikkate almış, sabit ve hareketli durumlarda paket trafiklerini önceliklendirerek servis kalitesini artırmıştır.

Çalışmada araçsal tasarsız ağlardaki araçtan araca olan uygulamalar incelenmiş ve bu uygulamalara yönelik gerçek hayatta kullanılabileceği öngörülen trafik kaynakları ve paket tipleri geliştirilmiştir. Bu yönüyle uygulamalarda kullanılacak kaynaklar ve paketler çeşitlendirilmiştir.

Geliştirilen trafik türlerine göre paket önceliklerini dikkate alan ve ağ trafiğini istikrarlı bir şekilde yöneten yeni bir öncelikli kuyruk yapısı geliştirilmiştir. Bu sayede literatürde yer alan kuyruk yapılarına yeni bir alternatif daha sunulmuştur.

Çalışmada geliştirilen bileşenlerin VanetPriQ kuyruk yapısıyla birlikte kullanımı araçsal ağlarda ortaya çıkan sorunların çözümüne yönelik önemli katkılar sağlayacaktır.

KAYNAKLAR

1. Zeadally, S., Hunt, R., Chen, Y.-S., Irwin, A., & Hassan, A. (2010). Vehicular ad hoc networks (VANETS): status, results, and challenges. *Telecommunication Systems*, *50*(4), 217–241.

2. Taha, M. M. I. (2008). *Broadcasting Protocols in Vehicular Ad-Hoc Networks (VANETs)*, M. Sc. Thesis, Assiut University Department of Electrical Engineering, Assiut, EGYPT.

3. Shinde, S. S., & Patil, S. P. (2010). Various Issues in Vehicular Ad hoc Networks : A Survey, *International Journal of Computer Science & Communication (IJCSC), 1*(2), 399–403.

4. Lee, K.C., Lee, U., Gerla, M. (2010). Survey of Routing Protocols in Vehicular Ad Hoc Networks. In Watfa,M., Klinger, K., & Snavely, J. (Eds.), *Advances in Vehicular Ad-Hoc Networks : Developments and Challenges*, International Science Reference,Hershey,Newyork, pp. 149-170.

5. Rawashdeh, Z. Y., & Mahmud, S.M. (2011). Communications in Vehicular Networks. In Prof. Wang, X. (Ed.), *Mobile Ad-Hoc Networks: Applications,* InTech, (p. 514).

6. Sevimli, K. K., Soytürk, M.(2008). *Düşük Yoğunluklu Araçsal Ağlarda Zaman Gecikmeli Veri İletişimi*, IEEE 18. Sinyal İşleme ve İletişim Uygulamaları Kurultayında sunuldu, Diyarbakır.

7. Çağrıcı, G., ve Ayav, T.(2002) *İzmir yüksek teknoloji enstitüsü'nün bilgisayar ağı için bir servis kalitesi uygulamasi* , Türkiye Bilişim Derneği 19. Bilişim çalıştayında sunuldu,İstanbul.

8. Kumar,R.A., and Varshini, K.M. (2014). Multilevel priority packet scheduling scheme for wireless networks, *International Journal of Distributed and Parallel Systems (IJDPS,) 5*(1), 69–76.

9. Xu, H., Wang, X., Toh, C. K. (2005). *Comparative Analysis of Scheduling Algorithms in Ad Hoc Mobile Networking*. Paper presented at Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies *(PDCAT'05)*, 639–643.

10. Chun, B. G., and Baker, M. (2002). Evaluation of packet scheduling algorithms in mobile ad hoc networks. *Mobile Computing and Communications Review*, *6*(3), 36-49.

11. Metin, B., & Deliç, H. (2002,18-22 Aralık). *IP ağlarında kuyruklama yöntemlerinin servis kalitesine etkisi* Bursa Elektrik, Elektronik ve Bilgisayar Mühendisliği Sempozyumu ve Fuarında sunuldu, Bursa.

12. Qian, Y., Lu, K., & Moayeri, N. (2008, November 30–December 4). *A secure vanet mac protocol for dsrc applications*. Paper presented at Global Telecommunications Conference, New Orleans,USA,

13. Mi, J., Liu, F., Xu, S., & Li, Q. (2008). *A Novel Queue Priority Algorithm for Real-Time Message in VANETs*. Paper presented at 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA*)*, 919–923.

14. Mu'azu, A. A., Tang, L. J., Hasbullah, H., Lawal, I. A., & Shah, P. A. (2014). *Real-time massage differentiation with priority data service flows in VANET*. Paper presented at 2014 International Conference on Computer and Information Sciences (ICCOINS), 1–6.

15. Al-Sultan, S., Al-Doori, M. M., Al-Bayatti, A. H., & Zedan, H. (2014). A comprehensive survey on vehicular Ad Hoc network. *Journal of Network and Computer Applications*, *37*, 380–392

16. Tayal, S., & Triphathi, M. R. (2012). *VANET-Challenges in Selection of Vehicular Mobility Model.* Paper presented at 2012 Second International Conference on Advanced Computing & Communication Technologies, 231–235.

17. Misra, S., Woungang, I., & Chandra Misra, S. (Eds.). (2009). *Guide to Wireless Ad Hoc Networks*, London: Springer London.

18. Sharef, B. T., Alsaqour, R. a., & Ismail, M. (2014). Vehicular communication ad hoc routing protocols: A survey. *Journal of Network and Computer Applications*, *40*, 363–396.

19. Koçkan, C. (2008). *Taşıtlar Arası Haberleşme*. Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

20. Sivasakthi, M., and Suresh, S. R. (2013). Research on vehicular ad hoc networks (VANETs):An overview, *Journal of Applied Sciences and Engineering Research*, 2(1), 23–27.

21. Bi, J. (2009, June). *Research on Vehicular Ad Hoc Networks*. Paper presented at Chinese Control and Decision Conference, China, 4430–4435.

22. Sevimli, K. K., & Soytürk, M.(2010,Şubat), *Araçsal Ağlar,* Akademik Bilişim 2010 Konferansında sunuldu, Muğla.

23. Yilmaz, E., & Öztürk, E. (2007, 18-19 Ekim). *Yeni Nesil Kablosuz İletişim Teknolojileri Karşılaştırmalı Analizi*, EMO III. İletişim Teknolojileri Ulusal Sempozyumunda sunuldu (İTUSEM 2007), Adana.

24. Li, Y.J. (2012). An Overview of the DSRC / WAVE Technology. In *Quality, Reliability, Security and Robustness in Heterogeneous Networks,* (pp. 544–558).

26. Sevimli, K. K., (2010). *Düşük Yoğunluklu Araçsal Ağlarda Gecikmeli İletişimin Sağlanması*, IEEE 18.Sinyal isleme ve iletisim uygulamalari kurultayinda sunuldu,Diyarbakır.

27. Carter, A. (2005). Vehicle Safety Communications Project Task 3 Final Report Identify Intelligent Vehicle Safety Applications Enabled by DSRC, *National Technical Information Service, Springfield, Virginia*,150.

28. Alotaibi, E., & Mukherjee, B. (2012). A survey on routing algorithms for wireless Ad-Hoc and mesh networks. *Computer Networks*, *56*(2), 940–965.

29. Li, F., Wang, Y., & Carolina, N. (2007). Routing in Vehicular Ad Hoc Networks : A Survey, (June), *IEEE Vehıcular Technology Magazine,* 12–22.

30. Paul, B. (2011). VANET Routing Protocols : Pros and Cons, *Internatiomal Journal of Computer Applications*, *20*(3), 28–34.

31. Kara, R. (2009, 13-15 Mayıs). *Pozisyon Tabanlı Ad Hoc Yönlendirme Algoritmalarında Bulanık Mantık İle Yol Seçimi*, 5. Uluslararası İleri Teknolojiler Sempozyumunda sunuldu, Karabük.

32. Bilal, S. M., Bernardos, C. J., & Guerrero, C. (2013). Position-based routing in vehicular networks: A survey. *Journal of Network and Computer Applications*, *36*(2), 685–697.

33. Pan, H., Jan, R., Jeng, A., Chen, C., & Tseng, H. (2011). *Mobile gateway routing for vehicular networks*. Proceedings of the 8th IEEE Asia.

34. Hassan, A. (2009). *VANET Simulation*. Master's Thesis in Electrical Engineering Halmstad Üniversitesi.

35. Grzybek, A., Seredynski, M., Danoy, G., & Bouvry, P. (2012). Aspects and trends in realistic VANET simulations. *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 1–6.

36. Martinez, F. J., Toh, C. K., Cano, J., Calafate, C. T., & Manzoni, P. (2009). A survey and comparative study of simulators for vehicular ad hoc networks (VANETs), *Wireless Communications and Mobile Computing*,11(7)813-828.

37. Stanica, R., Chaput, E., & Beylot, A.-L. (2011). Simulation of vehicular ad-hoc networks: Challenges, review of tools and recommendations. *Computer Networks*, *55*(14), 3179–3188.

38. Çavuşoğlu, Ü., & Zengin, A. (2012). NS-2 ve NS- 3 Ağ Simülatörlerinin Ölçeklenebilirlik Analizi ve Karşılaştırma, *Bilişim Teknolojileri Dergisi*, 5(3) 41–49.

39. Zeng, X., Bagrodia, R., & Gerla, M. GloMoSim: a library for parallel simulation of large-scale wireless networks. *Proceedings. Twelfth Workshop on Parallel and Distributed Simulation PADS '98* , 154–161.

40. İnternet: Vehicular in Network Simulation. http://veins.car2x.org/documentation/, Son Erişim Tarihi: 17.11.2014.

41. Noori, H. (2012, November). *Realistic Urban Traffic Simulation as Vehicular Ad-Hoc Network (VANET) via Veins Framework,* Presented at Proceedıng Of The 12th Conference Of Fruct Association.

42. Mangharam, R., Walker, D., Rajkumar, R., Mudalige, P., & Bai, F. (2006). GrooveNet : A Hybrid Simulator for Vehicle-to-Vehicle Networks. *School of Engineering and Applied Science Real-Time and Embedded Systems Lab.*

43. Tanyeri, U. (2009). *Mobil Kablosuz Ağlarda Veri Kayıplarının Simulasyon Ortamında Farkedilebilir Hale Getirilmesi*. Yüksek Lisans Tezi, Gazi Üniversitesi Bilişim Enstitüsü, Ankara.

44. Ros, F. J., Martinez, J. A., & Ruiz, P. M. (2014). A survey on modeling and simulation of vehicular networks: Communications, mobility, and tools. *Computer Communications*, 43, 1-15.

45. Issariyakul, T., & Hossain, E. (2012). *Introduction to Network Simulator NS2* (Second Edition) New York USA, Springer, pp. 535..

46. US National Highway Traffic Safety Administration (2011). *Vehicle Safety Communications-Applications (VSC-A)* Final Report : Appendix Volume 3 Security, 8-72.

47. IEEE Vehicular Technology Society (2010). *IEEE Standard for Wireless Access in Vehicular Environments (WAVE)-Networking Services IEEE Vehicular Technology Society* (Vol. 2010).

48. Campolo, C., Molinaro, A., & Vinel, A. (2011). Understanding the Performance of Short-lived Control Broadcast Packets in 802.11p / WAVE Vehicular Networks, 102–108.

49. İnternet: OpenStreetMap, http://www.openstreetmap.org/, Son Erişim Tarihi: 10.01.2015

EKLER

EK- 1. Ns-2.35 (Network simülator 2.35) kurulumu

1. Adım: NS-2 simülatörünü Ubuntu 12.04 LTS işletim sistemi üzerinde kurabilmek için öncelikle http://www.isi.edu/nsnam/ns/ adresinden Ns-2.35 sürümünün indirilmesi gerekmektedir.

2. Adım: İndirilen ns-allinone-2.35.tar.gz isimli sıkıştırılmış klasör bilgisayarın home dizinine kopyalanarak burada açılmalıdır. (/home/ibrahim/)

3. Adım: Terminal ekranı (Ctrl + Alt + T) açılarak öncelikle güncellemeler yapılmalı ve gerekli kütüphanelerin kurulumu için sırasıyla aşağıdaki komutlar işletilmelidir.

```
sudo apt-get update
sudo apt-get install tcl8.5-dev tk8.5-dev
sudo apt-get install build-essential autoconf automake
sudo apt-get install perl xgraph libxt-dev libx11-dev
libxmu-dev
```

4. Adım: Bu adımda Ns-2 klasörünün zipten çıkarıldığı yere terminalden ***cd*** komutlarıyla erişilmeli ve ***install*** komutu işletilmelidir.

```
cd /home/ibrahim/ns-allinone-2.35
sudo ./install
```

5. Adım: Ortam değişkenlerinin ve kütüphane yollarının ayarlanması gerekmektedir. Bu ayarların yapılacağı ***.bashrc*** dosyasına ulaşmak için aşağıdaki komut çalıştırılır.

```
gedit ~/.bashrc
```

Daha sonra ***.bashrc*** dosyasının sonuna aşağıdaki kodlar eklenerek kütüphane yolları ayarlanır.

```
# LD_LIBRARY_PATH
OTCL_LIB=/your/path/ns-allinone-2.35/otcl-1.13
NS2_LIB=/your/path/ns-allinone-2.35/lib
X11_LIB=/usr/X11R6/lib
USR_LOCAL_LIB=/usr/local/lib
```

EK- 1. (devam) Ns-2.35 (Network simülator 2.35) kurulumu

```
export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:
$X11_LIB:$USR_LOCAL_LIB

# TCL_LIBRARY

TCL_LIB=/your/path/ns-allinone-2.35/tcl8.4.18/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIB:$USR_LIB

# PATH

XGRAPH=/your/path/ns-allinone-2.35/bin:/your/path/ns-
allinone-2.35/tcl8.4.18/unix:/your/path/ns-allinone-
2.35/tk8.4.18/unix
NS=/your/path/ns-allinone-2.35/ns-2.35/
NAM=/your/path/ns-allinone-2.35/nam-1.14/
PATH=$PATH:$XGRAPH:$NS:$NAM
```

6. Adım: Ns-2.35 ana klasörüne girilir (ns-allinone-2.35/ns-2.35$) ve ***validate*** komutu çalıştırılır.

```
./validate
```

7. Adım: Son adımda Ns-2 nin çalışması test edilir, bu işlem için ns-allinone-2.35/ns-2.35/bin içerisinde ***ns*** komutu işletilir

```
ns/home/ibrahim/ns-allinone-2.35/bin/ns
```

Ekrana **%** işareti geldiğinde kurulum başarılı bir tamamlanmıştır.

EK- 2. Paket izi format metodu örnek kodu (format_lcm)

1. Adım: Paket iz formatı metot tanımlamaları */trace/cmu_trace.h* dosyasına eklendi

../cmu_trace.h

```
#ifndef __cmu_trace__
#define __cmu_trace__

#include "trace.h"

#include "god.h"

…….. // Arada kalan kod blokları

class CMUTrace : public Trace {
public:
     CMUTrace(const char *s, char t);

     ……. // diğer kod tanımlamaları
private:

     void format_rtp(Packet *p, int offset);
     void format_lcm(Packet *p, int offset); // format lcm
     void format_wm(Packet *p, int offset); // format wm
     void format_tc(Packet *p, int offset); // format tc
     void format_mm(Packet *p, int offset); // format mm
     void format_tora(Packet *p, int offset);
            void    format_aodv(Packet *p, int offset);


};

#endif /* __cmu_trace__ */
```

EK- 2. (devam) Paket izi format metodu örnek kodu (format_lcm)

2. Adım: Metodlar */trace/cmu-trace.cc* dosyasına yazıldı.(format_lcm)

```
void
CMUTrace::format_lcm(Packet *p, int offset)
{
      struct hdr_cmn *ch = HDR_CMN(p);
      struct hdr_rtp *rh = HDR_RTP(p);
      struct hdr_ip *ih = HDR_IP(p);
         Node* thisnode = Node::get_node_by_address(src_);

      //
         int dst = Address::instance().get_nodeaddr(ih->daddr());

      if (dst == src_){
            //
            if (thisnode->energy_model() &&
                thisnode->energy_model()->powersavingflag()) {
                  thisnode->energy_model()->set_node_state
(EnergyModel::INROUTE);
            }
         }
              //
      if (pt_->tagged()) {
            sprintf(pt_->buffer() + offset,
                  "-lcm:s %d -lcm:f %d -lcm:o %d ",
                  rh->seqno_,
                  ch->num_forwards(),
                  ch->opt_num_forwards());
      } else if (newtrace_) {
            sprintf(pt_->buffer() + offset,
                  "-Pn lcm -Pi %d -Pf %d -Po %d ",
                  rh->seqno_,
                  ch->num_forwards(),
                  ch->opt_num_forwards());
      } else {
            sprintf(pt_->buffer() + offset,
                  "[%d] %d %d",
                  rh->seqno_,
                  ch->num_forwards(),
                  ch->opt_num_forwards());
      }
}
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı,adı	: KÖK, İbrahim
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 01.01.1984, K.Maraş
Medeni hali	: Bekâr
Telefon	: 0(312)202 3813
e-mail	: ikok@gazi.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek lisans	Gazi Üniversitesi Bilgisayar Bilimleri A.B.D.	Devam ediyor
Lisans	Gazi Üniversitesi Elektronik ve Bilgisayar	2010

İş Deneyimi

Yıl	Yer	Görev
2014 – Devam ediyor	Gazi Üniversitesi Bilişim Enstitüsü	Araştırma Görevlisi
02/06–2014	Pamukkale Üniversitesi Bilgisayar Mühendisliği	Araştırma Görevlisi
2011 – 2014	Yargıtay Başkanlığı Yargıtay Cumhuriyet Başsavcılığı	Programcı

Yabancı Dil

İngilizce

Programlar

Erasmus Staj Hareketliliği Programı, Participant, Berlin, ALMANYA (Haziran-Ağustos 2009)

İlgi Alanları

Mobil Kablosuz Ağlar, Araçsal Tasarsız Ağlar, Bulut Bilişim

GAZİ GELECEKTİR..